NB-GPC3

USER MANUAL

NB-GPC3 *User Manual*

Part Number 1E-04-00-0124

© 2007 American Auto-Matrix™

This document is protected by copyright and is the property of American Auto-Matrix. It may not be used or copied in whole or in part for any purpose other than that for which it is supplied without authorization. This document does not constitute any warranty, expressed or implied.

Every effort has been made to ensure that all information was correct at the time of publication. American Auto-Matrix reserves the right to alter specifications, performance, capabilities and presentation of this product at any time.

American Auto-Matrix and Auto-Matrix are trademarks of American Auto-Matrix and are not to be used for publication without the written consent of American Auto-Matrix.

All other brand names or product names are trademarks or registered trademarks of their respective companies or organizations.

WORLD HEADQUARTERS

American Auto-Matrix
One Technology Lane
Export, Pennsylvania 15632-8903 USA
Tel (1) 724-733-2000
Fax (1) 724-327-6124
Email aam@aamatrix.com
www.aamatrix.com

Updated 6/29/07

Updated Remote I/O portion of §2 to discuss STAT Display Sequence limits on loaded STATbus networks.

Updated 3/16/2007

- Corresponds to v1.30 series firmware
- Document uses new date revision scheme for Technical Documentation
- §3.19.1 Added Zone Scheduling Feature
- Fixed various bit position errors in bitmap definitions through user documetation
- Appendix A - Added Zone Scheduling attributes to F900 Schedule Summary Channel

Version 1.1

(Corresponds to firmware version 1.10)

- Removed SBC-STAT Features chapter. Information on using STATs is now included as part of the STAT User Manual (part #1E-04-00-0103).
- Changed the number of digital inputs to 8 in §1.1.1, §1.2.2, §2.4, §3.6.14, §3.7, §A.7, and Appendix C.
- §1.3.1 - Corrected resoltuion of outputs.
- §1.3.2 - Clarified electrical ratings for outputs on the STATbus.
- §1.4 - Added SBC-RH1, SBC-RH3, and SBC-RHT to the list of STATbus device which can be used with the NB-GPC3.
- §1.5.3 - Required power supply changed to NEC class 2, 22-29 VAC, 50/60 Hz, 4.16 A maximum.
- Removed §1.5.4 as the *NB*-GPC3 does not have any outputs.
- Revised §2.2.2 to reflect changes in PTC values and explain power supply options.
- §2.3 - Removed SSB-DI1 from the list of IOX modules that can be used with UIs.
- §2.3 - Corrected thermistor listing to be Precon type III.
- Corrected Note in §2.3.1, §2.3.2, §2.3.3, §2.3.4, and §2.3.5.
- §2.4 - Remove reference to onboard Digital Inputs. Changed outputs to inputs in opening paragraph.
- §2.4.1 Added Note regarding **MD** and counting run hours.
- §2.5.1 - Corrected figures 2-20, 2-21, 2-22, and 2-23. Revised text to clarify the function of description of **MN**, **MX**, **min_pres_value**, **max_pres _value**.
- §2.7 - Added introductory paragraph.
- §2.7.1 - Changed all L1,L2 references to X1, X2 to match product label.
- §2.7.3.1 - Added humidity stats to list of stats.
- §2.7.3.1 - Added list of IOX modules.
- §2.7.4 - Corrected GPC STATbus power limits in Note.
- §2.7.6 - Clarified steps for assigning GID numbers.
- Changed default baud rate to 38400 in §3.1 and §A.1.
- §3.1 - Added description of **object_identifier** and note regarding its usage.
- §3.1 - Corrected name for **PF**.
- §3.1 - Clarified the meaning of **CR**=2.
- §3.1 - Added statement that fire broadcasts are sent every ten seconds.
- §3.3 - Added Note regarding maximum program file size of 8k.
- Added description for **reliability** to tables in §3.3 and §A.3.
- Added **description** to tables in §3.4, §A.4, §3.5, and §A.5.
- Added descirption of correlation between **object_name** and the type of file loaded in §3.5 and §A.5.
- Added options for **file_type** to tables in §3.5 and §A.5.
- Removed **event_time_stamps** in §3.6, §3.7, §A.6, and §A.7.
- Added new options for **SM** in §3.6, §3.9, §3.16.3, §3.17, §3.18, §A.6, §A.9, §A.16, §A.17, and §A.18.
- §3.8 - Removed **deadband**, **event_enable**, **high_limit**, **low_limit**, **notification_class**, **limit_enable**, **notify_type**, and **event_time_stamps** properties.
- §3.8 - Added restriction that **MN** < **present_value** < **MX**.
- Removed **time_delay** from §3.8, §A.8, §3.9, and §A.9.

- **DT** from §3.8 and §A.8.
- Added **OU**, **resolution**, and **acked-transitions** to §3.8 and §A.8.
- Added options to select DIs as inputs in **IL** in §3.8, §A.9, §3.16, §A.8, §A.9, and §A.16.
- §3.11 - Added description of **OI** to table.
- Added note regarding interlocking and assignment to a control loop in §3.8 and §3.15.
- Removed **RS** from porpoerty list in §3.9.
- Corrected number of Digital Inputs to 8 in **CV**, **IP**, **LP**, **OI**, and **RE** in §3.12 and §A.12.
- Added description of **CF** in §3.13 and §A.13.
- Removed all referenes to **GI** in §3.14 and §A.14.
- §3.14 - Added information about removing assigned channel or attribute. Removed note.
- Added description of **BO** to §3.16, and §A.16.
- §3.16.2 - Corrected Figure 3-8.
- §3.18 - Corrected Figure 3-11.
- §3.19 - Clarified the meaning of **C1** through **C8**.
- Corrected entry for **CV** and **C1** through **C8** in tables in §3.19 and §A.19.
- §3.22 - Revised description of the procedure for calculating **X1** through **XB** for Piecewise Curves.
- §3.23 - Corrected the description of **OP**=3.
- Corrected the description of **IC**, **IA**, and **RB** in §3.27 and §A.27 through §3.34.
- Corrected descriptions of **AV**, **HV**, and **LV** in §3.25 and §A.25.
- Added description of **CE** to table in §3.27 and §A.29.
- §4.1 - Corrected Figure 4-1.
- §4.2 - Corrected reference to table. Fixed formatting in Table 4-1. Removed PRINT statement from Table 4-1. Corrected FIX syntax.
- Appendix A - fixed typos in description of **GI** for inputs and outputs.
- Appendix A - fixed all descriptions of bitmaps to begin with bit 0.
- §A.1 - Corrected default for **object_identifier**, **time_synchroniazation_recipients**.
- §A.7 - Removed **AS**.
- §A.11 - Added **OI**. Corrected options for **IP**, **LP**, and **RE**.
- §A.11 - Corrected description for **V1** through **VO**.
- §A.13 - Corrected descriptions for **AM**, **CF**, and **RE**.
- §A.17 - Corrected descriptions for **OB**.
- Appendix B - Replaced SPL Error Table.
- Appendix C - Corrected property lists in the Standard Object Types Supported section and Proprietary Propoerties section.

Version 1.0 - Initial Manual Release

This manual describes the installation and operation of the American Auto-Matrix *NB*-GPC3 controller. This document is divided into the following sections:

- One: Overview, describing the features of the *NB*-GPC3 and presenting the specifications for the controller
- Two: Wiring, Installation & Usage, detailing the wiring and installation procedures as well as configuration information
- Three: Object & Property Descriptions, listing all properties present in each object in the controller and describing their use
- Four: SPL Support, giving a brief overview of SPL and how SPL programs are used by the *NB*-GPC3
- Appendix A: Proprietary Objects & Properties, enumerating the proprietary properties in every object along with their object identifier, object type, storage and default values (if any)
- Appendix B: SPL Error Codes, describing the error codes that may be encountered when working with SPL programs
- Appendix C: PICS, describing the conformance of the *NB*-GPC3 to the BACnet standard

This document contains certain style and formatting conventions for conveying information in a clear and concise manner:

- Property names are shown in **bold**. For example: **present_value**.
- Menu commands appear with a “>” symbol between levels. For example: File>Open.
- *Italics* indicate a section of this manual or another publication.
- The following formats are used to highlight important information:

NOTE
Notes indicate important information and appear in boxes with this format separated from the running text.

CAUTION
Cautions indicate information that may prevent serious system or user problems and appear in boxes with this format separated from the running text.

WARNING
Warnings indicate information that may prevent personal injury or equipment damage and appear in boxes with this format separated from the running text.

SECTION 1: OVERVIEW

This section describes the NB-GPC3, outlines key components and features, and provides operating specifications for the controller.

IN THIS SECTION

1.1 WHAT IS THE *NB*-GPC3?

As the ultimate Native Series® controller, the *NB*-GPC3 offers complete stand-alone control as well as full peer-to-peer capabilities with other devices on the same physical BACnet MS/TP network. The flexibility of inputs, outputs and programmable capabilities allows the *NB*-GPC3 to be used in a wide variety of applications, including large built up air handling units, central plant control, multiple boilers, optimization, pump control, and load shed algorithms to name a few.

1.1.1 FEATURES OF THE *NB*-GPC3

- Standalone or networked operation
- Twenty-four (24) universal inputs
- Eight (8) digital inputs
- Twelve (12) analog outputs
- Twelve (12) digital outputs
- Four (4) STATbus channels
- Easy configuration and firmware updates via software tools such as *NB*-Pro
- RS-485 communications rate up to 115.2k
- Battery-backed real time clock and memory
- Self-diagnostic circuits and LED indicators for power, I/O, network activity, processor and programs.

1.1.2 INPUTS, OUTPUTS VS. POINTS

The *NB*-GPC3 has a number of features that make it different from previous American Auto-Matrix controllers, the most notable of which is its ability to use remote I/O devices connected via STATbus. Remote I/O behaves in the same way as on-board I/O, except that it is located remotely from the controller. Because they are not on-board, each remote device requires a unique address, known as a Global Identification (GID) number so that the controller may recognize and direct communications to it. The *NB*-GPC3 internally maps these GID numbers to its inputs and outputs.

This means that a distinction must be made between the internal representation of an input or output and the physical connection where the device is connected to the controller, be it on-board or remote. When a reference is made to an input or output, it is assumed to be the internal representation of that input or output in the controller. When a reference is made to the physical connection for a given input or output, the term “point” will be added. For example, “Digital Input Point” would refer to the connection where the digital input was connected. In this way, “Universal Input 14” would refer to the fourteenth universal input object in the controller while “Universal Input Point 7” would be the seventh physical connection for a universal input. The *NB*-GPC3 internally associates physical connections, such as “on-board universal input point 7”, with internal representation, such as “Universal Input 7”.

1.2 INPUTS

The *NB*-GPC3 has no on-board I/O. All inputs and outputs to the *NB*-GPC3 will be connected via the four (4) STATbus channels. Each STATbus channel is capable of handling a maximum of thirteen (13) devices each. The *NB*-GPC3 has twenty-four (24) universal inputs and can map input points on the STATbus to these addresses. This mapping assigns an input to an address.

1.2.1 UNIVERSAL INPUTS

Within the *NB*-GPC3, there are twenty-four (24) universal inputs. The controller has the ability to map input points on the STATbus onto these twenty-four universal inputs. Using NB-Pro, the controller can be configured to associate the input signal from a given hardware point to a universal input.

Each universal input may be configured to behave as a digital (on/off) input, 0-5 volt, 0-10 volt, or 0-20mA scaled linear input, thermistor, or piecewise linear approximation. All ranges may be inverted for reverse-reading sensors. Thermistor input points may be configured to read either Fahrenheit or Celsius degrees over a temperature ranges from -30.0 to 230.0°F (-34.0 to 110.0°C). Piecewise linear approximation uses a programmable table for scaling. Two separate tables are available, each being defined by ten linear segments which approximate a characteristic curve. Input polarity is selectable for points configured as digital inputs. Each point configured as an analog input has programmable high and low alarm limits. If point is used as a digital input, alarms can be generated for off-to-on transitions, on-to-off transitions or both.

1.2.2 DIGITAL INPUTS

The *NB*-GPC3 has eight (8) digital inputs which are updated every 100 ms and are capable of measuring a signal with a pulse width as short as 50ms in pulse counter mode. The digital inputs have configuration settings similar to those of a universal input configured to read a digital signal. The digital inputs, however, operate at a much higher frequency allowing them to be used for such application as flow totalization and flow metering.

1.3 OUTPUTS

1.3.1 ANALOG OUTPUTS

The *NB*-GPC3 has a total of twelve (12) analog outputs that allow analog output points to be mapped from the STATbus.

The outputs can be configured for either a 0-10V or 0-20mA signal with 12 bits of resolution. In addition, each output can be configured as either normal or reverse acting. The outputs can be controlled automatically, manually or through an SPL program. All outputs allow for interlocking and interlock failure alarming.

1.3.2 DIGITAL OUTPUTS

The *NB*-GPC3 has a total of twelve (12) digital outputs that allow digital output points to be mapped from the STATbus.

Outputs provide on/off control for devices such as fans, valves using SSB-DO1, SSB-DO2, SSB-DO1-I, and SSB-DO2-I modules which use relays rated at 250 VAC/DC at up to 10 A. All digital outputs can be configured to enforce a minimum cycle time and each output allows for runtime alarming.

1.4 STATBUS

STATbus is an open-topology network that allows flexible connection of up to thirteen STATbus devices per channel using a non-polar, twisted pair cable. This provides unprecedented flexibility in the installation and wiring of I/O devices to the *NB*-GPC3. Substantial saving can be realized in both wiring and installation costs as compared to conventional sensors. STATbus uses digital communications signals, giving it a higher level of noise immunity than conventional, analog sensors.

NOTE

The *NB*-GPC3 may only use digital STATs or digital relative humidity sensors on it's STATbus. Analog STATs may not be used with the *NB*-GPC3. You may use any combination of SBC-STAT1D, SBC-STAT2D, SBC-STAT3, SBC-RH1, SBC-RH3, or SBC-RHT devices.

1.4.1 INPUTS ON THE STATBUS

The *NB*-GPC3 has twenty four Universal Inputs and four Pulse Inputs available for mapping STATbus devices. By connecting different IOX modules, the *NB*-GPC3 can be configured to have a nearly limitless combination of inputs up to the limits stated above.

1.4.2 OUTPUTS ON THE STATBUS

Up to twelve analog and twelve digital outputs may be connected to the *NB*-GPC3 via STATbus channels. These devices will behave, in all respects, like a normal output. The only difference will be to which network the device is physically connected.

1.5 SPECIFICATIONS

1.5.1 NETWORKING

- **line signaling:** RS-485
- **wiring:** shielded, twisted pair
- **network protection:** dual tranzorbs, PTC
- **communications speed:** up to 115.2k baud, selectable termination resistor
- **network configuration:** multidrop bus, per RS-485 specification and practice
- **communications protocol:** BACnet MS/TP

1.5.2 TERMINATIONS

- Pluggable terminal blocks for inputs, outputs, power, network, and STATbus connections.

1.5.3 POWER REQUIREMENTS

- NEC Class 2 transformer
- 22-29 VAC, 50/60 Hz, 4.16 A max
- PTC protection

1.5.4 OPERATING ENVIRONMENT

- **temperature range:** 32-122°F (0-50°C)
- **humidity range:** 0-80% RH, non-condensing
- **altitude:** up to 2000m

1.5.5 BATTERY BACKUP

- Only Sanyo CR-1/3N or Duracell CL-1/3N battery rated 3.0 VDC is to be used. (Not user replaceable).

1.5.6 DIMENSIONS

- **size:** 8.407 x 6.5 x 1.25 in. (20.83x16.51x3.18 cm)
- **shipping weight:** 3 lb. (1.36 kg)

1.5.7 AGENCY APPROVALS

- UL Listed Management Equipment, Energy (PAZX) UL standard 916
- UL Recognized Temperature-Indicating and Regulating Equipment – Component (XAPX2) UL standard 873
- Complies with FCC Part 15, subpart B, for Class B Computing Device
- Complies with CE Directive and Standards

SECTION 2: WIRING, INSTALLATION & USAGE

This section describes how to make the necessary connections for power and network communications to the NB-GPC3. Connection of input and output devices, as well as the use of remote I/O on the STATbus is covered as well.

IN THIS SECTION

2.1 INTRODUCTION

This section will explain the steps necessary to successfully mount the *NB*-GPC3, supply power to the controller, connect the controller to the communications network and connect and configure the controller's inputs and outputs.

2.2 INSTALLATION

IMPORTANT

If the equipment is used in a manner not specified by AAM, the protection provided by the equipment may be impaired.

There are three main steps to installing the *NB*-GPC3. First the controller must be physically mounted. Second, the controller must be connected to a suitable power supply. Third, the controller must be connected to the RS-485 communications network.

NOTE

All equipment must be installed by a licensed electrician in accordance with the NEC/CEC and local codes.

2.2.1 MOUNTING THE *NB*-GPC3

The *NB*-GPC3 should be mounted to a site which provides access to a stable 24 VAC power transformer from which the controller will draw its power. The temperature of the mounting location must be between 32º F and 122º F (0º C to 50º C) with a relative humidity of 0-80% non-condensing.

The mounting area should be flat and unobstructed by other equipment or machinery, free of moisture, and located away from potential leakage.

IMPORTANT

The case of the *NB*-GPC3 must be connected to a reliable earth ground.

The mounting dimensions for the *NB*-GPC3 are shown in Figure 2-1.

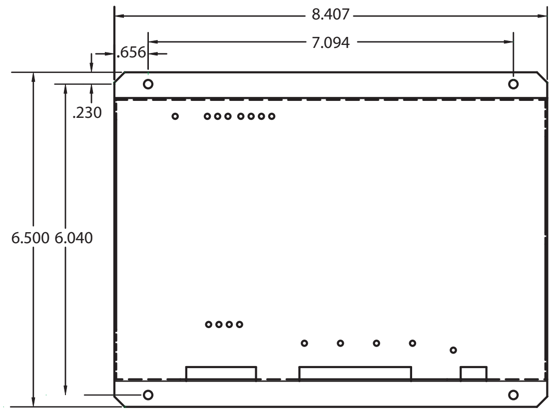

Figure 2-1: Mounting Dimensions for the NB-GPC3 (all dimensions in inches)

NOTE
When installing the *NB*-GPC3 make sure that there is sufficient room to allow insertion and removal of the terminal block plugs.

2.2.2 POWERING THE *NB*-GPC3

The *NB*-GPC3 can be used in a variety of configurations. Because of this, the power requirements of the controller can vary greatly and are very much application specific. There are two main parts of the *NB*-GPC3 which require power: the controller itself and the power outputs on the STATbus ports.

WARNING
The *NB*-GPC3 must be used with an approved external disconnect device rated for at least 24 VAC, .65 A. The disconnect device must be located near the equipment.

NOTE
All American Auto-Matrix products are listed as UL Class 2 devices. To maintain conformance with that listing, all transformers connected to them must be UL listed, Class 2 devices.

The *NB*-GPC3 requires a 22-29 VAC, 50/60 Hz NEC Class 2 transformer capable of providing a minimum of .65 A. The transformer is connected to the *NB*-GPC3 at pins 43 and 44 on terminal block TB12 as shown in Figure 2-2.

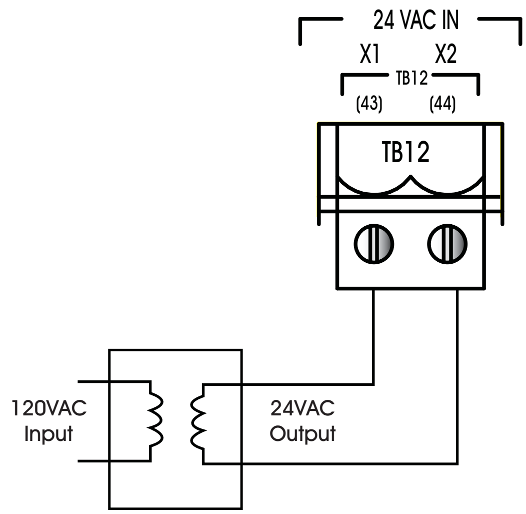

Figure 2-2: Connecting Power to the NB-GPC3

A .65 A power supply will provide only enough current to power the controller itself. Using a transformer of this size you should not use the power outputs on the STATbus connectors. If your application includes STATbus device which require power, then you will require a power supply capable of delivering more than the .65 A minimum. The *NB*-GPC3 is can be used with an NEC Class 2 power supply up to a maximum of 3.65 A (88 VA).

A current limit of 3.0 A total from all of the STATbus ports is enforced through a power management circuit incorporated in the controller. This may be used to provide power for IOX modules and any devices connected to them.

If you choose to use powered STATbus devices and do not want to simply use the largest rated transformer, you must determine whether the *NB*-GPC3's power supply will be sufficient and if you will need to supplement that power with an external transformer.

You need to determine the total current draw for all the powered STATbus devices and any devices, such as actuators, relays, sensors, etc., that you plan to connect to your STATbus devices.

STATbus device current = Current to power device + Current draw of actuator(s)/relay(s)/sensor/etc.

The current required to power the various STATbus devices is given in Table 2-1. The current draw for the input or output device(s) you connect to the STATbus devices should be found in the manufacturer's data provided with the device.

Table 2-1 Current to Power Various STATbus Modules

Module	Nominal Current Draw
SSB-UI1	75 mA
SSB-AO1	75 mA
SSB-DI1	75 mA
SSB-IOX1	125 mA
SSB-DO1, SSB-DO1-I, SSB-DO2, SSB-DO2-I	(See NOTE below)

NOTE

The power connection on the SSB-DO1, SSB-DO1-I, SSB-DO2, and SSB-DO2-I modules is only used to energize the relay coil(s) on those modules. They draw their operating power from the SSB pins and, for the purpose of calculating the total current draw, you only need to consider the power required to actuate the relay coils. Each relay coil requires 50 mA so an SSB-DO1 and SSB-DO1-I would add 50 mA per device to the total current requirement whereas the SSB-DO2 and SSB-DO2-I would add 100 mA per device.

The power rating of the transformer is found by simply adding together the power required by all devices on the STATbus and adding the .65 A required to run controller. In other words:

Required Supply Current (Amps)= .65 + STATbus device current

If the total current draw for all devices connected via the STATbus is greater than 3.0 A, then your application will require a separate, external transformer to provide the additional power needed. This transformer should be rated to handle the total current draw for all of your STATbus devices.

If you are using an external transformer to power STATbus devices, you can still simplify your wiring by mounting the additional transformer near the GPC and using a single 18 ga., 4-conductor, twisted pair cable to carry both communications and power just as you would if you were using the power outputs from the STATbus. The wiring diagram in Figure 2-3 shows how you can connect an external transformer to the power wires that would otherwise be connected to the power outputs pins on the STATbus port.

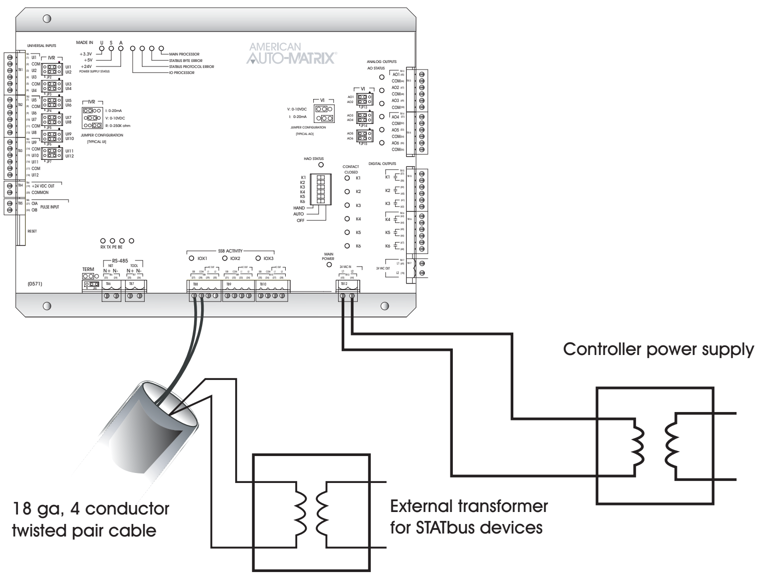

Figure 2-3: Using an External Transformer with 4-wire STATbus Cabling

WARNING
You should **never** connect an external transformer to a STATbus cable that is also connected to the power outputs on the *NB*-GPC3.

Once power is connected to the *NB*-GPC3, the Main Power light, located to the left of TB12 as shown in Figure 2-4, should be lit.

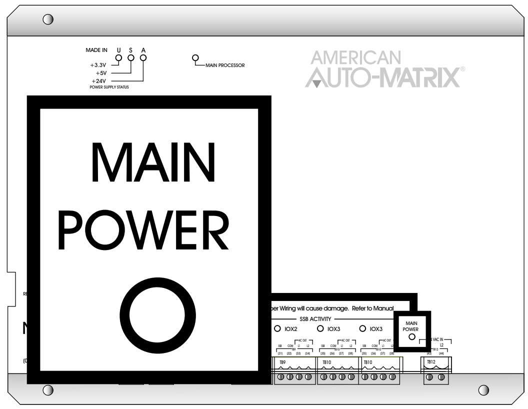

Figure 2-4: Location of the Main Power indicator

The Power Supply Status lights, shown in Figure 2-5, indicate that the 3.3 V, 5 V, and 24 V on-board power supplies are functioning properly. The Main Processor light will begin to blink. When the processor is functioning normally, this light will cycle on and off approximately every half second.

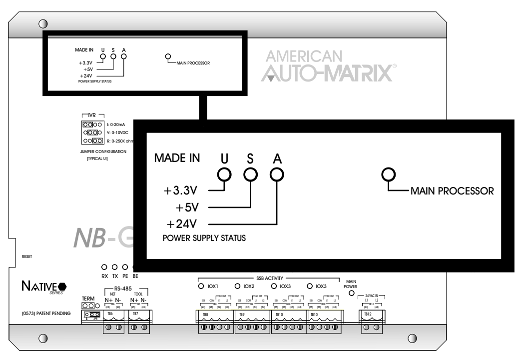

Figure 2-5: Location of the Power Supply Status Indicators and the Main Processor Indicator

2.2.3 Connecting the RS-485 Network

The *NB*-GPC3 communicates over an RS-485 network. As shown in Figure 2-6, the *NB*-GPC3 has two network connections at terminal blocks TB6 and TB7. The RS-485 network is connected to terminal block TB6 using 14-22 AWG, shielded, twisted pair wiring making sure to maintain polarity between devices.

The shield wire run to the first controller in the network should be connected to a reliable earth ground. At each connection along the network, i.e. from the first to the second, the second to the third, etc., the shield wires should be connected together and taped back. At the last controller, the shield wire should simply be taped back.

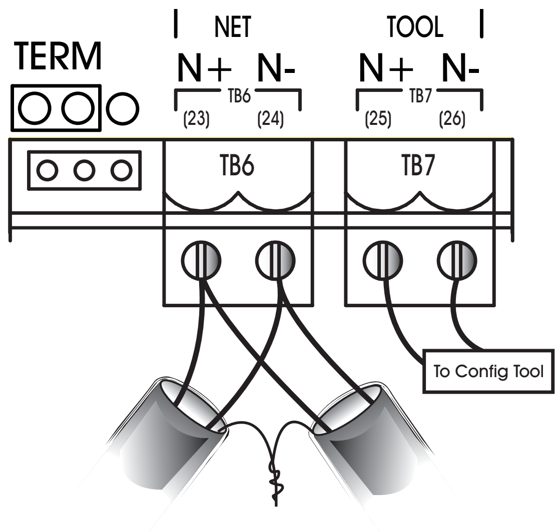

Figure 2-6: Proper Connection of the NB-GPC3 to an RS-485 Network

IMPORTANT
All RS-485 communications networks should be installed using shielded, twisted pair wiring. Each twisted pair must be individually shielded. Unshielded cables must be placed in metal conduit alone. Communications wiring should not be routed together with, or close to, wiring carrying DC switching, AC lines, fluorescent lighting or any other RFI/EMI emitting source. Failure to meet these requirements may result in various communications problems, such as excessive network retries, noise susceptibility, and/or a complete loss of communications.

The second network port on the *NB*-GPC3, terminal block TB7, is intended for the temporary connection of a configuration device. This device can be used to change the values of properties within the *NB*-GPC3, enabling the user to configure the *NB*-GPC3 without having it be connected to a host computer. The connection at TB7 is NOT intended for connection of devices that are part of the building RS-485 network. That way, if the *NB*-GPC3 were to be removed for some reason, the RS-485 network would be left intact.

IMPORTANT
The second network connection, located at terminal block TB7, is not to be used for communications between BACnet devices. It is intended for the temporary connection of a configuration tool to the *NB*-GPC3. BACnet devices on the network should not be connected to terminal block TB7.

For *NB*-GPC3 units that are electrically last on the network, it may be necessary to connect a termination resistor across the N+ and N- terminals to minimize unwanted communications line effects, such as signal reflection. Only the end units on the multi-drop network should be terminated. The *NB*-GPC3 provides the option of connecting an internal, jumper-selectable, 249 Ω termination resistor across the N+ and N- terminals. To connect the termination resistor, the TERM jumper, located to the left of TB6, should be moved to the left two pins as shown in Figure 2-7a. To disconnect the internal termination resistor, move the TERM jumper to the right two pins as shown in Figure 2-7b. If necessary, additional and termination resistance can be added by connecting a resistor between the N+ and N- pins.

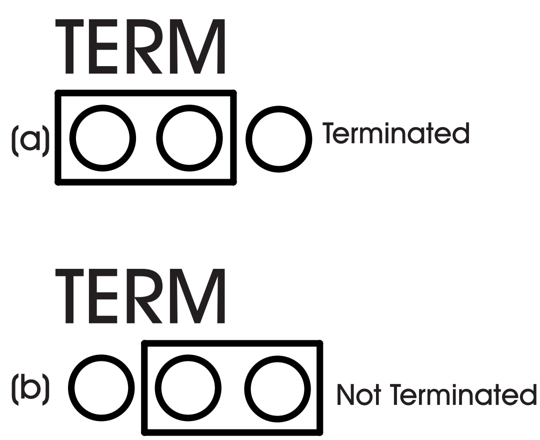

Figure 2-7: Configuring the TERM Jumper for RS-485 Communications Termination

Once power is supplied to the *NB*-GPC3 and the unit is connected to the RS-485 network, the RX and TX lights, located just above terminal blocks TB6 and TB7, shown in Figure 2-8, should blink rapidly, indicating that the controller is communicating over the network.

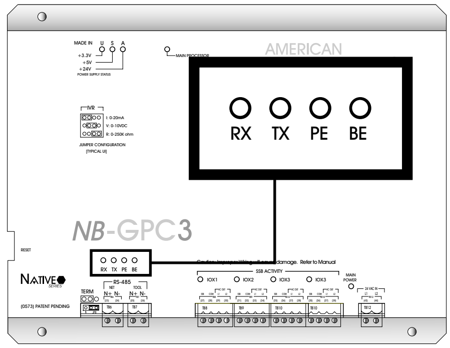

Figure 2-8: Network Status Indicator Lights

The BE light indicates that a Byte Error has occurred. This type of error occurs when information is being transferred over the network and data is lost. A Byte Error occurs if a problem is detected in the start, stop, or data bit timings. This type of error would also be generated if communication baud rates do not match. In this case, the *NB*-GPC3 detects that the error has occurred, lights the BE light and sends a request over the network for the information to be resent.

A packet of data is a number of bytes of data that are transmitted over the network together. Each packet of data includes a checksum value used to validate the data. A Packet Error occurs when *NB*-GPC3 detects a mismatch between the calculated checksum and the value received with the packet. Packet errors can also be caused by faulty transmission or reception of a data packet. When this happens, the PE light will be lit, indicating that such an error has occurred. A request will then be sent for the data to be retransmitted.

2.3 UNIVERSAL INPUTS

The *NB*-GPC3 provides a total of twenty four universal inputs (UIs). All of these inputs are exclusively for use with STATbus devices and can be configured to make use of remote inputs on SSB-FI1, SSB-UI1, and SSB-IOX1 modules as well as the dry contact inputs on SSB-DO1-I and SSB-DO2-I modules.

Each universal input may operate as either a digital or analog input. Each input may be configured individually to read any of the following:

- digital (on/off)
- linear inputs (0-5 V, 0-10 V, 0-20 mA, 4-20 mA, etc.) scaled between a programmable minimum and maximum value
- non-linear inputs with response provided via two programmable piecewise curves
- thermistor (Precon type III 77º @ 10kΩ)
- SMARTStat device

2.3.1 CONFIGURING CURRENT INPUTS

Any sensor which generates a signal in the form of a current is classified as a current sensor. Ranges of 0-20 mA and 4-20 mA are common in sensors. The current produced by these sensors is directly proportional to the value being measured. For example, if a pressure sensor measured 0 to 5 inches of water gauge and had an output range of 0 to 20 mA, then a reading of 10 mA would correspond to a pressure of 2.5 inches of water gauge.

To specify the sensor type, you must open the Universal Input object corresponding to the input you are configuring. Next, you want to set the **(ST) Sensor Type** property to a value of 3, for 4-20 mA inputs, or 8, for 0-20 mA inputs. That tells the *NB*-GPC3 that a current sensor is connected and specifies its range.

NOTE
When switching from a digital sensor type to an analog sensor type, you must reset the *NB*-GPC3 after changing the sensor type. Once done, you then have to rediscover the associated object. This process allows you to view other properties associated to the object, such as minimum and maximum scaled values. Refer to NB-Pro/NB-Link User Manual for more information.

You also need to specify the range over which the sensor operates. This is necessary so that the *NB*-GPC3 can calculate the measured value from the input signal. The **min_pres_value** property should be set to the lowest value that your sensor can measure. The **max_pres_value** property should be set to the maximum scaled value for your input.

As an example, sensors which measure relative humidity are often current sensors that operate in the 4-20 mA range. For a sensor of this type you would set **ST**=3 because the sensor measures 4-20 mA. Relative humidity ranges from 0 to 100% so you would set **min_pres_value**=0 and **max_pres_value**=100 to represent the limits of the sensor's output. In this case, a raw value of 4 mA would be scaled to a value of

0% in engineering units. The relative humidity sensor would read 100% if the input were reading a signal of 20mA.

2.3.2 CONFIGURING VOLTAGE INPUTS

Any sensor which puts out a voltage in response to a measured value is classified as a voltage sensor. Voltage sensors behave in very much the same way as current sensors. The primary differences have to do with the internal circuitry of the *NB*-GPC3 and how the signal is read.

To specify the sensor type, you must to go to Universal Input object corresponding to the input you are configuring. Next, you want to set the **(ST) Sensor Type** property to a value of 2, for 0-5 V inputs, or 6, for 0-10 V inputs.

NOTE
When switching from a digital sensor type to an analog sensor type, you must reset the *NB*-GPC3 after changing the sensor type. Once done, you then have to rediscover the associated object. This process allows you to view other properties associated to the object, such as minimum and maximum scaled values. Refer to NB-Pro/NB-Link User Manual for more information.

You also need to specify the range over which the sensor operates. This is necessary so that the *NB*-GPC3 can calculate the measured value from the input signal. The **min_pres_value** property should be set to the lowest value that your sensor can measure. This would correspond to the reading at zero volts. The **max_pres_value** property should be set to the maximum scaled value for your input. For example, if a 0-10 V carbon dioxide sensor measuring from 0-5000 ppm would have **min_pres_value** would be set to 0 and **max_pres_value** would be set to 5000.

2.3.3 CONFIGURING THERMISTOR INPUTS

The thermistor is one of the most common types of resistive sensors for temperature measurement. The thermistor's combination of high accuracy over a wide range coupled with its low cost makes it one of the most popular temperature sensors used. Because of the thermistor's popularity, the *NB*-GPC3 has the response curve for a **precon type III** thermistor built in.

For a thermistor input, you want to set the **(ST) Sensor Type** property equal to 7, the value which corresponds to a thermistor. The *NB*-GPC3 will then automatically set the **min_pres_value** to -30.0 and the **max_pres_value** to 230.0, the minimum and maximum values that can be read by this type of sensor. The temperature will now be displayed in the **present_value** property of this object. The value of **present_value** will also be displayed in the Universal Input Summary object.

NOTE
When switching from a digital sensor type to an analog sensor type, you must reset the *NB*-GPC3 after changing the sensor type. Once done, you then have to rediscover the associated object. This process allows you to view other properties associated to the object, such as minimum and maximum scaled values. Refer to NB-Pro/NB-Link User Manual for more information.

At this point, it is helpful to give the object a name so it can be easily identify in the future. You can name the object by setting the value of the **object_name** property. You should then check the reading and adjust any disagreement between the sensor and a known reading using the **(OF) Input Offset** property, This specifies a fixed offset for the sensor and would be used if the sensor reading was off by a fixed amount. For example, if a sensor was reading three degrees below the actual room temperature. In that case, you would set **OF**=3.0 to correct the reading.

If you are not going to configure any alarming function for this input, then you have set all the parameters which need to be set for the thermistor to provide a reading. For more information on configuring alarming, see Section 2.8, “Alarming”.

2.3.4 CONFIGURING NON-LINEAR INPUTS

The thermistors discussed previously are just one example of a sensor who’s output characteristics are well defined. Because precon type III thermistors are so prevalent, the output response curve is included with the controller. However, there are a number of other common sensors who’s response is non-linear. For these types of inputs, the *NB*-GPC3 provides the option of using one of the two Piecewise Curve objects to specify the response of the input device.

To use an input with a non-linear response, you must define the piecewise curve and then set the input to use the curve you have defined to scale its readings. The process for defining the Piecewise Curve is given in Section 3.22, “Piecewise Curves 1-2”. The response data necessary to construct the curve will usually be available in the catalog from which the sensor was ordered or on the data sheet accompanying the sensor.

You then set the **(ST) Sensor Type** property equal to 4 or 5. This tells the *NB*-GPC3 to use Piecewise Curve 1 or 2 with this object. The next step is to set the minimum and maximum scaled values using the **min_pres_value** and **max_pres_value** properties. The **min_pres_value**, should be set to the lowest scaled value from your piecewise curve. For most application this will correspond to the value of **Y1** in the Piecewise Curve object. Similarly, **max_pres_value** should be set to the maximum scaled value for the input. This will most likely be the value of **YB** in the Piecewise Curve object.

NOTE
When switching from a digital sensor type to an analog sensor type, you must reset the *NB*-GPC3 after changing the sensor type. Once done, you then have to rediscover the associated object. This process allows you to view other properties associated to the object, such as minimum and maximum scaled values. Refer to NB-Pro/NB-Link User Manual for more information.

At this point, you should give the object a name so that you can easily identify it in the future. You can name the object by setting the value of **ON**.

2.3.5 CONFIGURING DIGITAL INPUTS

An input that only has two signal states is considered a digital input. The most basic digital inputs are switches or contacts. The switch is either on or off, the contact is closed or open. Inputs of this type have many uses in a building automation system.

For a digital sensor, you will set the **(ST) Sensor Type** property to zero.

NOTE
When switching from a digital sensor type to an analog sensor type, you must reset the *NB*-GPC3 after changing the sensor type. Once done, you then have to rediscover the associated object. This process allows you to view other properties associated to the object, such as minimum and maximum scaled values. Refer to NB-Pro/NB-Link User Manual for more information.

Digital inputs can take one of two states, but you can tell the controller how you want it to treat those states. By setting the **polarity** property you can specify whether the *NB*-GPC3 should display **present_value**=1 for a high signal (normal operation, **polarity**=0) or a low signal (reverse operation, **polarity**=1).

Digital inputs have an input filter delay, **IF**, which specifies the time, in seconds, which the input's state must remain steady for the reading to be seen as reliable. This means that a signal that does not remain steady for **IF** seconds will be seen as a spurious signal and will cause the input reliability, **RE**, to be set to

“7 unreliable other”. This means that the signal is changing too rapidly to be considered reliable and should not be used for control functions.

Digital inputs have different alarming options than analog inputs. See Section 2.8.3, “Configuring Alarming for Digital Inputs” for a complete discussion of the alarming properties for inputs configured to read a digital signal.

2.4 DIGITAL INPUTS

The *NB*-GPC3 has eight digital intput objects with dedicated pulse counting features. These digital inputs are capable of detecting signals in the range 3-40 VDC peak to peak or 2-29 VAC at 50/60 Hz.

The digital inputs operate at a much higher frequency than a Universal Input. This makes it possible for the input to not only detect whether a signal is on or off, but to detect rapid pulses from the input. These pulses would be used primarily for demand metering applications as part of an energy management system. Digital, pulse counting inputs can also be used in flow metering applications.

Regardless of the specific application, all pulse counting digital inputs operate on the same principle. The device will generate a pulse for a given quantity of the value that is being measured for a demand metering application. One pulse might correspond to one kilowatt-hour of power whereas, for a flow metering setup, a single pulse might correspond to one gallon of liquid. The important piece of information is the correlation between pulses and measured values.

2.4.1 CONFIGURING DIGITAL INPUTS

To configure a digital pulse counting input, you must set the **(MD) Pulse Counter Mode** property to “Rising Edge”, “Falling Edge”, or “Both”. This tells the *NB*-GPC3 that you want the input to operate as pulse counter and not a simple digital input. When **MD** is set to “Rising Edge”, “Falling Edge”, or “Both”, the controller will not track the run hours for the input.

NOTE
MD must be set to “3=Disabled” for the input to track the run hours and generate run limit alarms.

To correlate pulses with the value being measured, you must enter a value for the **(SF) Pulse Multiplier** property. This multiplier specifies the amount of the measured quantity that is accumulated for each pulse. For example, if a demand meter sends out a pulse for each kilowatt-hour of power used, then you would set **SF**=1.0 since one pulse corresponds to one kilowatt-hour. If your input device were a flow meter that sent a pulse for every ten gallons of liquid that passed through a pipe, then you would set **SF**=10.0.

The total number of pulses accumulated will be displayed in the **(NP) Number of Pulses Accumulated** property. While this can be useful, you will be more concerned with the scaled pulse count stored in the **(SV) Scaled Pulse Count** property. This is the value of the total number of pulses, **NP,** multiplied by the scaling factor, **SF**. **SV** gives you the total amount of the value that you are measuring. For example, if the *NB*-GPC3 has accumulated 1250 (**NP**=1250) pulses and each pulse corresponds to 2.5 gallons of liquid that has been pumped through a pipe (**SF**=2.5), then the total amount of liquid pumped (**NP**x**SF**) would be displayed in **SV**. In this case, **SV** would have a value of 3125, meaning that 3125 gallons of liquid had been measured.

2.5 ANALOG OUTPUTS

2.5.1 CONFIGURING ANALOG OUTPUTS

Both voltage and current outputs can be configured to work as either normal- or reverse-acting. The analog outputs on the *NB*-GPC3 can operate in automatic mode, where the output is controlled by the *NB*-GPC3's control loops, or manual mode, where the value of the output is set by the user or by an SPL program. The minimum and maximum desired positions can be specified and the output can be interlocked using one or more inputs.

Properties **min_pres_value** and **max_pres_value** specify the minimum and maximum scaled values for the outputs, expressed as a percentage of the full scale. **MN** and **MX** are used to specify the display range for the **present_value**. **MN** is the value of **present_value** that corresponds to an output of **min_pres_value**. Similarly, **MX** is the value of **present_value** that will result in an output of **max_pres_value** .

For example, if the **present_value** is to be displayed as a percentage (0-100%) of a 10 VDC output range, set **min_pres_value** to 0 and **max_pres_value** to 100 (a display range of 0%-100% of full scale). Then set **MN**=0.0 and **MX**=100.0, so that **present_value**=0 represents 0.0% of the output range and **present_value**=100 represents 100.0% of the output range. This is shown in Figure 2-9.

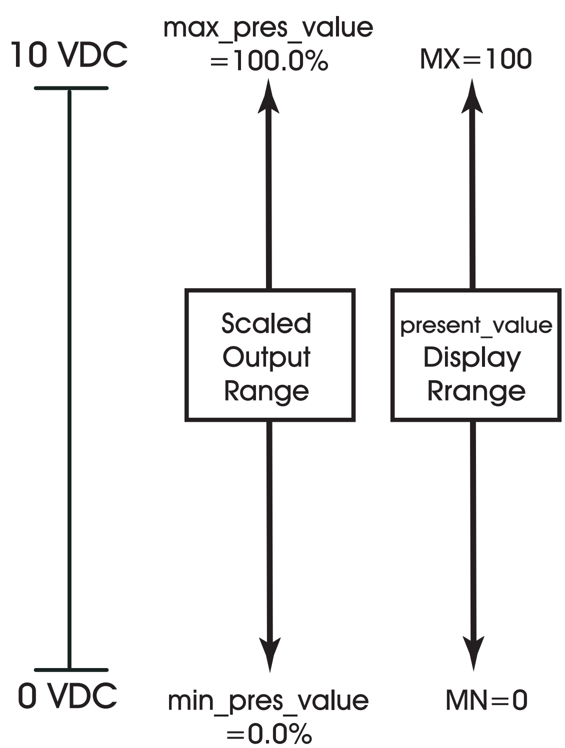

Figure 2-9: 0 to 10 V Displayed as a Percentage

If the output device in the previous example only operated from 2-10 V instead of 0-10 V, you would simply change the value of **min_pres_value** to be 20.0 becasue 2 V is 20% of the 10 V maximum. Everything else from the previous example would remain the same. This can be seen in Figure 2-10.

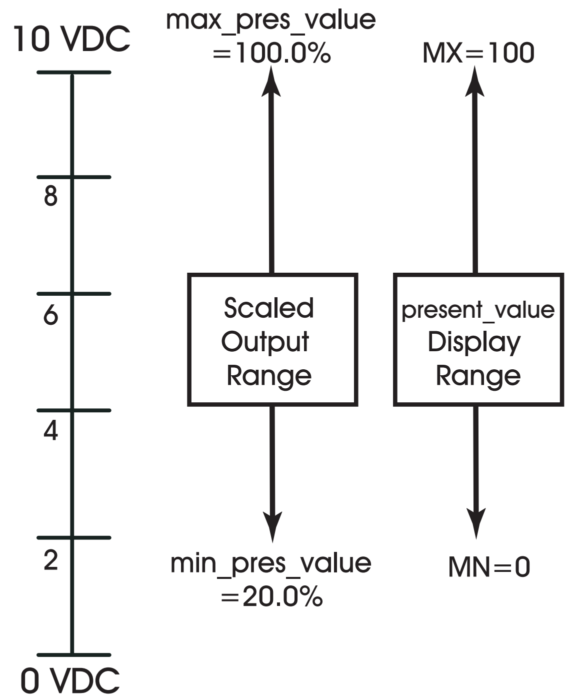

Figure 2-10: 2 to 10 V Displayed as a Percentage

The 2-10 V output in the last example could also be configured to display the actual output voltage rather than using a percentage. By setting **MN**=2 and **MX**=10, **present_value**=2 would correspond to 20% of the output (2 V) and a **present_value** of 10 would correspond to 100% (10 V). This is shown in Figure 2-11.

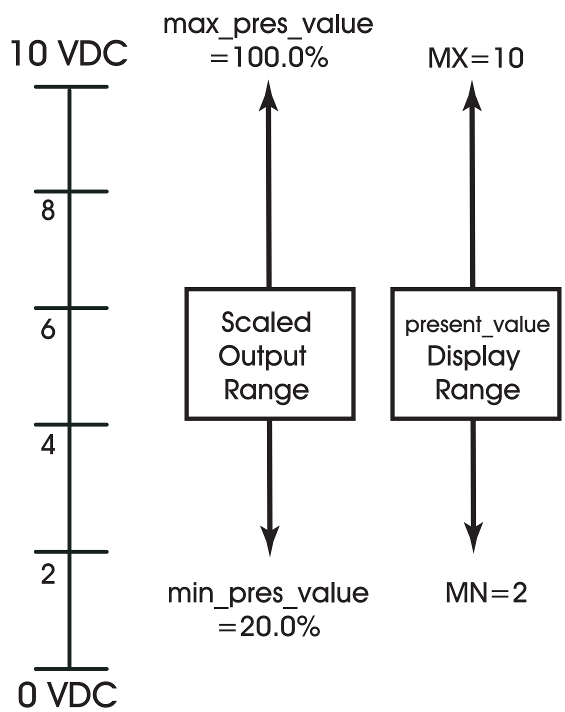

Figure 2-11: 2-10 V Displayed as a Voltage

Finally, any output can be configured to be reverse-reading. For a reverse-reading output, the value on **MN** is set greater than **MX**. For example, if a 4-20 mA output is controlling an actuator that is fully open at 4 mA and fully closed at 20 mA, you would set **min_pres_value**=20.0 (4mA = 20.0% x 20 mA) and **max_pres_value**=100.0 to specify the scaled output range. If you wanted to read **present_value** as the percentage that the actuator was open, you would set **MN**=100 and **MX**=0. This is shown in Figure 2-12. Here a fully closed actuator, output at 20mA, would read 0 and a fully open actuator, output at 4 mA, would read 100.

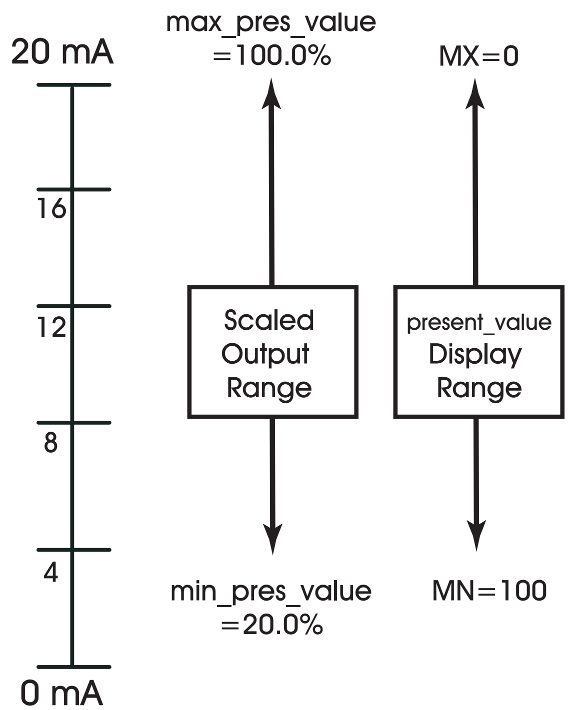

Figure 2-12:2 to 10 V Reverse Reading Input

If an output is set to manual control, the *NB*-GPC3 gives you the option to have that output go to a pre-determined position if the communications are lost with a host that is controlling the output. If you wish to have the communications failure position for the output, you must first set the **(CF) Communications Failure Enable?** property to 1. Then you should set the **(FP) Interlock/Communications Failure Position** property. This specifies the desired position you wish the output to take in the event of a failure.

The *NB*-GPC3 may also interlock the output to one or a number of inputs. This is used as a safety feature and can prevent equipment damage by immediately changing the output's position when a fault is detected. To specify which inputs, configured as digital inputs, are to be used for interlocks, set the appropriate bits in the **(IL) Inputs for Interlocking** property, located in the Analog Output object. A fault condition is generated when any of the specified inputs assumes the value specified by the **(LP) Interlock Polarity** for that input. When the interlock fault is detected, the output will assume the position specified in **FP**.

The *NB*-GPC3 can also set the position of an output when a fire alarm is detected. If the *NB*-GPC3 detects a fire alarm, the output will assume the position specified in the (**FI) Fire Position** property.

2.6 DIGITAL OUTPUTS

The *NB*-GPC3 has twelve (12) digital output objects which are exclusively for use with STATbus devices. They can be configured to make use of remote outputs on SSB-DO1, SSB-DO1-I, SSB-DO2, SSB-DO2-I and SSB-IOX1 modules.

Digital outputs are contacts that the *NB*-GPC3 can open and close based on its control loops. These are used to control devices that are either on or off and do not have any intermediate values. Digital outputs can be controlled automatically by the *NB*-GPC3 or they can have their outputs state set manually, either by the user or by a remote host or SPL program.

2.6.1 CONFIGURING DIGITAL OUTPUTS

The *NB*-GPC3 gives you the option to have an output that it is being controlled by a remote host take a pre-determined state if the communications are lost with that host. If you wish to have the output revert to a communications failure position in the event communications are lost, you must set the corresponding bit in the **(CF) Communications Failure Enable** property in the Digital Output Summary object (**CF** bit #=1). The desired state you wish the output to take in the event of a failure is determined by the (**FS) Communications State Failure Bitmap** property in the Digital Output Summary object. Setting a bit to 1 will cause the corresponding output to turn on in the event of a communications failure.

Using the Digital Output Summary object, it is also possible to set the state you wish various outputs to take in the event of a fire. By setting the corresponding bits in the **(FE) Outputs Enabled for Fire** property you can specify which outputs you wish to set the state for when a fire alarm is detected. Setting the bit to 1 indicates that the *NB*-GPC3 should set the state of that output to the state specified in the Fire State Bitmap property (**FI**). For a complete description of the properties in the Digital Output Summary object, see Section 3.15, "Digital Output Summary".

For each digital output there are a number of properties that are used to control how the output turns on and off and for how long. The *NB*-GPC3 allows the user to specify an interval to wait after power is applied before the outputs are turned on. This is set using the **(SI) Power-On Stagger Interval** property. This defines a wait period after power is applied, either initially or because of a power failure, before an output can be energized. This prevents outputs from snapping on upon powering up, before the controller's control loops have been given a chance to normalize.

You can specify a pulse width (**PW**) for the output using the **(PW) Pulse Width when Output is ON** property. This is a highly precise (nominally accurate to a tenth of a second) way to set the duration for which the output will be energized. This can be used for equipment that reads pulses of a certain duration as control signals.

Within the *NB*-GPC3, it is also possible to set a runtime limit for the output. The **(RH) Run Hours** property stores the number of hours that the output is energized (**present_value**=1). After a predetermined amount of time specified in the **(RL) Run Limit** property, a run limit alarm is generated. This is used to signal regular maintenance is required. For example, if a motor needed to be lubricated after 1000 hours of use, **RL** would be set to this limit (**RL**=1000) and would generate an alarm when **RH** exceeded the value of **RL**.

2.7 REMOTE I/O

The *NB*-GPC3 is designed to work with STATbus expansion modules known as IOX modules which allow you to add remote inputs and outputs to the controller. This section gives a brief overview of how to connect and configure those devices. For more details, please refer to the IOX Modules User Manual, part #1E-04-00-0126.

2.7.1 CONNECTING DEVICES TO THE STATBUS

STATbus allows I/O devices to be located remotely from the controller. Devices physically located on the STATbus differ from conventional inputs and outputs only in how they must be connected to the *NB*-GPC3. Instead of having a single input or output device wired to a pair of terminals on the controller, the STATbus allows for an open-topology network of devices to be created.

To use a STATbus device with the *NB*-GPC3, you must start by connecting the twisted pair wire to be used for STATbus network communications to the terminals marked SSB and COM on the corresponding terminal block on the *NB*-GPC3 (TB8,TB9, TB10, or TB11). These wires would then be connected to the two SSB terminals on the STATbus device. For STATbus devices which require power, the terminal labelled X1 and X2 can be used or an external power supply may be connected.

NOTE
Unlike the RS-485 network, STATbus connections have no polarity, meaning you need not concerned yourself with maintaining polarity from one device to the next.

To connect additional devices, the wires are simply daisy-chained from the SSB terminals of the first device to the SSB terminals of the second device as shown in Figure 2-13.

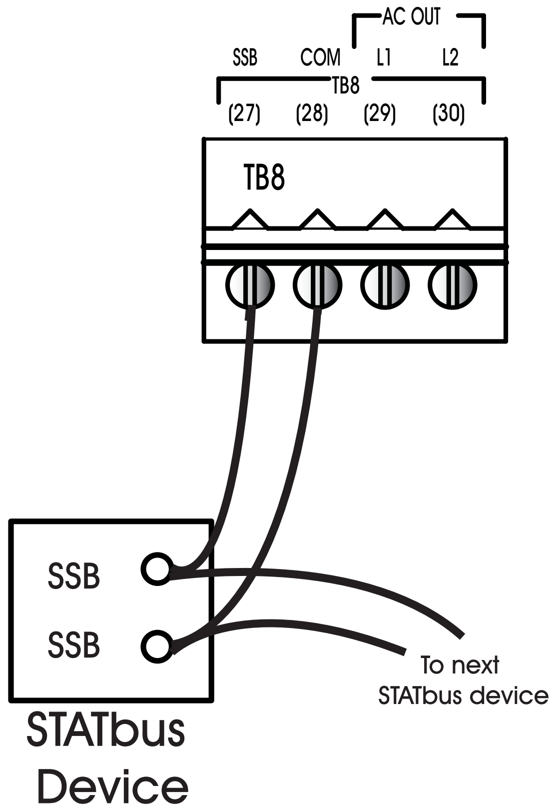

Figure 2-13: Connecting STATbus devices to the NB-GPC3

2.7.2 LENGTH OF THE NETWORK

The distance measured from the controller to the STATbus device on the network located furthest away from it should not exceed 1000' in length. The STATbus shown in Figure 2-14a is a valid configuration because the distance from the controller to the most distant device is less than 1000’ whereas the configuration shown in Figure 2-14b is not valid because the total length to the most distant device is 1150’, exceeding the 1000’ maximum.

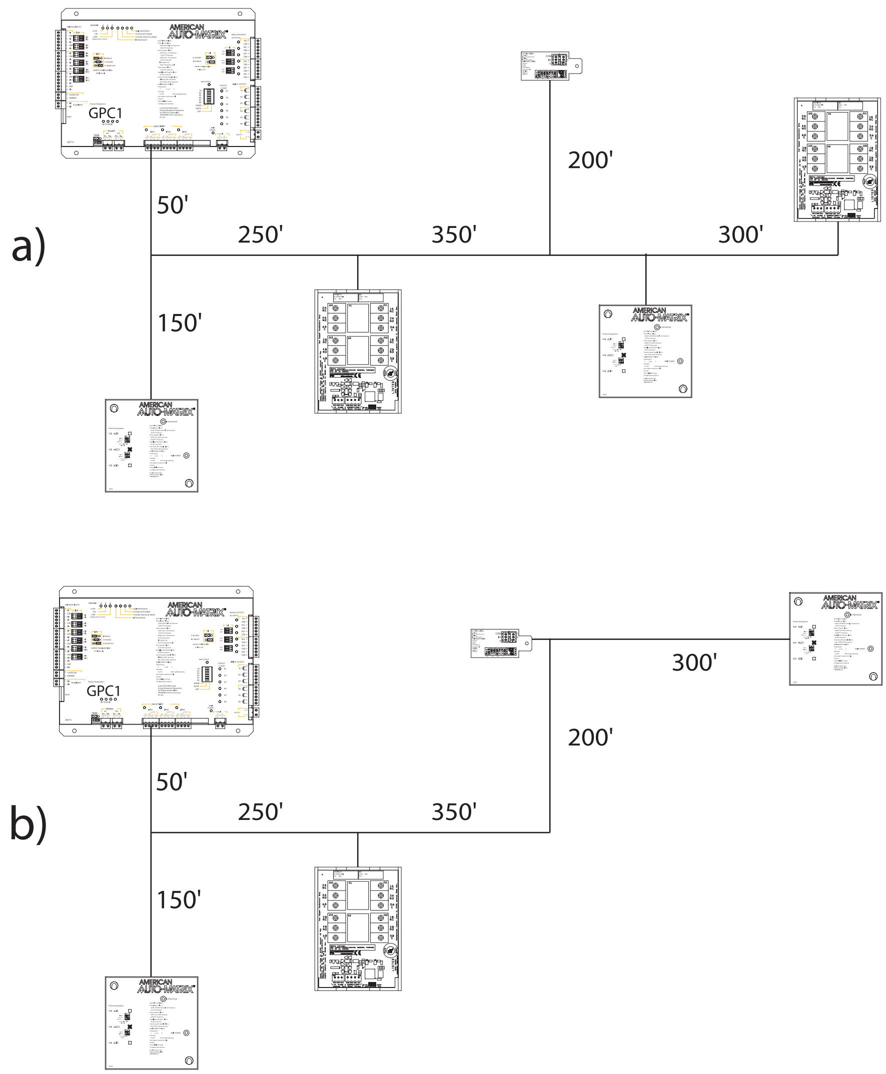

Figure 2-14 Determining Maximum STATbus Length

2.7.3 NUMBER OF DEVICES

Each STATbus channel on the controller will support a maximum of thirteen (13) devices.

CAUTION

A maximum of thirteen (13) devices can be connected to a single STATbus channel. If more than thirteen devices are connected, only thirteen will be enumerated by the controller. If more than thirteen devices are connected, there is no way to predict which devices will be enumerated and which will be left off.

2.7.3.1 COMMUNICATIONS LIMITS

While the STATbus protocol allows up to thirteen devices to be connected to a single network, certain devices reduce the maximum number of other devices that may be used on a single channel. In particular, STAT1D, STAT2D, and STAT3 have a higher power requirement than other STATbus devices and limit the total number of devices that can be put on the network and still communicate. When one or more STAT devices are included on the network, the maximum number of devices allowed on the network will be reduced.

NO STATS ON THE STATBUS

If your STATbus channel does not have any STATs (SBC-STAT1D, SBC-STAT2D, SBC-STAT3, SBC-RH1, SBC-RH3, or SBC-RHT) on it, then you may have up to thirteen devices in any combination on the network. This may include SSB-FI1, SSB-UI1, SSB-AO1, SSB-DI1, SSB-DO1, SSB-DO1-I, SSB-DO2, SSB-DO2-I, and SSB-IOX1 modules.

ONE OR MORE STATS ON THE STATBUS

If one or more STATs are being used, the total number of device that can communicate on a single STATbus channel will be reduced. Table 2-2 lists the number of additional STATbus devices that can be connected for a given number of STATs.

Table 2-2 Number of Devices Allowed on a STATbus

Number of STATs	Number of other STATbus Devices
1	11
2	9
3	7
4	5
5	2

EXAMPLE: NO STATS ON THE STATBUS

With no STATs on the Bus, you may use any combination of SSB devices, up to a maximum of thirteen devices total. This means any of the following would be valid:

- 13 SSB-FI1s to read a number of different inputs

- 6 SSB-FI1s and 6 SSB-AO1s to provide simple damper control for six zones
- 3 SSB-UI1s, 3 SSB-AO1s, and 3 SSB-DO1-Is, giving three zones with a zone temperature input, control for a damper and a reheat coil with supervisory monitoring

EXAMPLE: STATS ON THE STATBUS

For a STATbus design that contains STATs, you must refer to the Table 2-2 above to determine the number of devices that can be used in addition to the STATs.

For a system with 4 STATs, for example, Table 2-2 indicates that up to five additional devices can be connected to the bus. You could use four SSB-AO1s to create four zones with damper control

NOTE
If you are implementing custom display sequences on STAT displays, AAM recommends that STATbus devices be distributed across available IOX connections whenever possible. This practice will provide higher reliability against erroneous pixelation.

2.7.4 POWERING STATBUS DEVICES

The connection for STATbus is a four pin connector with two pins for communication and two for optionally providing 24 VAC power to devices on the STATbus. This power output can eliminate the need for dedicated transformers for remote devices, allowing potentially faster, easier installation at a lower cost. If you plan to power STATbus devices using this output, you must be aware that there are limitations imposed by the ability to transmit that power to remote devices as well as the power capacity of the controller.

Though STATBus communication signals can be carried up to the network maximum of 1000’ over 22 gauge cabling, due to power losses, this cable is not sufficient to transmit the 24 VAC used to power remote devices. For a longer network runs, the voltage drop across such a cable will leave less than the require voltage to operate many sensors and actuators. Therefore, when using devices which draw their power from the STATbus, you must use 18 gauge cabling to carry the power.

Using 18 gauge cabling to carry power, the controller should be able to provide the voltage and current necessary to operate most normal sensors, powered outputs, and smaller actuators. If you are using a large number of powered devices or devices which are known to have high power requirements, then you should supplement the power provided by the controller by using dedicated transformers one some of those units. Units with high power consumption would be the first candidates for dedicated transformers in this case.

You should never try to power large actuators using the STATbus as, in most cases, they simply require more power than the controller can provide. You should always use a dedicated power transformer when using these devices.

NOTE

The STATbus power outputs on the GPC3 controllers are protected and are capable of providing a maximum of 24 VAC at 3.0 A before tripping the PTC. This is the total for all STATbusses on the device.

2.7.5 DISTRIBUTION OF DEVICES

Whenever possible, you should try to distribute your STATbus devices over all the available STATbus channels on the controller. For example, if you have a total of twelve STATbus devices connected to your controller, it is better to have 4 devices on each of the three STATbus channels than to have all twelve devices on a single channel. This reduces the communications load on any one STATbus channel and will result in better overall system performance.

2.7.6 MAPPING REMOTE I/O DEVICES TO OBJECTS

Because remote I/O devices are located away from the controller, there is an additional step of mapping these remote points back to the controller so that the *NB*-GPC1 may recognize them. Once these I/O devices are mapped, they behave the same as an input or output point that is physically located on the *NB*-GPC1.

To map a STATbus device to an input or output, you will need to assign the device's global identification (GID) number to the input or output. To do this, you must perform the following steps.:

1. Set the **(CR) Configure Remote I/O** property in the Device Object to "Edit I/O GIDs" to allow the GPC to edit configuration information.
2. Set the **(GI) GID of I/O Device** property of the desired input or output to the GID of the device to be assigned.
3. If additional device need to be assigned, repeat steps 2 and 3 until all devices are assigned.
4. Set the **(CR) Configure Remote I/O** property in the Device Object to "GPC to Bus" to write the configuration information to the device(s).

Some STATbus devices have more than one input or output of a given type. This can lead to confusion when assigning the GID number for these modules because all inputs and outputs on a STATbus device have the same GID number.

NOTE

All input and output points on a STATbus device have the same GID number.

If the STATbus device has more than one input or output point, assigning its GID number to the **GI** property of a object will cause the GID number to be assigned to the **GI** property of a number of consecutive input or output objects equal to the number of points of that type on the module, starting with the object in which the GID was entered. For example, if the GID of an STATbus device with four universal inputs was assigned to UI16, that GID number would automatically be assigned to UI17, UI18 and UI19 as well, erasing any previous assignments. For example, if a STATbus device had previously been assigned to UI18, it's GID would be overwritten with the GID of the STATbus device. This can cause previously configured inputs to have their configuration information overwritten.

CAUTION

Assigning the GID number of a STATbus device with more than one input or output may cause the mapping of previously assigned STATbus devices to be lost.

When GID number of STATbus devices with multiple inputs or outputs are assigned, the lowest numbered input or output will correspondingly be mapped to the lowest numbered object. In the previous example, the first input on the STATbus device would be mapped to UI16, the second to UI17, the third to UI18, etc.

Once the GID number has been assigned, the controller's Configure Remote I/O property should be set to "GPC to Bus" (**CR**=1). This will write the configuration information to the remote I/O device. Once the data has been written, the GPC will return to "Normal" mode (**CR**=0).

NOTE

The GID number **must** be assigned and written to the remote I/O device before any any other configuration is performed.

With the device now mapped to the object, the input or output can be configured normally (i.e., setting the sensor type, defining any minimums or maximums, configuring alarming, etc.).

2.8 ALARMING

2.8.1 ENABLING ALARMING ON THE *NB*-GPC3

To enable the alarming functions of the *NB*-GPC3, you must choose the type of alarming desired. The **(IA) Intrinsic Alarming** property in the Device object is used to enable alarming and determine where and how alarm messages are to be sent. The choices for **(IA) Intrinsic Alarming** are given in Table 2-3.

Table 2-3 : Options for (IA) Intrinsic Alarming

IA	Alarming Option
0	None
1	Track Alarms
2	Broadcast
3	Unconfirmed to Device
4	Confirmed to Device

When **IA** is set to “1 = Track Alarms”, alarms will be generated but not sent to other BACnet devices on the same network.

When **IA** is set to “2 = Broadcast”, alarms will be broadcast to all other BACnet devices on the same network.

When **IA** is set to “3 = Unconfirmed to Device”, alarms will be sent to a device on the same network, defined using the **(UD) Use Device** and **(DD) Device Instance** properties, but will not require the device to acknowledge the alarm.

When **IA** is set to “4 = Confirmed to Device”, alarms will be sent to a device, defined using the **UD** and **DD** properties, and the device will be required to acknowledge the alarm.

When **IA)** is set to 3 or 4, you must configure the device that is to receive the alarm broadcasts. The **(UD) Use Device** property specifies whether the recipient will be an MSTP address, when **UD** is set equal to “0 = MSTP Address”, or a device instance, when **UD** is set equal to “1 = Device Instance”. The MSTP address or the device instance for the recipient must then be entered into the Device Instance property **(DD) Device Instance**.

You may also control the priorities for the different alarm conditions. These values influence how the alarms are handled relative to each other. The priorities for off-normal, normal and fault alarms are defined in the **(PO) Priority for Off Normal Alarms,** (**PN) Priority for Normal Alarms**, and **(PF) Priority for Fault Alarms** properties and have default values of 2, 4 and 3 respectively. The limits and alarming options specific to each input or output are specified in the respective object.

2.8.2 CONFIGURING ALARMING FOR ANALOG INPUTS

To enable alarming for a Universal Input configured to read an analog signal, you must first select the desired type of alarming using the (**AE) Alarm Function** property. Analog inputs can be configured for high limit alarming, low limit alarming, or both high and low limit alarming. The values of the (**AE) Alarm Function** property corresponding to these options are given in Table 2-4.

Table 2-4 : (AE) Alarm Function for Analog Inputs

AE	Alarming Function
0	Disabled
4	Low Limit
5	High Limit
6	Low and High Limit

Next you must enter values for the **low_limit** and **high_limit** properties. These specify the limits of normal operation. If the **present_value** of the analog input drops below the **low_limit**, and Low limit alarming (**AE**=4) is enabled, an alarm will be generated. Similarly, high limit alarming (**AE**=5) will generate an alarm if the **present_value** exceeds the high alarm limit (**high_limit**). If **AE**=6, an alarm will be generated if **present_value** drops below the low alarm limit (**low_limit**) or exceeds the high alarm limit (**high_limit**).

NOTE

After setting the value of the (**AE) Alarm Function** property, you must refresh the object in *NB*-Pro to activate the properties used to configure the alarming options.

Next you need to enter a value for the **deadband** property which defines an alarm limit hysteresis. This value determines when the controller will return from a high- or low-limit alarm condition. If the controller is currently in a high limit alarm, then the **present_value** must drop below **high_limit** by an amount specified in the **deadband** property before a limit return is generated. Similarly if the controller is in a low limit alarm, then the **present_value** must rise above **low_limit** by an amount specified in the **deadband** property before controller function returns to normal. The effects of **low_limit**, **high_limit**, and **deadband** are shown in Figure 2-15.

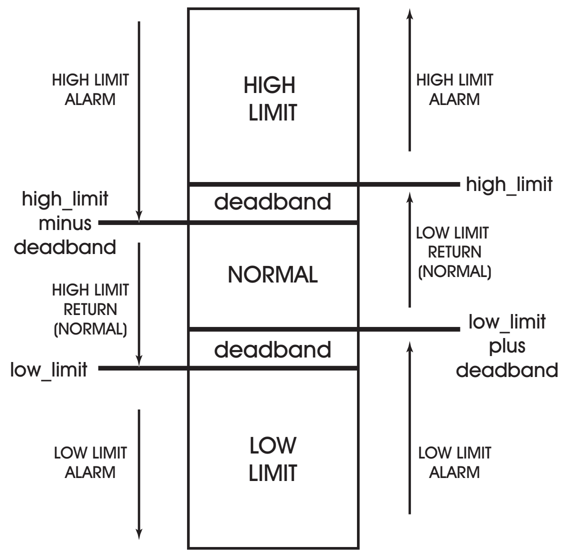

Figure 2-15: Alarm States and Thresholds for Alarming

Next you must specify the amount of time, in seconds, that the input must exceed a limit before an alarm condition is generated in the **time_delay** property. The **time_delay** property is used to compensate for noisy signal that may fluctuate in and out of a state that would cause an alarm to be generated. The delay can be any value between 1 and 255 seconds. Setting **time_delay**=0 will disable the alarm delay.

The **event_enable** property determines which alarm messages, when generated for this input, will be broadcast. You can choose to broadcast to-offnormal, to-normal and to-fault alarms by checking the appropriate box.

2.8.3 CONFIGURING ALARMING FOR DIGITAL INPUTS

To enable alarming for a Universal Input configured to read an digital signal, you must select the desired type of alarming using the (**AE) Alarm Function** property. Digital inputs can be configured to generate an alarm when the **present_value** transitions from 0 to 1, transitions from 1 to 0, or both. The input can be configured as a fire sensor and will generate a fire alarm based on the value read. The input may also be configured as a supervisory monitor of digital outputs. The values of **AE** corresponding to these options are given in Table 2-5.

Table 2-5 : (AE) Alarm Function for Digital Inputs

AE	Function
0	Disabled
1	Contact 0 > 1
2	Contact 1 > 0
3	Change of State
8	Fire
11	Supervise Digital Output 1
12	Supervise Digital Output 2

Table 2-5 : (AE) Alarm Function for Digital Inputs

AE	Function
13	Supervise Digital Output 3
14	Supervise Digital Output 4
15	Supervise Digital Output 5
16	Supervise Digital Output 6
17	Supervise Digital Output 7
18	Supervise Digital Output 8
19	Supervise Digital Output 9
20	Supervise Digital Output 10
21	Supervise Digital Output 11
22	Supervise Digital Output 12

When **AE**=1, the digital input will generate an alarm on a transitions from 0 to 1. Similarly, **AE**=2 will generate an alarm on a transition from 1 to 0. For **AE**=3, an alarm will be generated for any change of state (0 to 1 or 1 to 0) on the input.

When configured for fire alarming (**AE**=8), an alarm will be generated if the input, configured for fire detection, indicates a value of 1. By adjusting the **polarity** property, the fire alarming can be configured to operate so that the fire alarm state (**present_value**=1) will occur either when the signal is open (**polarity**=0) or closed (**polarity**=1). This is to accommodate different kinds of fire sensors.

For **AE**=11 through **AE**=22, the *NB*-GPC3 will supervise DO1 through DO12 respectively. A supervisory alarm is generated when a digital input fails to detect a transition and the supervised output indicates that a transition has been made. When supervising an output, the controller will wait a number of seconds (0-255) specified in the **(SD) Supervisory Delay** property and then compare the state of the supervisory input to that of the supervised output. If they do no match, a supervisory alarm will be generated.

The **time_delay** property specifies the amount of time, in seconds, that the input must remain in an alarm state before an alarm condition is generated. This delay can be set to any value between 1 and 255 seconds. Setting **time_delay**=0 will disable the alarm delay.

The **event_enable** property determines which alarm messages, when generated for this input, will be broadcast. You can choose to broadcast to-offnormal, to-normal and to-fault alarms by checking the appropriate box.

SECTION 3: OBJECT & PROPERTY DESCRIPTIONS

This section describes the objects and properties in the NB-GPC3 and explains how they are used.

IN THIS SECTION

3.1 DEVICE

The Device object is used to control and configure a number of general controller features of the *NB*-GPC3. This is the place where the controller manufacturer, controller type, serial number and unit ID number can be found. It is also where the firmware version and type along with the flash release code and the flash update count are located. The Device object contains the following properties: **apdu_timeout, app_software_version, database_revision, daylight_savings_status, device_add_binding, firmware_version, local_date, local_time, max_apdu_length_accepted, max_info_frames, max_master, model_name, number_of_apdu_retries, object_identifier, object_list, object_name, object_type, protocol_obj_types_supported, protocol_services_supported, protocol_version, segmentation_supported, system_status, time_synch_recipients, utc_offset, vendor_identifier, vendor_name, BF, BS, BT, CF, CM, CP, CR, CT, DA, DD, DE, EM, ET, FA, FC, FT, IA, ID, ND, NM, OS, PD, PF, PI, PN, PO, PS, RD, RM, RS, SN, SR, ST, TF, UD, VE** and **ZN.**

The **object_identifier** property is a unique number used to identify the controller. The **object_identifier** must be a unique number from 0 to 4194302. By default, American Auto-Matrix assigns an **object_identifier** in such a way that it is unique to AAM products. This unique number is based on the unit's serial number.

NOTE
The user is responsible for ensuring that the device's **object_identifier** is unique on the job site's network.

The **(CM) Controller Manufacturer** property identifies the factory-set manufacturer number for the controller. (**CM** for American Auto-Matrix controllers are always 255). Similarly the **(CT) Controller Type** property identifies the factory-set controller type number for the controller. **CT** for the *NB*-GPC3 is 205.

The **(SN) Serial Number** property stores the controller's serial number. This number is set at the factory and cannot be changed.

The **(ID) Unit Number** property is the unit ID number. This is a number that uniquely identifies the controller on the network. Each controller must have a unique **ID** or communications problems will occur. By default, **ID** is set to the last two digits of the unit's serial number. **ID** may be set to any value between 0 and 127.

Information about the *NB*-GPC3's firmware and flash updates are also found in the Device object. The **firmware_version** property identifies the firmware that is currently installed on the *NB*-GPC3. This number is checked when flashing updates to determine if updates are available. The version of the kernel running within the controller is also displayed in the device object.

Property **(OS) Kernel Version**, shows the current version of the kernel.

The Device object also keeps a record of the firmware type installed in the controller. This information is stored in the **(FT) Firmware Type** property. For the *NB*-GPC3, **FT** will be equal to 7.

When the firmware is updated by flashing the controller, the flash update count property, **FC**, is incremented. This keeps track of the total number of times the controller has been flashed.

Property **object_name** is a user definable string that can be used to help identify the controller or its location. This can be helpful if multiple *NB*-GPC3 controllers are on the same network. In this case, **object_name** can be used to quickly identify each one.

To enable the alarming functions of the *NB*-GPC3, you must choose the type of alarming desired. The **(IA) Intrinsic Alarming** property in the Device object is used to enable alarming and determine where and how alarm messages are to be sent. The choices for **IA** are given in Table 3-1.

Table 3-1 : Options for (IA) Intrinsic Alarming

IA	Alarming Option
0	None
1	Track Alarms
2	Broadcast
3	Unconfirmed to Device
4	Confirmed to Device

When the **IA** property is set to “1 = Track Alarms”, alarms will be generated but not sent to other BACnet devices on the same network.

When **IA** is set to “2 = Broadcast”, alarms will be broadcast to all other BACnet devices on the same network.

When **IA** is set to “3 = Unconfirmed to Device”, alarms will be sent to a device on the same network, defined using the **(UD) Use Device** and **(DD) Device Instance** properties, but will not require the device to acknowledge the alarm.

When **IA** is set to “4 = Confirmed to Device”, alarms will be sent to a device, defined using the **UD** and **DD** properties, and the device will be required to acknowledge the alarm.

When **IA** is set to 3 or 4, you must configure the device that is to receive the alarm broadcasts.

The **(UD) Use Device** property specifies whether the recipient will be an MSTP address, **UD** set equal to “0 = MSTP Address”, or a device instance, **UD** set equal to “1 = Device Instance”. The MSTP address or the device instance for the recipient must then be entered into the **(DD) Device Instance** property.

You may also control the priorities for the different alarm conditions. These values influence how the alarms are handled relative to each other. The priorities for off-normal, normal and fault alarms are defined in the **(PO) Priority for Off Normal Alarms,** (**PN) Priority for Normal Alarms**, and **(PF) Priority for Fault Alarms** properties and have default values of 2, 4 and 3 respectively. The limits and alarming options specific to each input or output are specified in the respective object

A number of the *NB*-GPC3’s communications parameters are defined in the Device object. The communication rate is define by the **(CP) Communications BAUD** property. Table 3-2 shows the communication rates available in the *NB*-GPC3. The default baud rate is 38.4k (**CP**=6).

Table 3-2 : Available Communications Rates

CP	Baud Rate
0	9600
6	38.4k
7	19.2k
8	115.2k
9	57.6k

Along with a baud rate, a communications timeout must be specified. This is the amount of time that the controller will wait to be polled before entering a communication fault state. The communications failure timeout, given in seconds, is specified by the **(CF) Communications Failure Timeout** property. If the communications timeout is exceeded, a communication fault will be generated (**FA**, bit 6=1). This only needs to be set if a host or peer controller has an SPL program controlling the *NB*-GPC3's outputs.

The **(CR) Configure Remote I/O** property defines how remote I/O devices on STATbus objects are configured. The possible values of **CR** are listed in Table 3-3.

Table 3-3 : Remote I/O Configuration Options

CR	Option
0	Normal
1	GPC to Bus
2	Edit I/O GIDs
3	Bus to GPC

When remote I/O configuration is set to Normal mode (**CR**=0), editing I/O configuration properties (**ST**, **reliability**, **high_limit**, etc.) will update the configuration information in the remote I/O device.

When **CR**=1, The *NB*-GPC3 writes all of the object I/O configuration data to the STATbus devices in one atomic (unstoppable) write. The controller then reverts to Normal mode (**CR**=0) upon completion.

NOTE
Only STATbus devices physically connected will be written to.

This would be used if you wanted to configure the *NB*-GPC3 offsite. All of the assignments and configurations could be made before the *NB*-GPC3 was installed and then **CR** could be set to 1 to write all of the configuration information to the STATbus devices which had already been installed.

If **CR**=2, the *NB*-GPC3 will operate in GID Edit mode. This enables the ability to directly edit the **GI** property of a object, allowing you to manually designate STATbus devices for use as inputs and outputs. None of the changes made while in edit mode are sent to the STATbus remote I/O devices until the "GPC to Bus" option (**CR**=1) is selected.

CR=3 is very similar to **CR**=2, except that the *NB*-GPC3 reads all of the object I/O configuration data from the STATbus devices in one atomic (unstoppable) write. The controller reverts to Normal mode upon completion. This would be used if all of the STATbus devices were already configured but the *NB*-GPC3 did not have that information. If, for example, a *NB*-GPC3 in the field were replaced by a different unit or if the controller had all of its properties reset to their default values.

The *NB*-GPC3 has a number of broadcast capabilities that are controlled by the Device object. Some of these are triggered by faults detected by the *NB*-GPC3. Any faults that are detected are displayed in the **(FA) Faults Detected** property. **FA** is a bitmap with the meanings of each bit shown in Table 3-4. When a fault is detected, the appropriate bit is set to 1. If no fault is detected, the value of the associated bit is 0. If a fault is detected, the controller can be configured to broadcast that information to other units on the network.

Table 3-4 : Detected faults

FA	Fault Detected
0	Fire
1	Emergency
6	Communications failure

By setting the appropriate properties, the *NB*-GPC3 can broadcast change to fire mode commands or time synchronization signals. This is done by setting the **(BF) Broadcast Fire?** or **(BS) Broadcast Sync Time?** properties to 1. The frequency with which time sync broadcasts are repeated can be controlled by the **(BT) Broadcast Time Interval** property. Fire broadcasts are sent out every ten (10) seconds. The value of **BT** determines the time, in minutes, between broadcasts. Setting **BT** to 255 disabled broadcasts.

If a fire condition is detected, either by the *NB*-GPC3 itself or as a result of a fire broadcast from another controller on the network), the controller will switch into a fire mode (**FA**, bit 0=1). It will stay in fire mode for an amount of time specified in the **(TF) Time to Remain in Fire Mode** property. The *NB*-GPC3 can also broadcast the fact that it has entered a fire mode to other units on the network. If the broadcast fire property is enabled (**BF**=1) then a broadcast will be sent when the *NB*-GPC3 when a fire condition is detected (**FA**, bit 1=1).

The *NB*-GPC3 also has the ability to synchronize the time of other units on the network. This is accomplished by enabling the broadcast time sync property (**BS**=1). When time synchronization is selected, the current time (**local_time**) will be broadcast when the *NB*-GPC3 has the token. The time between each broadcast, in minutes, is specified in **BT**.

Broadcasts sent from the *NB*-GPC3 can be set to either broadcast globally or to a specific zone. Whether a particular broadcast is sent globally or only to a designated zone is specified in the appropriate Broadcast object. If zone broadcast is enabled then the broadcasts will be sent to all of the units with the same **(ZN) Zone Number** property as the *NB*-GPC3. The zone number is a number from 0 to 65,535 used to group controllers so that they may be controlled together and can communicate exclusively with each other, if that is desired.

The Device object has registers for a number of system properties such as the current time, current date, and day of the week. The current time is displayed in the **local_time** property and takes the format HH:MM:SS. The current date is stored in **local_date** and has the from MM/DD/YYYY. The day of the week is displayed in the **(DA) Day of Week** property. The options for **DA** are given in Table 3-5.

Table 3-5 : Day of the Week

DA	Day of the Week
0	Monday
1	Tuesday
2	Wednesday
3	Thursday
4	Friday
5	Saturday
6	Sunday

The *NB*-GPC3 can be adjusted for daylight savings time. To configure daylight savings, you must specify when daylight savings time begins and ends. The starting month is specified in the **(RM) Daylight Saving Start Month** property. The option for **RM** are given in Table 3-6.

Table 3-6 : Daylight Savings Start Month

RM	Month
0	None
1	January
2	February
3	March
4	April
5	May
6	June
7	July
8	August
9	September
10	October
11	November
12	December

Once the starting month has been set, you must specify which day on which daylight savings time begins using the **(RD) Daylight Saving Start Day** property. The options for **RD** are given in Table 3-7.

Table 3-7 : Daylight Savings Start Day

RD	Day of the Week
0	None
1	First Sunday
2	First Friday
3	Second Saturday
4	Third Sunday
5	Third Saturday
6	Last Sunday
7	Last Thursday
8	Last Friday

The time at which daylight savings time begins is specified in the **(ST) Daylight Saving Start Time** property.

The end of daylight savings time is specified in a very similar manner. The month is which daylight saving time ends is specified in the **(NM) Daylight Saving End Month** property. The options for **NM** are given in Table 3-8.

Table 3-8 : Daylight Savings End Month

NM	Month
0	None
1	January
2	February
3	March
4	April
5	May
6	June
7	July
8	August
9	September
10	October
11	November
12	December

The day on which daylight savings time ends is specified in the **(ND) Daylight Saving End Day** property.The options for **ND** are given in Table 3-9.

Table 3-9 : Daylight Savings End Day

ND	Day of the Week
0	None
1	First Sunday
2	First Friday
3	Second Saturday
4	Third Sunday
5	Third Saturday
6	Last Sunday
7	Last Thursday
8	Last Friday

The time at which daylight savings time ends is specified in the **(ET) Daylight Saving End Time** property.

Property **(EM) Engineering Units** is used to specify the units to be used by the controller. Using this property the *NB*-GPC3 can be set to operate using either english (**EM**=0) or metric (**EM**=1) units.

The **(PD) Control and Alarm Power-on Delay** property specifies the control and alarm power-on delay. This is the time delay (in seconds) that must elapse after the controller is reset before it begins control and alarming functions.

The **(PS) Power-up State** property defines the power-up state of the *NB*-GPC3. This is the schedule state that the controller will operate in when it is first powered up or after power is restored following a power failure. Table 3-10 shows the possible states in which the controller can start up.

Table 3-10 : Power-up States

PS	Power-up State
0	Unoccupied
1	Warmup
2	Occupied
3	Night Setback

The **(RS) Reset the *NB*-GPC?** property is used to reset the controller. Setting property **RS**=1 will reset the *NB*-GPC3. When the controller is reset, the processor is reset, clearing the memory and stopping any programs that are running. **RS** will then return to a value of 0. Once the *NB*-GPC3 has finished resetting, loaded programs will be started again. This will also occur if power is removed and replaced to the *NB*-GPC3.

The **(DE) Default Enable** property is used if you wish to return the *NB*-GPC3 to the factory default values for all properties. Enabling the default enable property (**DE**=197) will immediately clear all properties and return them to their default values. Setting **DE**=197 also clears all the programs loaded into the *NB*-GPC3.

Device Properties

Property	Description
object_ identifier	a numeric code that is used to identify the object.
object_name	represents a name for the device that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
system_status	indicates the current physical and logical status of the BACnet Device.
vendor_name	identifies the manufacturer of the BACnet Device.
vendor_ identifier	a unique vendor identification code, assigned by ASHRAE, which is used to distinguish proprietary extensions to the protocol.
model_name	indicates the vendor's name used to represent the model of the device.
firmware_ revision	indicates the level of firmware installed in the device.
application_ software_ version	identifies the version of application software installed in the device.
protocol_ version	indicates the version of the BACnet protocol supported by this BACnet Device.
protocol_ revision	indicates the minor revision level of the BACnet standard.
protocol_ services_ supported	indicates which standardized protocol services are supported by this device's protocol implementation.
protocol_ object_types_ supported	indicates which standardized object types are supported by this device's protocol implementation.
object_list	a list of each object within the device that is accessible through BACnet services.
max_apdu_ length_ accepted	specifies the maximum number of information frames the node may send before it must pass the token.
segmentation_ supported	indicates whether the device supports segmentation of messages and, if so, whether it supports segmented transmission, reception, or both.
local_time	indicates the time of day to the best of the device's knowledge.
local_date	indicates the date to the best of the device's knowledge.

Property	Description
utc_offset	indicates the number of minutes (-780 to +780) offset between local standard time and Universal Time Coordinated.
daylight_ savings_status	indicates whether daylight savings time is in effect or not.
apdu_timeout	indicates the amount of time, in milliseconds, between retransmissions of an APDU requiring acknowledgment for which no acknowledgment has been received.
number_of_ apdu_retries	indicates the maximum number of times that an APDU shall be retransmitted.
time_ synchronization _recipients	a list of devices to which the device may automatically send a TimeSynchronization request.
max_master	specifies the highest possible address for master nodes and shall be less than or equal to 127.
max_info_ frames	specifies the maximum number of information frames the node may send before it must pass the token.
device_ address_ binding	a list of the device addresses that will be used when the remote device must be accessed via a BACnet service request.
BF	**Broadcast Fire?** specifies whether the "change to fire mode" command should be broadcast 0=No 1=Yes
BS	**Broadcast Sync Time?** specifies whether to broadcast time 0=No 1=Yes
BT	**Broadcast Time Interval** specifies the time (in minutes) between network broadcasts Defaults to 255 (off).
CF	**Communication Failure Timeout** specifies the amount of time, in seconds, that the controller will wait to be polled before entering a communication fault state.
CM	**Manufacturer** indicates the factory-set manufacturer number for the controller. (**CM** for American Auto-Matrix controllers is always 255)
CP	**Communication Speed** the rate at which the controller communicates. 0=9600 6=38400 (default) 7=19200 8=115.2k 9=57.6k

Property	Description
CR	**Configure Remote I/O** specifies options for the configuration of remote I/O device located on the STATbus. 0=Normal 1=GPC to bus 2=Edit I/O GIDs 3=GPC from bus
CT	**Controller Type** factory-set controller type number for the controller. **CT** for the *NB*-GPC3 is 205.
DA	**Day of Week** specifies the current day of the week. 0=Monday 1=Tuesday 2=Wednesday 3=Thursday 4=Friday 5=Saturday 6=Sunday
DD	**Device Instance To Send Alarms To** specifies the MSTP address or device instance to which alarm messages will be sent when **IA** is set to
DE	**Default Enable** used to return the controller to a default state. 0=Normal operation 197=set device to default
EM	**Engineering Units** specifies the units to be used when returning values 0=English 1=Metric
ET	**Daylight Saving End Time** specifies the time at which daylight saving time ends.
FA	**Faults Detected** a bitmap indicating the current fault conditions. bit 0=Fire bit 1=Emergency bit 6=Communications failure 0=Normal 1=Fault condition
FC	**Flash update count** indicates the number of times the controller has been flashed.
FT	**Firmware Type** indicates which firmware is installed on the controller 7=GPC3 8=GPC2 9=GPC3

Property	Description
IA	**Intrinsic Alarming** determines what alarm messages are generated and how they are routed. 0=None 1=Track Alarms 2=Broadcast 3=Unconfirmed to Device 4=Confirmed to Device
ID	**Unit Number (ID)** specifies the controller's identification number. The value of ID defaults to the last four digits of the unit's serial number.
ND	**Daylight Saving End Day** specifies the day on which daylight savings time ends 0 = None 1 = First Sunday 2 = First Friday 3 = Second Saturday 4 = Third Sunday 5 = Third Saturday 6 = Last Sunday 7 = Last Thursday 8 = Last Friday
NM	**Daylight Saving End Month** specifies the month in which daylight savings tim ends 0 = None 1 = January 2 = February 3 = March, 4 = April 5 = May 6 = June 7 = July 8 = August 9 = September 10 = October 11 = November 12 = December
OS	**Kernel Version** indicates the version number of the kernel.
PD	**Power-Up Delay** time delay (in seconds) that must elapse after the controller is reset before it begins control and alarming functions. 0=No delay 1-255=# of seconds
PF	**Priority for Fault Alarms** specifies where in the priority array to-fault alarms will be listed.
PI	**Process ID for Alarms**
PN	**Priority for Normal Alarms** specifies where in the priority array to-normal alarms will be listed.

Property	Description
PO	**Priority for Off-normal Alarms** specifies where in the priority array to-offnormal alarms will be listed.
PS	**Power-up state** schedule state that the controller will operate in when it is first powered up or after power is restored following a power failure. 0=unoccupied 1=warmup 2=occupied 3=night setback
RD	**Daylight Savings Start Day** specifies the day on which daylight saving time begins 0 = None 1 = First Sunday 2 = First Friday 3 = Second Saturday 4 = Third Sunday 5 = Third Saturday 6 = Last Sunday 7 = Last Thursday 8 = Last Friday
RM	**Daylight Savings Start Month** specifies the month in which daylight savings time begins 0 = None 1 = January 2 = February 3 = March, 4 = April 5 = May 6 = June 7 = July 8 = August 9 = September 10 = October 11 = November 12 = December
RS	**Reset the Controller?** used to reset the controller. Setting **RS** to 1 resets the controller. 0=No 1=Yes
SN	**Serial Number** the factory-set serial number.
SR	**Flash Release Code** the release code of the firmware currently flashed on the controller, used primarily for technical support purposes.
ST	**Daylight Saving Start Time** specifies the time at which daylight savings time begins.
TF	**Time to Remain in Fire Mode** The time (in minutes) that the controller will remain in fire mode.
UD	**Use Device** specifies whether the address specified in **DD** to which alarm messages will be sent is an MSTP address of a device instance. 0=MSTP Address 1=Device Instance
VE	**Firmware Version** contains the version of the controller’s firmware.
ZN	**Zone Number** specifies the zone number for the controller. All zone broadcasts will only be sent to controllers with matching zone numbers.

3.2 NOTIFICATIONCLASS1

The NOTIFICATIONCLASS1 is a Notification Class object which defines the distribution of event notifications (alarm events) within a BACnet system. Notification Class objects are useful when multiple event-initiating objects have identical needs as to how the generated notifications should be routed, what the designation(s) for the notifications should be, how the notifications should be acknowledged, etc.

The broadcasting and reporting of alarms and events in the *NB*-GPC3 is handled by the properties in the Device object associated with Intrinsic Alarming. Changes made to these properties are internally delegated to the Notification Class object.

The NOTIFICATIONCLASS1 object has the following properties: **ack_required**, **notification_class**, **object_identifier**, **object_name**, **object_type**, **priority**, and **recipient_list**.

NOTE
The NOTIFICATIONCLASS1 object contains no properties which are configurable by the user.

NOTIFICATIONCLASS1 Properties

property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
notification_ class	specifies the notification class to be used when handling and generating event notifications for this object.
priority	specifies the priority to be used for event notifications for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively.
ack_required	three separate flags that indicate whether acknowledgment shall be required in notifications generated for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL event transitions, respectively.
recipient_list	a list of one or more recipient destinations to which notifications shall be sent when event-initiating objects using this class detect the occurrence of an event.

3.3 PROGRAMS 1-8

The Program objects contain properties that are used to monitor and control the execution of the SPL programs that have been downloaded to the *NB*-GPC3. These properties are useful for debugging SPL programs as well as monitoring the progress of running programs. Each of the Program objects contain the following properties: **decription_of_halt**, **object_identifier, object_name**, **object_type**, **out_of_service**, **program_location**, **program_change**, **program_state**, **reason_for_halt**, **reliability status_flags**, **$1**, **$D**, **$E**, **$N**, **$W**, **%A**, **%B**, **%C**, **%D**, **%E**, **%F**, **%G**, **%H**, **%I**, **%J** , **%K**, **%L**, **%M**, **%N**, **%O** and **%P**.

Property **object_name** stores the name of the object. This is a user definable string that can be used to help identify the object or the program that is loaded.

Property **$1** enables single-step mode. **$1** is used as a debugging tool for SPL programs, allowing the program to execute one line at a time. When single-step mode is enabled (**$1**=1), the program will execute a single line and then stop. The program must then be manually restarted by setting **program_change** to “2=Run” in order to execute the next statement. Setting **$1**=0 returns the controller to normal operation.

Property **$D** is the delay time remaining. It indicates the number of seconds remaining when an **SWAIT** or **MWAIT** statement is encountered in the SPL program.

Property **$E** is the error code. When an SPL program aborts (**$$**=3), an error code is generated and placed in **$E**. A complete list of the possible error code that can be generated is given in Appendix B: SPL Error Codes.

Property **$W** indicates what action to take in the event of a trappable error. When this occurs, the program can be configured to abort (**$W**=0) or wait until the error no longer exists (**$W**=1). Changing this property has the same results as including **ERRORABORT** and **ERRORWAIT** statements in the SPL program code.

Properties **%A** through **%P** are program registers. Each one shows the current value of the associated register. Each register has a 32-bit value and a data type that is determined automatically. The values of the registers can be changed by entering different values for each of these properties.

NOTE
User-defined properties are not visible in the associated Program object. The values of these properties can be read using the Manual Read/Write option in *NB*-Pro.

NOTE
The maximum program file size is 8k.

Program Properties

property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
program_state	indicates the current logical state of the process executing the application program this object represents.
program_change	used to request changes to the operating state of the process this object represents.
reason_for_halt	indicates the reason why the program was halted.
description_of_ halt	describes the reason why a program has been halted.
program_ location	indicate the current location within the program code, for example, a line number or program label or section name.
description	describes the application being carried out by this program or other descriptive information. Note: the **description** is only enabled if it is declared as property of type CHARSTRING in the SPL program.
status_flags	four flags that indicate the general "health" of the program.
reliability	indicates whether the program is running/waiting (no fault detected) or is unreliable (process-error)
out_of_service	indicates whether or not the process this object represents is not in service.
$1	**Enable Single-Step Mode?** specifies whether the single-step, line by line debugging mode is enabled. 0=No 1=Yes
$D	**Delay Time Remaining** specifies the number of seconds remaining when an **SWAIT** or **MWAIT** statement is encountered in the SPL program.
$E	**Error Code** indicates the SPL error code that is returned when the program aborts.
$N	**Number of Program Attributes**
$W	**Trappable Error Action** specifies how the SPL program should handle trappable errors. 0=Abort on Error 1=Wait on Error

property	Description
%A	**Register A Value** indicates the value of program register A.
%B	**Register B Value** indicates the value of program register B.
%C	**Register C Value** indicates the value of program register C.
%D	**Register D Value** indicates the value of program register D.
%E	**Register E Value** indicates the value of program register E.
%F	**Register F Value** indicates the value of program register F.
%G	**Register G Value** indicates the value of program register G.
%H	**Register H Value** indicates the value of program register H.
%I	**Register I Value** indicates the value of program register I.
%J	**Register J Value** indicates the value of program register J.
%K	**Register K Value** indicates the value of program register K.
%L	**Register L Value** indicates the value of program register L.
%M	**Register M Value** indicates the value of program register M.
%N	**Register N Value** indicates the value of program register N.
%O	**Register O Value** indicates the value of program register O.
%P	**Register P Value** indicates the value of program register P.

3.4 FILE0

The FILE0 object is a file object similar to the PLB objects but is used for a different purpose It is used for uploading flash updates directly to the controller. This alternative method of updating the controller can be used rather than using the flash update options in *NB*-Pro, making it possible to update the controller in situations where *NB*-Pro may not be available. This also allows flash updates to be performed using third party front ends which support file transfers.

To update the controller using the FILE0 object, you must download the appropriate .aff file to the FILE0 object. The flash update file will run automatically and install the update. Once the installation is complete, the *NB*-GPC3 will reset itself. The flash update is complete after the controller is reset.

NOTE
Unlike the PLB objects, the FILE0 object may not be cleared by setting the **file_size** property to zero.

The FILE0 object has the following properties: **archive**, **description**, **file_access_method, file_size**, **file_type**, **modification_date**, **object_identifier**, **object_name**, **object_type**, and **read_only**.

NOTE
You should not change the values of any properties in the FILE0 object.

File0 Properties

property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
description	provides a description of the object. This object is intended for flash updates. The **description** is the same as the **file_type**.
file_type	identifies the intended use of this file.
file_size	indicates the size of the file data.
modification_ date	indicates the last time this object was modified.
archive	indicates whether the File object has been saved for historical or backup purposes.
read_only	indicates whether or not the file data may be changed through the use of a BACnet AtomicWriteFile service.
file_access_ method	indicates the type(s) of file access supported for this object.

3.5 PLB1-8

PLB1 through PLB8 are file objects and are used to store compiled SPL code on the *NB*-GPC3. The options presented in the PLB objects refer to the file itself while the options for controlling the execution of the running program are found in the corresponding Program objects. Each PLB object has the following properties: **archive**, **description**, **file_access_method, file_size**, **file_type**, **modification_date**, **object_identifier**, **object_name**, **object_type**, and **read_only**.

The **object_name** property is the descriptive name given to the file object. The object will have a name automatically assigned to it based on what, if anything, is loaded into it. Empty file objects will be named "RAM" followed by a number representing which file object it is. For example, if the third file object were empty, its **object_name** would be automatically se to "RAM3". File objects into which programs are loaded will be named "PLB" followed by a number. A program loaded into the fourth file object would cause the **object_name** to be set to "PLB4". If the file object has a custom display sequence loaded into it, that file object will have a **object_name** set to "LOGO" followed by a number. In this way, loading a custom display sequence into the second file objetc would cause its **object_name** to be set to "LOGO2".

The **file_size** property is the only property in the PLB objects whose value you should change. When a file is loaded, the **file_size** property indicates how much memory the file is taking up. If you wish to remove the file from memory, you must set the **file_size** property to zero. This will clear the file stored in the object.

NOTE
Setting the **file_size** property to zero will remove the stored program from memory.

Clearing a PLB removes that program from the controller. If a program is loaded and not removed it will run every time the controller is reset or power is applied. You would want to clear the PLB file to avoid undesired programs from executing when the controller is powered.

PLB Properties

property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
description	provides a description of the object. This object is intended for flash updates. The **description** is the same as the **file_type**.
file_type	identifies the intended use of this file. The possible file types are: Empty Region *n* System File *n* Trend File *n* Table File *n* Program Logic Block *n* Program Reference Block *n* Program Control Block *n* Display List *n* Custom Logo *n*
file_size	indicates the size of the file data.
modification_ date	indicates the last time this object was modified.
archive	indicates whether the File object has been saved for historical or backup purposes.
read_only	indicates whether or not the file data may be changed through the use of a BACnet AtomicWriteFile service.
file_access_ method	indicates the type(s) of file access supported for this object.

3.6 UNIVERSAL INPUTS 1-24

The Universal Input objects specify the configuration and scaling information for each of the universal inputs on the *NB*-GPC3. The properties in each Universal Input object are identical. Each Universal Input object has the following properties: **aked_transitions**, **deadband**, **event_enable**, **high_limit**, **low_limit**, **max_pres_value**, **min_pres_value**, **notification_class**, **notify_type, object_identifier**, **object_name**, **object_type**, **out_of_svc**, **units**, **present_value**, **reliability**, **resolution, status_flags**, **time_delay**, **AE**, **DF, GI**, **IF**, **OF**, **RH**, **RL**, **SM**, **ST**, **SU,** and **TM**.

The **present_value** property displays the current measured input value of the associated universal input.

The **reliability** property displays the object reliability. This property indicates whether or not the value of an input is questionable. An input is considered questionable if it is outside of its range. A good example of this would be a 4-20 mA sensor. Sensor readings from 0-4 mA would be considered questionable because the sensor should normally never have a reading in that range. In this case, the controller would clamp the readings below 4 mA at 4 mA and set the corresponding reliability property to questionable (**reliability**=1). In the case of a digital input, the reliability can still be questionable. **reliability** will be set equal to 1 while the input filter delay (**IF**) elapses or in the event of a communications failure when a host is controlling the state of the output.

The **polarity** property is the input polarity bitmap for digital inputs on the *NB*-GPC3. This property specifies how the controller represents the input internally. A value of **polarity**=0 indicates normal polarity. In this case low voltage is displayed as **present_value**=0 and high voltage is displayed as **present_value**=1. If **polarity**=1, the input is configured to operate using reverse polarity. Here, low voltage is displayed as **present_value**=1 and high voltage is displayed as **present_value**=0.

The **(ST) Sensor Type** property specifies the type of sensor connected to the input. **ST** lets you define the type of sensor as well as the type of scaling to be used when displaying the **present_value** for that sensor. The available types of sensors that can be used with the *NB*-GPC3 are listed in Table 3-11.

Table 3-11 : Sensor Types

ST	Sensor Type
0	Digital
2	MN..MX 0 to 5V
3	MN..MX 4 to 20mA
4	Curve 1
5	Curve 2
6	MN..MX 0 to 10V
7	Thermistor -30.0 to 230.0
8	MN..MX 0 to 20mA
9	SmartStat Temp
10	SmartStat Humidity

When the input is configured as a digital input (**ST**=0), the value of **present_value** (either 0 or 1) is determined by the value of the input voltage with respect to a voltage threshold and input polarity

(**polarity**). An input filter delay (**IF**) is included to minimize the bouncing that may occur with digital inputs.

For linear, analog input devices that operate in the range 0-5 V (**ST**=2), 4-20 mA (**ST**=3), 0-10 V (**ST**=6), or 0-20 mA (**ST**=8) the minimum and maximum values are scaled to **min_pres_value** and **max_pres_value** respectively. For example, if you wanted a 0-10 V input to be displayed as a percentage, you would set **min_pres_value**=0 and **max_pres_value**=100. The controller would then scale the raw counts (0-16,777,215) to the range 0-100. Here zero counts would correspond to a value of **present_value**=0 and 16,777,215 counts would correspond to a **present_value**=100.

NOTE
By making the value of **min_pres_value** greater than **max_pres_value**, you can configure the input for reverse scaling.

The *NB*-GPC3 is also capable of handling nonlinear sensor types. The most common type of non-linear sensor is a precon-type 3 thermistor. When the input is configured as a thermistor input (**ST**=7), it is assumed to be a precon-type 3 thermistor with a range of -30º to 230º F. The scaling for this type of input is performed using a built in table that is automatically referenced when this sensor type is selected.

If a different type of non-linear sensor is needed, it is possible to use one of the Piecewise Curve objects to scale the input (**ST**=4). This option works much like the thermistor input, but instead of referencing a built-in table, the table that correlates input counts to an output value is defined in a Piecewise Curve object.

As mentioned previously, properties **min_pres_value** and **max_pres_value** define the minimum and maximum scaled values, in Engineering Units, for the associated input configured to operate as a linear analog input (**ST**=2, 3, 6, or 8).

The **(DF) Display Format** property is used to determine how a STAT associated with the Universal Input will display the temperature. You have the option to include one decimal place in the display or simply display the nearest integer value for the temperature. Also, you have the option to display a degree symbol or a degree symbol and either a capital “F” or a capital “C” depending on the engineering units selected in the Device object. The options for **DF** are given in Table 3-12.

Table 3-12 : Display Formats

DF	Display Format
0	##º
1	##.#º
2	##ºF
3	##.#ºF
4	None

The **alarm_value, deadband**, **event_enable**, **limit_enable**, **high_limit**, **low_limit**, **notification_class**, **time_delay**, **(AE) Alarm Function** and **(SU) Amount to Setup/Setback Alarm Limit** properties are used to configure the controller's alarming capability for the associated input.

The **(AE) Alarm Function** property enables the various types of alarming for the associated input. This property can specify one of 20 alarming types to be used. Analog inputs can be configured for high limit alarming, low limit alarming, or both high and low limit alarming. Digital inputs can be configured to generate an alarm when the **present_value** transitions from 0 to 1, transitions from 1 to 0, or both. A digital input can be configured as a fire sensor and will generate a fire alarm based on the value read. A digital input may also be configured as a supervisory monitor of digital outputs. The complete list of the types of alarming are given in Table 3-13.

Table 3-13 : Alarm Functions

AE	Function
0	Disabled
1	Contact 0 > 1
2	Contact 1 > 0
3	Change of State
4	Low Limit
5	High Limit
6	Low & High Limit
8	Fire
11	Supervise Digital Output 1
12	Supervise Digital Output 2
13	Supervise Digital Output 3
14	Supervise Digital Output 4
15	Supervise Digital Output 5
16	Supervise Digital Output 6
17	Supervise Digital Output 7
18	Supervise Digital Output 8
19	Supervise Digital Output 9
20	Supervise Digital Output 10
21	Supervise Digital Output 11
22	Supervise Digital Output 12

If **AE**=0, then alarming has been disabled for the corresponding input object.

When **AE**=1, an input configured as a digital input will generate an alarm when its value transitions from 0 to 1. Similarly **AE**=2 will generate an alarm for a digital input that transitions from 1 to 0. For **AE**=3, an alarm will be generated for any change of state (0 to 1 or 1 to 0) on the digital input.

Low limit alarming (**AE**=4) will generate an alarm when the **present_value** of the analog input drops below the low alarm limit (**low_limit**). Similarly, high limit alarming (**AE**=5) will generate an alarm if the current value of the analog input exceeds the high alarm limit (**high_limit**). If **AE**=6, an alarm will be generated if **present_value** drops below the low alarm limit (**low_limit**) or exceeds the high alarm limit (**high_limit**).

When configured for fire alarming (**AE**=8), an alarm will be generated if a digital input configured for fire detection indicates a value of 1. By adjusting the input polarity (**polarity**), the fire alarming can be configured to operate so that the fire alarm state (**present_value**=1) will occur either when the signal is open (**polarity**=0) or closed (**polarity**=1). This is to accommodate different kinds of fire sensors.

For **AE**=11 through **AE**=22, the *NB*-GPC3 will supervise DO1 through DO12 respectively. When supervising an output, the controller will wait an amount of time specified in **SD** and then compare the supervisory input state to that of the output. If they do no match, a supervisory alarm will be generated.

The **low_limit** and **high_limit** properties define the low and high limits for alarming for Universal Inputs configured as analog inputs. If the **present_value** of the analog input rises above **high_limit** then a high limit alarm will be generated. If the *NB*-GPC3 is configured to broadcast alarms or to send a to-offnormal alarm to a specific device then the alarm notification will be generated. Similarly, if the **present_value** of the analog input falls below **low_limit** then a low limit alarm will be generated and the appropriate alarm events will be generated.

The **deadband** property defines an alarm limit hysteresis. This value determines when the controller will return from a high- or low-limit alarm condition. If the controller is currently in a high limit alarm, then the **present_value** must drop below **high_limit** by an amount **deadband** before a limit return is generated. Similarly if the controller is in a low limit alarm, then the **present_value** must rise above **low_limit** by an amount **deadband** before controller function returns to normal. The effects of **low_limit**, **high_limit**, and **deadband** are shown in Figure 3-1.

Alarm limits may be setup or setback during scheduled unoccupied or night setback periods when control can be less stringent. The amount of setup/setback is specified in the **(SU) Amount to Setup/Setback Alarm Limit** property which specifies a value (0.0 to 25.5) which is added to **high_limit** and subtracted from **low_limit** during scheduled unoccupied and night setback periods.

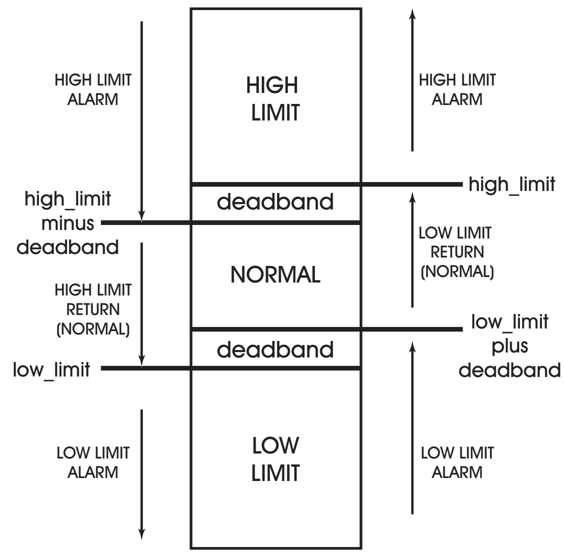

Figure 3-1: Alarm States and Thresholds for Alarming

The **time_delay** property specifies the amount of time, in seconds, that the input must remain in an alarm state before an alarm condition is generated. The **time_delay** is used to compensate for noisy signal that may fluctuate in and out of a state that would cause an alarm to be generated. The delay can be any value between 1 and 255 seconds. Setting **time_delay**=0 will disable the alarm delay.

When configured as a supervisory sensor, the Universal Input will generate a supervisory alarm when a digital input fails to make a transition when the corresponding outputs makes a transition. In this case, **time_delay** specifies a delay time, in seconds (0-255), that must elapse after the output changes state before the corresponding digital input is checked for supervision.

Regardless of the type of alarming chosen, there is an alarm delay property (**time_delay**) which specifies the amount of time, in seconds, that the input must exceed a limit before an alarm condition is generated. This delay can be any value between 1 and 255 seconds. Setting **time_delay**=0 will disable the alarm delay.

The **notification_class** property specifies the NOTIFICATIONCLASS object through which all alarm messages are routed.

WARNING
Do not change the value of the **notification_class** property. The *NB*-GPC3 has a single NOTIFICATIONCLASS object and changing the value of the **notification_class** property for an input will result in alarm messages being improperly routed.

The **(IF) Input Filter Delay** property specifies the debounce time (in seconds) during which a digital input must remain stable to avoid the signal being read as a digital bounce. In the case of a bounce, the **reliability** is set to 1. For analog inputs, **IF** specifies a weighted gain used to smooth the values from a fluctuating input.

The **(SM) Schedules to Follow** property determines which, if any, schedules the associated input will follow. By setting one or more of the bits in **SM** to 1, you can enable schedule control on the associated input. When all of the active schedules in **SM** enter an unoccupied or night setback mode, the high- and low-alarms limits will be setup/setback by an amount **SU**. Each bit of **SM** and its corresponding schedule are given in Table 3-14.

Table 3-14 : Schedules to Follow

SM bit	Schedule
0	Schedule 1
1	Schedule 2
2	Schedule 3
3	Schedule 4
4	Schedule 5
5	Schedule 6
6	Schedule 7
7	Schedule 8
8	SMARTStat 1
9	SMARTStat 2
10	SMARTStat 3
11	SMARTStat 4
12	SMARTStat 5
13	SMARTStat 6
14	SMARTStat 7
15	SMARTStat 8
16	SMARTStat 9
17	SMARTStat 10
18	SMARTStat 11
19	SMARTStat 12
20	Host Schedule
21	Schedule Summary
22	Occupancy

Bits 0-7 of **SM** are used to enable Schedule objects 1-8 respectively on the input.

Bits 8-19 enable SmartStat override. When the controller is in unoccupied mode and the *NB*-GPC1 detects that a button on the corresponding SmartStat has been pressed, it will switch to occupied mode for an amount of time specified by **ED**.

Bit 20 is used to enable host schedule override for the corresponding object. When this bit is selected, if the host schedule is enabled (**HE**=1) and the host schedule status (**HO**) indicates unoccupied or night setback mode, then the alarm limits will be setup/setback.

The **(RH) Run Hours** and **(RL) Run Limit** properties are used for runtime monitoring and alarming. **RH** indicates the number of hours **present_value**=1 for the associated digital input. **RL** is the run limit for the input. If **RH** exceeds **RL**, then a run limit alarm will be generated. Setting **RL**=0.0 disables run limit alarms for the input.

The **(GI) GID of I/O Device** indicates the global identification number of the STATbus device associated with the universal input. **GI** indicated the GID number of the STATbus device associated with the input, if there is one, and is used to assign a remote I/O device to the object when the *NB*-GPC3 is set to edit mode (STATbus:**CR**=2). If the inputs does not have a STATbus device mapped to it, **GI** will be zero.

The **(OF) Input Offset** property specifies an offset amount to be added to the **current_value** of the input. This can be used to correct the value read from sensor connected to the associated input.

The **(TM) Thermostat Multiplier** property specifies the magnitude of the user setpoint offset. Each press of the up or down buttons on a stat will increment the setpoint by an amount **TM**. There are five steps above and below the setpoint, providing for a maximum offset amount of 5x**TM** above or below the setpoint.

The **object_name** property stores the name of the object. This is a user definable string that can be used to help identify the object or, for example, the type of sensor connected to the input or the sensor's location.

Universal Input Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
present_value	indicates the current value, in engineering units, of the object.
status_flags	four flags that indicate the general "health" of the program.
event_state	provides a way to determine if this object has an active event state associated with it.
reliability	indicates whether the **present_value** is "reliable" as far as the device or operator can determine.
out_of_service	indicates whether or not the process this object represents is not in service.
polarity	indicates the relationship between the physical state of the output and the logical state represented by the **present_value** property. If the **polarity** property is NORMAL, then the ACTIVE state of the **present_value** property is also the ACTIVE or ON state of the physical output as long as **out_of_service** is FALSE. If the Polarity property is REVERSE, then the ACTIVE state of the **present_value** property is the INACTIVE or OFF state of the physical output as long as **out_of_service** is FALSE.
units	indicates the measurement units of this object.
min_pres_value	indicates the lowest number that can be reliably used for the **present_value** property of this object.
max_pres_value	indicates the highest number that can be reliably used for the **present_value** property of this object.
resolution	indicates the smallest recognizable change in **present_value** in engineering units (read-only).
time_delay	specifies the minimum period of time in seconds during which the **present_value** must be different from the **alarm_value** property before a TO-OFFNORMAL event is generated or must remain equal to the **alarm_value** property before a TO-NORMAL event is generated.
notification_class	specifies the notification class to be used when handling and generating event notifications for this object.
high_limit	specifies a limit that the **present_value** must exceed before an event is generated.
low_limit	specifies a limit below which the **present_value** must fall before an event is generated.

Property	Description
deadband	specifies a range between the **high_limit** and **low_limit** properties within which the **present_value** must remain for a TO-NORMAL event to be generated
limit_enable	enables and disables reporting of HighLimit and LowLimit offnormal events and their return to normal.
alarm_value	specifies the value that the Present_Value property must have before a TO-OFFNORMAL event is generated.
event_enable	three flags that separately enable and disable reporting of TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.
acked_transitions	three flags that separately indicate the receipt of acknowledgments for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.
notify_type	specifies whether the notifications generated by the object should be Events or Alarms.
AE	**Alarm Enable** specifies the type of alarming to be used by the input. 0=Disabled 1=Contact 0>1 2=Contact 1>0 3=Change of State 4=Low Limit 5=High Limit 6=Low & Hi Limit 8=Fire 11=Supervise Digital Output 1 12=Supervise Digital Output 2 13=Supervise Digital Output 3 14=Supervise Digital Output 4 15=Supervise Digital Output 5 16=Supervise Digital Output 6 17=Supervise Digital Output 7 18=Supervise Digital Output 8 19=Supervise Digital Output 9 20=Supervise Digital Output 10 21=Supervise Digital Output 11 22=Supervise Digital Output 12
DF	**Display Format** specifies the way in which the stat will display the temperature. 0=##d 1=##.#d 2=##dF 3=##.#dF 4=None
GI	**GID of I/O Device** indicated the global identification number of the STATbus device associated with the universal input. If the inputs does not have a STATbus device mapped to it, **GI** will be 0.

Property	Description
IF	**Input Filter Delay or Weighted Gain** the debounce time (in seconds) during which the input must remain stable to avoid the signal being read as a digital bounce. In the case of a bounce, the object **reliability** is set to 1. For analog inputs, **IF** specifies a weighted gain used to smooth the values from a fluctuating input.
OF	**Input Offset** specifies an offset amount to be added to the current value.
RH	**Run Hours** indicates the number of hours **present_value**=1 for the input.
RL	**Run Limit** specifies a number of hours that **present_value**=1 after which a run limit alarm is generated. Setting **RL**=0.0 disabled run limit alarms for the input.
SM	**Schedules to Follow** enables scheduled alarm controlling for the thermostatic control loop by selecting one or more of the available schedule control objects. 0=schedule disabled 1=schedule enabled **SM** is a bitmap with bit 0=Schedule 1 bit 1=Schedule 2 bit 2=Schedule 3 bit 3=Schedule 4 bit 4=Schedule 5 bit 5=Schedule 6 bit 6=Schedule 7 bit 7=Schedule 8 bit 8= SMARTStat 1 bit 9= SMARTStat 2 bit 10= SMARTStat 3 bit 11= SMARTStat 4 bit 12= SMARTStat 5 bit 13= SMARTStat 6 bit 14= SMARTStat 7 bit 15= SMARTStat 8 bit 16= SMARTStat 9 bit 17= SMARTStat 10 bit 18= SMARTStat 11 bit 19= SMARTStat 12 bit 20= Host Schedule bit 21= Schedule Summary bit 22= Occupancy
ST	**Sensor Type** specifies the type of sensor connected to the input. 0=Digital 2=MN..MX 0 to 5V 3=MN..MX 4 to 20mA 4=Curve 1 5=Curve 2 6=MN..MX 0 to 10V 7=Thermistor -30.0 to 230.0 8=MN..MX 0 to 20mA 9=SMARTStat Temperature 10=SMARTStat Humidity

Property	Description
SU	**Amount to Setup/Setback Alarm Limit** specifies a value (0.0 to 25.5) which is added to **high_limit** and subtracted from **low_limit** during scheduled unoccupied periods.
TM	**Thermostat Multiplier** specifies the magnitude of the user setpoint offset. Each press of the up or down buttons on a stat will increment the setpoint by an amount **TM**. There are five steps above and below the setpoint, providing for a maximum offset amount of 5x**TM** above or below the setpoint.

3.7 DIGITAL INPUT 1-8

Digital inputs operate at a higher speed than a Universal Input configured to read a digital signal. This makes it possible to not only detect an on or off signal but also count the number of transitions from one state to the other. Each Digital Input object has the following properties: **acked_transitions, alarm_value, event_enable, event_state, notification_class, notify_type, object_identifier, object_name, object_type, out_of_svc, polarity, present_value, reliability, status_flags, time_delay, AE, GI, IF, LP, MD, NP, RH, RL, SF,** and **SV.**

The **present_value** property indicates the current state of the associated digital input.

The **(LP) Interlock Polarity** property specifies how the *NB*-GPC3 handles interlocking for the digital input. For normal operation (**LP**=0), the corresponding input is considered to be normal when **present_value**=0. When the corresponding bit is configured for reverse operation (**LP**=1), then the input will be considered normal when **present_value**=1.

Properties **(AE) Alarm Function**, **(IF) Input Filter Delay**, **(RH) Run Hours**, and **(RL) Run Limit** are used to define the alarming characteristic of the digital input.

The **(AE) Alarm Function** property enables the various types of alarming for the associated input. The complete list of the types of alarming available for a pule input are given in Table 3-15.

Table 3-15 : Alarm Functions

AE	Function
0	Disabled
1	Contact 0 > 1
2	Contact 1 > 0
3	Change of State
8	Fire
11	Supervise Digital Output 1
12	Supervise Digital Output 2
13	Supervise Digital Output 3
14	Supervise Digital Output 4
15	Supervise Digital Output 5
16	Supervise Digital Output 6
17	Supervise Digital Output 7
18	Supervise Digital Output 8
19	Supervise Digital Output 9
20	Supervise Digital Output 10
21	Supervise Digital Output 11
22	Supervise Digital Output 12

If **AE**=0, then alarming has been disabled for the corresponding input object.

When **AE**=1, the digital input will generate an alarm when its value transitions from 0 to 1. Similarly **AE**=2 will generate an alarm for a transition from 1 to 0. For **AE**=3, an alarm will be generated for any change of state (0 to 1 or 1 to 0) on the digital input.

When configured for fire alarming (**AE**=8), an alarm will be generated if a digital input configured for fire detection indicates a value of 1. By adjusting the input's **polarity** property, the fire alarming can be configured to operate so that the fire alarm state (**present_value**=1) will occur either when the signal is open (**polarity**=0) or closed (**polarity**=1). This is to accommodate different kinds of fire sensors.

For **AE**=11 through **AE**=22, the *NB*-GPC3 will supervise Digital Output 1 through Digital Output 12 respectively. When supervising an output, the controller will wait an amount of time specified in **SD** and then compare the supervisory input state to that of the output. If they do not match, a supervisory alarm will be generated.

For all types of alarming, there is a **time_delay** property which specifies the amount of time, in seconds, that the input must stay in an alarm state before an alarm condition is actually generated. This delay can be any value between 1 and 255 seconds. Setting **time_delay**=0 will disable the alarm delay. **time_delay** is used to compensate for noisy signal that may fluctuate in and out of a state that would cause an alarm to be generated.

The **(IF) Input Filter Delay** property is the debounce time (in seconds) during which a digital input must remain stable to avoid the signal being read as a digital bounce. In the case of a bounce, the object **reliability** is set to 1.

The **(RH) Run Hours** and **(RL) Run Limit** properties are used for runtime monitoring and alarming. **RH** indicates the number of hours **present_value**=1 for the associated digital input. **RL** is the run limit for the input. If **RH** exceeds **RL**, then a run limit alarm will be generated. Setting **RL**=0.0 disabled run limit alarms for the input.

The **reliability** property displays the object reliability. The **reliability** indicates whether an input is questionable. In the case of a digital input, the reliability will be 1 during the time when the input filter delay elapses.

The **polarity** property displays the input polarity of the associated digital input. **polarity** specifies how the controller internally represents the input. When an input is set to normal polarity (**polarity**=0), low voltage is displayed as **present_value**=0 and high voltage is displayed as **present_value**=1. If, however, the polarity is set to reverse polarity (**polarity**=1), then low voltage will be displayed as **present_value**=1 and high voltage as **present_value**=0.

The **(MD) Pulse Counter Mode**, **(NP) Number of Pulses Accumulated**, **(SF) Pulse Multiplier**, and **(SV) Scaled Pulse Count** properties are used when the digital input is set for pulse counting (**ST**=1).

The **(MD) Pulse Counter Mode** property defines how the *NB*-GPC3 will detect a pulse. There are two ways that a pulse can be detected, by sensing the rising edge of the pulse (**MD**=0), the falling edge of the pulse (**MD**=1), or both (**MD**=2). The differences between the rising edge and falling edge of the pulse are shown in Figure 3-2.

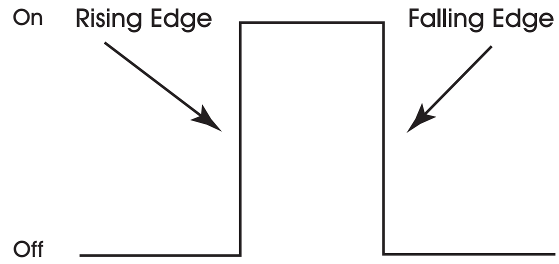

Figure 3-2: Rising and Falling Edges of a Pulse

The Pulse Counter Mode can also be set to "None" (**MD**=3). This option allows the *NB*-GPC3 to detect AC signals. When this mode is selected, the digital input will have a **present_value** of 1 when an AC voltage is detected on the input or a **present_value** of 0 if no AC voltage is detected No pulses will be accumulated when the Pulse counter mode is set to "None".

The **(NP) Number of Pulses Accumulated** property stores the raw number of pulses that have been detected by the controller. This value is also converted to a scaled pulse count, stored in the **(SV) Scaled Pulse Count** property, by multiplying the value of **NP** by the value of the **(SF) Pulse Multiplier** property. **SF** specifies a scaling factor (0.000 to 65.535) that is multiplied by **NP** to obtain a scaled count (**SV**). The final scaled pulse count **SV** is found using the formula **SV** = **NP** x **SF**

NOTE
To reset the scaled count, **SV**, you must write a value of zero to the number of pulses **NP**.

The **(GI) GID of I/O Device** property is the GID number of the STATbus device associated with the digital input. When a STATbus device is used as an digital input, the global identification number of that device will be set using in **GI**. If no STATbus device is being used, **GI** will display a value of zero.

The **object_name** property stores the name of the object. This is a user definable string that can be used to help identify the object or, for example, the type of sensor connected to the input or the sensor's location.

Digital Input Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
present_value	indicates the current value, in engineering units, of the object.
status_flags	four flags that indicate the general "health" of the program.
event_state	provides a way to determine if this object has an active event state associated with it.
reliability	indicates whether the **present_value** is "reliable" as far as the device or operator can determine.
out_of_service	indicates whether or not the process this object represents is not in service.
polarity	indicates the relationship between the physical state of the output and the logical state represented by the **present_value**e property. If the **polarity** property is NORMAL, then the ACTIVE state of the **present_value** property is also the ACTIVE or ON state of the physical output as long as **out_of_service** is FALSE. If the Polarity property is REVERSE, then the ACTIVE state of the **present_value** property is the INACTIVE or OFF state of the physical output as long as **out_of_service** is FALSE.
time_delay	specifies the minimum period of time in seconds during which the **present_value** must be different from the **alarm_value** property before a TO-OFFNORMAL event is generated or must remain equal to the **alarm_value** property before a TO-NORMAL event is generated.
notification_ class	specifies the notification class to be used when handling and generating event notifications for this object.
alarm_value	specifies the value that the Present_Value property must have before a TO-OFFNORMAL event is generated.
event_enable	three flags that separately enable and disable reporting of TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.
acked_ transitions	three flags that separately indicate the receipt of acknowledgments for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.
notify_type	specifies whether the notifications generated by the object should be Events or Alarms.

Property	Description
AE	**Alarm Enable** specifies the type of alarming to be used by the input. 0=Disabled 1=Contact 0>1 2=Contact 1>0 3=Change Of State 8=Fire 11=Supervise Digital Output 1 12=Supervise Digital Output 2 13=Supervise Digital Output 3 14=Supervise Digital Output 4 15=Supervise Digital Output 5 16=Supervise Digital Output 6 17=Supervise Digital Output 7 18=Supervise Digital Output 8 19=Supervise Digital Output 9 20=Supervise Digital Output 10 21=Supervise Digital Output 11 22=Supervise Digital Output 12
GI	**GID of I/O Device** indicated the global identification number of the STATbus device associated with the output. If the inputs does not have a STATbus device mapped to it, **GI** will be 0.
IF	**Input Filter Delay** the debounce time (in seconds) during which the input must remain stable to avoid the signal being read as a digital bounce. In the case of a bounce, the object **reliability** is set to 1.
LP	**Interlock Polarity**
MD	**Pulse Counter Mode** defines how the *NB*-GPC3 detects a pulse. 0=Rising edge 1=Falling edge 2=Both 3=None
NP	**Number of Pulses Accumulated** displays the current number of pulses detected on the digital input selected in **IC**.
RH	**Run Hours** indicates the number of hours **present_value**=1 for the input.
RL	**Run Limit** specifies a number of hours that **present_value**=1after which a run limit alarm is generated. Setting **RL**=0.0 disabled run limit alarms for the input.
SF	**Pulse Multiplier** specifies a scaling factor (0.000 to 65.535) that is multiplied by **NP** to obtain a scaled count.
SV	**Scaled Pulse Count** displays a scaled version of NP using the formula **SV** = **NP** x **SF**

3.8 ANALOG OUTPUTS 1-12

The Analog Output objects on the *NB*-GPC3 allow the user to configure the options for each analog output. The user can set minimum and maximum values, both in engineering units as well as scaled values. The user can also set up various alarming options for the analog output. Each Analog Output object has the following properties: **acked_transitions, event_state, max_pres_value, min_pres_value, object_identifier, object_name, object_type, out_of_svc, present_value, priority_array reliability, relinquish_default, resolution, status_flags, units, CF, FI, FP, GI, IL, MN, MX, OU,** and **UT.**

The **present_value** property indicates the current value of the associated analog output. The current value of the output can be manually controlled by a user directly writing a value to **present_value**, controlled by a host controller writing a value to **present_value**, or controlled automatically by the *NB*-GPC3.

The **(OU) actual output value** property displays the actual state of the output. Because of control loop delays and other factors, **OU** may be different from **present_value**.

The **(UT) update threshold** property is the threshold value by which **present_value** must change before the output is updated. This property can be used to reduce hunting when it is given a noisy signal.

The **min_pres_value** and **max_pres_value** properties are the minimum and maximum engineering units properties. These properties define the range of **present_value** that is displayed by the *NB*-GPC3. The output's value may be scaled to units that are more meaningful in the control application being used (e.g., percent, psi, voltage, current, etc.). The minimum and maximum values of the range can be set in properties **min_pres_value** and **max_pres_value**. Reverse scaling can be accomplished by setting **min_pres_value** > **max_pres_value**. **min_pres_value** and **max_pres_value** default to 0.0 and 100.0, respectively.

The **(MN) Minimum Scaled Value** and **(MX) Maximum Scaled Value** properties specify the range of the total output signal that is scaled across **min_pres_value** and **max_pres_value**. The **present_value** is restricted to the range **MN** < **present_value** < **MX**. Reverse scaling can be accomplished by setting **MN** > **MX**. **MN** and **MX** default to 0.0% and 100.0%, respectively.

For example, if the analog output value (**present_value**) is to be displayed as a percentage (0-100) of a 10 VDC output range, **min_pres_value** would be set to 0 and **max_pres_value** to 100 (a 0.0%-100.0% display range). **MN** would be set to 0.0% and **MX** to 100.0% (the full range of the output signal), because **present_value**=0 represents 0.0% of the output range and **present_value**=100 represents 100.0% of the output range.

The **reliability** property indicates object reliability for the analog outputs. The reliability is only used for outputs on the STATbus. When a control sequence within the *NB*-GPC3 changes the value of an output, the *NB*-GPC3 sends a command out over the STATbus network telling the output device to change its value. If the message to change its value is successfully received, then the STATbus device will send a message back to the *NB*-GPC3 acknowledging the request. If the *NB*-GPC3 sends such a message, but does not receive an acknowledgement, then the controller will flag the output as unreliable.

The **(CF) Communications Failure Enable?** property enables the communications failure feature for the associated analog output object of the *NB*-GPC3. If Device:**(CF) Communications Failure Timeout** is set to a non-zero value, a communications timeout occurs, and the **CF** property for the analog output is set to 1, then the **present_value** of the output will be set to the value stored in the programmed failure position (**FP**).

The **(IL) Inputs for Interlocking** property used in conjunction with **FP** for interlocking of the analog output. Each bit (0-24) in the **IL** bitmap corresponds to a universal input object of the *NB*-GPC3. Any universal input configured to read a digital signal as well as any of the digital input objects may be used for interlocking.

One or more of these inputs can be selected as interlocks by setting the corresponding bit in **IL** to 1. In the event that any of the digital inputs specified in **IL** has a value of 1, then the current value (**present_value**) assumes its interlock failure position (**FP**). All of the inputs specified in **IL** must have a current value of 0 before normal control is restored to **present_value**.

The **(FP) Interlock/Communications Failure Position** property specifies the **present_value** that the output should take in the event an interlock or communications failure occurs. An interlock failure occurs when the value of any of the bits set in **IL** equals 1. A communications failure occurs when a host or peer controller has an SPL program controlling the *NB*-GPC3's outputs and the communications timeout is exceeded.

The **(FI) Fire Position** property specifies the desired position of the output to which the output will revert in the event that a fire condition (Device Object:**FA** bit 0=1) is detected.

NOTE
If a communications failure or interlock failure occurs, then the desired position of the output reverts to **FP**. If a fire condition then occurs while the communications or interlock failure is still active, **FI** takes priority over **FP**.

NOTE
The output does not have to be assigned to a PID or Thermostatic Control loop to be set or reset by fire or interlock failures.

The **(GI) GID of I/O Device** property is the GID number of the STATbus device connected to the analog output. When a STATbus device is used as an analog output, the global identification number of that device will be displayed in **GI**. Otherwise, **GI** will display a value of zero.

The **object_name** property stores the name of the object. This is a user definable string that can be used to help identify the object or, for example, the type of output connected.

3.8.1 COMMAND PRIORITIZATION LIMITS

In the GPC, as well as BACnet standard, there are certain BACnet Priority Levels that are reserved and cannot be written with to the GPC. All priority levels can be written to with the exception of the following for Analog Outputs:

- Priority Level 4 - reserved for Fire Alarm functions
- Priority Level 7 - reserved for Input Interlocking functions
- Priority Level 11 - reserved for Control Algorithm (loops, schedules, etc.) interaction

While levels 4 and 7 cannot be commanded with in GPC, 6 and 11 may be used. Be aware that if your Analog Output has a control algorithm associated to it, the write command that you subject will likely be overwritten by the GPC's internal control algorithm.

Analog Output Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
present_value	indicates the current value, in engineering units, of the object.
status_flags	four flags that indicate the general "health" of the program.
event_state	provides a way to determine if this object has an active event state associated with it.
reliability	indicates whether the **present_value** is "reliable" as far as the device or operator can determine.
out_of_service	indicates whether or not the process this object represents is not in service.
units	indicates the measurement units of this object.
min_pres_value	indicates the lowest number that can be reliably used for the **present_value** property of this object.
max_pres_value	indicates the highest number that can be reliably used for the **present_value** property of this object.
resolution	indicates the smallest recognizable change in **present_value** in engineering units (read-only).
priority_array	contains prioritized commands that are in effect for this object.
relinquish_ default	the default value to be used for the **present_value** property when all command priority values in the **priority_array** property have a NULL value.
acked_ transitions	three flags that separately indicate the receipt of acknowledgments for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.
CF	**Communication Failure Enable?** 0=No 1=Yes
FI	**Fire Position** specifies the failure position (0-100%) for the output to be used when a fire event is received.
FP	**Interlock/Communication Failure Position** specifies the failure position (0-100%) for the output to be used in the event of an interlock or communications failure.

Property	Description
IL	**Inputs for Interlocking** specifies which inputs (if any) are used as interlocks for the associated output (a value of 1 indicates that input is used for interlocking). bit 0 = UI 1 bit 1 = UI 2 bit 2 = UI 3 bit 3 = UI 4 bit 4 = UI 5 bit 5 = UI 6 bit 6 = UI 7 bit 7 = UI 8 bit 8 = UI 9 bit 9 = UI 10 bit 10 = UI 11 bit 11 = UI 12 bit 12 = UI 13 bit 13 = UI 14 bit 14 = UI 15 bit 15 = UI 16 bit 16 = UI 17 bit 17 = UI 18 bit 18 = UI 19 bit 19 = UI 20 bit 20 = UI 21 bit 21 = UI 22 bit 22 = UI 23 bit 23 = UI 24 bit 24 = DI 1 bit 25 = DI 2 bit 26 = DI 3 bit 27 = DI 4 bit 28 = DI 5 bit 29 = DI 6 bit 30 = DI 7 bit 31 = DI 8
MN	**Minimum Scaled Voltage** specifies the percentage of the total output for **present_value=min_pres_value**.
MX	**Maximum Scaled Voltage** specifies the percentage of the total output for **present_value=max_pres_value**.
OU	**Actual Output Value** specifies the actual output state of the output. This may differ from the current value because of delays and other effects.
UT	**Update Threshold** specifies a threshold value by which **present_value** must change before the output is updated.
GI	**GID of I/O Device** indicated the global identification number of the STATbus device associated with the analog output. If the inputs does not have a STATbus device mapped to it, **GI** will be 0.

3.9 DIGITAL OUTPUTS 1-12

The Digital Output objects on the *NB*-GPC3 allow the user to configure the options for each digital output. The user can set minimum "on" and "off" times as well as set up various alarming options for the digital output. Each Digital Output object has the following properties: **acked_transitions, event_state, minimum_off_time, minumum_on_time, notify_type, object_identifier, object_name, object_type, out_of_svc, polarity, present_value, priority_array, reliability, relinquish_default, status_flags, GI, OU, PW, RH, RL, SI,** and **SM.**

The **present_value** property is the *requested state* (0 or 1), rather than the *actual state* of the output. The **present_value** can be set through schedule control, thermostatic control, or floating point control. The **present_value** can be also set manually or via SPL program. The **present_value** for each of the digital outputs can be viewed as a single bitmap as **CV** in the Digital Output Summary object.

The **(OU) Actual Output State** property displays the actual state of the digital outputs. It is useful to have both **OU** and **CV** because it is possible for the actual output state to differ from the current value because of various staging delays.

Properties **minimum_on_time** and **minimum_off_time** specify a minimum ON and OFF time for the digital input. **minimum_on_time** defines the minimum time, in seconds, that must elapse after the output is energized before it may be turned off. **minimum_off_time** specifies the minimum time, in seconds, that must elapse after the output is turned off before it may be re-energized.

The **(SI) Power-on Stagger Interval** property specifies a delay time (0-255 seconds) that must elapse after the controller is powered up or reset, before the corresponding output can be turned on. If a power on delay time is specified Device Object:**PD**>0, then the individual **SI** delays do not occur until Device Object:**PD** expires. Used in conjunction with Device Object:**PD**, multiple GPCs on the same network can consecutively stagger their outputs in the event of a network power failure.

The **(PW) Pulse Width when Output is On** property specifies a pulse width (0.0-25.5 seconds) that is applied to the digital output while **present_value**=1 (on). **PW** allows digital outputs to be in the on state for durations accurate to one-tenth of a second. For example, a pulse width of 5.8 seconds (**PW**=5.8) may have an actual pulse duration of anywhere from 5.7 (**PW**-0.1) to 5.9 seconds (**PW**+0.1). Setting **PW**=0 disables minimum cycle times for the digital outputs and disables pulsed output control.

The **(RH) Run Hours** and **((RL) Run Limit** properties are used for runtime monitoring and alarming. **RH** indicates the number of hours **present_value**=1 for the associated digital output. **RL** is the run limit for the input. If **RH** exceeds **RL**, then a run limit alarm will be generated. Setting **RL**=0.0 disables run limit alarms for the input.

The **(GI) GID of I/O Device** property is the GID of the I/O device used as a digital output. If a STATbus device is configured as a digital output, then the global identification number of that device will be displayed in **GI**. Otherwise, GI will display a value of zero.

The **(SM) Schedules to Follow** property determines which, if any, schedules the associated output will follow. By setting one or more of the bits in **SM** to 1, you can enable schedule control for the output. If any of the selected schedules in **SM** enter an occupied mode, the digital output will be enabled. Each bit of **SM** and its corresponding schedule are given in Table 3-16.

Table 3-16 : Schedules to Follow

SM bit	Schedule
0	Schedule 1
1	Schedule 2
2	Schedule 3
3	Schedule 4
4	Schedule 5
5	Schedule 6
6	Schedule 7
7	Schedule 8
8	SMARTStat 1
9	SMARTStat 2
10	SMARTStat 3
11	SMARTStat 4
12	SMARTStat 5
13	SMARTStat 6
14	SMARTStat 7
15	SMARTStat 8
16	SMARTStat 9
17	SMARTStat 10
18	SMARTStat 11
19	SMARTStat 12
20	Host Schedule
21	Schedule Summary
22	Occupancy

Bits 0-7 of **SM** are used to enable Schedule objects 1-8 respectively on the input.

Bits 8-19 enables SmartStat override. When the controller is in unoccupied mode and the *NB*-GPC1 detects that a button on the SmartStat has been pressed, it will switch to occupied mode for an amount of time specified by **ED**.

Bit 20 is used to enable host schedule override for the corresponding object. When this bit is selected, if the host schedule is enabled (**HE**=1) control will follow the host schedule status (**HO**).

The **(IL) Inputs for Interlocking** property used in conjunction with **FP** for interlocking of the digital output. Any universal input configured to read a digital signal as well as any of the digital input objects may be used for interlocking. **IL** is a bitmap with bits 0 through 23 corresponding to Universal Inputs 1 through 24 and bits 24 through 31 corresponding to Digital Inputs 1 through 8 of the *NB*-GPC3.

The **reliability** property indicates object reliability for each of the digital outputs. The reliability is only used for outputs on the STATbus. When a control sequence within the *NB*-GPC3 changes the value of an output, the *NB*-GPC3 sends a command out over the STATbus network telling the output device to change its value. If the message to change its value is successfully received, then the STATbus device will send a message back to the *NB*-GPC3 acknowledging the request. If the *NB*-GPC3 sends such a message, but does not receive an acknowledgement, then the controller will flag the output as unreliable and will set the **reliability** property to 1.

The **object_name** property stores the name of the object. This is a user definable string that can be used to help identify the object or, for example, to identify what the output is controlling.

3.9.1 COMMAND PRIORITIZATION LIMITS

In the GPC, as well as BACnet standard, there are certain BACnet Priority Levels that are reserved and cannot be written with to the GPC. All priority levels can be written to with the exception of the following for Binary Outputs:

- Priority Level 4 - reserved for Fire Alarm functions
- Priority Level 6 - reserved for Minimum On/Off functions
- Priority Level 7 - reserved for Input Interlocking functions
- Priority Level 11 - reserved for Control Algorithm (loops, schedules, etc.) interaction

While levels 4 and 7 cannot be commanded with in GPC, 6 and 11 may be used. Be aware that if your Digital Output has a control loop associated to it, the write command that you subject will likely be overwritten by the GPC's internal control algorithm.

Digital Output Properties

property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
present_value	indicates the current value, in engineering units, of the object.
status_flags	four flags that indicate the general "health" of the program.
event_state	provides a way to determine if this object has an active event state associated with it.
out_of_service	indicates whether or not the process this object represents is not in service.
polarity	indicates the relationship between the physical state of the output and the logical state represented by the **present_value**e property. If the **polarity** property is NORMAL, then the ACTIVE state of the **present_value** property is also the ACTIVE or ON state of the physical output as long as **out_of_service** is FALSE. If the Polarity property is REVERSE, then the ACTIVE state of the **present_value** property is the INACTIVE or OFF state of the physical output as long as **out_of_service** is FALSE.
relinquish_ default	the default value to be used for the **present_value** property when all command priority values in the **priority_array** property have a NULL value.
priority_array	contains prioritized commands that are in effect for this object.
minimum_off_ time	specifies the minimum number of seconds that the **present_value** shall remain in the INACTIVE state after a write to the **present_value** property causes that property to assume the INACTIVE state.
minimum_on_ time	indicates the minimum number of seconds that the **present_value** shall remain in the ACTIVE state after a write to the **present_value** property causes that property to assume the ACTIVE state.
reliability	indicates whether the **present_value** is "reliable" as far as the device or operator can determine.
notification_ class	specifies the notification class to be used when handling and generating event notifications for this object.
feedback_value	indicates the status of a feedback value from which the **present_value** must differ before an event is generated.
event_enable	three flags that separately enable and disable reporting of TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.

property	Description
acked_ transitions	three flags that separately indicate the receipt of acknowledgments for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.
notify_type	specifies whether the notifications generated by the object should be Events or Alarms.
event_time_ stamps	indicates the times of the last event notifications for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively.
GI	**GID of I/O Device** indicated the global identification number of the STATbus device associated with the digital output. If the inputs does not have a STATbus device mapped to it, **GI** will be 0.
IL	**Inputs for Interlocking** specifies which inputs (if any) are used as interlocks for the associated output (a value of 1 indicates that input is used for interlocking). bit 0 = UI 1 bit 1 = UI 2 bit 2 = UI 3 bit 3 = UI 4 bit 4 = UI 5 bit 5 = UI 6 bit 6 = UI 7 bit 7 = UI 8 bit 8 = UI 9 bit 9 = UI 10 bit 10 = UI 11 bit 11 = UI 12 bit 12 = UI 13 bit 13 = UI 14 bit 14 = UI 15 bit 15 = UI 16 bit 16 = UI 17 bit 17 = UI 18 bit 18 = UI 19 bit 19 = UI 20 bit 20 = UI 21 bit 21 = UI 22 bit 22 = UI 23 bit 23 = UI 24 bit 24 = DI 1 bit 25 = DI 2 bit 26 = DI 3 bit 27 = DI 4 bit 28 = DI 5 bit 29 = DI 6 bit 30 = DI 7 bit 31 = DI 8
OU	**Actual Output State** specifies the actual output state of the output. This may differ from the current value because of delays and other effects.
PW	**Pulse Width when Output is On** specifies the “on” time (**present_value**=1) in seconds (0.0 to 25.5) that the output should remain on after a transition from the off to on state. 0=Disabled 0.1-25.5=pulse “on” duration in seconds
RH	**Run Hours** indicates the number of hours **present_value**=1 for the input.

property	Description
RL	**Run Hours Limit** specifies a number of hours that **present_value**=1after which a run limit alarm is generated. Setting **RL**=0.0 disabled run limit alarms for the input.
SI	**Power-On Stagger Interval** specifies the time, in seconds, that must elapse after the controller is turned on or reset before the output can be turned on.
SM	**Schedules to Follow** enables scheduled alarm controlling for the thermostatic control loop by selecting one or more of the available schedule control objects. 0=schedule disabled 1=schedule enabled bit 0=Schedule 1 bit 1=Schedule 2 bit 2=Schedule 3 bit 3=Schedule 4 bit 4=Schedule 5 bit 5=Schedule 6 bit 6=Schedule 7 bit 7=Schedule 8 bit 8= SMARTStat 1 bit 9= SMARTStat 2 bit 10= SMARTStat 3 bit 11= SMARTStat 4 bit 12= SMARTStat 5 bit 13= SMARTStat 6 bit 14= SMARTStat 7 bit 15= SMARTStat 8 bit 16= SMARTStat 9 bit 17= SMARTStat 10 bit 18= SMARTStat 11 bit 19= SMARTStat 12 bit 20= Host Schedule bit 21= Schedule Summary bit 22= Occupancy

3.10 STATBUS 1-4

The *NB*-GPC3 is capable of using up to a total of thirteen STATbus devices connected to each of its four STATbus ports. Each of these ports has an associated STATbus object which are used to monitor the status of the STATbus network and the devices connected to it. Each STATbus object has the following properties: **object_identifier**, **object_name**, **object_type, BS**, **CD**, **CF**, **G1**, **G2**, **G3**, **G4**, **G5**, **G6**, **G7**, **G8**, **G9**, **GA**, **GB**, **GC**, **GD**, and **SM**.

Property **object_name** stores the name of the object. This is a user definable string that can be used to help identify the object.

Properties **G1** through **GD** store the global identification numbers of the devices attached to the STATbus network. After the *NB*-GPC3 has enumerated all of the STATbus devices attached to the associated STATbus object, the global identification numbers will be assigned to an internal routing table and then displayed in **G1** though **GD**. These values are read only.

The overall status of the STATbus can be monitored using the **(BS) Bus Status** property. If the STATbus object is functioning properly, is configured and is not experiencing an error, **BS** will equal 0. When **BS**=1, it indicates that the network needs to be configured. This means that a remote I/O device has been physically added and recognized by the *NB*-GPC3, but has not yet been assigned to a object. If the *NB*-GPC3 encounters an error on the STATbus, **BS** will take a value of 2 (**BS**=2).

The **(CD) Configure Device (GID)** property is used with the **(CF) Configure Function** property to help in field configuration of STATbus devices. **CF** can be used to locate a specific STATbus device in the field by causing an LED on the device to blink (**CF**=”1=Blink LED”). The device you wish to configure is selected by entering its GID number into **CD**.

The **(SM) Status Map** property is a bitmap which displays information about the current status of the STATbus. Bit 0 of **SM** is not used.

Bit 1 will be 1 if there is an unconfigured device on the network.

Bit 2 will take a value of 1 if there are duplicate configurations on the network. This could occur, for example, if an SSB-FI1 configured as UI14 was added to a network which already had a device configured as UI14.

Bit 3 indicates that the power level on the STATbus is low. It is used to determine if the bus is shorted, or loaded down to the point that it will function. When a STATbus network is heavily loaded, the power drawn by the devices may be so great as to disrupt communications. This can lead to devices not enumerating correctly or dropping communications. STAT3 devices are especially susceptible to this effect as they require additional power for the screen.

Bit 4 indicates that a re-enumeration has occurred. If bit 5 has a value of 1, it indicates that the *NB*-GPC3 had trouble enumerating a device on the network, not that enumeration failed to work and, as such, should not be a cause for concern.

STATbus Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
BS	**Bus Status** indicates the status of the SMARTStat Bus. 0=Ok 1=Needs configured 2=Error
CD	**Configure Device (GID)** specifies the device to be configured.
CF	**Configure Function** enables the blinking LED on the device specified in **CD**. 0=Disabled 1=Blink LED
G1	**GID Device 1** indicates the global identification number of device 1.
G2	**GID Device 2** indicates the global identification number of device 2.
G3	**GID Device 3** indicates the global identification number of device 3.
G4	**GID Device 4** indicates the global identification number of device 4.
G5	**GID Device 5** indicates the global identification number of device 5.
G6	**GID Device 6** indicates the global identification number of device 6.
G7	**GID Device 7** indicates the global identification number of device 7.
G8	**GID Device 8** indicates the global identification number of device 8.
G9	**GID Device 9** indicates the global identification number of device 9.
GA	**GID Device 10** indicates the global identification number of device 10.
GB	**GID Device 11** indicates the global identification number of device 11.
GC	**GID Device 12** indicates the global identification number of device 12.
GD	**GID Device 13** indicates the global identification number of device 13.
SM	**Status Map** a bitmap indicating the current status of the STATbus bit 0=not used bit 1=unconfigured device on network bit 2=duplicate configuration on network bit 3=low power bit 4=re-enumeration has occurred

3.11 UNIVERSAL INPUT SUMMARY

The Universal Input Summary object is a convenient way to monitor the states and set a number of options of options for the universal inputs on the *NB*-GPC3. From this object, you can check the current values as well as the object reliabilities for each universal input. This object will also display any inputs that are currently overridden. The input polarities as well as the interlock polarities for each object will also be displayed. The Universal Input Summary object has the following properties: **object_identifier**, **object_name**, **object_type, AT**, **ER**, **IP**, **LP**, **OI**, **PI**, **PS**, **PU**, **RE**, **SE, V1**, **V2**, **V3**, **V4**, **V5**, **V6**, **V7**, **V8**, **V9**, **VA**, **VB**, **VC**, **VD**, **VE**, **VF**, **VG**, **VH**, **VI**, **VJ**, **VK**, **VL**, **VM**, **VN**, and **VO**.

Properties **V1** through **VO** display the current measured of inputs 1 through 24, respectively. Each value is displayed separately. Each object's reliability is displayed in the **(RE) Channel Reliability Bitmap** property. This bitmap indicates which inputs have readings that are considered reliable. Inputs considered to be reliable inputs will have the corresponding bit in **RE**=0 while unreliable ones will have the corresponding bit in **RE**=1.

The **(OI) Overridden Inputs Bitmap** property displays which inputs currently have their **out_of_service** property set to 1. If an input is out of service, the corresponding bit in **OI** will be set equal to 1. If the input is reading normally, **OI** will be set to 0.

The **(IP) Input Polarities** property displays the polarities for all the universal inputs as a bitmap. Each bit of **IP** specifies how the controller internally represents the corresponding input when it is configured as a digital input. When an input is set to normal polarity (**polarity**=0), low voltage is displayed as **CV**=0 and high voltage is displayed as **CV**=1. If, however, the polarity is set to reverse polarity (**polarity**=1), then low voltage will be displayed as **CV**=1 and high voltage as **CV**=0.

The **(LP) Interlock Polarities** property displays the interlock polarities for all of the universal inputs. Each bit of **LP** indicates the individual interlock polarity states for the corresponding universal input configured as a digital input. For normal operation (**LP**=0), the corresponding digital input is considered to be normal when **CV**=0. When the corresponding bit is configured for reverse operation (**LP**=1), then the input will be considered normal when **CV**=1.

The **(AT) Delay for Alarms After Going Occupied** property is the alarm limit setup/setback time delay. **AT** specifies the time, in minutes (0-255), that must elapse after an input following a schedule makes a transition to occupied mode before alarm limits are no longer setup/setback. If a universal input has alarming enabled (**AE**>0), has an alarm limit setup/setback offset specified (**SU**>0.0), and is under schedule control (at least one bit of **SM** set to 1), then the input's alarm limits will remain setup/setback for **AT** minutes after the schedule indicates a changeover to occupied mode. If **AT**=0 minutes, the input's alarm limits will be returned to their normal occupied values as soon as the schedule indicates the change to occupied mode.

Properties **(PU) User P.I.N.**, **(PI) Installer P.I.N.**, and **(PS) Service P.I.N.** properties are used to set the P.I.N. numbers used when the universal input is configured for use with a SMARTStat. **PU** sets the P.I.N. number needed to unlock the User menu, **PI** sets the P.I.N. number needed to unlock the Installer menu, and **PS** sets the P.I.N. number needed to unlock the Service menu.

The **(SE) User Occupancy Override Enable** property is used to enable and disable the ability of the user to override scheduled unoccupied periods. When **SE** is set equal to 1, pressing the buttons on a STAT will cause the controller to switch from unoccupied to occupied mode for a predetermined amount of time.

The **(ED) Extended Occupancy Duration** property specifies the amount of time that control setpoints will be setup/setback when the SMARTStat indicates the zone is occupied during a scheduled unoccupied or night setback period.

The **(ER) Extended Occupancy Remaining** property indicates the amount of time, in minutes, that remains before the controller returns an overridden zone to unoccupied mode.

The **object_name** property stores the name of the object. This is a user definable string that can be used to help identify the object or its function.

Universal Input Summary Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
AT	**Alarm Limit setup/setback time delay** the time (in minutes) that must elapse after transitions from occupied to unoccupied or from unoccupied to occupied before alarm limits return to their previous (pre-setup/setback) values.
ED	**Extended Occupancy Duration** specifies the time, in minutes, occupancy is extended when the stat indicates an override
ER	**Extended Occupancy Remaining** indicates the amount of time remaining before the controller reverts to unoccupied mode.
IP	**Input Polarities** specifies how the controller represents the input internally. 0=Normal polarity. Low Voltage is displayed as **present_value**=0 and High Voltage is displayed as **present_value**=1. 1=Reverse polarity. Low Voltage is displayed as **present_value**=1 and High Voltage is displayed as **present_value**=0. **IP** is a bitmap with bit 0 corresponding to Universal Input 1, bit 2=Universal Input 2, etc. up to bit 24=Universal Input 24.
LP	**Interlock Polarities** indicates the individual interlock polarity states for each Universal Input. 0=Normal 1=Reverse **LP** is a bitmap with bit 0 corresponding to Universal Input 1, bit 1=Universal Input 2, etc. up to bit 23=Universal Input 24.
OI	**Overridden Inputs Bitmap** displays which inputs currently have their **out_of_service** property set to 1
PI	**Installer P.I.N.** specifies the P.I.N. code used to enter the Installer menu.
PS	**Service P.I.N.** specifies the P.I.N. code used to enter the Service menu.
PU	**User P.I.N.** specifies the P.I.N. code used to enter the User menu.
RE	**Inputs with Unreliable objects** indicates whether the reading from the corresponding input is considered reliable. 0=Reliable 1=Unreliable **RE** is a bitmap with bit 0 corresponding to Universal Input 1, bit 1=Universal Input 2, etc. up to bit 23=Universal Input 24.
SE	**User Occupancy Override Enable** specifies whether or not the user may override the schedule occupancy state.
V1	**Current Measured Input 1 Value** indicates the **present_value** of input 1.
V2	**Current Measured Input 2 Value** indicates the **present_value** of input 2.
V3	**Current Measured Input 3 Value** indicates the **present_value** of input 3.
V4	**Current Measured Input 4 Value** indicates the **present_value** of input 4.
V5	**Current Measured Input 5 Value** indicates the **present_value** of input 5.
V6	**Current Measured Input 6 Value** indicates the c**present_value** of input 6.
V7	**Current Measured Input 7 Value** indicates the **present_value** of input 7.
V8	**Current Measured Input 8 Value** indicates the **present_value** of input 8.
V9	**Current Measured Input 9 Value** indicates the**present_value** of input 9.
VA	**Current Measured Input 10 Value** indicates the**present_value** of input 10.
VB	**Current Measured Input 11 Value** indicates the **present_value** of input 11.
VC	**Current Measured Input 12 Value** indicates the**present_value** of input 12.
VD	**Current Measured Input 13 Value** indicates the **present_value** of input 13.
VE	**Current Measured Input 14 Value** indicates the **present_value** of input 14.
VF	**Current Measured Input 15 Value** indicates the **present_value**of input 15.
VG	**Current Measured Input 16 Value** indicates the **present_value** of input 16.
VH	**Current Measured Input 17 Value** indicates the c**present_value** of input 17.

Property	Description
VI	**Current Measured Input 18 Value** indicates the **present_value**of input 18.
VJ	**Current Measured Input 19 Value** indicates the **present_value** of input 19.
VK	**Current Measured Input 20 Value** indicates the **present_value** of input 20.
VL	**Current Measured Input 21 Value** indicates the**present_value** of input 21.
VM	**Current Measured Input 22 Value** indicates the **present_value** of input 22.
VN	**Current Measured Input 23 Value** indicates the **present_value** of input 23.
VO	**Current Measured Input 24Value** indicates the **present_value** of input 24.

3.12 DIGITAL INPUT SUMMARY

The Digital Input Summary object is a convenient way to monitor the states and many of the options for the digital inputs on the *NB*-GPC3. From this object, you can check the current values as well as the object reliabilities for each digital input. The input polarities as well as the interlock polarities for each object will also be displayed. The Digital Input Summary object has the following properties: **object_identifier, object_name, object_type, CV, IP, LP, OI** and **RE**.

The **(CV) current Value** property indicates the current value of the associated digital input. **CV** is a bitmap with bit 0 corresponding to DI1, bit 1=DI2, etc. up to bit 7=DI8.

The **(OI) Overridden Digital Inputs Bitmap** property indicates those inputs whose **out_of_service** property is true. **OI** is a bitmap with bit 0 corresponding to DI1, bit 1=DI2, etc. up to bit 7=DI8.

The **(RE) Channel Reliability Digital Inputs Bitmap** property displays the reliabilities for all of the digital inputs. **RE** indicates whether an input is questionable. In the case of a digital input, the reliability will be 1 during the time when the input filter delay (**IF**) elapses. **RE** is a bitmap with bit 0 corresponding to DI1, bit 1=DI2, etc. up to bit 7=DI8. A value of 0 indicates that the input is reliable while a value of 1 means that the input associated with that bit is unreliable.

The **(LP) Digital Interlock Polarities** property specifies how the *NB*-GPC3 handles interlocking. Each bit of **LP** indicates the individual interlock polarity states for the corresponding digital input. For normal operation (**LP**=0), the corresponding input is considered to be normal when **present_value**=0. When the corresponding bit is configured for reverse operation (**LP**=1), then the input will be considered normal when **present_value**=1. **LP** is a bitmap with bit 0 corresponding to DI1, bit 1=DI2, etc. up to bit 7=DI8.

The **(IP) Digital Input Polarities** property displays the input polarities for all of the digital inputs. Each bit of **IP** specifies how the controller internally represents the corresponding digital input. When an input is set to normal polarity (**polarity**=0), low voltage is displayed as **present_value**=0 and high voltage is displayed as **present_value**=1. If, however, the polarity is set to reverse polarity (**polarity**=1), then low voltage will be displayed as **present_value**=1 and high voltage as **present_value**=0. **IP** is a bitmap with bit 0 corresponding to DI1, bit 1=DI2, etc. up to bit 7=DI8.

Property **object_name** stores the name of the object. This is a user definable string that can be used to help identify the object.

Digital Input Summary Properties

property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
CV	**Current Digital Input Value** indicates the current state of the associated digital input. **CV** is a bitmap with bit 0 corresponding to DI1, bit 1=DI2, etc. up to bit 7=DI8.
IP	**Input Polarities** specifies how the controller represents the input internally. 0=Normal polarity. Low Voltage is displayed as **present_value**=0 and High Voltage is displayed as **present_value**=1. 1=Reverse polarity. Low Voltage is displayed as **present_value**=1 and High Voltage is displayed as **present_value**=0. **IP** is a bitmap with bit 0 corresponding to DI1, bit 1=DI2, etc. up to bit 7=DI8..
LP	**Interlock Polarities** indicates the individual interlock polarity states for each Digital Input. 0=Normal 1=Reverse **LP** is a bitmap with bit 0 corresponding to DI1, bit 1=DI2, etc. up to bit 7=DI8.
OI	**Overridden Digital Inputs Bitmap** indicates those inputs whose **out_of_service** property is true **OI** is a bitmap with bit 0 corresponding to DI1, bit 1=DI2, etc. up to bit 7=DI8.
RE	**Inputs with Unreliable Values** indicates whether the reading from the corresponding digital input is considered reliable. 0=Reliable 1=Unreliable **RE** is a bitmap with bit 0 corresponding to DI1, bit 1=DI2, etc. up to bit 7=DI8.

3.13 Analog Output Summary

The Analog Output Summary object is a convenient way to monitor the states and many of the options for the analog outputs on the *NB*-GPC3. From this object, you can check the current values for each output. This object also allows you to set whether the output will be controlled automatically or manually as well as determine the output's behavior in the case of a communications failure. The Analog Output summary object has the following properties: **object_identifier, object_name, object_type, AM, CF, RE, V1, V2, V3, V4, V5, V6, V7, V8, V9, VA, VB,** and **VC.**

Properties **V1** through **VC** display the current values for analog outputs 1 through 12 respectively.

NOTE
V1 through **VC** are read-only properties. You cannot set the value of the outputs by writing to these properties.

The **(AM) Outputs in Automatic Mode** property indicates which outputs are configured to operate in automatic mode. **AM** is a bitmap with bit 0 corresponding to AO1, bit 1=AO2, etc. up to bit 11=AOC. If the corresponding bit in **AM** is set equal to 1, then the corresponding output is being controller automatically by the *NB*-GPC3. If a bit in **AM** is set equal to 0, then the corresponding output is being controlled manually, either by a host controller or an SPL program.

The **(CF) Communications Failure Enable?** property enables the communications failure feature on an individual basis for each analog output object of the *NB*-GPC3. If the analog output object is in manual mode (**AM**=0), communications with the host is lost (bit 6 of Device Object:**FA**=1), and **CF** is enabled by setting **CF**=1, then the current output value (**present_value**) reverts to the programmed failure position (**FP**) for the corresponding output.

The **(RE) Channel Reliability Bitmap** property indicates the value of the **reliability** property for all of the analog outputs.

Property **object_name** stores the name of the object. This is a user definable string that can be used to help identify the object.

Analog Output Sumary Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
AM	**Outputs in Automatic Mode** specifies how the analog output is controlled. 0=Manual 1=Automatic **AM** is a bitmap with bit 0 corresponding to AO 1, bit 1=AO 2, etc. up to bit 11=AO12.
CF	**Outputs Enabled for Communication Failure indicates** which outputs will be enabled in the event that a communication failure is detected. **CF** is a bitmap with bit 0 corresponding to AO 1, bit 1=AO 2, etc. up to bit 11=AO12.
RE	**Outputs with Unreliable States** indicates whether the reading from the corresponding analog output is considered reliable. 0=Reliable 1=Unreliable **RE** is a bitmap with bit 0 corresponding to AO 1, bit 1=AO 2, etc. up to bit 11=AO12.
V1	**Current Value for Output 1** indicates the **present_value** of output 1.
V2	**Current Value for Output 2** indicates the **present_value** of output 2.
V3	**Current Value for Output 3** indicates the**present_value**of output 3.
V4	**Current Value for Output 4** indicates the **present_value** of output 4.
V5	**Current Value for Output 5** indicates the **present_value** of output 5.
V6	**Current Value for Output 6** indicates the **present_value** of output 6.
V7	**Current Value for Output 7** indicates the **present_value** of output 7.
V8	**Current Value for Output 8** indicates the **present_value** of output 8.
V9	**Current Value for Output 9** indicates the **present_value** of output 9.
VA	**Current Value for Output 10** indicates the **present_value** of output 10.

Property	Description
VB	**Current Value for Output 11** indicates the **present_value** of output 11.
VC	**Current Value for Output 12** indicates the **present_value** of output 12.

3.14 OCCUPANCY DETECTOR

The Occupancy Detector object allows you to define the circumstances under which the *NB*-GPC3 will switch to an extended occupied mode during unoccupied periods when an occupancy detector is used with the controller. The Occupancy Detector object has the following properties: **object_identifier, object_name, object_type, IA, IC, MD, MR, MS,** and **MT.**

The **(MS) Occupancy Detector Input Status** property is a read-only property which shows the status of the occupancy detector input. To enable occupancy detection, **MT** must be > 0 and the occupancy detector input specified in **IC** and **IA** must be configured as a digital input. If either of these two conditions are not met, **MS** will display 0. When occupancy in the zone is detected during unoccupied periods (**MS** = 1), the occupancy input extends occupancy time by the amount specified in **MT**.

The **(IC) Occupancy Detector Input Channel** and **(IA) Occupancy Detector Input Attribute** properties specify the object and property to be used for Occupancy Detection. A list of the most common choices for **IA** will be displayed. If you wish to use a property not listed, for example a user defined property from an SPL program, you can enter the two-letter code into **IA**. Any two letter property name may be entered in this way. To remove the selected channel or attribute, select the “Disable” option at the bottom of the list of choices in the drop-down menu.

The **(MD) Extended Occupancy Delay** property sets the amount of time, in seconds, during which the occupancy detector must remain on before the occupancy detector will override the zone. This prevents false triggers that might occur as others pass quickly through the zone.

The **(MT) Extended Occupancy Duration** property defines, in minutes, the length of time to override the zone whenever occupancy is detected.

The **(MR) Extended Occupancy Remaining** property displays the time remaining before the controller returns to unoccupied mode.

Occupancy Detector Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
IA	**Input property for Occupancy Duration** specifies the property associated with the object specified in **IC** to be used for occupancy detection.
IC	**Input object for Occupancy Duration** specifies the object of the input to be used for occupancy detection.
MD	**Extended Occupancy Delay** specifies the time, in minutes, that the occupancy detector status, **MS**, must be positive before the controller will switch to occupied mode.
MR	**Extended Occupancy Remaining** indicates the time, in minutes, before the controller reverts to unoccupied mode.
MS	**Occupancy Status** indicates whether the object and property specified in **IC** and **IA** indicate that the monitored zone is occupied.
MT	**Extended Occupancy Duration** specifies the time, in minutes, that the controller will stay in occupied mode once occupancy is detected.

3.15 DIGITAL OUTPUT SUMMARY

The Digital Output Summary object is a convenient way to monitor the states and many of the options for the digital outputs on the *NB*-GPC3. From this object, you can check the current values for each output as well as the actual output state. This object also allows you to determine the output's behavior in the case of a communications failure. You can specify which outputs are to be enabled in the case of fire as well as set the output's fire state position. The Digital Output Summary Object has the following properties: **object_identifier, object_name, object_type, CF, CV, FE, FI, FS, IP, IS, OU, RE,** and **SB.**

The **(CV) Digital Outputs Which are On** property displays the current state (on or off) for each of the digital outputs. **CV** is a bitmap with bit 0 corresponding to DO1, bit 1=DO2, etc. up to bit 11=DOC. If a digital output is currently energized, then the corresponding bit in **CV** will be set equal to 1.

NOTE
CV is read-only. The state of the outputs cannot be changes by writing values to **CV**.

The **(OU) Actual Output State** property displays the actual state of each of the digital outputs. It is useful to have both **OU** and **CV** because it is possible for the actual output state to differ from the current value because of various staging delays.

The **(CF) Outputs Enabled for Communication Fail** property enables the communications failure feature on an individual basis for each digital output object of the *NB*-GPC3. If communications with the host are lost (bit 6 of Device Object:**FA**=1), and **CF** is enabled for that output by setting the corresponding bit in **CF**=1, then the current output value (**present_value**) reverts to a programmed failure position (**FP**) for the corresponding output. **CF** is a bitmap with bit 0 corresponding to DO1, bit 1=DO2, etc. up to bit 11=DOC. Enabling a bit in **CF** will cause the corresponding output to assume the state defined in **FS** in the case of a communications failure.

The **(FS) Communication FailureState Bitmap** property defines the communication failure states for the digital outputs. **FS** functions in conjunction with **CF** and specifies the desired failure state for the corresponding input. **FS** is a bitmap with bit 0 corresponding to DO1, bit 1=DO2, etc. up to bit 11=DOC. The value of each bit in **FS** represents the desired failure state of the corresponding output (0=off, 1=on).

The **(FE) Outputs Enabled for Fire** property specifies which outputs will be enabled in the event of a fire. **FE** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12. If the *NB*-GPC3 enters fire mode (Device Object:**FA**=1), you may wish to turn certain outputs either on or off. Setting a bit in **FE** to 1 will cause the corresponding output to assume the state defines in **FI** when a a fire event is detected. If **FE**=0 for a given bit, then the output will not take any action when a fire is detected.

The **(FI) Fire State Bitmap** defines the states the digital outputs should assume in case a fire is detected. **FI** functions in conjunction with **FE** and specifies the desired failure state for the corresponding input. **FI** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12. The value of each bit in **FI** represents the desired fire state of the corresponding output (0=off, 1=on).

NOTE
The output does not have to be assigned to a PID or Thermostatic Control loop to be set or reset by fire or interlock failures.

The **(IP) IO Polarity** property is a bitmap which indicates the polarities for the digital outputs. Each bit of **IP** corresponds to a digital output and its value reflects the value of the **polarity** property for that output.

The **(RE) Data Reliability** property indicates object reliability for the digital outputs. The reliability is only used for outputs on the STATbus. When the a control sequence within the *NB*-GPC3 changes the value of an output, the *NB*-GPC3 sends a command out over the STATbus network telling the output device to change its value. If the message to change its value is successfully received, then the STATbus device will send a message back to the *NB*-GPC3 acknowledging the request. If the *NB*-GPC3 sends such a message, but does not receive an acknowledgement, then the controller will flag the output as unreliable. The object reliability, **RE**, is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12. If an output is unreliable, the *NB*-GPC3 will set the corresponding bit in **RE** to 1. A value of zero indicates that the output is reliable.

The **(SB) Outputs Enabled for Staging** property defines which outputs are available for staging. **SB** is a bitmap with bit #1 corresponding to Digital Output 1, bit #2 corresponding to Digital Output 2, etc. If a bit in **SB** is set to 1, then the associated output is enabled for staging.

The **(IS) Inter-stage ON Delay** specifies the minimum time, in seconds (0-255), that must elapse after a staged output is energized before subsequent stages may be energized.

The **object_name** property stores the name of the object. This is a user definable string that can be used to help identify the object.

Digital Output Summary Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
CF	**Outputs Enabled for Communication Fail** specifies which outputs will be enabled in the event that a communication failure is detected. **CF** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
CV	**Digital Outputs which are On** indicates which outputs are currently on. **CV** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 12=Digital Output 12.
FE	**Outputs Enabled for Fire** specifies which outputs will be enabled in the event that a fire event is detected. **FE** is a bitmap with bit 1 corresponding to Digital Output 1, bit 2=Digital Output 2, etc. up to bit 11=Digital Output 12.
FI	**Fire Positions** indicates which digital outputs are in fire mode. **FI** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
FS	**Communication Failure Positions** indicates which outputs are currently experiencing communications failures. **FS** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
IP	**IO Polarities** indicates the value of the **polarity** property for each of the digital outputs. **IP** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
IS	**Inter-Stage Delay** specifies a minimum delay, in seconds (0-255) that must elapse after a staged output is energized before subsequent stages may be energized.
OU	**Actual Output States** indicates the actual output states of the digital outputs. **OU** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
RE	**Outputs with Unreliable Values** indicates whether the reading from the corresponding digital output is considered reliable. 0=Reliable 1=Unreliable **RE** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
SB	**Outputs Enabled for Staging** selects two or more digital outputs to be grouped together for staging. **SB** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.

3.16 FLOATING POINT CONTROL 1-2

The Floating Point Control loops are used to control such devices as fans, pumps, and blowers. Each loop performs either PI or PID control while providing calibration and alarming functions. Floating Point Control objects 1 and 2 contain the following properties: **object_identifier, object_name, object_type, BO, CE, CF, CP, CR, CS, DB, DL, DO, DP, FI, FP, IA, IC, IL, MR, PB, PE, RA, RC, RI, RL, RP, RS, SG, SM, SP, SU, TS,** and **TT.**

In floating point control applications, each floating point control object controls the position of a motor actuator using two digital outputs (an increase output and a decrease output).

The **(CE) Enable Control Loop?** property is used to determine how the desired position of the control loop is determined. **CE**=0 means that **DP** is set manually. **CE**=1 means that **DP** is calculated automatically by the PI algorithm.

If the **(DP) Desired Position** of the motor is greater than the current position, the controller will drive the motor open by turning on the “increase” output for a calculated period of time. If the desired position is less than the current position, the controller will drive the motor closed by turning on the “decrease” output for a calculated period of time.

The desired position of the floating point control object can be set manually or calculated automatically by the PI algorithm. The automatic floating point control algorithm operates as follows. When the value of the selected measured variable is within the control loop’s deadband, no control action is taken by the PI loop. When the value of the measured variable is outside the deadband, but within a programmable proportional band, the output is modulated using PI control according to the setpoint of the control loop. When the value of the measured variable is outside the deadband and beyond (either above or below) the proportional band, the output is set to either 0% or 100%, as appropriate.

NOTE
All values written by the Floating Point Control loops are written at a Priority Array Level of 11.

The **(TT) Travel Time** property is used to specify the total time in seconds (0 to 65,535 seconds) that it takes the motor actuator to go full stroke (from fully open to fully closed). **TT** is used to determine the current position (**CP**) of the motor. **TT** defaults to a value of 0 seconds.

NOTE
The travel time of a motor depends on the load that is applied to the motor. For accuracy, it is suggested that you determine **TT** when the motor is *loaded*.

NOTE
For spring loaded motors, the full stroke travel time from 0% to 100% may be different than the 100% to 0% travel time. You may choose to use the higher of the two travel times for **TT**. In this case, it is recommended that you perform regular calibrations on the motor.

The **(PE) Floating Point Control Pair Enable** property is used to enable the pair of digital outputs for floating point control. If **PE**=0, the corresponding floating point control pair is disabled. If **PE**=1, then the corresponding floating point control pair is enabled.

The **(DO) Digital Output Pair Selection** property is used to select the pair of digital outputs to be controlled by the floating point control loop. Output pairs are often used with floating point actuators which respond to increase/decrease signals. The options for **DO** are given in Table 3-17.

Table 3-17 : Digital Output Pair Selection Options

DO	Output Pair Selected
0	none
1	DO1+DO2
2	DO3+DO4
3	DO5+DO6
4	DO7+DO8
5	DO9+DO10
6	DO11+DO12

The first output listed is the “increase” signal and the second output listed is the “decrease” signal.

The **(BO) Digital Outputs which are On** property is a bitmap which indicates the digital outputs that are currently energized. **BO** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.

3.16.1 PI CONTROL

PI stands for *proportional* + *integral* and represents a method of control which controls equipment according to a setpoint in proportion to the value of a measured variable. It also accounts for the amount of error (difference between the measured variable and the setpoint). PI control can be used in floating point control loops by enabling the control enable property (**CE**=1). PI control is disabled by setting **CE**=0. PI control is a subset of PID control (PID minus the derivative control). Refer to the descriptions of the PID Control objects for an in depth study of PI (and PID) control.

The **(SP) Loop Setpoint** property specifies the control setpoint for the floating point control loop. In PI control, the setpoint is defined in **SP**, and the measured input variable is the analog sensor referenced by the universal input object specified in the **IC** and **IA** properties. The setpoint is expressed in the same kind of measurement units (Engineering Units) that the measured variable uses (e.g., degrees, cfm, inches of WC, etc.). For example, if you are using the floating point control object to adjust cooling dampers to control a temperature value that the input sensor measures in degrees, the setpoint for the floating point control object must also be expressed in degrees. The data type of **SP** is the same as the data type of the selected measured variable.

The **(CS) Calculated Control Setpoint** property is a read only value that reflects any setup or setback that may be applied (during unoccupied or night setback modes), as well as any reset or setpoint adjustments made through a stat. **CS** is the actual setpoint that is used by the control loop. Refer to properties **SU** and **SM** for more information about schedule controlled setup/setback. **CS** is expressed in the same kind of measurement units (Engineering Units) that the measured variable uses (e.g., degrees, cfm, inches of WC, etc.). The data type of **CS** is the same as the data type of the selected measured variable.

The **(DP) Desired Position** property specifies the desired position of the floating setpoint motor output in percent (e.g., 50%). **DP** is set automatically by the floating point control loop when it is enabled (**CE**=1). **DP** can be set manually if **CE**=0. **DP** specifies the desired motor position to which the floating point control loop should control the outputs in order to bring the measured input value (specified by **IC** and **IA**) closer to the calculated setpoint value (**CS**).

The **(CP) Current Position** property is the current (actual) position of the motor in percent (e.g., 50%). This value is calculated based on **TT**, the actuator itself has no feedback. To ensure that the actual motor position is the same as **CP**, the motor must occasionally be recalibrated. For information about calibration, refer to properties **TT**, **CR** and **RI**.

The **(IC) Input Channel** and **(IA) Input Attribute** properties select the object and property that is used as the measured variable in the control loop. A list of the most common choices for **IA** will be displayed. If you wish to use a property not listed, for example a user defined property from an SPL program, you can enter the two-letter code into **IA**. Any two letter property name may be entered in this way.

NOTE
To clear **IC** and remove the association with its target object, you must write a value of 0000 (four zeroes) to **IC**.

The **(PB) Proportional Control Band** defines an input variable range over which the output signal is proportional (i.e., changes in the measured variable result in proportional changes in the desired output). **PB** is expressed in the same kind of measurement units (Engineering Units) that the measured variable uses (e.g., degrees, cfm, inches of WC, etc.). The data type of **PB** is the same as the data type of the selected measured variable.

To determine **PB**, you must first decide how close the *NB*-GPC3 must control the output to the setpoint. For instance, if the setpoint is 72ºF., an acceptable control range might be within two degrees of the setpoint. This control range can be expressed as a band centered on the setpoint: from 70º to 74º, or four degrees, the *proportional band* (**PB**). Refer to Figure 3-3 and Figure 3-4.

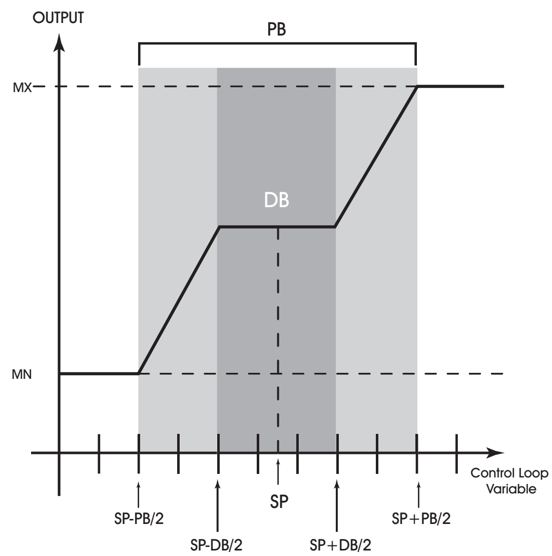

*Figure 3-3: Proportional Band For Normal Acting Control (**SG**=0)*

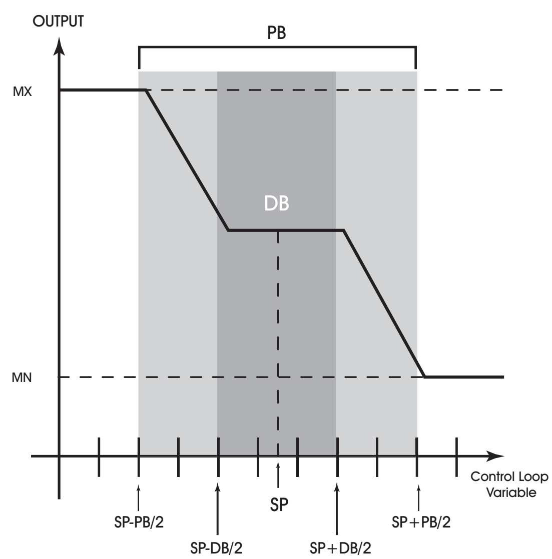

*Figure 3-4: Proportional Band For Reverse Acting Control (**SG**=1)*

The **(DB) Desired Control Deadband** property is used to specify an input variable range within the proportional band **PB**. The size of **DB** should be based on the type of sensor input that is selected for the input object specified in **IC** and **IA**. When the value of the measured variable is within this deadband, there is no change in the proportional output response.

DB is centered on the setpoint **SP** to create the actual control deadband. When the value of the control variable (specified by **IC** and **IA**) is within ±**DB**/2 of the setpoint **SP**, the *NB*-GPC3 assumes that it has reached the setpoint.

By specifying a **DB** that is greater than or equal to the resolution of the sensor specified in **IC** and **IA**, you eliminate the possibility of cycling around the setpoint. The value of **DB** should never exceed the proportional band **PB**. If **DB** is greater than **PB**, then the control loop will not have proportional control.

DB is expressed in the same kind of measurement units (Engineering Units) that the measured variable uses (e.g., degrees, cfm, inches of WC, etc.). The data type of **DB** is the same as the data type of the selected measured variable. **DB** defaults to a value of 0.

CAUTION

*You should never change **DB** to a value greater than half of the proportional band **PB**. Doing so will eliminate the effects of PI control, resulting in on/off control.*

The **(SG) Control Action** property refers to the control sign of the floating point control loop. The control sign is used to determine whether the control loop is *normal acting* or *reverse acting.* A normal acting control loop causes an increase in output position when the value of the input variable *increases* (see Figure 3-3). A reverse acting control loop causes an increase in output position when the value of the input variable *decreases* (see Figure 3-4).

Property **SG** is also used during schedule control to determine whether **SU** is added to **SP** (**SG**=0) or subtracted from **SP** (**SG**=1) during unoccupied periods. For more information, refer to properties **SU** and **SM**.

With just these properties configured, the *NB*-GPC3 will provide simple closed loop feedback *proportional* control. This means that the actual measured performance of the control (from the measured variable input) is fed back to the controller and compared with the effective setpoint for the loop. Any difference between the actual value of the measured variable (**MV**) and effective setpoint values is called *error*.

An analogy is helpful in explaining the effects of error. Figure 3-5 shows a simple lever and fulcrum. A change in the lever position on the error side produces a proportional change in the lever on the output side. Depending on the position of the fulcrum, a change on the error side will have a greater or lesser effect on the output side. The fulcrum position changes the ratio of error to output.

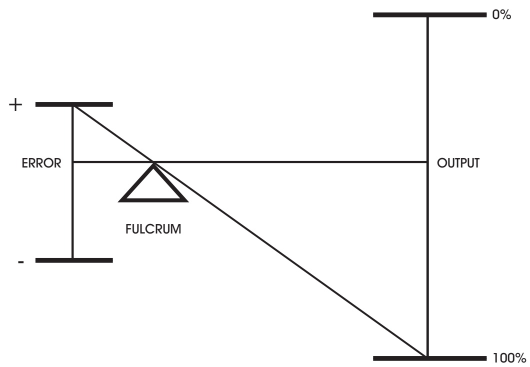

Figure 3-5: Ratio of Input to Output

One problem with proportional-only control is that loop performance changes when the condition being measured by the input sensor changes (e.g., the measured temperature changes when a door is opened and the room or space is flooded with cold air). As the loop environment changes, the proportional only control loop begins to cycle around an offset from the setpoint. Figure 3-6 illustrates the performance of a typical loop under proportional only control.

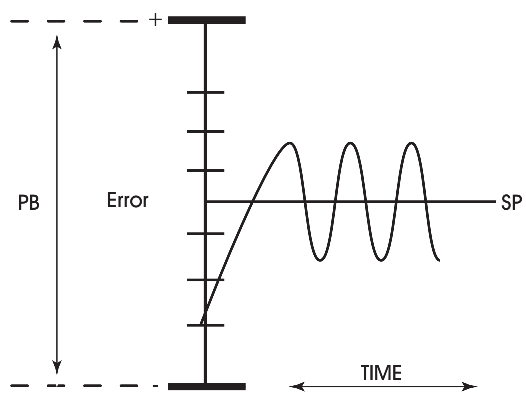

Figure 3-6: Proportional Only Control

Proportional-only control produces cycling, and its performance changes when the measured environment changes. The way to eliminate cycling and to compensate for load changes is to use *integral* action, the *I* part of PI control.

Rather than responding exclusively to the loop error from moment to moment, as with proportional action, integral action is based on a summation of the error which has occurred over some period of time. This error sum is used to reset, or modify, the response of the control loop (output) based on a running average of the error. The amount of time over which the error averaging is accumulated is called the *reset period*.

The **(RP) Reset Period** property specifies the reset period (in seconds) over which the error averaging is accumulated. The longer **RP** is, the less effect it has on the control response. Figure 3-7 shows the response of a typical control loop when integral action is used in addition to proportional action (PI control). A value of zero disables the reset period.

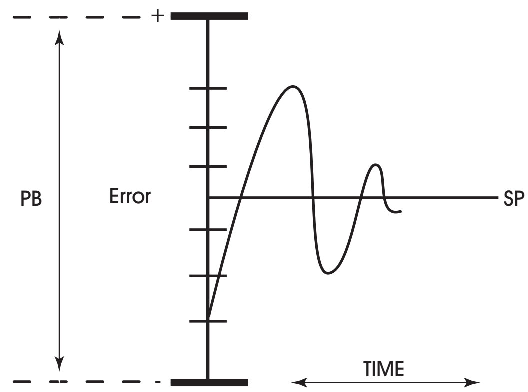

Figure 3-7: Proportional + Integral (PI) Control

The proportional control action of the loop has a major effect on integral action. Increasing **PB** results in a smaller integral effect for a given value of **RP**. In general, decreasing the proportional band, **PB**, will increase the amount of overshoot. On the other hand, the larger **PB** is, the slower the loop will respond.

3.16.2 RESET FEATURE

The **(MR) Maximum Reset**, **(RC) Reset Variable**, **(RA) Reset Attribute**, **(RS) Setpoint at which Reset Action Begins** and **(RL) Reset Limit** properties control the reset feature (not to be confused with the reset period) of the associated floating point control object.

The **(MR) Maximum Reset** property allows you to specify a maximum amount to reset the loop setpoint **SP**. Property **CS** takes into effect the use of maximum reset specified in **MR**.

NOTE
The data type of the **MR** property is the same as the data type of the referenced input variable specified by **IC** and **IA**.

The **(RC) Reset Variable** and **(RA) Reset Attribute** properties allows you to specify the reset variable that is to be used by the control loop. This property gives you the ability to control a loop using one input while resetting the loop using a different input. Throughout this section, V_R refers to the actual value of the reset variable selected by these properties.

NOTE
To clear **RC** and remove the association with its target object, you must write a value of 0000 (four zeroes) to **RC**.

The **(RS) Setpoint at which Reset Action Begins** property allows you to specify the reset setpoint value at which reset action will occur. Just as **SP** is the proportional control setpoint for the measured variable of the floating point control loop, **RS** is the reset control setpoint for V_R. The data type of **RS** is the same as the data type of the reset variable specified by **RC** and **RA**.

The **(RL) Reset Limit** property specifies the reset limit of the floating point control loop. When the reset variable specified by **RC** and **RA** reaches a value of **RL**, the control loop setpoint will be reset by the maximum amount **MR**.

NOTE
The data type returned for properties **RS** and **RL** are determined by the data type of the referenced reset variable specified by **RC** and **RA**.

NOTE
It may not be possible to use a negative value for **MR** if the data type of the control loop's input object, specified in **IC** and **IA** uses an unsigned data type.

The relationship between **RL** and **RS**, as well as the sign (+ or -) of **MR**, determines how changes in the reset variable specified in **RC** and **RA** affect the setpoint of the loop **SP**. Refer to figure Figure 3-8.

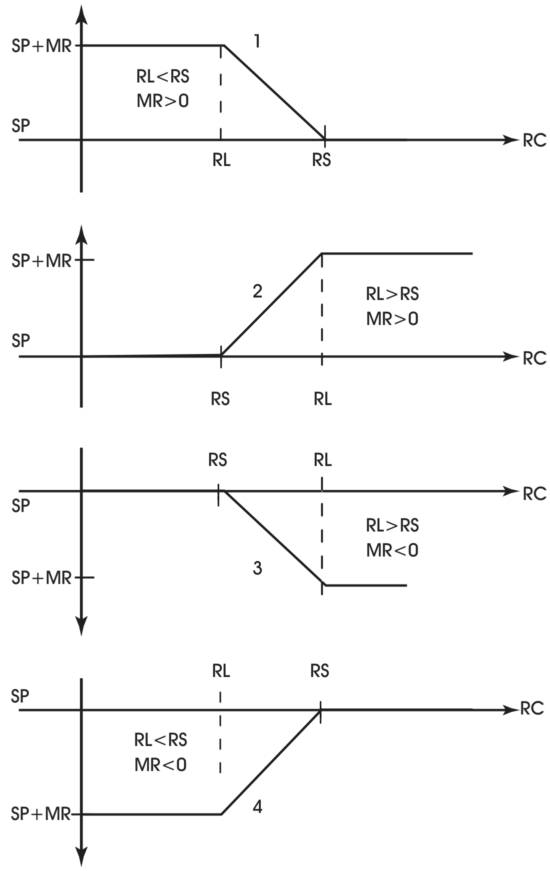

Figure 3-8: Four Forms Of Reset Action

3.16.3 Schedule Control

The floating point control objects of the *NB*-GPC3 can be controlled using Schedule objects 1 through 8, as well as the SmartStat occupancy override and a host override feature. When enabled, schedule control increases or decreases the control loop setpoint by a programmable amount when the selected schedule objects enter *unoccupied mode*.

Unoccupied mode is a programmable time period during selected days of the week when control constraints are typically less stringent. For example, in a typical office building, the occupied period might be from 8:00 am to 6:00 pm, Monday through Friday. The other times during the week (Monday through Friday from 12:00 midnight to 7:59 am, from 6:01 pm to 11:59 pm, and all day Saturday and Sunday) would be the unoccupied periods. The control constraints can be less stringent at these times because people are not expected to be in the building.

The **(SU) Unoccupied Setup/Setback property** specifies a setpoint offset for unoccupied mode. During unoccupied periods (as specified by schedule objects that are selected through the **SM** property), the **SP** is either setup or setback by an amount specified in **SU**. The sign of the control loop (**SG**) determines whether **SU** is a setup (**SG**=0) or setback (**SG**=1) amount. **CS** (the effective setpoint) incorporates any setup/setback that may exist, as well as any reset or setpoint adjustment from the SmartStat Module. The data type of the value specified in **SU** is the same as the data type of the referenced measured variable specified by **IC** and **IA**.

The **(SM) Schedules to Follow** property is used to select which schedules, if any, are being used to control the setpoint adjustment of the floating point control loop. In addition to using schedules, you can select SmartStat override control and/ or host override control. These option are also selected using the **SM** bitmap. Each bit in **SM** corresponds to a desired control function. These bits are summarized in Table 3-18.

Table 3-18 : Schedules to Follow

SM bit	Schedule
0	Schedule 1
1	Schedule 2
2	Schedule 3
3	Schedule 4
4	Schedule 5
5	Schedule 6
6	Schedule 7
7	Schedule 8
8	SMARTStat 1
9	SMARTStat 2
10	SMARTStat 3
11	SMARTStat 4
12	SMARTStat 5
13	SMARTStat 6
14	SMARTStat 7
15	SMARTStat 8
16	SMARTStat 9
17	SMARTStat 10
18	SMARTStat 11
19	SMARTStat 12
20	Host Schedule
21	Schedule Summary
22	Occupancy

You select one or more of the control options in Table 3-18 by setting the appropriate bit(s) to 1. If you do not want to use schedule control, set all of the bits in **SM** to 0.

If one or more of the schedule bits (bits 0-7) are set in **SM**, then the unoccupied mode setpoint adjustment is based on Schedule objects 1-8. If all of the selected schedules are in unoccupied mode (Schedule X:**present_value**=0), then the corresponding **SU** value is incorporated into the effective setpoint of the control loop (**CS**).

If a SmartStat override bit (bits 8-19) is set in **SM**, then the SmartStat can be used to extend occupied mode.

If the host override bit (bit 20) is set in **SM**, then you can control the schedule state of the floating point control loop from the Schedule Summary object. In this case, the host schedule enable property must be enabled (Schedule Summary Object:**HE**=1).

If the SmartStat Override bit and multiple schedule bits are set in **SM**, the *NB*-GPC3 checks the states of the selected schedules and SmartStat. If any of the selected schedules has its **present_value**=2 or if the SmartStat indicates the zone is occupied, then the control loop uses its occupied mode setpoint. If all of these selected control options indicate that the control loop should be in unoccupied or night setback mode, then the control loop incorporates the appropriate setup/setback value into its effective setpoint.

If the Host Schedule option is selected (bit 9 of **SM**=1) and Host Override is enabled (Schedule Summary Object:**HE**=1), then the schedule state of the control loop will be controlled by the Schedule Summary Object:**HO**, regardless of the settings of other selected schedules or the state of the SmartStat.

3.16.4 INTERLOCK, COMMUNICATIONS AND FIRE FAILURE POSITIONING

The **(IL) Input Interlock Bitmap** property is used in conjunction with the **(FP) Failure Position** property for floating point control interlocking. The first twenty four bits of **IL** correspond to the Universal Input 1 through Universal Input 24 objects of the *NB*-GPC3. The next four bits correspond to the Digital Input 1 through Digital Input 4 objects.

NOTE
To be used for interlocking, the universal input must be configured as a digital input.

One or more of these inputs can be selected as interlocks by setting the corresponding bit in **IL** to 1. In the event that any of the digital inputs specified in **IL** has a value of 1, then the desired position (**DP**) assumes its interlock failure position (**FP**). All of the inputs specified in **IL** must have a current value of 0 before normal control is restored to **present_value**.

Property **CF** is the communications failure enable property. In the event that the floating point control loop is being controlled by a host, you can enable communications failure checking. If **CF**=0 (disabled), then host communications failure checking is disabled. If **CF**=1, then host communications failure checking is enabled.

If **CF**=1 and a host communications failure occurs (after Device Object:**CF** has expired), the floating point control loop reverts to the position specified in the failure position property (**FP**).

The **(FP) Failure Position** specifies the desired position (**DP**) should an interlock or communications failure occur. An interlock failure occurs when the value of any of the bits set in **IL** equals 1. For more information, refer to **IL**. A communications failure occurs when a timeout error is detected.

The **(FI) Fire Position** property specifies the fire position for the control loop. In the event of a fire condition (Device Object:**FA** bit 0=1), the desired position (**DP**) reverts to the position specified in the fire position property (**FI**).

NOTE
If a communications failure or interlock failure occurs, then the desired position of the floating point control loop reverts to **FP**. If a fire condition then occurs while the communications or interlock failure is still active, **FI** takes priority over **FP**.

3.16.5 CALIBRATION

The actuator can be manually calibrated by enabling floating point control pair enable (**PE**=1), disabling PI control (**CE**=0) and setting **DP** to 0% or 100%. When the actuator is at the programmed position (after approximately **TT** seconds), set **CP** to 0% or 100% accordingly. Finally, be sure to return PI control (**CE**=1) if you want **DP** to be set automatically.

Floating point control loops can be calibrated automatically by the *NB*-GPC3 at programmable intervals. This is done using the recalibrate interval. The **(RI) Recalibration Interval** property specifies how often (if at all) the associated floating point control object is to be recalibrated.

RI is given in hours (0-255 hours). If **RI**=0, then recalibration of floating point control loops does not occur. If **RI**>0, recalibration of the associated floating point control loops occurs every **RI** hours.

The *NB*-GPC3 recalibrates the floating point control loops by driving the desired position (**DP**) to the fully closed position (0%) for the amount of time specified in the travel time property (**TT**). The *NB*-GPC3 then sets the current position to 0%, after which the recalibration is complete and the controller returns the desired position to its original value.

For floating point control objects, you can enable an automatic creep feature using property **CR**, the creep enable property. This feature is used to automatically calibrate the output when its desired position is either 0% or 100%. The automatic creep feature is performed in one of two ways: (1) the appropriate output is left on when the output signal is at 0% or 100%, or (2) the output is *creeped* (pulsed) at a rate of 1% per minute (the current position is set to 1% or 99%) when the output signal is at 0% or 100%. The value of the creep enable property (**CR**) selects the desired method.

These two methods of output correction (continuous on and automatic creep) are illustrated in Figure 3-9. This example shows a floating point control loop with a desired position of 100%.

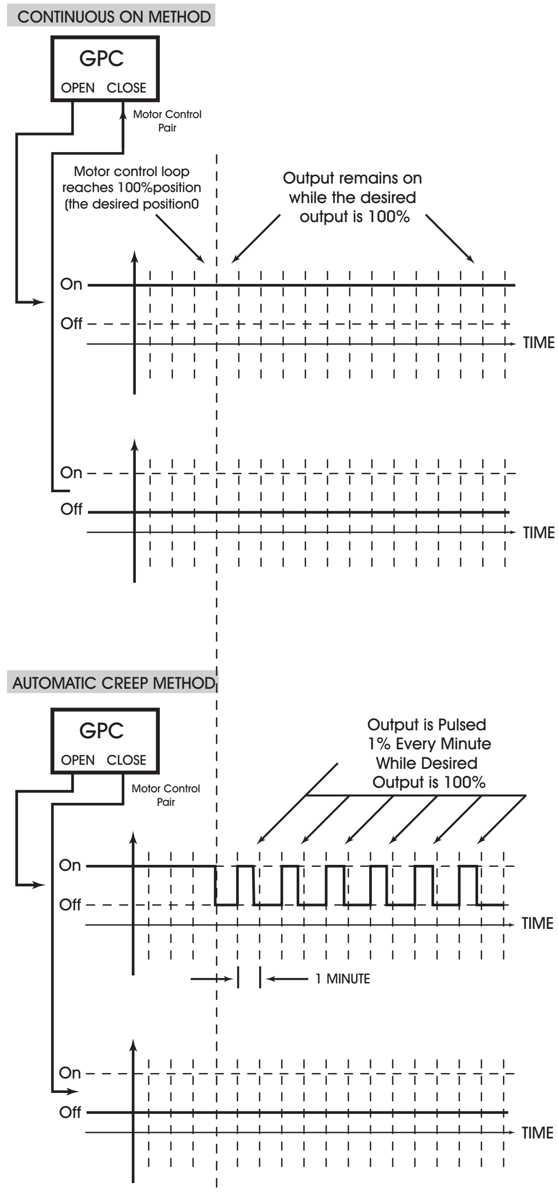

Figure 3-9: Output Correction, Continuous and Automatic Creep

Output correction is the same for desired positions of 0%, only the “close” output (rather than the “open” output) of the pair is used.

Floating Point Control Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
BO	Digital Outputs which are On indicates the digital outputs that are currently energized. **BO** is a bitmap with bit 0 corresponding to BO1, bit 1=BO2, etc. up to bit 11=BO12.
CE	**Enable Control Loop?** enables/disables floating point control for the associated control loop. 0=No 1=Yes
CF	**Communication Failure Enable?** specifies what action to take in the event that a communication failure is detected. 0=No 1=Yes
CP	**Current Position** indicates the current position of the motor.
CR	**Motor Creep Function** specifies how the controller handles automatic calibrations at minimum and maximum positions. 0=Drive motor constantly if **DP**=0% or **DP**=100% 1=Creep motor output by 1% per minute if **DP**=0% or **DP**=100%
CS	**Calculated Control Setpoint** indicates the calculated control setpoint. This value accounts for any reset or setup/setback action on the loop setpoint.
DB	**Desired Control Deadband** specifies the deadband that is used to control cycling around the setpoint. If the current value of the input object is between **SP**-(**DB**/2) and **SP**+(**DB**/2), the measured variable is considered to be at its setpoint
DL	**Demand Load** indicates the heating/cooling demand in terms of the temperature separation from setpoints

Property	Description
DO	**Digital Output Pair Selection** specifies which Digital Output pair is to be used for floating point control. 0=none 1=Digital Outputs 1 & 2 2=Digital Outputs 3 & 4 3=Digital Outputs 5 & 6 4=Digital Outputs 7 & 8 5=Digital Outputs 9 & 10 6=Digital Outputs 11 & 12
DP	**Desired Position** specifies the desired output position (0-100%) of the associated motor.
FI	**Fire Position** specifies the failure position (0-100%) to use when a fire event is detected.
FP	**Interlock/Comm Failure Position** specifies the failure position (0-100%) to use when an input interlock failure occurs. **FP** is used when the current value of any of the inputs specified by **IL** has a value of 1.
IA	**Input property** specifies the object to be used for floating point control.
IC	**Input object** specifies the property associated with the object specified in **IC** to be used for floating point control.
IL	**Input for Interlock** specifies which inputs are to be used for interlocking of the associated floating point control loop. bit 0 = UI 1 bit 1 = UI 2 bit 2 = UI 3 bit 3 = UI 4 bit 4 = UI 5 bit 5 = UI 6 bit 6 = UI 7 bit 7 = UI 8 bit 8 = UI 9 bit 9 = UI 10 bit 10 = UI 11 bit 11 = UI 12 bit 12 = UI 13 bit 13 = UI 14 bit 14 = UI 15 bit 15 = UI 16 bit 16 = UI 17 bit 17 = UI 18 bit 18 = UI 19 bit 19 = UI 20 bit 20 = UI 21 bit 21 = UI 22 bit 22 = UI 23 bit 23 = UI 24 bit 24 = DI 1 bit 25 = DI 2 bit 26 = DI 3 bit 27 = DI 4 bit 28 = DI 5 bit 29 = DI 6 bit 30 = DI 7 bit 31 = DI 8

Property	Description
MR	**Maximum Amount to Reset Setpoint** specifies the maximum amount o reset **SP**.
PB	**Proportional Control Band** specifies a range, centered around the loop setpoint **SP**, where the output signal is proportional.
PE	**Floating Point Control Pair Enable** specifies whether the motor is to be controlled using pulsed pairs. 0=No 1=Yes
RA	**Reset property** specifies the property associated with the object specified in **RV** to determine reset.
RC	**Reset Variable** specifies the object to be used to determine reset.
RI	**Motor Recalibrate Interval** specifies a time interval in hours (0-255) the defines how often the associated floating point control loop is recalibrated. 0=Calibration disabled **RI** > 0 =Recalibrate every **RI** hours
RL	**Limit for Maximum Reset** specifies the reset limit of the control loop. When the reset variable specified in **RV** and **RA** reaches a value of **RL**, the control loop setpoint will be reset by the maximum amount **MR.**
RP	**Reset Period** specifies a time, in seconds (0 to 65,535) over which the output of the control loop should be adjusted (reset). 0=Diabled 1 to 65,535=Reset period, in seconds
RS	**Setpoint at which Reset Action Begins** specifies the setpoint of the control loop at which reset action begins.
SG	**Control Action** specifies whether the controller's output should be increased or decreased when the control signal is positive. 0=Normal (increase for positive error) 1=Reverse (decrease for positive error)

Property	Description
SM	**Schedules to Follow** enables scheduled alarm controlling for the thermostatic control loop by selecting one or more of the available schedule control objects. 0=schedule disabled 1=schedule enabled **SM** is a bitmap with bit 0=Schedule 1 bit 1=Schedule 2 bit 2=Schedule 3 bit 3=Schedule 4 bit 4=Schedule 5 bit 5=Schedule 6 bit 6=Schedule 7 bit 7=Schedule 8 bit 8= SMARTStat 1 bit 9= SMARTStat 2 bit 10= SMARTStat 3 bit 11= SMARTStat 4 bit 12= SMARTStat 5 bit 13= SMARTStat 6 bit 14= SMARTStat 7 bit 15= SMARTStat 8 bit 16= SMARTStat 9 bit 17= SMARTStat 10 bit 18= SMARTStat 11 bit 19= SMARTStat 12 bit 20= Host Schedule bit 21= Schedule Summary bit 22= Occupancy
SP	**Loop Setpoint** specifies the desired setpoint for the floating point control loop.
SU	**Unoccupied Setup/Setback** specifies a value (0.0 to 25.5) which is added to (if **SG**=1) or subtracted from (if **SG**=0) the control loop setpoint during scheduled unoccupied periods
TS	**Thermostat Setpoint Adjustment** indicates the offset, read from a Stat3, to be applied to control setpoints.
TT	**Motor Travel Time** specifies the time, in seconds (0-3000), that it takes the motor to move from its fully closed to its fully open positions.

3.17 THERMOSTATIC CONTROL 1-12

When thermostatic control is enabled, the associated digital output will control the measured variable based on a user-defined setpoint. By calculating a control setpoint, which takes into account a possible setup/setback during unoccupied and night setback periods, and comparing it with the measured variable, the controller can determine the amount of output that will be necessary to maintain the desired setpoint. The control loop will enforce a control deadband to prevent hysteresis and can be configured to operate based on a pre-defined schedule. Each Thermostatic Control object has the following properties: **object_identifier, object_name, object_type, CV, CE, CS, DB, IC, IA, SG, SM, SP, SU, OB, DL,** and **TS.**

The **(IC) Input Channel** and **(IA) Input Attribute** properties are used to specify the property which is to be used as the measured value for the thermostatic control loop. When **IC** is selected, a list of the most common choices for **IA** will be displayed. If you wish to use a property not listed, for example a user defined property from an SPL program, you can enter the two-letter code into **IA**. Any two letter property may be entered in this way.

NOTE
To clear **IC** and remove the association with its target object, you must write a value of 0000 (four zeroes) to **IC**.

The **(CE) Enabled Control Loop?** property is used to determine whether or not thermostatic control will be active. When **CE**=1, thermostatic control is enabled. If **CE**=0, then the thermostatic control object will be disabled and no control functions will be performed.

Once enabled, the thermostatic control object will control based on the **(CS) Calculated Control Setpoint** property. This property represents the desired temperature in the area being controlled. The controller will begin with the value in **(SP) Loop Setpoint**. When in unoccupied or night setback mode, GPC will apply a specified amount of setup/setback, defined in **(SU) Unoccupied Setup/Setback**. This value is then stored in the **(CS) Calculated Control Setpoint** property.

NOTE
All values written by the Thermostatic Control loops are written at a Priority Array Level of 11.

The **(TS) Thermostat SP Adjustment** property indicates any setpoint adjustments read in from a SmartSTAT. This value is used in calculating **CS**.

The value of **CS** is compared to the measured value of the input, specified in **IC** and **IA**. The difference between **CS** and the measured variable will be stored in the **(DL) Demand Load** property. If the measured variable does not equal the calculated setpoint and is outside of the specified control deadband, then action will be taken to correct the measured variable.

The **(SG) Control Sign** property specifies the control action for the control loop. When **SG** = 0 (normal or cooling), a positive error causes an increase in output. When **SG** = 1 (reverse or heating), a positive error causes a decrease in output. This point determines the response of the loop output to the kind of error. If the output action is to be increased (toward max) when the error is positive, set **SG** to normal (0). If the output action is to be decreased (toward min) for positive error, set **SG** to reverse (1). (Property **SG** is also used during schedule control to determine whether the setup/setback is added to **SP** [**SG** = 0] or subtracted from **SP** [**SG** = 1] during unoccupied and night setback periods.)

The **(CV) current value** property indicates the current state of the control loop. Control loop conditions are true (typically ON) and false (typically OFF).

The **(SU) Unoccupied Setup/Setback** property specifies a setpoint offset for unoccupied and night setback modes. During unoccupied and night setback periods (as specified by schedule objects that are selected through the **SM** property), the **SP** is either setup or setback by an amount specified in **SU**. The sign of the control loop (**SG**) determines whether **SU** is a setup (**SG**=0) or setback (**SG**=1) amount. **CS** (the effective setpoint) incorporates any setup/setback that may exist.

The **(DB) Desired Control DeadBand** property specifies a control deadband for the thermostatic control loop. For a normal action control, this specifies the amount by which the temperature must drop below the cooling setpoint before the output is de-energized (**SP**-**DB)**. For a reverse action control. this specifies the amount by which the temperature must rise above the heating setpoint before the output is de-energized (**SP**+**DB)**. This response is illustrated in Figure 3-10.

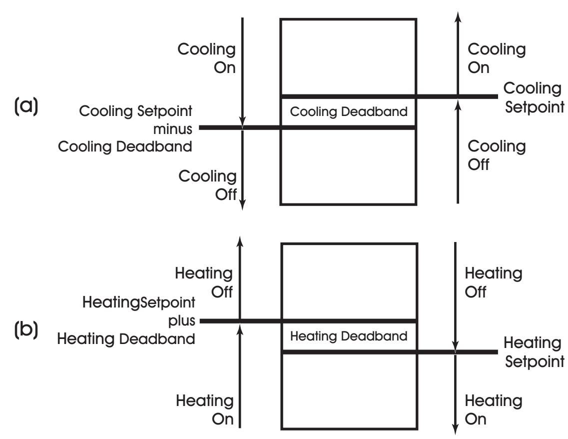

Figure 3-10: Deadband for a Normal Acting (a) and Reverse Acting (b) Thermostatic Control Loop

The **(SM) Schedules to Follow** property is used to select which schedules are used to control the setpoint adjustment of the thermostatic control loop. In addition to using schedules, you can select SmartStat override control and/or host override control. Each bit in **SM** corresponds to a desired control function. These bits are summarized in Table 3-19.

Table 3-19 : Schedules to Follow

SM bit	Schedule
0	Schedule 1
1	Schedule 2
2	Schedule 3
3	Schedule 4
4	Schedule 5
5	Schedule 6
6	Schedule 7
7	Schedule 8
8	SMARTStat 1
9	SMARTStat 2
10	SMARTStat 3
11	SMARTStat 4
12	SMARTStat 5
13	SMARTStat 6
14	SMARTStat 7
15	SMARTStat 8
16	SMARTStat 9
17	SMARTStat 10
18	SMARTStat 11
19	SMARTStat 12
20	Host Schedule
21	Schedule Summary
22	Occupancy

You select one or more of the control options in Table 3-19 by setting the appropriate bit(s) in **SM** to 1. If you do not want to use schedule control, set all of the bits in **SM** to 0.

The **(OB) Output Channel Bitmap** property is used to specify which digital outputs will be controlled by the thermostatic control loop. **OB** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. Setting a bit of **OB** allows the thermostatic control loop to control the corresponding digital output.

NOTE
The control loop will not enable unless at least one Digital Output is selected in **OB**

Thermostatic Control 1-12 Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
CE	**Enable Control Loop?** enables/disables thermostatic control for the associated control loop. 0=Disabled 1=Enabled
CS	**Calculated Control Setpoint** specifies the calculated (actual) control setpoint that is used by the thermostatic control loop. **CS** accounts for the effects of setup/setback (**SU**) during scheduled unoccupied periods and TS
CV	**Current Value of Selected Output** indicates the current output of the control loop.
DB	**Desired Control DeadBand** specifies a control hysteresis that is used to keep **present_value** from toggling when the value is on the border between two states.
DL	**Demand Load** indicates the heating/cooling demand in terms of the measured variable separation from setpoints
IA	**Input Attribute** specifies the property associated with the object specified in **IC** to be used for thermostatic control.
IC	**Input Channel** specifies the object to be used for thermostatic control.
OB	**Output Bitmap** specifies which digital outputs will be controlled by the thermostatic control loop. **OB** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
SG	**Control Action** specifies the control sign for the thermostatic control application. 0=Cooling 1=Heating
SM	**Schedules to Follow** enables scheduled alarm controlling for the thermostatic control loop by selecting one or more of the available schedule control objects. 0=schedule disabled 1=schedule enabled **SM** is a bitmap with bit 0=Schedule 1 bit 1=Schedule 2 bit 2=Schedule 3 bit 3=Schedule 4 bit 4=Schedule 5 bit 5=Schedule 6 bit 6=Schedule 7 bit 7=Schedule 8 bit 8= SMARTStat 1 bit 9= SMARTStat 2 bit 10= SMARTStat 3 bit 11= SMARTStat 4 bit 12= SMARTStat 5 bit 13= SMARTStat 6 bit 14= SMARTStat 7 bit 15= SMARTStat 8 bit 16= SMARTStat 9 bit 17= SMARTStat 10 bit 18= SMARTStat 11 bit 19= SMARTStat 12 bit 20= Host Schedule bit 21= Schedule Summary bit 22= Occupancy
SP	**Loop Setpoint** specifies the desired setpoint for the thermostatic control loop.
SU	**Unoccupied Setup/Setback** specifies a value (0.0 to 25.5) which is offset from the control setpoint during scheduled unoccupied periods.
TS	**Thermostat Setpoint Adjustment** indicates the offset, read from a Stat, to be applied to control setpoints.

3.18 PID CONTROL 1-12

Proportional + Integral + Derivative (PID) represents a method of control that controls equipment according to a setpoint in proportion to the value of a measured variable. It accounts for the amount of error (difference between the measured variable and the setpoint) and the continued presence of error. Each PID Control object has the following properties: **object_identifier, object_name, object_type, AO, CE, CS, DB, DL, IA, IC, MR, OH, OL, PB, PO, RA, RC, RL, RP, RS, RT, SG, SM, SP, SR, SU,** and **TS.**

The **(CE) Enable Control Loop?** property enables and disables the PID loop. When **CE** = 0, the loop output is not updated but may be set manually. When **CE** = 1, the loop output is updated by the PID control loop and the corresponding analog output is controlled.

NOTE
All values written by the PID Control loops are written at a Priority Array Level of 11.

The **(SG) Control Sign** property specifies the control action for the control loop. When **SG** = 0 (normal), a positive error causes an increase in output. When **SG** = 1 (reverse), a positive error causes a decrease in output. This point determines the response of the loop output to the kind of error. If the output action is to be increased (toward max) when the error is positive, set **SG** to normal (0). If the output action is to be decreased (toward min) for positive error, set **SG** to reverse (1). (Property **SG** is also used during schedule control to determine whether the setup/setback is added to **SP** [**SG** = 0] or subtracted from **SP** [**SG** = 1] during unoccupied and night setback periods.)

The **(IC) Input Channel** and **(IA) Input Attribute** properties specify the object and property to be used as the Loop Measured Variable. It specifies the input to be used for the control loop's measured variable. A list of the most common choices for **IA** will be displayed. If you wish to use a property not listed, for example a user defined property from an SPL program, you can enter the two-letter code into **IA**. Any two letter property name may be entered in this way.

NOTE
To clear **IC** and remove the association with its target object, you must write a value of 0000 (four zeroes) to **IC**.

The **(SP) Loop Setpoint** property specifies the desired setpoint for the control loop. The measured input variable is the analog sensor specified in the **IC** and **IA** properties. The setpoint is expressed in the same kind of measurement units (engineering units) that the measured variable uses (e.g., degrees, cfm, inches of WC, etc.). This value is used with the setup/setback value and any reset to calculate the actual setpoint used to control the loop.

The **(TS) Thermostat SP Adjustment** property indicates any setpoint adjustments read in from a SmartSTAT. This value is used in calculating the final control setpoint.

The **(SM) Schedules to Follow** property allows the for scheduled setpoint setup/setback and is used to select which schedules are used to control the setpoint adjustment of the PID control loop. In addition to using schedules, you can select SmartStat override control and/or host override control. Each bit in **SM** corresponds to a desired control function. These bits are summarized in Table 3-20.

Table 3-20 : Bit options for SM

Bit	Schedule to Follow
0	Schedule 1
1	Schedule 2
2	Schedule 3
3	Schedule 4
4	Schedule 5
5	Schedule 6
6	Schedule 7
7	Schedule 8
8	SMARTStat 1
9	SMARTStat 2
10	SMARTStat 3
11	SMARTStat 4
12	SMARTStat 5
13	SMARTStat 6
14	SMARTStat 7
15	SMARTStat 8
16	SMARTStat 9
17	SMARTStat 10
18	SMARTStat 11
19	SMARTStat 12
20	Host Schedule
21	Schedule Summary
22	Occupancy

You select one or more of the control options in Table 3-20 by setting the appropriate bit(s) in **SM** to 1. If you do not want to use schedule control, set all of the bits in **SM** to 0.

The **(SU) Unoccupied SetUp/Setback** property specifies the amount to add (if **SG** = 0) or subtract (if **SG** = 1) from the setpoint during an unoccupied or night setback period. The adjusted setpoint will be displayed in **CS**. The property **CS** (the effective setpoint incorporates any setup/setback that may exist as well as any reset or setpoint adjustment from the *SBC*-STAT.

The **(CS) Calculated Control Setpoint** property shows the actual loop control setpoint. This property reflects the setup/setback as well as any reset and/or SmartSTAT setpoint adjustment. This point is expressed in the same kind of measurement units (engineering units) that the measured variable uses (e.g., degrees, cfm, inches of WC, etc.).

The **(DL) Demand Load** property indicates the amount by which **CS** differs from the loop measured variable specified in **IC** and **IA**.

The **(PO) Percent Output** property shows the output value of the control loop. The value is calculated based on the error, change in error and past error for the control loop. **PO** can be set manually if the control loop is disabled (**CE** = 0).

The **(OL) Minimum Output Limit** and **(OH) Maximum Output Limit** properties define the minimum and maximum limits of the output range for the control loop. The value of **PO** will be scaled between **OL** and **OH** before being sent to the analog output.

The **(AO) Scaled Analog Output Value** property shows the value of **PO,** scaled between **OL** and **OH,** which is passed to the **present_value** of the associated analog output.

The **(MR) Maximum Amount to Reset Setpoint** property specifies the maximum amount to reset the loop setpoint (**SP**) based on when reset is being used. Property **CS** takes into account the use of the maximum reset specified in **MR**.

The **(RC) Reset Variable** and **(RA) Reset Attribute** specify the object and property to be used as the Reset Variable.

The **(RS) Reset Setpoint** property specifies the value at which the reset action begins. When the value of the reset variable exceeds **RS**, reset action will be used in determining the calculated setpoint. Just as **SP** is the proportional control setpoint for the measured variable specified in **IC** and **IA**, **RS** is the reset control setpoint for the value of the reset variable selected by **RC** and **RA**.

The **(RL) Reset Limit** property specifies the value at which maximum reset is used. When the value of the reset variable is equal to **RL**, the maximum reset (**MR**) is used in determining the calculated setpoint (**CS**).

The relationship between **RL** and **RS**, as well as the sign (+ or –) of **MR**, determines how changes in the reset variable specified by **RC** and **RA** affect the calculated control setpoint **CS**. Refer to Figure 3-11.

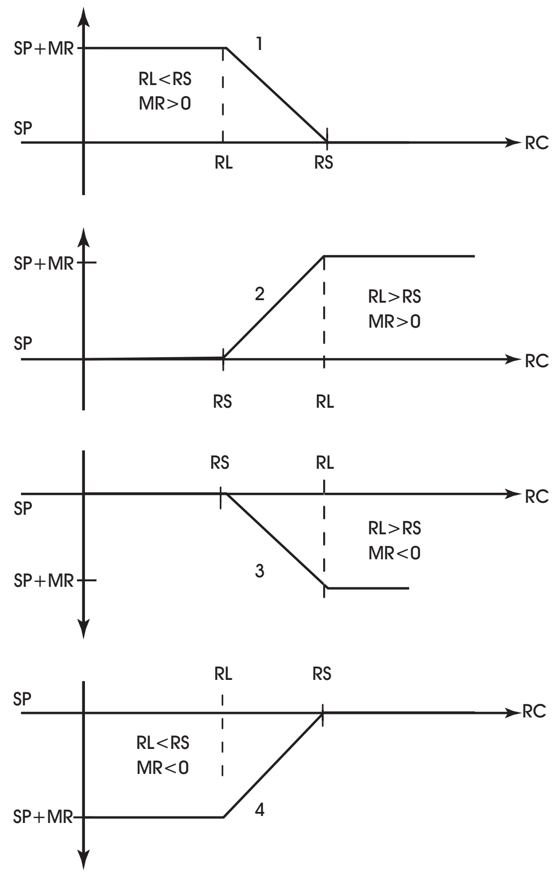

Figure 3-11: Four Forms of Reset Action

The **(DB) Desired Control Deadband** property specifies the deadband within the proportional control band in which the output remains constant at a point midway between maximum output and minimum output. By specifying a **DB** that is greater than or equal to the resolution of the sensor specified in **IC** and **IA**, you eliminate the possibility of cycling around the setpoint. The value of **DB** should never exceed the proportional band **PB**. If **DB** is greater than **PB**, then the control loop will not have proportional control.

Property **DB** is used to specify an input variable range within the proportional band **PB**. The size of **DB** should be based on the type of sensor input selected for the input specified in **IC** and **IA**. When the value of the measured variable is within this dead band, the output signal remains constant at the midpoint of the minimum/maximum range.

The point **DB** is centered on the setpoint **SP** to create the actual control dead band. When the value of the control variable (specified by **IC**) is within ±**DB**/2 of the setpoint **SP**, the *NB*-GPC3 assumes that it has reached the setpoint. Refer to Figure 3-12.

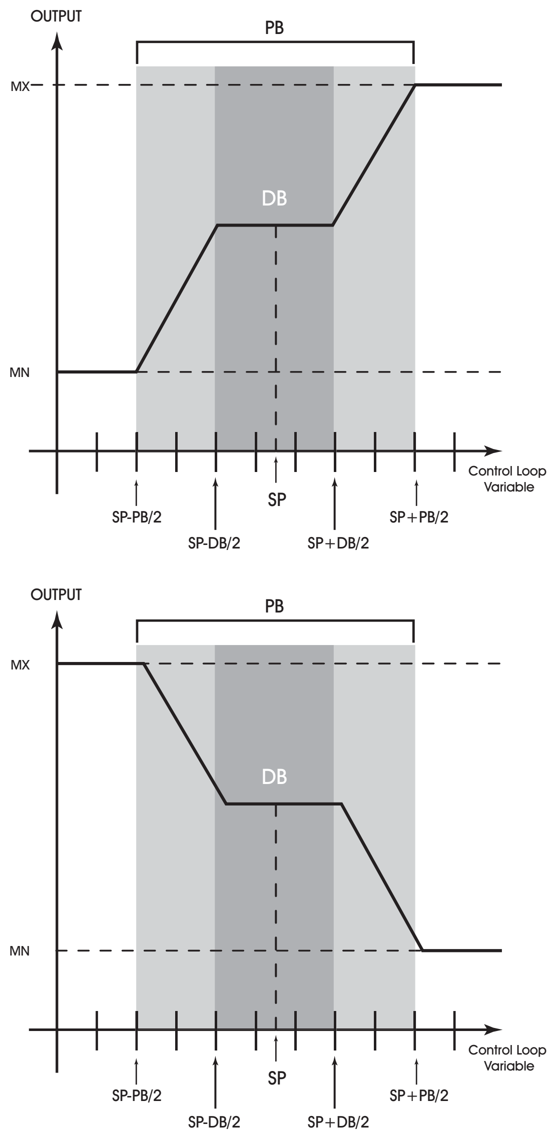

Figure 3-12: Normal Acting (above) and Reverse Acting (below), Proportional Control Output Response Showing a Dead Band Centered Around the Setpoint (SP)

By entering a value in **DB** that is greater than the resolution of the measured variable sensor, you create a deadband that allows the *NB*-GPC3 to effectively reach setpoint.

Be sure that the **DB** selected does not exceed the size of the proportional band (**PB**). The property **DB** is expressed in the same kind of measurement units (engineering units) that the measured variable uses (e.g., degrees, cfm, inches of WC, etc.). The data type of **DB** is the same as the data type of the selected measured variable. The point **DB** defaults to 0.

CAUTION

*Never change **DB** to a value greater than half of the proportional band **PB**. Doing so will eliminate the effects of PID control, resulting in on/off control.*

With appropriate values entered for these properties, the *NB*-GPC3 will provide simple closed loop feedback proportional control. This means that the actual measured performance of the control (from the measured variable input) is fed back to the controller and is compared with the effective setpoint for the loop. Any difference between the actual value of the measured variable and effective setpoint values is called error.

An analogy is helpful in explaining the effects of error. Figure 3-13 shows a simple lever and fulcrum. A change in the lever position on the error side produces a proportional change in the lever on the output side. Depending on the position of the fulcrum, a change on the error side will have a greater or lesser effect on the output side. The fulcrum position changes the ratio of error to output.

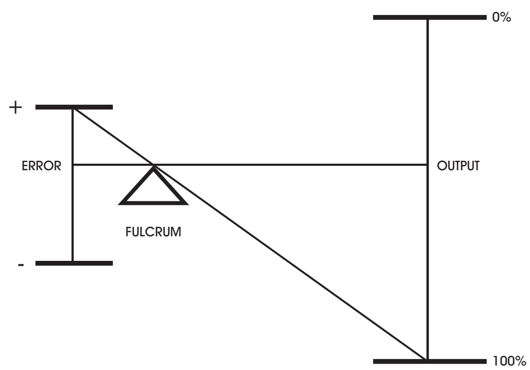

Figure 3-13: Input/Output Ratio

One problem with proportional only control is the changes in loop performance that occur when the condition being measured by the input sensor changes (e.g., the measured temperature changes when a door is opened and the room or space is flooded with cold air). As the loop environment changes, the proportional only control loop begins to cycle around an offset from the setpoint. Figure 3-14 illustrates the performance of a typical loop under proportional only control.

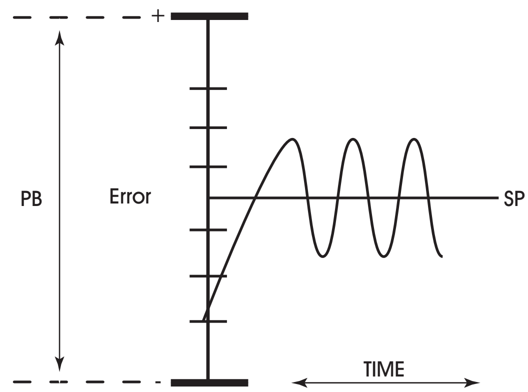

Figure 3-14: Proportional Only Control

The **(PB) Proportional Control Band** specifies the input variable range over which the output value is proportional to the error value (i.e., changes in the measured variable result in proportional changes in the output signal). The proportional band is centered around setpoint for the loop. This point is expressed in the same kind of measurement units (engineering units) that the measured variable uses—for example: degrees, cfm, inches of WC.

To determine **PB**, first decide how closely the *NB*-GPC3 must control the output to the setpoint. For instance, if the setpoint is 72°F, then an acceptable control range might be within two degrees of the setpoint. This control range can be expressed as a band centered on the setpoint: from 70° to 74°, or 4 degrees—the *proportional band* (**PB**). Refer to Figure 3-15 and Figure 3-16.

For normal acting control loops (see Figure 3-15), the **(PO) Percent Output** property is set to maximum output when the input variable equals the setpoint plus half of the proportional band (**CS** + **PB**/2). The percent output is set to minimum output when the input variable equals the setpoint minus half of the proportional band (**CS** - **PB**/2). These associations are reversed for reverse acting control loops. **PO** will be midway between minimum and maximum output when the measured variable is equal to the control setpoint **CS**. The opposite would be true for reverse acting control loop as shown in Figure 3-16.

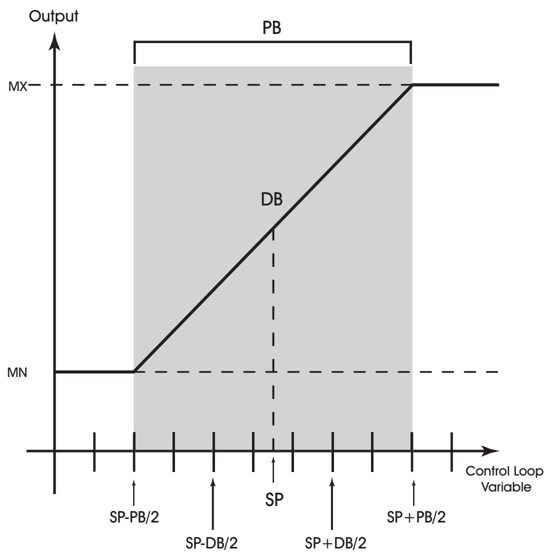

Figure 3-15: Proportional Band for Normal Acting Control (SG = 0)

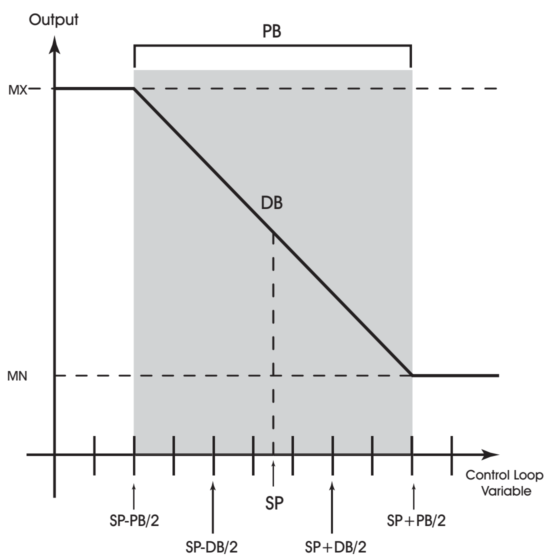

*Figure 3-16: Proportional Band for Reverse Acting Control (**SG** = 1)*

Proportional only control produces cycling, and its performance changes when the measured environment changes. The way to eliminate cycling and to compensate for load changes is to use *integral* action, the “I” part for PID control.

Rather than responding exclusively to the loop error from moment to moment as is the case with proportional action, integral action is based on a summation of the error that has occurred over some period. This error sum is used to reset, or modify, the response of the control loop (output) based on a running average of the error. The amount of time over which the error averaging is accumulated is called the *reset period*.

The **(RP) Reset Period** property specifies the reset period (in seconds) over which the error history is accumulated. If **RP** = 10 seconds with a constant error of 2.0, then the error history would increase by 0.2 every second. In five seconds, the error history would be 1.0. At the end of ten seconds, the error history would be 2.0. Setting **RP** to 0 disables integral action making the loop proportional only. The longer **RP** is, the less effect it has on the control response. Figure 3-17 shows the response of a typical control loop when integral action is used in addition to proportional action (PI control). A value of 0 disables the reset period.

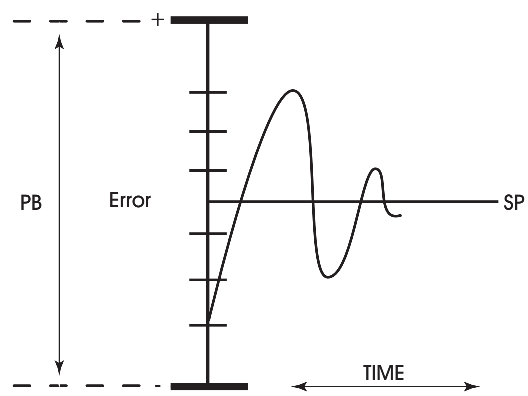

Figure 3-17: Proportional + Integral (PI) Control

At the start-up of the loop or following a change in setpoint (see Figure 3-17), the error is fairly large. Proportional action causes the loop output to accelerate toward the setpoint. However by the time the loop response reaches the setpoint value, it has gained inertia from the preceding proportional action. This causes the loop to overshoot the setpoint. As the loop exceeds the setpoint moving toward its first peak, the error sum is accumulating. This slows down the acceleration, eventually causing the downturn in response.

As the error falls and then drops below the setpoint, the error sum will be reduced because now the error is in the opposite direction. The cycle continues in diminishing peaks until it finally converges at the setpoint as shown in Figure 3-17.

The proportional control action of the loop has a major effect on integral action. Increasing **PB** results in a smaller integral effect for a given value of **RP**. In general, decreasing the proportional band, **PB**, will increase the magnitude of the changes in **PO**.

Several important factors may not be obvious to inexperienced users of these DDC techniques.

First, whenever the error falls outside of the proportional band—that is, ±**PB**/2 from the setpoint, two important things happen: the controller's output is fully pegged in the appropriate direction, and the error sum stops accumulating. The control produces its maximum output because it must bring the error within the proportional band again. The error sum stops accumulating so that it does not "wind up" a massive error sum that would take many control cycles to dissipate. This feature is called antireset windup.

Antireset windup also makes the loop recover quickly when it reenters the proportional band. Another feature of antireset windup is that the error history is limited to **PB**/2 because that is all that required to produce maximum output. Additional error accumulation would only slow down loop recovery.

To quicken loop response while eliminating overshoot, derivative action must be taken. Derivative action takes into account the rate of change in error and allows the *NB*-GPC3 to counter the effects of the error's rate of change on the control output. To find the change in error, subtract the current error (read every second by the PID loop) from the previous second's error. A percentage of this change (specified by **RT**) becomes the derivative contribution to the PID output.

The **(RT) Derivative Rate** property specifies a percentage of change in error that is to be used in calculating **PO**. The value is specified in percent per second. The point **RT** can have any value from 0.0 to 25.5%/second. The effect of adding derivative action to the output response appears in Figure 3-18.

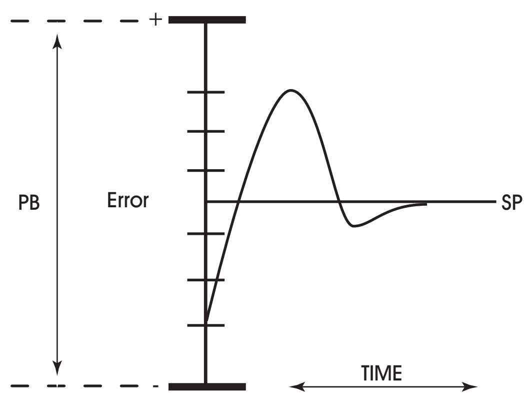

Figure 3-18: Proportional + Integral + Derivative (PID) Control

The **(SR) Soft Start Ramp** property specifies the maximum percentage change per minute for the associated output under he following conditions: when the controller is initially powered up or reset; upon transitions from unoccupied to occupied mode, upon cancellation of an interlock failure or fire condition, or when a control loop is initially enabled. These situations can cause the control loop to peg to 100% which can cause the output to spike and, in turn, could lead to equipment damage. To prevent this, the output will be limited to changing **SR** percent per minute. The effects of **SR** are seen in **AO**, but do not effect **PO**.

PID Control Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
AO	**Analog Output Value** displays the output of the PID loop.
CE	**Enable Control Loop?** enables/disables PID control. 0=No 1=Yes
CS	**Calculated Control Setpoint** specifies the effective (calculated) setpoint accounting for setup/setback, manual setpoint adjustments, etc.
DB	**Desired Control Deadband** specifies the deadband that is used to control cycling around the setpoint. If the current value of the input object is between **SP**-(**DB**/2) and **SP**+(**DB**/2), the measured variable is considered to be at its setpoint.
DL	**Demand Load** indicates the heating/cooling demand in terms of the temperature separation from setpoints
IA	**Loop Measured property** specifies the property associated with the object specified in **IC** to be used as the measured variable for PID control loop.
IC	**Loop Measured object** specifies the input object to be used as the measured variable for the PID control loop.
MR	**Maximum Amount to Reset Setpoint** the maximum amount to reset the control loop setpoint.
OH	**Output High Limit** defines the maximum output for the PID control loop.
OL	**Output Low Limit** defines the minimum output for the PID control loop.
PB	**Proportional Control Band** specifies a range, centered around the loop setpoint **SP**, where the output signal is proportional. If the value of the selected input object is outside the proportional band, the proportional component of the PID calculation is clamped at **OL** or **OH** as appropriate.
PO	**Percent Output Value** displays the calculated output of the PID control loop. **PO** ranges from **OL** to **OH**.

Property	Description
RA	**Reset property** specifies the attribute associated with the object specified in **RV** to be used as the reset variable for the PID control loop.
RC	**Reset object** specifies the input object to be used as the reset variable for the PID control loop.
RL	**Limit for Maximum Reset** specifies the reset limit of the control loop. When **RV** reaches a value of **RL**, the control loop setpoint will be reset by the maximum amount **RV**.
RP	**Reset Period** specifies a time, in seconds (0 to 65,535) over which the output of the control loop should be adjusted (reset) using integral action. 0=Disabled 1 to 65,535=Integral reset period, in seconds
RS	**Setpoint at which Reset Action Begins** specifies the setpoint of the control loop at which reset action begins.
RT	**Derivative Rate** specifies a a percentage of the amount of derivative error that is contributed each second to the PID output of the control loop (0.0 to 25.5%). 0.0=Disable 0.1 to 25.5=Derivative rate in %/second
SG	**Control Action** specifies whether the controller's output should be increased or decreased when error is positive. 0=Normal (increase for positive error) 1=Reverse (decrease for positive error)

Property	Description
SM	**Schedules to Follow** enables scheduled alarm controlling for the associated PID control loop by selecting one or more of the available schedule control objects. 0=Schedule disabled 1=Schedule enabled **SM** is a bitmap with bit 0=Schedule 1 bit 1=Schedule 2 bit 2=Schedule 3 bit 3=Schedule 4 bit 4=Schedule 5 bit 5=Schedule 6 bit 6=Schedule 7 bit 7=Schedule 8 bit 8= SMARTStat 1 bit 9= SMARTStat 2 bit 10= SMARTStat 3 bit 11= SMARTStat 4 bit 12= SMARTStat 5 bit 13= SMARTStat 6 bit 14= SMARTStat 7 bit 15= SMARTStat 8 bit 16= SMARTStat 9 bit 17= SMARTStat 10 bit 18= SMARTStat 11 bit 19= SMARTStat 12 bit 20= Host Schedule bit 21= Schedule Summary bit 22= Occupancy
SP	**Loop Setpoint** specifies the desired value of the variable selected in **IC** and **IA**.
SR	**Soft Start Ramp** specifies the maximum percentage change per minute for the associated output under he following conditions: when the controller is initially powered up or reset; upon transitions from unoccupied to occupied mode, upon cancellation of an interlock failure or fire condition, or when a control loop is initially enabled.
SU	**Unoccupied Setup/Setback** specifies a value (0.0 to 25.5) which is added to (if **SG**=1) or subtracted from (if **SG**=0) the control loop setpoint during scheduled unoccupied periods.
TS	**Thermostat Setpoint Adjustment** indicates the offset, read from a Stat3, to be applied to control setpoints.

3.19 SCHEDULE SUMMARY

The Schedule Summary object is a convenient way to monitor the schedules on the *NB*-GPC3. The Schedule Summary object has the following properties: **object_identifier, object_name, object_type, AS, C1, C2, C3, C4, C5, C6, C7, C8, CV, DH, H0, H1, H2, H3, H4, H5, H6, H7, H8, H9, HE, HO, IS,** and **ZE.**

The **(CV) Occupied Schedules** displays the highest priority active schedule state. Also, the schedule state for Schedule 1 through Schedule 8 are individually displayed in properties **C1** though **C8**.

The **(AS) Active Schedule** property is a bitmap which displays the schedules that are active for the current day.

Properties **H0** through **H9** allow you specify scheduled holidays.

The **(DH) Today is Holiday** property, indicates if the current date corresponds to any of the scheduled holidays.

The **(IS) Inactive Schedule State** property specifies the mode that the controller will enter if no schedules are active. This could occur, for example, if today is listed as a holiday, but there are no schedules which include holidays.

Using the **(HE) Host Schedule Override Enable** property, the *NB*-GPC3 can be set to use a schedule which is broadcast by a host unit.

The **(HO) Host Schedule Value** property displays current status of the host schedule.

The **object_name** property can be used to provide a descriptive name for the object.

3.19.1 ZONE SCHEDULING

The GPC provides a set of attributes that can be used with an SPL program to provide a robust zone scheduling application.

Property **SO** shows the state of each SBC-STAT's local override flag. The state of the flag is controlled locally at the SBC-STAT interface through the User Menu or through the use of the setpoint keys. An occerride command from the local interface will only be allowed if the corresponding bit for the STAT from **EA** is set to 0 ro disable the remote access button and the **AM** bit is seto to 0 for Auto mode. The **SO** attribute will not reflect a manual override from **OM**.

Attribute **AB** displays the state of the remote access control buttons. Writes to this attribute are always permitted regardless of the value displayed in **EA**

Attribute **EA** is read-write point used to enable or disable the ability to enable Override mode from a corresponding SBC-STAT.

Attribute **OM** displays whether or not an SBC-STAT is in an override state. Each bit set in **OM** attribute will reflect the value of the corresponding bit from the local STAT interface **SO** whenever access buttons **EA** are disabled and auto mode **AM** is enabled. If access buttons are enabled, **OM** will be controlled by the access buttons states **AB**; a value of 1 on an access button will initiate the STAT into override. Writes to these bits are permitted only when the corresponding bit in **AM** is set to a value of 1 (manual mode).

The **AM** attribute is used to enable or disable manual writes to the **OM** attribute. A bit with a value of 0 disables write access; where a bit with a value of 1 enables write access.

Schedule Summary Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
AS	**Active Schedule** indicates which schedule is currently active.
C1	**Schedule 1** displays the schedule state of Schedule 1.
C2	**Schedule 2** displays the schedule state of Schedule 2.
C3	**Schedule 3** displays the schedule state of Schedule3.
C4	**Schedule 4** displays the schedule state of Schedule 4.
C5	**Schedule 5** displays the schedule state of Schedule 5.
C6	**Schedule 6** displays the schedule state of Schedule 6.
C7	**Schedule 7** displays the schedule state of Schedule 7.
C8	**Schedule 8** displays the schedule state of Schedule 8.
CV	**Occupied Schedules** displays the highest priority schedule state of all active schedules.
DH	**Holiday?** specifies whether today is one of the holidays specified in **H0**-**H9**.
H0	**Programmed Holiday 1** specifies the date of the first scheduled holiday.
H1	**Programmed Holiday 2** specifies the date of the second scheduled holiday.
H2	**Programmed Holiday 3** specifies the date of the third scheduled holiday.
H3	**Programmed Holiday 4** specifies the date of the fourth scheduled holiday.
H4	**Programmed Holiday 5** specifies the date of the fifth scheduled holiday.
H5	**Programmed Holiday 6** specifies the date of the sixth scheduled holiday.

Property	Description
H6	**Programmed Holiday 7** specifies the date of the seventh scheduled holiday.
H7	**Programmed Holiday 8** specifies the date of the eighth scheduled holiday.
H8	**Programmed Holiday 9** specifies the date of the ninth scheduled holiday.
H9	**Programmed Holiday 10** specifies the date of the tenth scheduled holiday.
HE	**Enable Host Schedule** specifies whether to use the schedule broadcast by the host. 0=No 1=Yes
HO	**Host Schedule Value** specifies the desired host schedule override state. 0=Unoccupied 1=Warmup 2=Occupied 3=Night setback
IS	**Inactive Schedule State** indicates the state the controller will assume when no schedules are currently active 0=Unoccupied 1=Warmup 2=Occupied 3=Night setback
AS	**Active Schedules** lists the schedules which are currently active. bit #0=Schedule 1 bit #1=Schedule 2 bit #2=Schedule 3 bit #3=Schedule 4 bit #4=Schedule 5 bit #5=Schedule 6 bit #6=Schedule 7 bit #7=Schedule 8
SO	**Digital Stat Override Status** displays the state of each STAT’s local override flag. bit #0=SMARTStat 1 bit #1=SMARTStat 2 bit #2=SMARTStat 3 bit #3=SMARTStat 4 bit #4=SMARTStat 5 bit #5=SMARTStat 6 bit #6=SMARTStat 7 bit #7=SMARTStat 8 bit #8=SMARTStat 9 bit #9=SMARTStat 10 bit #10=SMARTStat 11 bit #11=SMARTStat 12

Property	Description
AB	**Access Buttons States** displays the state of the remote access control buttons. bit #0=Button 1 bit #1=Button 2 bit #2=Button 3 bit #3=Button 4 bit #4=Button 5 bit #5=Button 6 bit #6=Button 7 bit #7=Button 8 bit #8=Button 9 bit #9=Button 10 bit #10=Button 11 bit #11=Button 12
EA	**Enable Access Buttons** enables remote access button presses. bit #0=Enable Button 1 bit #1=Enable Button 2 bit #2=Enable Button 3 bit #3=Enable Button 4 bit #4=Enable Button 5 bit #5=Enable Button 6 bit #6=Enable Button 7 bit #7=Enable Button 8 bit #8=Enable Button 9 bit #9=Enable Button 10 bit #10=Enable Button 11 bit #11=Enable Button 12
OM	**Override Map** displays whether or not a STAT is in an override state. bit #0=Override 1 bit #1=Override 2 bit #2=Override 3 bit #3=Override 4 bit #4=Override 5 bit #5=Override 6 bit #6=Override 7 bit #7=Override 8 bit #8=Override 9 bit #9=Override 10 bit #10=Override 11 bit #11=Override 12
AM	**Manual Control** enables write commands to OM. bit #0=Enable Button 1 bit #1=Enable Button 2 bit #2=Enable Button 3 bit #3=Enable Button 4 bit #4=Enable Button 5 bit #5=Enable Button 6 bit #6=Enable Button 7 bit #7=Enable Button 8 bit #8=Enable Button 9 bit #9=Enable Button 10 bit #10=Enable Button 11 bit #11=Enable Button 12

3.20 SCHEDULES 1-8

The Schedule objects on the *NB*-GPC3 are used to set the occupied, unoccupied, night setback and warmup periods for the eight schedules available in the *NB*-GPC3. Each of the Schedule objects has the following properties: **object_identifier, object_name, object_type, AD, AO, CV, NS, OC, UN,** and **WO.**

The **(CV) Current Schedule Value** property shows the current state for that particular schedule (0=unoccupied, 1= warm-up, 2=occupied, 3=night setback).

The *NB*-GPC3 operates in one of four scheduled control states: occupied, unoccupied, warm-up and night setback.

Unoccupied mode, **CV**=0, is the period of time when people are not expected to be in the zone and temperature control is not as strict. During unoccupied mode, the *NB*-GPC3 maintains cooling comfort levels at setup values and heating comfort levels at setback values. These setup and setback values are used to broaden the control range between the heating and cooling setpoints in order to provide less stringent control. The properties used to define the offsets are located in the PID Control, Thermostatic Control, and Floating Point Control objects, but the time of implementation is set in the individual Schedule objects. Unoccupied mode usually ends when night setback begins.

Warmup, **CV**=1, is the period of time before occupancy. During this period, the central air handler unit supplies warm air to the VAV boxes. Warmup provides special control action to bring the zone temperature to its desired setpoint for the occupied mode, based on the heating setpoint. In time based warmup, the warmup period ends when occupied mode begins.

Occupied mode, **CV**=2, is the period of time when the zone is occupied by people and the *NB*-GPC3 must maintain appropriate comfort levels in the zone. The heating and cooling setpoints define a desired zone temperature range. Occupied mode ends when unoccupied mode time begins.

Night setback, **CV**=3, is the period of time during unoccupied mode when the entire building is usually unoccupied and the air handler may be shut down. During night setback mode, the damper would be closed completely (The *NB*-GPC3 assumes that air flow should be 0). The *NB*-GPC3 provides the option to set up and set back the night setback control temperature (as does the standard unoccupied mode), and when these offsets are reached or exceeded damper control of air flow resumes. During the night setback period, control setpoints are setup/setback just as they are for unoccupied periods.

NB-GPC3 schedules can be activated based on the values assigned to properties in the Schedule objects. When the current day of the week matches the setting of the active days property, **AD**, from one of the eight schedule objects, that object's schedule becomes active. More than one schedule can be active at any given time, but they are prioritized so that the schedule with the highest mode priority dictates the control mode. Priority is determined in the following order:

- occupied (highest priority)
- warmup
- unoccupied
- night setback (lowest priority)

Control loops of the *NB*-GPC3 may be configured to use the current schedule state of selected schedules to adjust setpoints. In occupied mode, for example, a setpoint value is used to determine when a certain control action occurs. In unoccupied and night setback modes, a setup (or setback) amount is added to (or subtracted from, depending on its sign) the control loop setpoint, altering the point at which the control action occurs. In unoccupied mode, the control action is less stringent since fewer (if any) people are in the building during that time.

The schedule mode properties define four windows for a schedule which is active for a set of days of the week. When the current day of the week matches one of the active schedule days, the time of day determines which of the four available modes that will dictate control strategy.

The **(AD) Active Days** property specifies a set of the eight possible days of the week (seven days plus holiday) during which the schedule will run in one of the four available modes at any given time of that particular active day.

By configuring the **(OC) Time to go Occupied**, **(UN) Time to go Unoccupied**, **(WO) Time to go Warmup**, **(NS) Time to go Night Setback** and **(AD) Active Days** properties, the controller can be configured to adjust the heating and cooling setpoints based on the time of day and day of the week.

Figure 3-19 shows and example of how **AD**, **WO**, **OC**, **UN** and **NS** can work together to define a schedule. The schedule shown would require 3 schedule objects to define. The first would have **AD** set active for Monday through Friday. **WO** would be 5:00, **OC** would be 8:00, **UN** would be 18:00, and **NS** would be 0:00. This would define the behavior for the weekdays.

The second schedule would be used for the weekends. The active days would be Saturday and Sunday and would be set in **AD**. **WO** would be 1:00, **OC** would be 6:00, **UN** would be 14:00, and **NS** would be 0:00.

The third schedule would be the holiday schedule. Since you want the schedule to be unoccupied during the entire day, you would use the **(AO) All Day Override** property. Setting **AO** tells the controller to stay in the specified mode for the entire day. In this case, you would set **AO** to “1=Unoccupied” and **AD** for holidays.

To configure the schedule to work this way would require three schedules. The first schedule would control the weekdays and would have **AD** set for monday through friday. To set the times for the various schedule states, **OC** would be set to 8:00, **UN** to 18:00, **NS** to 0:00, and **WO** to 5:00. The second schedule would cover the weekends and would have **AD** set for saturday and sunday. for this schedule **OC** would be set to 6:00, **UN** to 14:00, **NS** to 0:00, and **WO** to 1:00. Finally, the third schedule would have **AD** set for holidays and **AO** set to “1=Unoccupied”.

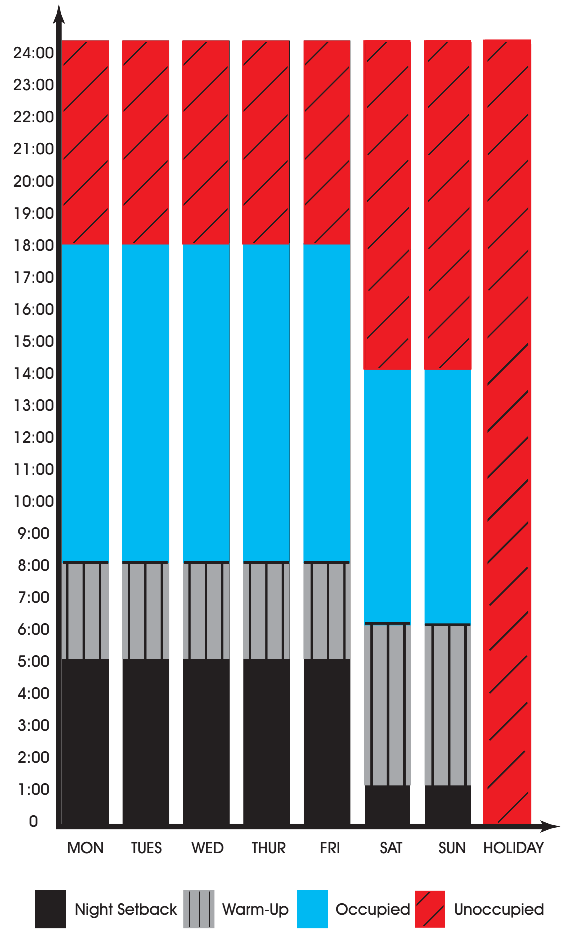

Figure 3-19: Example of schedule modes

When the *NB*-GPC3 is configured to receive schedules from a host controller, holiday schedules refer to the host system which defines the holidays in each month of the year. The *NB*-GPC3 also has the ability to define holidays in properties **HO** through **H9** in the Schedule Summary object. If the *NB*-GPC3 has bit 7 set for a currently active schedule, then the controller will follow that schedule when the holiday bit is sent from the host or if the current day corresponds to one of the holidays set in the *NB*-GPC3. If the host broadcasts a holiday, and the *NB*-GPC3 does not have a schedule with a holiday schedule, then no schedules will be active until the host clears the holiday.

All schedules are independent of each other. One or more schedules may be active when the rest are inactive.

A descriptive name can be stored in the name property (**object_name**) to help identify the schedule and when it should be used.

Schedule Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
AD	**Active Days** bitmap that specifies which days the schedule is active. Setting the appropriate bit to 1 indicates the schedule will b active on that day. bit 0=Monday bit 1=Tuesday bit 2=Wednesday bit 3=Thursday bit 4=Friday bit 5=Saturday bit 6=Sunday bit 7=Holiday
AO	**All Day Override** 0=No 1=Yes
CV	**Current Schedule Value** indicates the current state of the schedule. 0=Unoccupied 1=Warm-up 2=Occupied 3=Night setback
NS	**Time to go Night Setback** specifies the time when night setback mode should begin.
OC	**Time to go Occupied** specifies the time when occupied mode should begin.
UN	**Time to go Unoccupied** specifies the time when unoccupied mode should begin.
WO	**Time to go Warm-Up** specifies the time when warm-up mode should begin.

3.21 SCALES 1-4

Each Scale object performs a linear interpolation between two known points which can be used to scale a value within the controller by looking up values that lie along this linear segment. Each Scale object has the following properties: **object_identifier, object_name, object_type, CV, IA, IC, X1, X2, Y1,** and **Y2.**

The **(X1) Input Range X1 Value** and **(Y1) Input Range Y1 Value** and **(X1) Input Range X2 Value** and **(Y2) Input Range Y2 Value** properties to specify the starting and ending point respectively, the controller calculates the value for any point along the line connecting them.

Properties **X1**, **X2** and **Y1**, **Y2** indicate the x- and y-coordinate values to be used for the starting and ending points of the line segment which. Both x- and y-coordinate values are given in engineering units. Once these values are specified, the line is completely defined and the *NB*-GPC3 can freely interpolate values.

NOTE
If the value to be scaled is below **X1** or above **X2**, the output will be pegged at **Y1** or **Y2** respectively.

The **(IC) Input Channel** and **(IA) Input Attribute** properties specify the object and property which contains the x-value to be scaled. The scaled y-value will then be stored in the **(CV) Current Value** property. If you wish to use a property not listed, for example a user defined property from an SPL program, you can enter the two-letter code into **IA**. Any two letter property name may be entered in this way.

NOTE
To clear **IC** and remove the association with its target object, you must write a value of 0000 (four zeroes) to **IC**.

Property **object_name** stores the name of the object. This is a user definable string that can be used to help identify the object or, for example, to identify what the scaling is to be used for.

Scale Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
CV	**Current Value** specifies the calculated scaled value.
IA	**Input property** specifies the property associated with the object specified in **IC** to be scaled.
IC	**Input object** specifies the object to be scaled.
X1	**Input range X1 value** specifies the minimum value of the input.
X2	**Input range X2 value** specifies the maximum value of the input.
Y1	**Output range Y1 value** specifies the minimum value of the scaled output.
Y2	**Output range Y2 value** specifies the maximum value of the scaled output.

3.22 PIECEWISE CURVES 1-2

The *NB*-GPC3 can accommodate non-linear sensors by using built-in tables to define the response characteristics of the sensor. Each tables requires eleven points to define ten linear segments. These ten segments approximate the response of the sensor. The controller will perform a linear interpolation to 'look up' values that lie along an individual segment much like the calculations performed by the Scale objects. Each of the Piecewise Curve objects contain the following properties: **object_identifier, object_name**, **object_type, X1**, **X2**, **X3**, **X4**, **X5**, **X6**, **X7**, **X8**, **X9**, **XA**, **XB**, **Y1**, **Y2**, **Y3**, **Y4**, **Y5**, **Y6**, **Y7**, **Y8**, **Y9**, **YA** and **YB.**

The **object_name** property stores the name of the object. This is a user definable string that can be used to help identify the object or the sensor type it is approximating.

Properties **(X1) Point 1's value in % Full Scale** through **(XB) Point 11's value in % Full Scale** represent the sensor readings for eleven chosen points on a sensor curve. The acceptable range for **X1** through **XB** depend on the chosen sensor type. For a voltage input, the 0-10 V input range is mapped to **X1** through **XB** values from 0 through 100. A current input, with a range of 0-20 mA, can have **X1** through **XB** values from 0 through 50. The 0-250 kΩ range for a resistive input can have **X1** through **XB** values from 0 through 25. The values of **X1** through **XB** must be entered in increasing order (i.e. **X1** < **X2** < **X3** etc.).

NOTE
The Piecewise Curve will only interpolate values for input values between **X1** and **XB**. If the input is below **X1**, the Piecewise Curve will be pegged at the value associated with **X1**. If the input is above **XB**, the Piecewise Curve will be pegged at the value associated with **X1**.

Properties **(Y1) Point 1's value in engineering units** through **(YB) Point 11's value in engineering units** are the Engineering Unit values (e.g., 70 degrees, 72 degrees, etc.), corresponding to the sensor readings entered into **X1** through **XB**. These values, coupled with the corresponding sensor readings, define the line segments which make up the piecewise curve

3.22.1 PIECEWISE CURVES FOR VOLTAGE INPUTS

To program a piecewise curve for a nonlinear sensor, you need to know the response characteristics of the sensor. These response characteristics are usually supplied by the manufacturer and may be in the form of a graph or table. Figure 3-20 shows an example of what a curve for a temperature sensor may look like. Though the sensor response could extend beyond this range, you will get more accurate results if you limit the range of your Piecewise Curve to the range of values you expect to see from the sensor. Figure 3-20 only contains the portion of the sensor's response that would be needed for zone temperature monitoring.

Once you have the response data, in either graph or table form, you must choose the points which define the line segments that approximate the response curve in the expected response region. When choosing the points to use, you can use fewer points in areas of the curve that are mostly linear and concentrate the points to better approximate the more non-linear portions of the response. In Figure 3-20, you can see that more points are chosen near the 'bends' in the characteristic response curve.

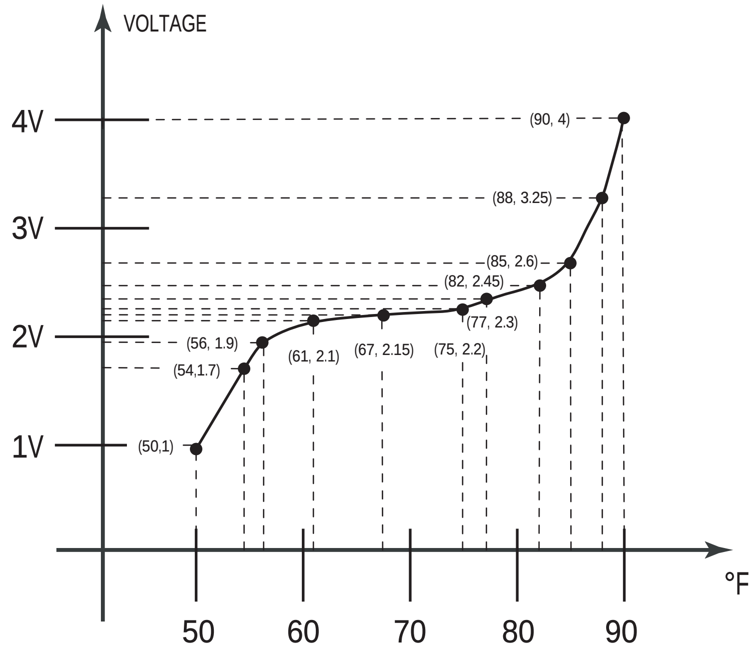

Figure 3-20: An Example of a Sensor Response Curve

From the graph, the following values are selected to represent the curve:

Table 3-21 : Sensor Response Points

Voltage	Temperature (°F)
1.00	50
1.70	54
1.90	56
2.10	61
2.15	67
2.20	75
2.30	77
2.45	82
2.60	85
3.25	88
4.00	90

Next you must convert the voltage values into a percentage of full scale to be used to define the x-coordinates of your piecewise curve. Since the voltage input has a range of 0-10 V, each volt measured corresponds to ten percent of the full scale. The formula for the percentage of full scale output for a voltage sensor is simply:

$$\%\text{ Full Scale} = \text{Voltage} \times 10$$

The calculated percentages correspond to **X1** through **XB** and the temperature reading correspond to **Y1** through **YB**. For the sensor described above, this gives the following assignments:

Table 3-22 : Assigning Sensor Response Points to the Piecewise Curve

		% Full Scale	Temperature (°F)		
X1	⇦	10.0	50	⇨	**Y1**
X2	⇦	17.0	54	⇨	**Y2**
X3	⇦	19.0	56	⇨	**Y3**
X4	⇦	21.0	61	⇨	**Y4**
X5	⇦	21.5	67	⇨	**Y5**
X6	⇦	22.0	75	⇨	**Y6**
X7	⇦	23.0	77	⇨	**Y7**
X8	⇦	24.5	82	⇨	**Y8**
X9	⇦	26.0	85	⇨	**Y9**
XA	⇦	32.5	88	⇨	**YA**
XB	⇦	40.0	90	⇨	**YB**

3.22.2 PIECEWISE CURVES FOR CURRENT INPUTS

Where the full scale of the voltage sensor is represented internally as a full 0 to 100%, a current sensor is represented in slightly less than 50% of the full scale readable by the controller. This means that the 0 to 20 mA full scale sensor reading range is mapped to the range of 0 to slightly less than 50% of the full scale readable by the controller. Calculating the Piecewise Curve for a current input is the same as for the resistive input, except that you would instead apply a different formula to calculate the percentage of full scale. Here, 1 mA read in from the sensor corresponds to 2.49% of the full scale. You can simply multiply the current value from the sensor's characteristic response, or you can use the following formula to calculate the percentage of full scale:

$$\%\text{ Full Scale} = \frac{\text{Current(mA)} \times 249}{100}$$

The values for **Y1** through **YB** are entered in Engineering Units in exactly the same way as for the voltage sensor.

3.22.3 PIECEWISE CURVES FOR RESISTANCE INPUTS

Like the current sensor, the resistance sensor is represented inside the controller a fraction of the full scale range possible in the controller. The 0 to 250 kΩ resistance range is represented internally as 0 to slightly less than 25% of the full scale readable by the controller. Calculating the a Piecewise Curve for a resistive input is also slightly different than for the voltage or current sensors because the controller measures the voltage drop across the input and that response is inherently non-linear. Because of this, there is no simple multiplier that can be used to convert resistance to full scale percentage as for the voltage or current sensors. Instead, you will have to use the following equation:

$$\%\ \text{Full Scale} = 25 \times \frac{\text{Resistance}(\Omega)}{\text{Resistance}(\Omega) + 20000}$$

The calculated full scale values are then entered into **X1** through **XB**. The values for **Y1** through **YB** are entered in Engineering Units in exactly the same way as for the voltage and current sensors.

Piecewise Curve Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
X1	**Point 1's value in raw counts** specifies the x coordinate of point 1.
X2	**Point 2's value in raw counts** specifies the x coordinate of point 2.
X3	**Point 3's value in raw counts** specifies the x coordinate of point 3.
X4	**Point 4's value in raw counts** specifies the x coordinate of point 4.
X5	**Point 5's value in raw counts** specifies the x coordinate of point 5.
X6	**Point 6's value in raw counts** specifies the x coordinate of point 6.
X7	**Point 7's value in raw counts** specifies the x coordinate of point 7.
X8	**Point 8's value in raw counts** specifies the x coordinate of point 8.
X9	**Point 9's value in raw counts** specifies the x coordinate of point 9.
XA	**Point 10's value in raw counts** specifies the x coordinate of point 10.
XB	**Point 11's value in raw counts** specifies the x coordinate of point 11.
Y1	**Point 1's value in engineering units** specifies the y coordinate of point 1.
Y2	**Point 2's value in engineering units** specifies the y coordinate of point 2.
Y3	**Point 3's value in engineering units** specifies the y coordinate of point 3.
Y4	**Point 4's value in engineering units** specifies the y coordinate of point 4.
Y5	**Point 5's value in engineering units** specifies the y coordinate of point 5.
Y6	**Point 6's value in engineering units** specifies the y coordinate of point 6.
Y7	**Point 7's value in engineering units** specifies the y coordinate of point 7.
Y8	**Point 8's value in engineering units** specifies the y coordinate of point 8.
Y9	**Point 9's value in engineering units** specifies the y coordinate of point 9.
YA	**Point 10's value in engineering units** specifies the y coordinate of point 10.
YB	**Point 11's value in engineering units** specifies the y coordinate of point 11.

3.23 LOGIC 1-4

The Logic objects on the *NB*-GPC3 are used to perform logical operations using selectable input objects and the user's choice of operator. For simple operations these objects can be used in place of SPL programs. Each of the Logic objects contain the following properties: **object_identifier, object_name, object_type, A1, A2, A3, A4, A5, A6, A7, A8, CV, I1, I2, I3, I4, I5, I6, I7, I8,** and **OP.**

The **object_name** property stores the name of the object. This is a user definable string that can be used to help identify the object or the operation it is performing.

The **(CV) Current Value** property is the current value of the performed operation. The logic object behaves as an 8-input logic gate. It indicates the result of applying the operand specified in **OP** to the values of the inputs properties specified in **I1**;**A1** through **I8**;**A8**. It will apply the selected operation to every input specified and then output the result, i.e. a logical AND will only return "true" if all of the specified inputs are "true". If only two inputs are specified, the Logic object will perform a simple logic operation on those inputs.

The **(I1) Input Channel 1** through **(I8) Input Channel 8** and **(A1) Input Attribute 1** through **(A8) Input Attribute 8** properties specify the input object and properties, respectively, for the first through eighth inputs for the associated object.

NOTE

You do not need to use all eight inputs when using the Logic objects. The NOT operator can be selected with only a single input, all other logical operations will be performed if two or more inputs are specified.

The **(OP) Operation** property specifies the logic operation to be performed on the selected properties. The operation available are listed in Table 3-23.

Table 3-23 : Logic Object operation codes

OP	Operation
0	Disabled
1	OR
2	AND
3	NOT
4	XOR

If OP=1, the object will perform a logical OR on all of the selected objects (i.e. **I1:A1** OR **I2:A2** OR **I3:A3** OR **I4:A4** OR **I5:A5** OR **I6:A6** OR **I7:A7** OR **I8:A8**). This operation will return "true" if any of the selected inputs are "true". If all of the inputs are "false", then **CV** will be set to "false".

If OP=2, the object will perform a logical AND on all of the selected objects (i.e. **I1:A1** AND **I2:A2** AND **I3:A3** AND **I4;A4** AND **I5;A5** AND **I6;A6** AND **I7;A7** AND **I8;A8**). This operation will return “true” only if all of the selected inputs are “true”. If any of the inputs are “false”, then **present_value** will be set to “false”.

If **OP**=3,the object will perform a logical NOT. The NOT operator will be applied to the first input only (i.e. **I1:A1**).

If OP=4, the object will perform a logical XOR on all of the selected objects (i.e. **I1;A1** XOR **I2;A2** XOR **I3;A3** XOR **I4;A4** XOR **I5;A5** XOR **I6;A6** XOR **I7;A7** XOR **I8;A8**). This operation will return “true” if only one of the selected inputs is “true”. Otherwise, the result will be “false”.

Logic Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
A1	**Input property 1** specifies the property associated with the first input object.
A2	**Input property 2** specifies the property associated with the second input object.
A3	**Input property 3** specifies the property associated with the third input object.
A4	**Input property 4** specifies the property associated with the fourth input object.
A5	**Input property 5** specifies the property associated with the fifth input object.
A6	**Input property 6** specifies the property associated with the sixth input object.
A7	**Input property 7** specifies the property associated with the seventh input object.
A8	**Input property 8** specifies the property associated with the eighth input object.
CV	**Current Value** indicates the result after the operand specified in **OP** has been applied to the inputs.
I1	**Input object 1** specifies the first input object.
I2	**Input object 2** specifies the second input object.
I3	**Input object 3** specifies the third input object.
I4	**Input object 4** specifies the fourth input object.
I5	**Input object 5** specifies the fifth input object.
I6	**Input object 6** specifies the sixth input object.
I7	**Input object 7** specifies the seventh input object.
I8	**Input object 8** specifies the eighth input object.
OP	**Operation** specifies the logic operation to be performed on the selected objects. 0=Diabled 1=OR 2=AND 3=NOT 4=XOR

3.24 MATH 1-2

The Math objects on the *NB*-GPC3 are used to perform mathematical calculations using selectable input objects and the user's choice of operator. The result can then be used as the measured variable input for any of the control loops within the controller or used in other Math, Scale, or Min/Max/Avg objects. Each of the Math objects contain the following properties: **object_identifier, object_name, object_type, A1, A2, CV, I1, I2,** and **OP.**

The **(I1) Input Channel 1** and **(A1) Input Attribute 1** properties specify the input object and property respectively. This becomes the first term in the mathematical calculation performed by the object. Similarly, the **(I2) Input Channel 2** and **(A2) Input Property 2** properties represent the object and property to be used as the second term in the calculation.

The **(OP) Operation** specifies the operation to be performed on the objects selected in **I1**, **I2**, **A1** and **A2**. The values of OP and the corresponding operations are listed in Table 3-24.

Table 3-24 : Math Object operation codes

OP	Operation
0	Disabled
1	Addition
2	Subtraction
3	Multiplication
4	Division

If **OP**=1, the object adds the values selected in **I1**, **A1**, **I2** and **A2** (i.e. **CV**=**I1:A1**+**I2:A2**).

If **OP**=2, the object subtracts the value selected in **I2**, **A2** from the value selected in **I1** and **A1** (i.e. **CV**=**I1:A1**-**I2:A2**).

If **OP**=3, the object multiplies the values selected in **I1**, **A1**, **I2** and **A2**(ie. **CV**=**I1:A1**×**I2:A2**).

If **OP**=4, the object divides the value selected in **I1**, **A1** by the value selected in **I2** and **A2** (i.e. **CV**=**I1:A1**÷**I2:A2**).

The **(CV) Current Value** property indicates the numeric result of applying the operand specified in **OP** to the values of the inputs properties specified in **I1** and **A1** and **I2** and **A2** respectively.

The **object_name** property stores the name of the object. This is a user definable string that can be used to help identify the object or the calculation it is performing.

Math Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
A1	**Input property 1** specifies the property associated with the first input object.
A2	**Input property 2** specifies the property associated with the second input object.
CV	**Current Value** indicates the numeric result of applying the operand specified in **OP** to the inputs.
I1	**Input object 1** specifies the first input object.
I2	**Input object 2** specifies the second input object.
OP	**Operation** specifies the operation to be performed on the selected objects. 0=Disabled 1=Addition 2=Subtraction 3=Multiplication 4=Division

3.25 MIN/MAX/AVG 1-3

The Min/Max/Avg objects on the *NB*-GPC3 are used to calculate the minimum, the maximum, and the average of a number of selectable input objects. The result can then be used as the measured variable input elsewhere within the controller. Each of the Min/Max/Avg objects contain the following properties: **object_identifier, object_name, object_type, A1, A2, A3, A4, AV, HV, I1, I2, I3, I4,** and **LV.**

The **(I1) Input Channel 1** through **(I4) Input Channel 4** and **(A1) Input Attribute** through **(A4) Input Attribute 4** properties are used to specify the first through fourth objects and properties to be used in the mix/max/avg calculation.

The **(AV) Average Value** property displays the arithmetic mean of the selected properties.

The **(HV) High Value** property indicated the maximum value of the selected properties.

The **(LV) Low Value** property indicates the minimum value of the selected properties.

The **object_name** property stores the name of the object. This is a user definable string that can be used to help identify the object or the function it is programmed to perform.

Min/Max/Avg Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
A1	**Input Property 1** specifies the property in the object selected in **I1** to be used as the first input min/max/avg calculations.
A2	**Input Property 2** specifies the property in the object selected in **I2** to be used as the second input min/max/avg calculations.
A3	**Input Property 3** specifies the property in the object selected in **I3** to be used as the third input min/max/avg calculations.
A4	**Input Property 4** specifies the property in the object selected in **I4** to be used as the fourth input min/max/avg calculations.
AV	**Average Value** displays the arithmetic mean of the inputs selected.
HV	**High Value** displays the highest value of the inputs selected.
I1	**Input Object 1** specifies the object from which the first property to be used for min/max/avg calculations can be selected.
I2	**Input Object 2** specifies the object from which the second property to be used for min/max/avg calculations can be selected.
I3	**Input Object 3** specifies the object from which the third property to be used for min/max/avg calculations can be selected.
I4	**Input Object 4** specifies the object from which the fourth property to be used for min/max/avg calculations can be selected.
LV	**Low Value** displays the lowest value of the inputs selected.

3.26 INPUT SELECT 1-4

The Input Select objects allow you to choose one of two user-defined objects based on a predetermined selection criteria. Each Input Select object has the following properties: **object_identifier, object_name, object_type, A1, A2, CV, I1, I2, SA,** and **SC.**

To use the Input Select object, two objects are specified using the **(I1) Input Channel 1** and **(A1) Input Attribute 1** and by **(I2) Input Channel 2** and **(A2) Input Attribute 2** properties. One of those two values will be selected based on the value of a selection criteria specified by the user.

The **(SC) Selection Channel** and **(SA) Selection Attribute** properties are used to pick the property that will be used as a selection criteria. If the chosen property has a value of 0, the property specified by **I1** and **A1** will be selected. If the chosen property has a value of 1, the property specified by **I2** and **A2** will be selected.

The value of the property that is selected will be stored in the **(CV) Current Value** property.

The Input Select object has a user definable **object_name** property that can be set to help identify the object or the function it is performing.

Input Select Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
A1	**Input property 1** specifies the property associated with the first input object.
A2	**Input property 2** specifies the property associated with the second input object.
CV	**Current Value** indicates the value of the property which has been selected.
I1	**Input object 1** specifies the object from which the first input property will be chosen.
I2	**Input object 2** specifies the object from which the second input property will be chosen.
SA	**Selection attribute** specifies the property to be used as the selection criteria. If the specified property has a value of 0, **CV** will take the value of the property specified in **I1** and **A1**. If the specified property has a value of 1, **CV** will take the value of the property specified in **I2** and **A2**. In either case the value will be written to the property specified by **OA** and **OC**.
SC	**Selection Channel** specifies the object from which the property used for selection will be chosen.

3.27 BROADCAST 0-7

The Broadcast objects of the *NB*-GPC3 allow the controller to send and receive values over the network. Each of the Broadcast objects contain eight properties: **object_identifier, object_name, object_type, BE, BZ, CE, CV, IA, IC,** and **RB.**

The **(IC) Input Channel** and **(IA) Input Attribute** properties are used to specify the property that is to be broadcast over the network. If you wish to use a property not listed, for example a user defined property from an SPL program, you can enter the two-letter code into **IA**. Any two letter property name may be entered in this way.

NOTE
To clear **IC** and remove the association with its target object, you must write a value of 0000 (four zeroes) to **IC**.

Property **(CV) Current Value** indicates the value of the property specified in **IC** and **IA,** if they are set. This is the value that will be broadcast by the object if broadcasting is enabled. If **RB** is set to receive broadcasts, then **CV** will display the received value.

The **(BE) Broadcast Enable** property specifies whether the controller should broadcast the value of **CV** over the network. If **BE**=1, the controller will broadcast the value of **CV**. If **BE**=0, no value will be broadcast.

The **(RB) Receive Broadcast?** property specifies whether the controller should accept broadcasted values for **CV**. If **RB**=1, then the *NB*-GPC1 will receive broadcast values for **CV**.

NOTE
If **RB** is set to receive broadcasts, then you should clear **IC** and **IA**. If **IC** and **IA** are set, then that value will take precedence over the received value.

The **(BZ) Broadcast Zone/Global** property specifies whether the information broadcast will be sent to the entire network or just to units having the same zone number as the *NB*-GPC3. If **BZ**=1, the information will be broadcast globally. If **BZ**=0, the value of **CV** will only be sent to controllers having the same zone number a the *NB*-GPC3. In this case, only those controllers that are capable of receiving broadcasts on the same object will be able to use the information.

Property **object_name** stores the name of the object. This is a user definable string that can be used to help identify the object or, for example, the value the object is set to broadcast.

Broadcast 2 can also be configured to calculate and broadcast the outside air enthalpy. The **(CV) Current Value** property of Broadcast 2 may also be calculated from the **(CV) Current Value** properties of Broadcast 0 and Broadcast 1.

NOTE
The outside air temperature and outside air humidity, as displayed in the **(CV) Current Value** properties of Broadcast 0 and Broadcast 1 respectively, must be configured to display a value, whether from a measured input or received as a broadcast from the network, before the enthalpy can be calculated. The outside air temperature must be stored in the **(CV) Current Value** property of Broadcast 0 and the outside air humidity must be stored in the **(CV) Current Value** property of Broadcast 1.

The enthalpy calculation is enabled by setting the **(CE) Enthalpy Calculation Enable** property to “Enabled” after configuring the **(CV) Current Value** properties of Broadcast 0 and Broadcast 1 as noted above.

Broadcast 5 is intended to broadcast schedules exclusively. Because of this, the **CV** for Broadcast 5 is an unsigned integer rather than a floating point value, as is the case for the other Broadcast objects. **IA** for Broadcast 5 is fixed as “**CV**” and cannot be changed. **IC** may only be set to Schedule 1-8, or the Schedule Summary object.

Broadcast Properties

Property	Description
object_identifier	a numeric code that is used to identify the object.
object_name	represents a name for the object that is unique internetwork-wide.
object_type	indicates membership in a particular object type class.
BE	**Broadcast Enable** specifies whether the controller should broadcast the value of CV over the network. 0=No 1=Yes
BZ	**Broadcast Zone/Global** specifies whether the controller's broadcasts are sent to the zone or broadcast globally. 0=Zone broadcast 1=Global broadcast
CE	**Enthalpy Calculation Enable** specifies whether Broadcast 2 should calculate the enthallpy based on the outside air temperature and humidity readings in Broadcast 0 and Broadcast 1.
CV	**Current Value** indicates the current value of the referenced object specified in **IC** and **IA**.
IA	**Input property** specifies the property associated with the object specified in **IC** to be broadcast.
IC	**Input object** specifies the input object to be broadcast.
RB	**Receive Broadcast?** specifies that the controller should accept broadcasted values for **CV**.

SECTION 4: SPL SUPPORT

This section gives an overview of the SPL programming language and describes the features of SPL specific to the NB-GPC3.

IN THIS SECTION

4.1 SPL: AN OVERVIEW

4.1.1 WHAT IS SPL?

The SAGE Programming Language (SPL) is a powerful, rich control programming language. It is a BASIC-like programming language that has been adapted for real-time control applications. SPL's features include custom report generation, data logging, manipulation of database objects, floating point math support, trigonometric functions, mixed-mode math support, programmable attributes and registers, six-level expression nesting, and other features.

SPL allows you to create simple or complex, customized control programs using a rich set of features that support all types of real-time control and monitoring applications. The source code for these control programs are created offline on a personal computer using a standard text editor. The source code is translated into program code that is downned to the *NB*-GPC3.

4.1.2 SPL AND THE *NB*-GPC3

SPL can be used to create programs that can be used by the *NB*-GPC3. A subset of all SPL features are available for creating an SPL program that will reside in a *NB*-GPC3. A complete list of the SPL statements that are supported by the *NB*-GPC3 are listed later in this section.

The *NB*-GPC3 has a number of special purpose objects that can be used to control various inputs and outputs. There are also a number of objects specifically designed to efficiently accomplish tasks which would normally require SPL. These objects should be used whenever possible. The goal of any SPL program should be to write the least amount of code needed to accomplish the task at hand. This is done by utilizing as much of the built-in functionality of the *NB*-GPC3 as possible.

For customized applications, those that cannot be accomplished using a combination of the objects and properties in the controller, the *NB*-GPC3 offers the ability to accept SPL programs into a portion of its memory. These RAM-resident, customized programs can be used to enhance existing control applications or to create new applications.

4.1.3 CREATING AN SPL PROGRAM FOR AN *NB*-GPC3

There are several steps involved in creating an SPL program and downloading it to the *NB*-GPC3. There are several tools that you should have at your disposal to accomplish this. These include:

- an ASCII text editor (such as WordPad)
- a Windows-based personal computer
- a connection to the *NB*-GPC3 via an *NB*-Link interface device

SPL program creation begins by entering source code into a source file This source code is a logical and sequential list of SPL program statements (and compiler directives) that defines a desired process. This process may begin as a simple flowchart or pseudocode for the desired application.

Next, the source file is translated into program code that can be interpreted by the *NB*-GPC3. This is called compiling and is accomplished using the SPL Editor, included with the *NB*-Pro software package. The editor may either be selected from the Windows *Start* menu and run as a stand-alone application or it can be called from *NB*-Pro.

After the source code has been successfully compiled, the resulting program code can be used by the *NB*-GPC3. The compiled program code is downloaded to the *NB*-GPC3 using *NB*-Pro.

A brief summary of the SPL program development process is illustrated in Figure 5-2. For a complete discussion of writing SPL, refer to the SPL User Manual (refer to part number 1E-04-00-0082).

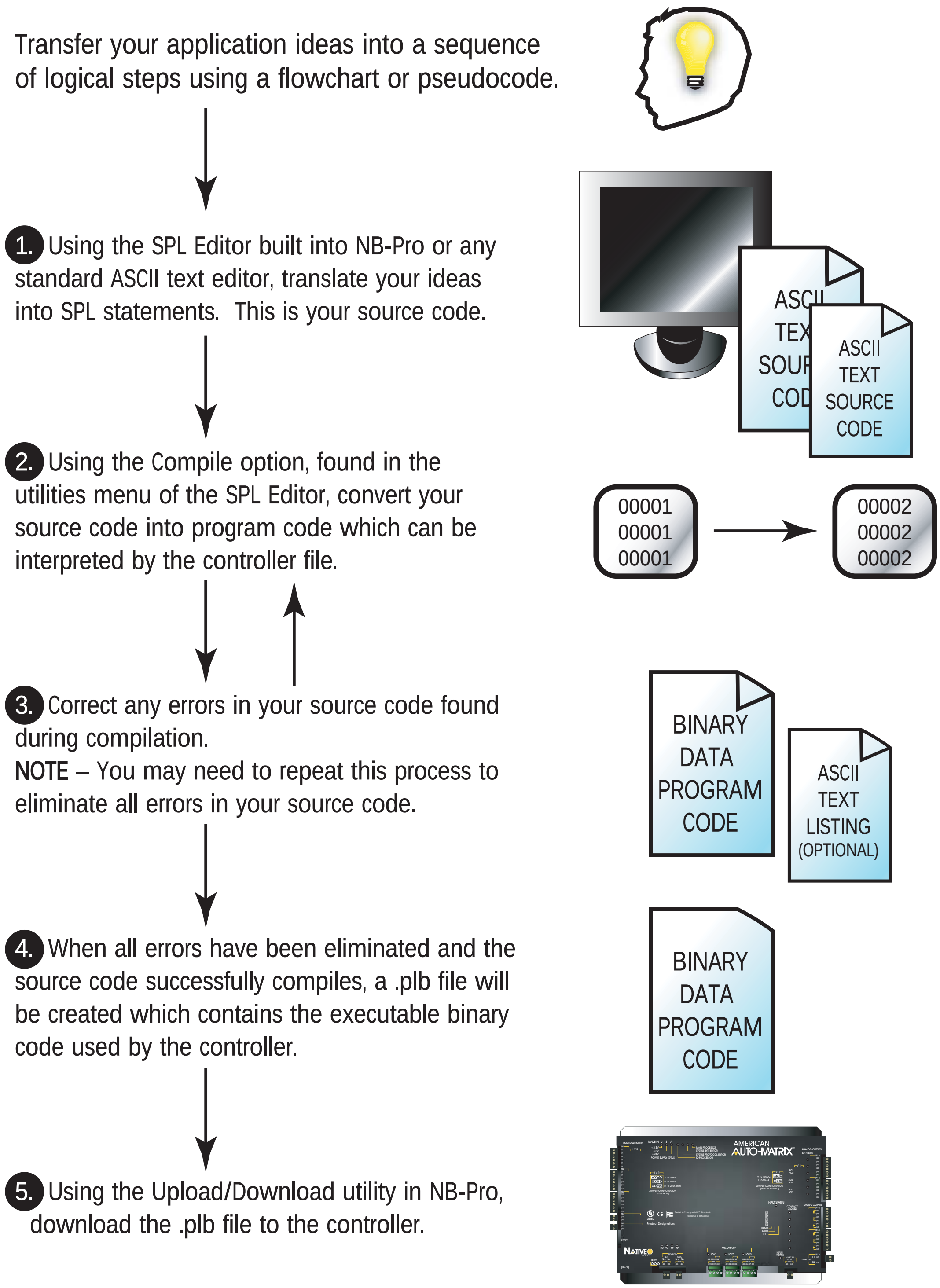

Figure 4-1 SPL Program Development Summary

4.2 SPL FEATURES SUMMARY

This section is intended to be a brief overview of the features (i.e., compiler directives, statements, terms/ functions and operators) that are available for creating SPL programs to be downloaded into a *NB*-GPC3. Refer to the SPL User Manual for more information about SPL and unitary controllers.

When you write your SPL program for the *NB*-GPC3, you must include a special SPL compiler directive at the beginning of the source code. The #GPC compiler directive should appear at the beginning (typically the first line) of any SPL program that has the *NB*-GPC3 as its target platform. This alerts the SPL compiler to accept only those SPL statements and features that are supported by the *NB*-GPC3.

The #ONESEC and #ENDONESEC compiler directives may also be used in any SPL program that has the *NB*-GPC3 as its target platform. These directives are used to define the portion of the program that is the once-per-second subroutine (that portion of code which is enclosed by the two directives). The once-per-second routine is executed one time every second to provide a precise time-based mechanism for executing portions of code at regular intervals. This routine is called automatically by the *NB*-GPC3 and cannot be called from the main program. The #ENDONESEC compiler directives may be eliminated if the once-per-second routine extends to the end of the program logic block (PLB).

Other compiler directives (not specific to *NB*-GPC3) are listed in Table 4-1.

Table 4-1 lists the SPL statements that are supported by the *NB*-GPC3. A brief description of each statement is included. Table 4-2 lists the SPL terms and functions that are supported by the *NB*-GPC3. A brief description of each is included. Table 4-3 lists the SPL expression operators that are supported by the *NB*-GPC3. A brief description of each is included.

For a complete description of SPL and the utilities used to write and compile SPL programs, refer to the SPL User Manual.

Table 4-1 Program Statements

Format	Decription
variable=expression	assignment statement
ALARM *classexpr, "formatstring", x,x,x...x,x,x*	generates an alarm
BACNET(*dexpr,obexpr,pexpr,iexpr*)	read or write a BACnet property
CALL *PLBname*	go to external subprogram's logic block then load, execute and possibly unload it
DATA *v1,v2,v3,v4....*	initialize the individual elements of a ram-based table
DREF *unit,channel;attr*	adds entries to the attribute table at the end of the PLB
ERRORABORT	trap condition - abort on errors
ERRORWAIT	trap condition - wait until no error
GOSUB *label*	go to internal subroutine
GOTO *label*	unconditional branch
IF *expression* THEN *label*	conditional branch if expr is true
IF *expression* THEN *label1* ELSE *label2*	conditional branches if expr is true or false
LOOP *register,label*	iteration control
MWAIT *expression*	wait a certain amount of minutes
NOP	no operation used for debugging
ON *expression* GOTO *label0,label1...labeln*	indexed conditional branches

Table 4-1 Program Statements

Format	Decription
ONERROR *label*	trap condition - branch if error occurs
PROP *propertyidentifier,dataytpe*	define a property
RETURN	return from a subroutine
SECTION *number*	section marker used for debugging
SWAIT *expression*	wait a certain amount of seconds
WAIT (*expression*)	wait until an expression is true, then go on

Table 4-2 SPL Functions

Function	Description
ABS(x)	Absolute value
ARCTAN(x)	Arctangent value
BETWEEN(tx,tx)	Between two times
COS(x)	Cosine value
EXP(x)	e^x
FIX(fx)	Convert float to fixed
FLOAT(ix)	Convert integer to float
INT(x)	Convert to integer
LN(x)	Natural logarithm
LOG(x)	Logarithm
MAX(x1,x2,..x8)	Maximum value from list
MEAN(x1,x2,..x8)	Mean value from list
MIN(x1,x2,..x8)	Minimum value from list
OID(*type,expr*)	Convert type/instance to objectid
RETYPE(x1,x2)	Convert to a specified datatype
ROUND(x)	Round off
SIN(x)	Sine value
SQRT(x)	Square root
TAN(x)	Tangent value
TODAY(ix)	Compare today's day of week

Table 4-3 Expression Operators

Operator	Description
-	Unary negation
NOT	Unary ones complement
**	Exponential
*	Multiplication
/	Division
MOD	Remainder after division
+	Addition
-	Subtraction
==	Equality
<>	Inequality
>	Greater than
>=	Greater than or equal
<	Less than
<=	Less than or equal
AND	Bitwise And
OR	Bitwise Or
XOR	Bitwise Exclusive-Or
SHL	Bitwise Shift Left
SHR	Bitwise Shift Right

Appendix A: Objects & Properties

The following tables contain listings of the BACnet objects and property assignments for the NB-GPC3. Each property is listed with its identifier number, data type, access code, storage, default value (if any) and a brief description of its functionality.

In this section

A.1 DEVICE

NOTE

The Device object is represented in *NB*-Pro as follows:
AAM GPC xxxxxxxxxx
(where xxxxxxxxxx is the Unitary Controller serial number)

The instance must be a unique number from 0 to 4194302. By default, AAM sets the value in such a way that it is unique to AAM products based off the unit's serial number, however the user must ensure the device instance is unique on the job site's network.

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_ identifier	75	BACnet ObjID	RW	EEPROM Device (8), Instance *serial number*	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM AAM NB-GPC *serial number*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Device (8)	indicates membership in a particular object type class.
system_status	112	BACnet ObjID	RO	- 0	indicates the current physical and logical status of the BACnet Device.
vendor_name	121	CharStr	RO	NRAM American Auto-Matrix	identifies the manufacturer of the BACnet Device.
vendor_ identifier	120	Unsigned	RO	- 6	a unique vendor identification code, assigned by ASHRAE, which is used to distinguish proprietary extensions to the protocol.
model_name	70	CharStr	RO	NRAM NB-GPC3	indicates the vendor's name used to represent the model of the device.
firmware_ revision	44	CharStr	RO	NRAM *revision number*	indicates the level of firmware installed in the device.
application_ software_ version	12	CharStr	RO	NRAM *version number*	identifies the version of application software installed in the device.
protocol_ version	98	Unsigned	RO	- 1	indicates the version of the BACnet protocol supported by this BACnet Device.
protocol_ revision	139	Unsigned	RO	- 2	indicates the minor revision level of the BACnet standard.
protocol_ services_ supported	97	BACnet Services Supported	RO	-	indicates which standardized protocol services are supported by this device's protocol implementation.
protocol_ object_ types_ supported	96	BACnet Object Types Supported	RO	-	indicates which standardized object types are supported by this device's protocol implementation.

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_list	76	BACnet Array	RO	-	a list of each object within the device that is accessible through BACnet services.
max_apdu_ length_ accepted	62	Unsigned	RO	NRAM 480	specifies the maximum number of information frames the node may send before it must pass the token.
segmentation_ supported	107	BACnet Segment ation	RO	- 3	indicates whether the device supports segmentation of messages and, if so, whether it supports segmented transmission, reception, or both.
local_time	57	Time	RW	-	indicates the time of day to the best of the device's knowledge.
local_date	56	Date	RW	-	indicates the date to the best of the device's knowledge.
utc_offset	119	Integer	RW	NRAM -300	indicates the number of minutes (-780 to +780) offset between local standard time and Universal Time Coordinated.
daylight_ savings_ status	24	Boolean	RO	NRAM 0	indicates whether daylight savings time is in effect or not.
apdu_timeout	11	Unsigned	RW	NRAM 3000	indicates the amount of time, in milliseconds, between retransmissions of an APDU requiring acknowledgment for which no acknowledgment has been received.
number_of_ apdu_retries	73	Unsigned	RW	NRAM 1	indicates the maximum number of times that an APDU shall be retransmitted.
time_ synchronization _recipients	116	List of BACnet recipients	RW	NRAM {}	a list of one device to which the device may automatically send a TimeSynchronization request.
max_master	64	Unsigned	RW	EEPROM 127	specifies the highest possible address for master nodes and shall be less than or equal to 127.
max_info_ frames	63	Unsigned	RW	NRAM 10	specifies the maximum number of information frames the node may send before it must pass the token.
device_ address_ binding	30	List	RW	- 30	a list of the device addresses that will be used when the remote device must be accessed via a BACnet service request.
BF	49734	Unsigned	RW	NRAM 0	**Broadcast Fire?** specifies whether the “change to fire mode” command should be broadcast 0=No 1=Yes
BS	49747	Unsigned	RW	NRAM 0	**Broadcast Sync Time?** specifies whether to broadcast time synchronization information 0=No 1=Yes
BT	49748	Unsigned	RW	NRAM 255	**Broadcast Time Interval** specifies the time (in minutes) between network broadcasts 1-254=time in minutes 255=disabled
CF	49990	Unsigned	RW	NRAM 0	**Communication Failure Timeout** specifies the amount of time, in seconds, that the controller will wait to be polled before entering a communication fault state.

Property	Identifier #	Data Type	Access	Storage & Default	Description
CM	49997	Unsigned	RO	RAM 255	**Manufacturer** indicates the factory-set manufacturer number for the controller. (**CM** for American Auto-Matrix controllers is always 255)
CP	50000	Unsigned	RW	NRAM 6	**Communication Speed** the rate at which the controller communicates. 0=9600 6=38400 (default) 7=19200 8=115.2k 9=57.6k
CR	50002	Unsigned	RW	NRAM 0	**Configure Remote I/O** specifies options for the configuration of remote I/O device located on the STATbus. 0=Normal 1=GPC to bus 2=Edit I/O GIDs 3=GPC from bus
CT	50004	Unsigned	RO	RAM 105	**Controller Type** factory-set controller type number for the controller. **CT** for the *NB-GPC3* is 205.
DA	50241	Unsigned	RW	RAM	**Day of Week** specifies the current day of the week. 0=Monday 1=Tuesday 2=Wednesday 3=Thursday 4=Friday 5=Saturday 6=Sunday
DD	50244	Unsigned	RW	NRAM	**Device Instance To Send Alarms To** specifies the MSTP address or device instance to which alarm messages will be sent when IA is set to “3=Unconfirmed to Device” or “4=Confirmed to Device”.
DE	50245	Unsigned	RW	RAM 0	**Default Enable** used to return all properties in the controller to their default values. 0=Normal operation 197=set properties to their default values.
EM	50509	Boolean	RW	NRAM 0	**Engineering Units** specifies the units to be used when returning values 0=English 1=Metric
ET	50516	Time	RW	NRAM	**Daylight Saving End Time** specifies the time at which daylight saving time ends.
FA	50753	BitStr	RO	NRAM	**Faults Detected** a bitmap indicating the current fault conditions. bit 0=Fire bit 1=Emergency bit 6=Communications failure 0=Normal 1=Fault condition
FC	50755	Unsigned	RO	NRAM	**Flash update count** indicates the number of times the controller has been flashed.

Property	Identifier #	Data Type	Access	Storage & Default	Description
FT	50772	Unsigned	RO	RAM	**Firmware Type** indicates which firmware is installed on the controller 7=GPC1 8=GPC2 9=GPC3
IA	51521	Enum	RW	NRAM	**Intrinsic Alarming** determines what alarm messages are generated and how they are routed. 0=None 1=Track Alarms 2=Broadcast 3=Unconfirmed to Device 4=Confirmed to Device
ID	51524	Unsigned	RW	NRAM	**Unit Number (ID)** specifies the controller's identification number. The value of ID defaults to the last two digits of the unit's serial number.
ND	50500	Unsigned	RW	NRAM	**Daylight Saving End Day** specifies the day on which daylight savings time ends 0 = None 1 = First Sunday 2 = First Friday 3 = Second Saturday 4 = Third Sunday 5 = Third Saturday 6 = Last Sunday 7 = Last Thursday 8 = Last Friday
NM	52813	Unsigned	RW	NRAM	**Daylight Saving End Month** specifies the month in which daylight savings tim ends 0 = None 1 = January 2 = February 3 = March, 4 = April 5 = May 6 = June 7 = July 8 = August 9 = September 10 = October 11 = November 12 = December
OS	53075	Real	RO	RAM	**Kernel Version** indicates the version number of the kernel.
PD	53316	Unsigned	RW	NRAM 15	**Power-Up Delay** time delay (in seconds) that must elapse after the controller is reset before it begins control and alarming functions. 0=No delay 1-255=# of seconds
PF	53318	Unsigned	RW	NRAM 3	**Priority for Fault Alarms** specifies where in the priority array to-fault alarms will be listed.
PI	53321	Unsigned	RW	NRAM	**Process ID for Alarms**
PN	53326	Unsigned	RW	NRAM 4	**Priority for Normal Alarms** specifies where in the priority array to-normal alarms will be listed.

Property	Identifier #	Data Type	Access	Storage & Default	Description
PO	53327	Unsigned	RW	NRAM 2	**Priority for Off-normal Alarms** specifies where in the priority array to-offnormal alarms will be listed.
PS	53331	Unsigned	RW	NRAM 0	**Power-up state** schedule state that the controller will operate in when it is first powered up or after power is restored following a power failure. 0=unoccupied 1=warmup 2=occupied 3=night setback
RD	53828	Unsigned	RW	NRAM	**Daylight Savings Start Day** specifies the day on which daylight saving time begins 0 = None 1 = First Sunday 2 = First Friday 3 = Second Saturday 4 = Third Sunday 5 = Third Saturday 6 = Last Sunday 7 = Last Thursday 8 = Last Friday
RM	53837	Unsigned	RW	NRAM	**Daylight Savings Start Month** specifies the month in which daylight savings time begins 0 = None 1 = January 2 = February 3 = March, 4 = April 5 = May 6 = June 7 = July 8 = August 9 = September 10 = October 11 = November 12 = December
RS	53843	Unsigned	RW	RAM 0	**Reset the Controller?** used to reset the controller. Setting **RS** to 1 resets the controller. 0=No 1=Yes
SN	54094	Unsigned	RO	NRAM	**Serial Number** the factory-set serial number.
SR	54098	Unsigned	RO	RAM	**Flash Release Code** the release code of the firmware currently flashed on the controller, used primarily for technical support purposes.
ST	54100	Time	RW	NRAM	**Daylight Saving Start Time** specifies the time at which daylight savings time begins.
TF	54342	Unsigned	RO	NRAM 0	**Time to Remain in Fire Mode** The time (in minutes) that the controller will remain in fire mode.
UD	54596	Boolean	RW	NRAM	**Use Device** specifies whether the address specified in **DD** to which alarm messages will be sent is an MSTP address of a device instance. 0=MSTP Address 1=Device Instance

Property	Identifier #	Data Type	Access	Storage & Default	Description
VE	54853	Real	RO	RAM	**Firmware Version** contains the version of the controller's firmware.
ZN	55886	Unsigned	RW	NRAM 0	**Zone Number** specifies the zone number for the controller. All zone broadcasts will only be sent to controllers with matching zone numbers.

A.2 NOTIFICATIONCLASS1

Property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- Notification Class (15), Instance (1)	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM NOTIFICATIONCLASS1	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Notification Class (15)	indicates membership in a particular object type class.
notification_ class	17	Unsigned	RO	- 1	specifies the notification class to be used when handling and generating event notifications for this object.
priority	86	BACnet Array	RW	NRAM 2	specifies the priority to be used for event notifications for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively.
ack_required	1	BACnet Event Trans. Bits	RW	NRAM 1, 0, 1	three separate flags that indicate whether acknowledgment shall be required in notifications generated for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL event transitions, respectively.
recipient_list	102	BACnet List	RO		a list of one or more recipient destinations to which notifications shall be sent when event-initiating objects using this class detect the occurrence of an event.

A.3 PROGRAMS 1-8

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Program (16), Instance 1-8	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Program *N* (unloaded)	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Program (16)	indicates membership in a particular object type class.
program_state	92	BACnet Program State	RO	RAM 0	indicates the current logical state of the process executing the application program this object represents.
program_change	90	BACnet Program Request	RW	RAM 0	used to request changes to the operating state of the process this object represents.
reason_for_halt	100	BACnet Program Error	RO	RAM 0	indicates the reason why the program was halted.
description_of_ halt	29	CharStr	RO	RAM	describes the reason why a program has been halted.
program_ location	91	CharStr	RO	RAM Sec0:0x0000	indicate the current location within the program code, for example, a line number or program label or section name.
status_flags	111	BACnet Status Flags	RO	NRAM 0	four flags that indicate the general "health" of the program.
reliability	103	BACnet Reliability	RO	RAM 0	indicates whether the program is running/waiting (no fault detected) or is unreliable (process-error)
out_of_service	81	Boolean	RO	- 0	indicates whether or not the process this object represents is not in service.
$1	42033	Boolean	RW	NRAM 0	**Enable Single-Step Mode?** specifies whether the single-step, line by line debugging mode is enabled. 0=No 1=Yes
$D	42052	Unsigned	RW	NRAM 0	**Delay Time Remaining** specifies the number of seconds remaining when an **SWAIT** or **MWAIT** statement is encountered in the SPL program.
$E	42053	Unsigned	RW	NRAM 0	**Error Code** indicates the SPL error code that is returned when the program aborts.
$N	46062	Unsigned	RW	NRAM 0	**Number of Program Attributes**
$W	42071	Unsigned	RW	NRAM 0	**Trappable Error Action** specifies how the SPL program should handle trappable errors. 0=Abort on Error 1=Wait on Error

Property	Identifier #	Data Type	Access	Storage & Default	Description
%A	42305	Unsigned	RW	NRAM 0	**Register A Value** indicates the value of program register A.
%B	42306	Unsigned	RW	NRAM 0	**Register B Value** indicates the value of program register B.
%C	42307	Unsigned	RW	NRAM 0	**Register C Value** indicates the value of program register C.
%D	42308	Unsigned	RW	NRAM 0	**Register D Value** indicates the value of program register A.
%E	42309	Unsigned	RW	NRAM 0	**Register E Value** indicates the value of program register E.
%F	42310	Unsigned	RW	NRAM 0	**Register F Value** indicates the value of program register F.
%G	42311	Unsigned	RW	NRAM 0	**Register G Value** indicates the value of program register G.
%H	42312	Unsigned	RW	NRAM 0	**Register H Value** indicates the value of program register H.
%I	42313	Unsigned	RW	NRAM 0	**Register I Value** indicates the value of program register I.
%J	42314	Unsigned	RW	NRAM 0	**Register J Value** indicates the value of program register J.
%K	42315	Unsigned	RW	NRAM 0	**Register K Value** indicates the value of program register K.
%L	42316	Unsigned	RW	NRAM 0	**Register L Value** indicates the value of program register L.
%M	42317	Unsigned	RW	NRAM 0	**Register M Value** indicates the value of program register M.
%N	42318	Unsigned	RW	NRAM 0	**Register N Value** indicates the value of program register N.
%O	42319	Unsigned	RW	NRAM 0	**Register O Value** indicates the value of program register O.
%P	42320	Unsigned	RW	NRAM 0	**Register P Value** indicates the value of program register P.

A.4 FILE0

Property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- File (10), Instance 0	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM FILE0	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- File (10)	indicates membership in a particular object type class.
description	28	CharStr	RO	- Flash Upgrade	provides a description of the object. This object is intended for flash updates.
file_type	43	CharStr	RO	- Flash Upgrade	identifies the intended use of this file.
file_size	42	Unsigned	RO	NRAM 327680	indicates the size of the file data.
modification_ date	71	BACnet Date Time	RO	NRAM NULL	indicates the last time this object was modified.
archive	13	Boolean	RW	NRAM 1	indicates whether the File object has been saved for historical or backup purposes.
read_only	99	Boolean	RO	- 0	indicates whether or not the file data may be changed through the use of a BACnet AtomicWriteFile service.
file_access_ method	41	BACnet File Access Method	RO	- 1	indicates the type(s) of file access supported for this object.

A.5 PLB1-8

Property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- File (10), Instance 1-8	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM RAM*N*, PLB*N*, or LOGO*N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- File (10)	indicates membership in a particular object type class.
description	28	CharStr	RO	- Flash Upgrade	provides a description of the object. This object is intended for flash updates. The **description** is the same as the **file_type**.
file_type	43	CharStr	RO	- RAM*N*	identifies the intended use of this file. The possible file types are: Empty Region *n* System File *n* Trend File *n* Table File *n* Program Logic Block *n* Program Reference Block *n* Program Control Block *n* Display List *n* Custom Logo *n*
file_size	42	Unsigned	RO	NRAM -	indicates the size of the file data.
modification_ date	71	BACnet Date Time	RO	NRAM NULL	indicates the last time this object was modified.
archive	13	Boolean	RW	NRAM 0	indicates whether the File object has been saved for historical or backup purposes.
read_only	99	Boolean	RO	- 0	indicates whether or not the file data may be changed through the use of a BACnet AtomicWriteFile service.
file_access_ method	41	BACnet File Access Method	RO	- 1	indicates the type(s) of file access supported for this object.

A.6 UNIVERSAL INPUTS 1-24

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Analog Input (3), Instance *N* or Binary Input (3), Instance *N*	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Universal Input *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Analog Input (3) or Binary Input (3)	indicates membership in a particular object type class.
present_value	85	Real	RW	RAM 0	indicates the current value, in engineering units, of the object.
status_flags	111	BACnet Status Flags	RO	RAM 0	four flags that indicate the general "health" of the program.
event_state	36	BACnet Event State	RO	RAM 0	provides a way to determine if this object has an active event state associated with it.
reliability	103	BACnet Reliability	RO	RAM 0	indicates whether the **present_value** is "reliable" as far as the device or operator can determine.
out_of_service	81	Boolean	RW	NRAM 0	indicates whether or not the process this object represents is not in service.
polarity	84	BACnet Polarity	RW	NRAM 0	indicates the relationship between the physical state of the output and the logical state represented by the **present_value**e property. If the **polarity** property is NORMAL, then the ACTIVE state of the **present_value** property is also the ACTIVE or ON state of the physical output as long as **out_of_service** is FALSE. If the Polarity property is REVERSE, then the ACTIVE state of the **present_value** property is the INACTIVE or OFF state of the physical output as long as **out_of_service** is FALSE.
units	117	BACnet Eng. Units	RW	NRAM 95	indicates the measurement units of this object.
min_pres_value	69	Real	RW	NRAM	indicates the lowest number that can be reliably used for the **present_value** property of this object.
max_pres_value	65	Real	RW	NRAM	indicates the highest number that can be reliably used for the **present_value** property of this object.
resolution	106	Real	RO	- 0.001525	indicates the smallest recognizable change in **present_value** in engineering units (read-only).
time_delay	113	Unsigned	RW	NRAM 0	specifies the minimum period of time in seconds during which the **present_value** must be different from the **alarm_value** property before a TO-OFFNORMAL event is generated or must remain equal to the **alarm_value** property before a TO-NORMAL event is generated.
notification_ class	17	Unsigned	RO	- 1	specifies the notification class to be used when handling and generating event notifications for this object.
high_limit	45	Real	RW	NRAM 0.0	specifies a limit that the **present_value** must exceed before an event is generated.

Property	Identifier #	Data Type	Access	Storage & Default	Description
low_limit	59	Real	RW	NRAM 0.0	specifies a limit below which the **present_value** must fall before an event is generated.
deadband	25	Real	RW	NRAM 0.0	specifies a range between the **high_limit** and **low_limit** properties within which the **present_value** must remain for a TO-NORMAL event to be generated
limit_enable	52	BACnet Limit Enable	RW	NRAM 0	enables and disables reporting of HighLimit and LowLimit offnormal events and their return to normal.
alarm_value	6	BACnet BinaryPV	-	-	specifies the value that the Present_Value property must have before a TO-OFFNORMAL event is generated.
event_enable	35	BACnet Event Trans. Bits	RW	NRAM 0	three flags that separately enable and disable reporting of TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.
acked_ transitions	0	BACnet Event Trans. Bits	RW	NRAM 7	three flags that separately indicate the receipt of acknowledgments for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.
notify_type	72	BACnet Notify Type	RW	NRAM 0	specifies whether the notifications generated by the object should be Events or Alarms.
AE	49477	Unsigned	RW	NRAM 0	**Alarm Enable** specifies the type of alarming to be used by the input. 0=Disabled 1=Contact 0>1 2=Contact 1>0 3=Change of State 4=Low Limit 5=High Limit 6=Low & Hi Limit 8=Fire 11=Supervise Digital Output 1 12=Supervise Digital Output 2 13=Supervise Digital Output 3 14=Supervise Digital Output 4 15=Supervise Digital Output 5 16=Supervise Digital Output 6 17=Supervise Digital Output 7 18=Supervise Digital Output 8 19=Supervise Digital Output 9 20=Supervise Digital Output 10 21=Supervise Digital Output 11 22=Supervise Digital Output 12
DF	50246	Unsigned	RW	NRAM 0	**Display Format** specifies the way in which the stat will display the temperature. 0=##d 1=##.#d 2=##dF 3=##.#dF 4=None
GI	51017	Unsigned	RW	NRAM 0	**GID of I/O Device** indicates the global identification number of the STATbus device associated with the universal input. If the input does not have a STATbus device mapped to it, **GI** will be 0.

Property	Identifier #	Data Type	Access	Storage & Default	Description
IF	51526	Real	RW	NRAM 0	**Input Filter Delay or Weighted Gain** the debounce time (in seconds) during which the input must remain stable to avoid the signal being read as a digital bounce. In the case of a bounce, the object **reliability** is set to 1. For analog inputs, **IF** specifies a weighted gain used to smooth the values from a fluctuating input.
OF	53062	Real	RW	NRAM	**Input Offset** specifies an offset amount to be added to the current value.
RH	53832	Real	RW	RAM 0	**Run Hours** indicates the number of hours **present_value**=1 for the input.
RL	53836	Real	RW	NRAM 0	**Run Limit** specifies a number of hours that **present_value**=1 after which a run limit alarm is generated. Setting **RL**=0.0 disabled run limit alarms for the input.
SM	54093	BitStr	RW	NRAM 0	**Schedules to Follow** enables scheduled alarm controlling for the associated universal input by selecting one or more of the available schedule control objects. 0=Schedule disabled 1=Schedule enabled **SM** is a bitmap with bit 0=Schedule 1 bit 1=Schedule 2 bit 2=Schedule 3 bit 3=Schedule 4 bit 4=Schedule 5 bit 5=Schedule 6 bit 6=Schedule 7 bit 7=Schedule 8 bit 8= SMARTStat 1 bit 9= SMARTStat 2 bit 10= SMARTStat 3 bit 11= SMARTStat 4 bit 12= SMARTStat 5 bit 13= SMARTStat 6 bit 14= SMARTStat 7 bit 15= SMARTStat 8 bit 16= SMARTStat 9 bit 17= SMARTStat 10 bit 18= SMARTStat 11 bit 19= SMARTStat 12 bit 20= Host Schedule bit 21= Schedule Summary bit 22= Occupancy
ST	54100	Unsigned	RW	NRAM 0	**Sensor Type** specifies the type of sensor connected to the input. 0=Digital 2=MN..MX 0 to 5V 3=MN..MX 4 to 20mA 4=Curve 1 5=Curve 2 6=MN..MX 0 to 10V 7=Thermistor -30.0 to 230.0 8=MN..MX 0 to 20mA 9=SMARTStat Temperature 10=SMARTStat Humidity
SU	54101	Real	RW	NRAM 0	**Amount to Setup/Setback Alarm Limit** specifies a value (0.0 to 25.5) which is added to **high_limit** and subtracted from **low_limit** during scheduled unoccupied periods.

Property	Identifier #	Data Type	Access	Storage & Default	Description
TM	54349	Real	RW	NRAM 1	**Thermostat Multiplier** specifies the magnitude of the user setpoint offset. Each press of the up or down buttons on a stat will increment the setpoint by an amount **TM**. There are five steps above and below the setpoint, providing for a maximum offset amount of 5x**TM** above or below the setpoint.

A.7 DIGITAL INPUTS 1-8

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Binary Input (3), Instance 49-56	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Digital Input *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Binary Input (3)	indicates membership in a particular object type class.
present_value	85	Real	RW	RAM 0	indicates the current value, in engineering units, of the object.
status_flags	111	BACnet Status Flags	RO	NRAM 0	four flags that indicate the general "health" of the program.
event_state	36	BACnet Event State	RO	RAM 0	provides a way to determine if this object has an active event state associated with it.
reliability	103	BACnet Reliability	RO	RAM 0	indicates whether the **present_value** is "reliable" as far as the device or operator can determine.
out_of_service	81	Boolean	RW	NRAM 0	indicates whether or not the process this object represents is not in service.
polarity	84	BACnet Polarity	RW	NRAM 0	indicates the relationship between the physical state of the output and the logical state represented by the **present_value**e property. If the **polarity** property is NORMAL, then the ACTIVE state of the **present_value** property is also the ACTIVE or ON state of the physical output as long as **out_of_service** is FALSE. If the Polarity property is REVERSE, then the ACTIVE state of the **present_value** property is the INACTIVE or OFF state of the physical output as long as **out_of_service** is FALSE.
time_delay	113	Unsigned	RW	NRAM 0	specifies the minimum period of time in seconds during which the **present_value** must be different from the **alarm_value** property before a TO-OFFNORMAL event is generated or must remain equal to the **alarm_value** property before a TO-NORMAL event is generated.
notification_ class	17	Unsigned	RO	- 1	specifies the notification class to be used when handling and generating event notifications for this object.
alarm_value	6	BACnet BinaryPV	-	-	specifies the value that the Present_Value property must have before a TO-OFFNORMAL event is generated.
event_enable	35	BACnet Event Trans. Bits	RW	NRAM 0	three flags that separately enable and disable reporting of TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.
acked_ transitions	0	BACnet Event Trans. Bits	RW	NRAM 0	three flags that separately indicate the receipt of acknowledgments for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.
notify_type	72	BACnet Notify Type	RW	NRAM 0	specifies whether the notifications generated by the object should be Events or Alarms.

Property	Identifier #	Data Type	Access	Storage & Default	Description
AE	49477	Unsigned	RW	NRAM 0	**Alarm Enable** specifies the type of alarming to be used by the input. 0=Disabled 1=Contact 0>1 2=Contact 1>0 3=Change Of State 8=Fire 11=Supervise Digital Output 1 12=Supervise Digital Output 2 13=Supervise Digital Output 3 14=Supervise Digital Output 4 15=Supervise Digital Output 5 16=Supervise Digital Output 6 17=Supervise Digital Output 7 18=Supervise Digital Output 8 19=Supervise Digital Output 9 20=Supervise Digital Output 10 21=Supervise Digital Output 11 22=Supervise Digital Output 12
GI	51017	Unsigned	RW	NRAM 0	**GID of I/O Device** indicates the global identification number of the STATbus device associated with the output. If the input does not have a STATbus device mapped to it, **GI** will be 0.
IF	51526	Real	RW	NRAM 0	**Input Filter Delay** the debounce time (in seconds) during which the input must remain stable to avoid the signal being read as a digital bounce. In the case of a bounce, the object **reliability** is set to 1.
LP	51792	Boolean	RW	NRAM	**Interlock Polarity**
MD	52548	Unsigned	RW	NRAM	**Pulse Counter Mode** defines how the *NB*-GPC3 detects a pulse. 0=Rising edge 1=Falling edge 2=Both 3=None
NP	52816	Unsigned	RW	NRAM	**Number of Pulses Accumulated** displays the current number of pulses detected on the digital input selected in **IC**.
RH	53832	Real	RW	RAM	**Run Hours** indicates the number of hours **present_value**=1 for the input.
RL	53836	Real	RW	NRAM 0	**Run Limit** specifies a number of hours that **present_value**=1after which a run limit alarm is generated. Setting **RL**=0.0 disabled run limit alarms for the input.
SF	54086	Real	RW	NRAM	**Pulse Multiplier** specifies a scaling factor (0.000 to 65.535) that is multiplied by **NP** to obtain a scaled count.
SV	54102	Real	RO	RAM	**Scaled Pulse Count** displays a scaled version of NP using the formula **SV** = **NP** x **SF**

A.8 ANALOG OUTPUTS 1-12

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Analog Output (1), Instance 1-12	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Analog Output *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Analog Output (1)	indicates membership in a particular object type class.
present_value	85	Real	RW	RAM 0.0	indicates the current value, in engineering units, of the object.
status_flags	111	BACnet Status Flags	RO	NRAM 0	four flags that indicate the general "health" of the program.
event_state	36	BACnet Event State	RO	RAM 0	provides a way to determine if this object has an active event state associated with it.
reliability	103	BACnet Reliability	RO	RAM 0	indicates whether the **present_value** is "reliable" as far as the device or operator can determine.
out_of_service	81	Boolean	RW	NRAM 0	indicates whether or not the process this object represents is not in service.
units	117	BACnet Eng. Units	RW	NRAM 95	indicates the measurement units of this object.
min_pres_value	69	Real	RW	NRAM 0.0	indicates the lowest number that can be reliably used for the **present_value** property of this object.
max_pres_value	65	Real	RW	NRAM 100.0	indicates the highest number that can be reliably used for the **present_value** property of this object.
resolution	106	Real	RO	- 0.024414	indicates the smallest recognizable change in **present_value** in engineering units (read-only).
priority_array	87	BACnet Array	RO	RAM NULL	contains prioritized commands that are in effect for this object.
relinquish_ default	104	Real	RW	NRAM 0.0	the default value to be used for the **present_value** property when all command priority values in the **priority_array** property have a NULL value.
acked_ transitions	0	BACnet Event Trans. Bits	RW	NRAM 7	three flags that separately indicate the receipt of acknowledgments for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.
CF	49990	Boolean	RW	NRAM 0	**Communication Failure Enable?** 0=No 1=Yes
FI	50761	Real	RW	NRAM 0	**Fire Position** specifies the failure position (0-100%) for the output to be used when a fire event is received.

Property	Identifier #	Data Type	Access	Storage & Default	Description
FP	50768	Real	RW	NRAM 0.0	**Interlock/Communication Failure Position** specifies the failure position (0-100%) for the output to be used in the event of an interlock or communications failure.
GI	51017	Unsigned	RW	NRAM 0	**GID of I/O Device** indicates the global identification number of the STATbus device associated with the analog output. If the output does not have a STATbus device mapped to it, **GI** will be 0.
IL	51532	BitStr	RW	NRAM 0	**Inputs for Interlocking** specifies which inputs (if any) are used as interlocks for the associated output (a value of 1 indicates that input is used for interlocking). bit 0 = UI 1 bit 1 = UI 2 bit 2 = UI 3 bit 3 = UI 4 bit 4 = UI 5 bit 5 = UI 6 bit 6 = UI 7 bit 7 = UI 8 bit 8 = UI 9 bit 9 = UI 10 bit 10 = UI 11 bit 11 = UI 12 bit 12 = UI 13 bit 13 = UI 14 bit 14 = UI 15 bit 15 = UI 16 bit 16 = UI 17 bit 17 = UI 18 bit 18 = UI 19 bit 19 = UI 20 bit 20 = UI 21 bit 21 = UI 22 bit 22 = UI 23 bit 23 = UI 24 bit 24 = DI 1 bit 25 = DI 2 bit 26 = DI 3 bit 27 = DI 4 bit 28 = DI 5 bit 29 = DI 6 bit 30 = DI 7 bit 31 = DI 8
MN	52307	Real	RW	NRAM 0	**Minimum Scaled Voltage** specifies the percentage of the total output for **present_value=min_pres_value**.
MX	51283	Real	RW	NRAM 100	**Maximum Scaled Voltage** specifies the percentage of the total output for **present_value=max_pres_value**.
OU	53077	Boolean	RO	RAM	**Actual Output Value** specifies the actual output state of the output. This may differ from the current value because of delays and other effects.
UT	54612	Real	RW	NRAM 0	**Update Threshold** specifies a threshold value by which **present_value** must change before the output is updated.

A.9 DIGITAL OUTPUTS 1-12

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Binary Output (4), Instance 1-12	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Digital Output *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Binary Output (4)	indicates membership in a particular object type class.
present_value	85	Real	RW	RAM 0	indicates the current value, in engineering units, of the object.
status_flags	111	BACnet Status Flags	RO	NRAM 0	four flags that indicate the general "health" of the program.
event_state	36	BACnet Event State	RO	RAM 0	provides a way to determine if this object has an active event state associated with it.
out_of_service	81	Boolean	RW	NRAM 0	indicates whether or not the process this object represents is not in service.
polarity	84	BACnet Polarity	RW	NRAM 0	indicates the relationship between the physical state of the output and the logical state represented by the **present_value**e property. If the **polarity** property is NORMAL, then the ACTIVE state of the **present_value** property is also the ACTIVE or ON state of the physical output as long as **out_of_service** is FALSE. If the Polarity property is REVERSE, then the ACTIVE state of the **present_value** property is the INACTIVE or OFF state of the physical output as long as **out_of_service** is FALSE.
relinquish_ default	104	Real	RW	NRAM 7	the default value to be used for the **present_value** property when all command priority values in the **priority_array** property have a NULL value.
priority_array	87	BACnet Array	RO	RAM NULL	contains prioritized commands that are in effect for this object.
minimum_off_ time	66	Unsigned	RW	NRAM 0	specifies the minimum number of seconds that the **present_value** shall remain in the INACTIVE state after a write to the **present_value** property causes that property to assume the INACTIVE state.
minimum_on_ time	67	Unsigned	RW	NRAM 0	indicates the minimum number of seconds that the **present_value** shall remain in the ACTIVE state after a write to the **present_value** property causes that property to assume the ACTIVE state.
reliability	103	BACnet Reliability	RO	RAM 0	indicates whether the **present_value** is "reliable" as far as the device or operator can determine.
GI	51017	Unsigned	RW	NRAM 0	**GID of I/O Device** indicates the global identification number of the STATbus device associated with the digital output. If the output does not have a STATbus device mapped to it, **GI** will be 0.

Property	Identifier #	Data Type	Access	Storage & Default	Description
IL	51532	BitStr	RW	NRAM 0	**Inputs for Interlocking** specifies which inputs (if any) are used as interlocks for the associated output (a value of 1 indicates that input is used for interlocking). bit 0 = UI 1 bit 1 = UI 2 bit 2 = UI 3 bit 3 = UI 4 bit 4 = UI 5 bit 5 = UI 6 bit 6 = UI 7 bit 7 = UI 8 bit 8 = UI 9 bit 9 = UI 10 bit 10 = UI 11 bit 11 = UI 12 bit 12 = UI 13 bit 13 = UI 14 bit 14 = UI 15 bit 15 = UI 16 bit 16 = UI 17 bit 17 = UI 18 bit 18 = UI 19 bit 19 = UI 20 bit 20 = UI 21 bit 21 = UI 22 bit 22 = UI 23 bit 23 = UI 24 bit 24 = DI 1 bit 25 = DI 2 bit 26 = DI 3 bit 27 = DI 4 bit 28 = DI 5 bit 29 = DI 6 bit 30 = DI 7 bit 31 = DI 8
OU	53077	Boolean	RO	RAM	**Actual Output State** specifies the actual output state of the output. This may differ from the current value because of delays and other effects.
PW	53335	Real	RW	NRAM 0	**Pulse Width when Output is On** specifies the “on” time (**present_value**=1) in seconds (0.0 to 25.5) that the output should remain on after a transition from the off to on state. 0=Disabled 0.1-25.5=pulse “on” duration in seconds
RH	53832	Real	RW	NRAM 0	**Run Hours** indicates the number of hours **present_value**=1 for the input.
RL	53836	Real	RW	NRAM 0	**Run Hours Limit** specifies a number of hours that **present_value**=1after which a run limit alarm is generated. Setting **RL**=0.0 disabled run limit alarms for the input.
SI	54089	Unsigned	RW	NRAM 0	**Power-On Stagger Interval** specifies the time, in seconds, that must elapse after the controller is turned on or reset before the output can be turned on.

Property	Identifier #	Data Type	Access	Storage & Default	Description
SM	54093	BitStr	RW	NRAM	**Schedules to Follow** enables scheduled control of the digital output by selecting one or more of the available schedule control objects. 0=schedule disabled 1=schedule enabled **SM** is a bitmap with bit 0=Schedule 1 bit 1=Schedule 2 bit 2=Schedule 3 bit 3=Schedule 4 bit 4=Schedule 5 bit 5=Schedule 6 bit 6=Schedule 7 bit 7=Schedule 8 bit 8= SMARTStat 1 bit 9= SMARTStat 2 bit 10= SMARTStat 3 bit 11= SMARTStat 4 bit 12= SMARTStat 5 bit 13= SMARTStat 6 bit 14= SMARTStat 7 bit 15= SMARTStat 8 bit 16= SMARTStat 9 bit 17= SMARTStat 10 bit 18= SMARTStat 11 bit 19= SMARTStat 12 bit 20= Host Schedule bit 21= Schedule Summary bit 22= Occupancy

A.10 STATBus 1-4

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (255), Instance *N*	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM STATbus *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (255)	indicates membership in a particular object type class.
BS	49747	Unsigned	RO	RAM 0	**Bus Status** indicates the status of the SMARTStat Bus. 0=Ok 1=Needs configured 2=Error
CD	49988	Unsigned	RW	RAM 0	**Configure Device (GID)** specifies the device to be configured.
CF	49990	Unsigned	RW	RAM 0	**Configure Function** enables the blinking LED on the device specified in **CD**. 0=Disabled 1=Blink LED
G1	50993	Unsigned	RW	NRAM 0	**GID Device 1** indicates the global identification number of device 1.
G2	50994	Unsigned	RW	NRAM 0	**GID Device 2** indicates the global identification number of device 2.
G3	50995	Unsigned	RW	NRAM 0	**GID Device 3** indicates the global identification number of device 3.
G4	50996	Unsigned	RW	NRAM 0	**GID Device 4** indicates the global identification number of device 4.
G5	50997	Unsigned	RW	NRAM 0	**GID Device 5** indicates the global identification number of device 5.
G6	50998	Unsigned	RW	NRAM 0	**GID Device 6** indicates the global identification number of device 6.
G7	50999	Unsigned	RW	NRAM 0	**GID Device 7** indicates the global identification number of device 7.
G8	51000	Unsigned	RW	NRAM 0	**GID Device 8** indicates the global identification number of device 8.
G9	51001	Unsigned	RW	NRAM 0	**GID Device 9** indicates the global identification number of device 9.
GA	51009	Unsigned	RW	NRAM 0	**GID Device 10** indicates the global identification number of device 10.
GB	510010	Unsigned	RW	NRAM 0	**GID Device 11** indicates the global identification number of device 11.
GC	51011	Unsigned	RW	NRAM 0	**GID Device 12** indicates the global identification number of device 12.

Property	Identifier #	Data Type	Access	Storage & Default	Description
GD	51012	Unsigned	RW	NRAM 0	**GID Device 13** indicates the global identification number of device 13.
SM	54093	BitStr	RO	RAM 0	**Status Map** 1=unconfigured 2=duplicate

A.11 UNIVERSAL INPUT SUMMARY

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (254), Instance 0	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Universal Input Summary	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (254)	indicates membership in a particular object type class.
AT	49492	Unsigned	RW	NRAM 0	**Alarm Limit setup/setback time delay** the time (in minutes) that must elapse after transitions from occupied to unoccupied or from unoccupied to occupied before alarm limits return to their previous (pre-setup/setback) values.
ED	50500	Unsigned	RW	NRAM 0	**Extended Occupancy Duration** specifies the time, in minutes, occupancy is extended when the stat indicates an override
ER	50514	Unsigned	RW	RAM	**Extended Occupancy Remaining** indicates the amount of time remaining before the controller reverts to unoccupied mode.
IP	51536	BitStr	RW	NRAM 0	**Input Polarities** specifies how the controller represents the input internally. 0=Normal polarity. Low Voltage is displayed as **present_value**=0 and High Voltage is displayed as **present_value**=1. 1=Reverse polarity. Low Voltage is displayed as **present_value**=1 and High Voltage is displayed as **present_value**=0. **IP** is a bitmap with bit 0 corresponding to UI1, bit 1=UI2, etc. up to bit 23=UI24.
LP	52304	BitStr	RW	NRAM 0	**Interlock Polarities** indicates the individual interlock polarity states for each Universal Input. 0=Normal 1=Reverse **LP**is a bitmap with bit 0 corresponding to UI1, bit 1=UI2, etc. up to bit 23=UI24.
OI	E9	RW	NRAM	0	**Overridden Inputs** allows the current value of the corresponding input, V1 through VO, to be overridden. 0=Normal 1=Override **OI** is a bitmap with bit 0 corresponding to UI1, bit 1=UI2, etc. up to bit 23=UI24.
PI	53321	Unsigned	RW	NRAM 0	**Installer P.I.N.** specifies the P.I.N. code used to enter the Installer menu.
PS	53331	Unsigned	RW	NRAM 1100	**Service P.I.N.** specifies the P.I.N. code used to enter the Service menu.
PU	53333	Unsigned	RW	NRAM 0	**User P.I.N.** specifies the P.I.N. code used to enter the User menu.

Property	Identifier #	Data Type	Access	Storage & Default	Description
RE	53829	BitStr	RO	RAM	**Inputs with Unreliable objects** indicates whether the reading from the corresponding input is considered reliable. 0=Reliable 1=Unreliable **RE** is a bitmap with bit 0 corresponding to UI1, bit 1=UI2, etc. up to bit 23=UI24..
SE	54085	Boolean	RW	NRAM 0	**User Occupancy Override Enable** specifies whether or not the user may override the schedule occupancy state.
V1	54833	Real	RW	RAM	**Current Measured Input 1 Value** indicates the **present_value** of Universal Input 1.
V2	54834	Real	RW	RAM	**Current Measured Input 2 Value** indicates the **present_value** of Universal Input 2.
V3	54835	Real	RW	RAM	**Current Measured Input 3 Value** indicates the **present_value** of Universal Input 3.
V4	54836	Real	RW	RAM	**Current Measured Input 4 Value** indicates the **present_value** of Universal Input 4.
V5	54837	Real	RW	RAM	**Current Measured Input 5 Value** indicates the **present_value** of Universal Input 5.
V6	54838	Real	RW	RAM	**Current Measured Input 6 Value** indicates the c**present_value** of Universal Input 6.
V7	54839	Real	RW	RAM	**Current Measured Input 7 Value** indicates the **present_value** of Universal Input 7.
V8	54840	Real	RW	RAM	**Current Measured Input 8 Value** indicates the **present_value** of Universal Input 8.
V9	54841	Real	RW	RAM	**Current Measured Input 9 Value** indicates the**present_value** of Universal Input 9.
VA	54849	Real	RW	RAM	**Current Measured Input 10 Value** indicates the**present_value** of Universal Input 10.
VB	54850	Real	RW	RAM	**Current Measured Input 11 Value** indicates the **present_value** of Universal Input 11.
VC	54851	Real	RW	RAM	**Current Measured Input 12 Value** indicates the**present_value** of Universal Input 12.
VD	54852	Real	RW	RAM	**Current Measured Input 13 Value** indicates the **present_value** of Universal Input 13.
VE	54853	Real	RW	RAM	**Current Measured Input 14 Value** indicates the **present_value** of Universal Input 14.
VF	54854	Real	RW	RAM	**Current Measured Input 15 Value** indicates the **present_value**of Universal Input 15.
VG	54855	Real	RW	RAM	**Current Measured Input 16 Value** indicates the **present_value** of Universal Input 16.
VH	54856	Real	RW	RAM	**Current Measured Input 17 Value** indicates the c**present_value** of Universal Input 17.

Property	Identifier #	Data Type	Access	Storage & Default	Description
VI	54857	Real	RW	RAM	**Current Measured Input 18 Value** indicates the **present_value**of Universal Input 18.
VJ	54858	Real	RW	RAM	**Current Measured Input 19 Value** indicates the **present_value** of Universal Input 19.
VK	54859	Real	RW	RAM	**Current Measured Input 20 Value** indicates the **present_value** of Universal Input 20.
VL	54860	Real	RW	RAM	**Current Measured Input 21 Value** indicates the**present_value** of Universal Input 21.
VM	54861	Real	RW	RAM	**Current Measured Input 22 Value** indicates the **present_value** of Universal Input 22.
VN	54862	Real	RW	RAM	**Current Measured Input 23 Value** indicates the **present_value** of Universal Input 23.
VO	54863	Real	RW	RAM	**Current Measured Input 24Value** indicates the **present_value** of Universal Input 24.

A.12 DIGITAL INPUT SUMMARY

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (254), Instance 48	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Digital Input Summary	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (254)	indicates membership in a particular object type class.
CV	50006	BitStr	RW	RAM	**Current Digital Input Value** indicates the current state of the associated digital input. **CV** is a bitmap with bit 0 corresponding to Digital Input 1, bit 1=Digital Input 2, etc. up to bit 7=Digital Input8.
IP	51536	BitStr	RW	NRAM 0	**Input Polarities** specifies how the controller represents the input internally. 0=Normal polarity. Low Voltage is displayed as **present_value**=0 and High Voltage is displayed as **present_value**=1. 1=Reverse polarity. Low Voltage is displayed as **present_value**=1 and High Voltage is displayed as **present_value**=0. **IP** is a bitmap with bit 0 corresponding to Digital Input 1, bit 1=Digital Input 2, etc. up to bit 7=Digital Input 8.
LP	52304	BitStr	RW	NRAM 0	**Interlock Polarities** indicates the individual interlock polarity states for each Digital Input. 0=Normal 1=Reverse **LP** is a bitmap with bit 0 corresponding to Digital Input 1, bit 1=Digital Input 2, etc. up to bit 7=Digital Input 8.
OI	53065	BitStr	RW	NRAM 0	**Overridden Digital Inputs Bitmap** indicates those inputs whose **out_of_service** property is true. **OI** is a bitmap with bit 0 corresponding to Digital Input 1, bit 1=Digital Input 2, etc. up to bit 7=Digital Input 8.
RE	53829	BitStr	RO	RAM	**Inputs with Unreliable Values** indicates whether the reading from the corresponding digital input is considered reliable. 0=Reliable 1=Unreliable **RE** is a bitmap with bit 0 corresponding to Digital Input 1, bit 1=Digital Input 2, etc. up to bit 7=Digital Input 8.

A.13 ANALOG OUTPUT SUMMARY

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (253), Instance 0	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Analog Output Summary	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (253)	indicates membership in a particular object type class.
AM	49485	BitStr	RW	RAM	**Outputs in Automatic Mode** specifies how the analog output is controlled. 0=Manual 1=Automatic **AM** is a bitmap with bit 0 corresponding to AO 1, bit 1=AO 2, etc. up to bit 11=AO12.
CF	49990	Unsigned	RW	NRAM 0	**Outputs Enabled for Communication Failure** **indicates** which outputs will be enabled in the event that a communication failure is detected. **CF** is a bitmap with bit 0 corresponding to AO 1, bit 1=AO 2, etc. up to bit 11=AO12.
RE	53829	BitStr	RO	RAM	**Outputs with Unreliable States** indicates whether the reading from the corresponding analog output is considered reliable. 0=Reliable 1=Unreliable **RE** is a bitmap with bit 0 corresponding to AO 1, bit 1=AO 2, etc. up to bit 11=AO 12.
V1	54833	Real	RW	RAM	**Current Value for Output 1** indicates the **present_value** of output 1.
V2	54834	Real	RW	RAM	**Current Value for Output 2** indicates the **present_value** of output 2.
V3	54835	Real	RW	RAM	**Current Value for Output 3** indicates the**present_value**of output 3.
V4	54836	Real	RW	RAM	**Current Value for Output 4** indicates the **present_value** of output 4.
V5	54837	Real	RW	RAM	**Current Value for Output 5** indicates the **present_value** of output 5.
V6	54838	Real	RW	RAM	**Current Value for Output 6** indicates the **present_value** of output 6.
V7	54839	Real	RW	RAM	**Current Value for Output 7** indicates the **present_value** of output 7.
V8	54840	Real	RW	RAM	**Current Value for Output 8** indicates the **present_value** of output 8.
V9	54841	Real	RW	RAM	**Current Value for Output 9** indicates the **present_value** of output 9.

Property	Identifier #	Data Type	Access	Storage & Default	Description
VA	54849	Real	RW	RAM	**Current Value for Output 10** indicates the **present_value** of output 10.
VB	54850	Real	RW	RAM	**Current Value for Output 11** indicates the **present_value** of output 11.
VC	54851	Real	RW	RAM	**Current Value for Output 12** indicates the **present_value** of output 12.

A.14 OCCUPANCY DETECTOR

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (252), Instance 1	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Occupancy Detector	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	Proprietary (252)	indicates membership in a particular object type class.
IA	51521	CharStr	RW	NRAM	**Input property for Occupancy Duration** specifies the property associated with the object specified in **IC** to be used for occupancy detection.
IC	51523	CharStr	RW	NRAM	**Input object for Occupancy Duration** specifies the object of the input to be used for occupancy detection.
MD	52548	Unsigned	RW	NRAM 0	**Extended Occupancy Delay** specifies the time, in minutes, that the occupancy detector status, **MS**, must be positive before the controller will switch to occupied mode.
MR	52562	Unsigned	RO	NRAM	**Extended Occupancy Remaining** indicates the time, in minutes, before the controller reverts to unoccupied mode.
MS	52563	Unsigned	RO	NRAM	**Occupancy Status** indicates whether the object and property specified in **IC** and **IA** indicate that the monitored zone is occupied.
MT	52564	Unsigned	RW	NRAM 0	**Extended Occupancy Duration** specifies the time, in minutes, that the controller will stay in occupied mode once occupancy is detected.

A.15 DIGITAL OUTPUT SUMMARY

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (251), Instance 0	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Digital Output Summary	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (251)	indicates membership in a particular object type class.
CF	49990	BitStr	RW	NRAM 0	**Outputs Enabled for Communication Fail** specifies which outputs will be enabled in the event that a communication failure is detected. **CF** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
CV	50006	BitStr	RW	RAM 0	**Digital Outputs which are On** indicates which outputs are currently on. **CV** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
FE	50757	BitStr	RW	NRAM 0	**Outputs Enabled for Fire** specifies which outputs will be enabled in the event that a fire event is detected. **FE** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
FI	50761	BitStr	RO	NRAM 0	**Fire Positions** indicates which digital outputs are in fire mode. **FI** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
FS	50771	BitStr	RO	NRAM 0	**Communication Failure Positions** indicates which outputs are currently experiencing communications failures. **FS** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
IP	51536	BitStr	RW	NRAM	**IO Polarities** indicates the value of the **polarity** property for each of the digital outputs. **IP** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
IS	51539	Unsigned	RW	NRAM 0	**Inter-Stage Delay** specifies a minimum delay, in seconds (0-255) that must elapse after a staged output is energized before subsequent stages may be energized.
OU	53077	BitStr	RO	RAM	**Actual Output States** indicates the actual output states of the digital outputs. **OU** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.

Property	Identifier #	Data Type	Access	Storage & Default	Description
RE	53829	BitStr	RO	RAM	**Outputs with Unreliable Values** indicates whether the reading from the corresponding digital output is considered reliable. 0=Reliable 1=Unreliable **RE** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.
SB	54082	BitStr	RW	NRAM 0	**Outputs Enabled for Staging** selects two or more digital outputs to be grouped together for staging. **SB** is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.

A.16 FLOATING POINT CONTROL 1-2

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (250), Instance 33-34	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Floating Point Control *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (250)	indicates membership in a particular object type class.
BO	49743	BitStr	RO	RAM	**Digital Outputs which are On** **indicates the digital outputs that are currently energized.** BO **is a bitmap with bit 0 corresponding to Digital Output 1, bit 1=Digital Output 2, etc. up to bit 11=Digital Output 12.**
CE	49989	Unsigned	RW	NRAM 0	**Enable Control Loop?** enables/disables floating point control for the associated control loop. 0=No 1=Yes
CF	49990	Unsigned	RW	NRAM 0	**Communication Failure Enable?** specifies what action to take in the event that a communication failure is detected. 0=No 1=Yes
CP	50000	Real	RO	RAM 0	**Current Position** indicates the current position of the motor.
CR	50002	Unsigned	RW	NRAM 0	**Motor Creep Function** specifies how the controller handles automatic calibrations at minimum and maximum positions. 0=Drive motor constantly if **DP**=0% or **DP**=100% 1=Creep motor output by 1% per minute if **DP**=0% or **DP**=100%
CS	50003	Real	RO	RAM 0	**Calculated Control Setpoint** indicates the calculated control setpoint. This value accounts for any reset or setup/setback action on the loop setpoint.
DB	50242	Real	RW	NRAM 0	**Desired Control Deadband** specifies the deadband that is used to control cycling around the setpoint. If the current value of the input object is between **SP**-(**DB**/2) and **SP**+(**DB**/2), the measured variable is considered to be at its setpoint
DL	50252	Real	RO	NRAM	**Demand Load** indicates the heating/cooling demand in terms of the temperature separation from setpoints
DO	50255	Enum	RO	NRAM	**Digital Output Pair Selection** specifies which Digital Output pair is to be used for floating point control. 0=none 1=Digital Outputs 1 & 2 2=Digital Outputs 3 & 4 3=Digital Outputs 5 & 6 4=Digital Outputs 7 & 8 5=Digital Outputs 9 & 10 6=Digital Outputs 11 & 12

Property	Identifier #	Data Type	Access	Storage & Default	Description
DP	50256	Real	RW	NRAM 0	**Desired Position** specifies the desired output position (0-100%) of the associated motor.
FI	50761	Real	RW	NRAM 0	**Fire Position** specifies the failure position (0-100%) to use when a fire event is detected.
FP	50768	Real	RW	NRAM 0	**Interlock/Comm Failure Position** specifies the failure position (0-100%) to use when an input interlock failure occurs. **FP** is used when the current value of any of the inputs specified by **IL** has a value of 1.
IA	51521	CharStr	RW	NRAM	**Input property** specifies the object to be used for floating point control.
IC	51523	CharStr	RW	NRAM	**Input object** specifies the property associated with the object specified in **IC** to be used for floating point control.
IL	51532	BitStr	RW	NRAM 0	**Input for Interlock** specifies which inputs are to be used for interlocking of the associated floating point control loop. bit 0 = UI 1 bit 1 = UI 2 bit 2 = UI 3 bit 3 = UI 4 bit 4 = UI 5 bit 5 = UI 6 bit 6 = UI 7 bit 7 = UI 8 bit 8 = UI 9 bit 9 = UI 10 bit 10 = UI 11 bit 11 = UI 12 bit 12 = UI 13 bit 13 = UI 14 bit 14 = UI 15 bit 15 = UI 16 bit 16 = UI 17 bit 17 = UI 18 bit 18 = UI 19 bit 19 = UI 20 bit 20 = UI 21 bit 21 = UI 22 bit 22 = UI 23 bit 23 = UI 24 bit 24 = DI 1 bit 25 = DI 2 bit 26 = DI 3 bit 27 = DI 4 bit 28 = DI 5 bit 29 = DI 6 bit 30 = DI 7 bit 31 = DI 8
MR	52562	Real	RW	NRAM 0	**Maximum Amount to Reset Setpoint** specifies the maximum amount o reset **SP**.
PB	53314	Real	RW	NRAM 0	**Proportional Control Band** specifies a range, centered around the loop setpoint **SP**, where the output signal is proportional.
PE	53317	Unsigned	RW	NRAM 0	**Floating Point Control Pair Enable** specifies whether the motor is to be controlled using pulsed pairs. 0=No 1=Yes

Property	Identifier #	Data Type	Access	Storage & Default	Description
RA	53825	CharStr	RW	NRAM	**Reset property** specifies the property associated with the object specified in **RV** to determine reset.
RC	53827	CharStr	RW	NRAM	**Reset Variable** specifies the object to be used to determine reset.
RI	53833	Unsigned	RW	NRAM 0	**Motor Recalibrate Interval** specifies a time interval in hours (0-255) the defines how often the associated floating point control loop is recalibrated. 0=Calibration disabled **RI** > 0 =Recalibrate every **RI** hours
RL	53836	Real	RW	NRAM 0	**Limit for Maximum Reset** specifies the reset limit of the control loop. When the reset variable specified in **RV** and **RA** reaches a value of **RL**, the control loop setpoint will be reset by the maximum amount **MR.**
RP	53840	Unsigned	RW	NRAM 0	**Reset Period** specifies a time, in seconds (0 to 65,535) over which the output of the control loop should be adjusted (reset). 0=Diabled 1 to 65,535=Reset period, in seconds
RS	53843	Real	RW	NRAM 0	**Setpoint at which Reset Action Begins** specifies the setpoint of the control loop at which reset action begins.
SG	54087	Uint	RW	NRAM 0	**Control Action** specifies whether the controller's output should be increased or decreased when the control signal is positive. 0=Normal (increase for positive error) 1=Reverse (decrease for positive error)
SM	54093	BitStr	RW	NRAM 0	**Schedules to Follow** enables scheduled alarm controlling by selecting one or more of the available schedule control objects. 0=Schedule disabled 1=Schedule enabled **SM** is a bitmap with bit 0=Schedule 1 bit 1=Schedule 2 bit 2=Schedule 3 bit 3=Schedule 4 bit 4=Schedule 5 bit 5=Schedule 6 bit 6=Schedule 7 bit 7=Schedule 8 bit 8= SMARTStat 1 bit 9= SMARTStat 2 bit 10= SMARTStat 3 bit 11= SMARTStat 4 bit 12= SMARTStat 5 bit 13= SMARTStat 6 bit 14= SMARTStat 7 bit 15= SMARTStat 8 bit 16= SMARTStat 9 bit 17= SMARTStat 10 bit 18= SMARTStat 11 bit 19= SMARTStat 12 bit 20= Host Schedule bit 21= Schedule Summary bit 22= Occupancy
SP	54096	Real	RW	NRAM 0	**Loop Setpoint** specifies the desired setpoint for the floating point control loop.

Property	Identifier #	Data Type	Access	Storage & Default	Description
SU	54101	Real	RW	NRAM 0	**Unoccupied Setup/Setback** specifies a value (0.0 to 25.5) which is added to (if **SG**=1) or subtracted from (if **SG**=0) the control loop setpoint during scheduled unoccupied periods
TS	54355	Real	RO	NRAM	**Thermostat Setpoint Adjustment** indicates the offset, read from a Stat3, to be applied to control setpoints.
TT	54356		RW	NRAM 0	**Motor Travel Time** specifies the time, in seconds (0-3000), that it takes the motor to move from its fully closed to its fully open positions.

A.17 THERMOSTATIC CONTROL 1-12

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (250), Instance 17-28	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Thermostatic Control *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (250)	indicates membership in a particular object type class.
CE	49989	Boolean	RW	NRAM 0	**Enable Control Loop?** enables/disables thermostatic control for the associated control loop. 0=Disabled 1=Enabled
CS	50003	Real	RO	RAM 0	**Calculated Control Setpoint** specifies the calculated (actual) control setpoint that is used by the thermostatic control loop. **CS** accounts for the effects of setup/setback (**SU**) during scheduled unoccupied periods.
CV	50006	Unsigned	RO	RAM	**Current Value of Selected Output** indicates the current output of the control loop.
DB	50242	Real	RW	NRAM 0	**Desired Control DeadBand** specifies a control hysteresis that is used to keep **present_value** from toggling when the value is on the border between two states.
DL	50252	Real	RO	NRAM	**Demand Load** indicates the heating/cooling demand in terms of the temperature separation from setpoints
IA	51521	CharStr	RW	NRAM	**Input Attribute** specifies the property associated with the object specified in **IC** to be used for thermostatic control.
IC	51523	CharStr	RW	NRAM	**Input Channel** specifies the object to be used for thermostatic control.
OB	53058	Uint	RW	NRAM 0	**Output Bitmap** specifies which digital outputs will be controlled ny the thermostatic control loop. **OB** is a bitmap with bit 0 corresponding to BO1, bit 1=BO2, etc. up to bit 11=BO12.
SG	54087	Boolean	RW	NRAM 0	**Control Action** specifies the control sign for the thermostatic control application. 0=Cooling 1=Heating

Property	Identifier #	Data Type	Access	Storage & Default	Description
SM	54093	BitStr	RW	NRAM 0	**Schedules to Follow** enables scheduled alarm controlling for the thermostatic control loop by selecting one or more of the available schedule control objects. 0=schedule disabled 1=schedule enabled **SM** is a bitmap with bit 0=Schedule 1 bit 1=Schedule 2 bit 2=Schedule 3 bit 3=Schedule 4 bit 4=Schedule 5 bit 5=Schedule 6 bit 6=Schedule 7 bit 7=Schedule 8 bit 8= SMARTStat 1 bit 9= SMARTStat 2 bit 10= SMARTStat 3 bit 11= SMARTStat 4 bit 12= SMARTStat 5 bit 13= SMARTStat 6 bit 14= SMARTStat 7 bit 15= SMARTStat 8 bit 16= SMARTStat 9 bit 17= SMARTStat 10 bit 18= SMARTStat 11 bit 19= SMARTStat 12 bit 20= Host Schedule bit 21= Schedule Summary bit 22= Occupancy
SP	54096	Real	RW	NRAM 0	**Loop Setpoint** specifies the desired setpoint for the thermostatic control loop.
SU	54101	Real	RW	NRAM 0	**Unoccupied Setup/Setback** specifies a value (0.0 to 25.5) which is added to the control setpoint during scheduled unoccupied periods.
TS	54355	Real	RO	NRAM	**Thermostat Setpoint Adjustment** indicates the offset, read from a Stat3, to be applied to control setpoints.

A.18 PID CONTROL 1-12

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (250), Instance 1-12	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM PID Control *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (250)	indicates membership in a particular object type class.
AO	49487	Real	RO	RAM 0	**Analog Output Value** displays the output of the PID loop.
CE	49989	Boolean	RW	NRAM 0	**Enable Control Loop?** enables/disables PID control. 0=No 1=Yes
CS	50003	Real	RO	RAM 0	**Calculated Control Setpoint** specifies the effective (calculated) setpoint accounting for setup/ setback, manual setpoint adjustments, etc.
DB	50242	Real	RW	NRAM 0	**Desired Control Deadband** specifies the deadband that is used to control cycling around the setpoint. If the current value of the input object is between **SP**-(**DB**/2) and **SP**+(**DB**/2), the measured variable is considered to be at its setpoint.
DL	50252	Real	RO	NRAM	**Demand Load** indicates the heating/cooling demand in terms of the temperature separation from setpoints
IA	51521	CharStr	RW	NRAM	**Loop Measured property** specifies the property associated with the object specified in **IC** to be used as the measured variable for PID control loop.
IC	51523	CharStr	RW	NRAM	**Loop Measured object** specifies the input object to be used as the measured variable for the PID control loop.
MR	52562	Real	RW	NRAM 0	**Maximum Amount to Reset Setpoint** the maximum amount to reset the control loop setpoint.
OH	53064	Real	RW	NRAM 100	**Output High Limit** defines the maximum output for the PID control loop.
OL	53068	Real	RW	NRAM 0	**Output Low Limit** defines the minimum output for the PID control loop.
PB	53314	Real	RW	NRAM 0	**Proportional Control Band** specifies a range, centered around the loop setpoint **SP**, where the output signal is proportional. If the value of the selected input object is outside the proportional band, the proportional component of the PID calculation is clamped at **OL** or **OH** as appropriate.
PO	53327	Real	RW	NRAM 0	**Percent Output Value** displays the calculated output of the PID control loop. **PO** ranges from **OL** to **OH**.
RA	53825	CharStr	RW	NRAM 00	**Reset property** specifies the attribute associated with the object specified in **RV** to be used as the reset variable for the PID control loop.

Property	Identifier #	Data Type	Access	Storage & Default	Description
RC	53827	CharStr	RW	NRAM 0	**Reset object** specifies the input object to be used as the reset variable for the PID control loop.
RL	53836	Real	RW	NRAM 0	**Limit for Maximum Reset** specifies the reset limit of the control loop. When **RV** reaches a value of **RL**, the control loop setpoint will be reset by the maximum amount **RV**.
RP	53840	Unsigned	RW	NRAM 0	**Reset Period** specifies a time, in seconds (0 to 65,535) over which the output of the control loop should be adjusted (reset) using integral action. 0=Disabled 1 to 65,535=Integral reset period, in seconds
RS	53843	Real	RW	NRAM 0	**Setpoint at which Reset Action Begins** specifies the setpoint of the control loop at which reset action begins.
RT	53844	Real	RW	NRAM 0	**Derivative Rate** specifies a a percentage of the amount of derivative error that is contributed each second to the PID output of the control loop (0.0 to 25.5%). 0.0=Disable 0.1 to 25.5=Derivative rate in %/second
SG	54087	Unsigned	RW	NRAM 0	**Control Action** specifies whether the controller's output should be increased or decreased when error is positive. 0=Normal (increase for positive error) 1=Reverse (decrease for positive error)
SM	54093	BitStr	RW	NRAM 0	**Schedules to Follow** enables scheduled alarm controlling for the associated PID control loop by selecting one or more of the available schedule control objects. 0=Schedule disabled 1=Schedule enabled **SM** is a bitmap with bit 0=Schedule 1 bit 1=Schedule 2 bit 2=Schedule 3 bit 3=Schedule 4 bit 4=Schedule 5 bit 5=Schedule 6 bit 6=Schedule 7 bit 7=Schedule 8 bit 8= SMARTStat 1 bit 9= SMARTStat 2 bit 10= SMARTStat 3 bit 11= SMARTStat 4 bit 12= SMARTStat 5 bit 13= SMARTStat 6 bit 14= SMARTStat 7 bit 15= SMARTStat 8 bit 16= SMARTStat 9 bit 17= SMARTStat 10 bit 18= SMARTStat 11 bit 19= SMARTStat 12 bit 20= Host Schedule bit 21= Schedule Summary bit 22= Occupancy
SP	54096	Real	RW	NRAM 0	**Loop Setpoint** specifies the desired value of the variable selected in **IC** and **IA**.

Property	Identifier #	Data Type	Access	Storage & Default	Description
SR	54098	Unsigned	RW	NRAM 100	**Soft Start Ramp** specifies the maximum percentage change per minute for the associated output under he following conditions: when the controller is initially powered up or reset; upon transitions from unoccupied to occupied mode, upon cancellation of an interlock failure or fire condition, or when a control loop is initially enabled.
SU	54101	Real	RW	NRAM 0	**Unoccupied Setup/Setback** specifies a value (0.0 to 25.5) which is added to (if **SG**=1) or subtracted from (if **SG**=0) the control loop setpoint during scheduled unoccupied periods.
TS	54355	Real	RO	NRAM	**Thermostat Setpoint Adjustment** indicates the offset, read from a Stat3, to be applied to control setpoints.

A.19 SCHEDULE SUMMARY

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (249), Instance 0	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Schedule Summary	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (249)	indicates membership in a particular object type class.
AS	49491	Unsigned	RO	RAM	**Active Schedule** indicates which schedule is currently active.
C1	49969	Unsigned	RO	RAM	**Schedule 1** displays the schedule state of Schedule 1.
C2	49970	Unsigned	RO	RAM	**Schedule 2** displays the schedule state of Schedule 2.
C3	49971	Unsigned	RO	RAM	**Schedule 3** displays the schedule state of Schedule3.
C4	49972	Unsigned	RO	RAM	**Schedule 4** displays the schedule state of Schedule 4.
C5	49973	Unsigned	RO	RAM	**Schedule 5** displays the schedule state of Schedule 5.
C6	499704	Unsigned	RO	RAM	**Schedule 6** displays the schedule state of Schedule 6.
C7	49975	Unsigned	RO	RAM	**Schedule 7** displays the schedule state of Schedule 7.
C8	49976	Unsigned	RO	RAM	**Schedule 8** displays the schedule state of Schedule 8.
CV	50006	Unsigned	RO	RAM	**Occupied Schedules** indicates the current state of the controller. 0=Unoccupied 1=Warm-up 2=Occupied 3=Night setback
DH	50248	Boolean	RW	NRAM 0	**Holiday?** specifies whether today is one of the holidays specified in **H0**-**H9**.
H0	51248	Date	RW	NRAM 0	**Programmed Holiday 1** specifies the date of the first scheduled holiday.
H1	51249	Date	RW	NRAM 0	**Programmed Holiday 2** specifies the date of the second scheduled holiday.
H2	51250	Date	RW	NRAM 0	**Programmed Holiday 3** specifies the date of the third scheduled holiday.
H3	51251	Date	RW	NRAM 0	**Programmed Holiday 4** specifies the date of the fourth scheduled holiday.
H4	51252	Date	RW	NRAM 0	**Programmed Holiday 5** specifies the date of the fifth scheduled holiday.

Property	Identifier #	Data Type	Access	Storage & Default	Description
H5	512503	Date	RW	NRAM 0	**Programmed Holiday 6** specifies the date of the sixth scheduled holiday.
H6	51254	Date	RW	NRAM 0	**Programmed Holiday 7** specifies the date of the seventh scheduled holiday.
H7	51255	Date	RW	NRAM 0	**Programmed Holiday 8** specifies the date of the eighth scheduled holiday.
H8	51256	Date	RW	NRAM 0	**Programmed Holiday 9** specifies the date of the ninth scheduled holiday.
H9	51257	Date	RW	NRAM 0	**Programmed Holiday 10** specifies the date of the tenth scheduled holiday.
HE	51269	Unsigned	RW	NRAM 0	**Enable Host Schedule** specifies whether to use the schedule broadcast by the host. 0=No 1=Yes
HO	51279	Unsigned	RW	NRAM 0	**Host Schedule Value** specifies the desired host schedule override state. 0=Unoccupied 1=Warmup 2=Occupied 3=Night setback
IS	51539	Unsigned	RW	NRAM 0	**Inactive Schedule State** indicates the state the controller will assume when no schedules are currently active 0=Unoccupied 1=Warmup 2=Occupied 3=Night setback
AS	49491	Bitstring	RO	NRAM	**Active Schedules** lists the schedules which are currently active. bit #0=Schedule 1 bit #1=Schedule 2 bit #2=Schedule 3 bit #3=Schedule 4 bit #4=Schedule 5 bit #5=Schedule 6 bit #6=Schedule 7 bit #7=Schedule 8
SO	54095	Bitstring	RW	NRAM	**Digital Stat Override Status** displays the state of each STAT's local override flag. bit #0=SMARTStat 1 bit #1=SMARTStat 2 bit #2=SMARTStat 3 bit #3=SMARTStat 4 bit #4=SMARTStat 5 bit #5=SMARTStat 6 bit #6=SMARTStat 7 bit #7=SMARTStat 8 bit #8=SMARTStat 9 bit #9=SMARTStat 10 bit #10=SMARTStat 11 bit #11=SMARTStat 12

Property	Identifier #	Data Type	Access	Storage & Default	Description
AB	49474	Bitstring	RW	NRAM	**Access Buttons States** displays the state of the remote access control buttons. bit #0=Button 1 bit #1=Button 2 bit #2=Button 3 bit #3=Button 4 bit #4=Button 5 bit #5=Button 6 bit #6=Button 7 bit #7=Button 8 bit #8=Button 9 bit #9=Button 10 bit #10=Button 11 bit #11=Button 12
EA	50497	Bitstring	RW	NRAM	**Enable Access Buttons** enables remote access button presses. bit #0=Enable Button 1 bit #1=Enable Button 2 bit #2=Enable Button 3 bit #3=Enable Button 4 bit #4=Enable Button 5 bit #5=Enable Button 6 bit #6=Enable Button 7 bit #7=Enable Button 8 bit #8=Enable Button 9 bit #9=Enable Button 10 bit #10=Enable Button 11 bit #11=Enable Button 12
OM	53069	Bitstring	RW	NRAM	**Override Map** displays whether or not a STAT is in an override state. bit #0=Override 1 bit #1=Override 2 bit #2=Override 3 bit #3=Override 4 bit #4=Override 5 bit #5=Override 6 bit #6=Override 7 bit #7=Override 8 bit #8=Override 9 bit #9=Override 10 bit #10=Override 11 bit #11=Override 12
AM	49485	Bitstring	RW	NRAM	**Manual Control** enables write commands to OM. bit #0=Enable Button 1 bit #1=Enable Button 2 bit #2=Enable Button 3 bit #3=Enable Button 4 bit #4=Enable Button 5 bit #5=Enable Button 6 bit #6=Enable Button 7 bit #7=Enable Button 8 bit #8=Enable Button 9 bit #9=Enable Button 10 bit #10=Enable Button 11 bit #11=Enable Button 12

A.20 SCHEDULES 1-8

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (249), Instance 1-8	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Schedule *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (249)	indicates membership in a particular object type class.
AD	49476	Unsigned	RW	NRAM 0	**Active Days** bitmap that specifies which days the schedule is active. Setting the appropriate bit to 1 indicates the schedule will b active on that day. bit 0=Monday bit 1=Tuesday bit 2=Wednesday bit 3=Thursday bit 4=Friday bit 5=Saturday bit 6=Sunday bit 7=Holiday
AO	49487	Unsigned	RW	NRAM 0	**All Day Override** 0=No 1=Yes
CV	50006	Unsigned	RO	RAM 0	**Current Schedule Value** indicates the current state of the schedule. 0=Unoccupied 1=Warm-up 2=Occupied 3=Night setback
NS	52819	Time	RW	NRAM 19:00	**Time to go Night Setback** specifies the time when night setback mode should begin.
OC	53059	Time	RW	NRAM 08:00	**Time to go Occupied** specifies the time when occupied mode should begin.
UN	54606	Time	RW	NRAM 17:00	**Time to go Unoccupied** specifies the time when unoccupied mode should begin.
WO	55119	Time	RW	NRAM 07:30	**Time to go Warm-Up** specifies the time when warm-up mode should begin.

A.21 SCALES 1-4

Property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (247), Instance 17-20	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Scale *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (247)	indicates membership in a particular object type class.
CV	50006	Real	RO	RAM	**Current Value** specifies the calculated scaled value.
IA	51521	CharStr	RW	NRAM 0	**Input property** specifies the property associated with the object specified in **IC** to be scaled.
IC	51523	CharStr	RW	NRAM	**Input object** specifies the object to be scaled.
X1	55345	Real	RW	NRAM 0	**Input range X1 value** specifies the minimum value of the input.
X2	55346	Real	RW	NRAM 0	**Input range X2 value** specifies the maximum value of the input.
Y1	55601	Real	RW	NRAM 0	**Output range Y1 value** specifies the minimum value of the scaled output.
Y2	556012	Real	RW	NRAM 0	**Output range Y2 value** specifies the maximum value of the scaled output.

A.22 PIECEWISE CURVES 1-2

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (247), Instance 1-2	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Piecewise Curve *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (247)	indicates membership in a particular object type class.
X1	55345	Real	RW	NRAM 0	**Point 1's value in raw counts** specifies the x coordinate of point 1.
X2	55346	Real	RW	NRAM 0	**Point 2's value in raw counts** specifies the x coordinate of point 2.
X3	55347	Real	RW	NRAM 0	**Point 3's value in raw counts** specifies the x coordinate of point 3.
X4	55348	Real	RW	NRAM 0	**Point 4's value in raw counts** specifies the x coordinate of point 4.
X5	55349	Real	RW	NRAM 0	**Point 5's value in raw counts** specifies the x coordinate of point 5.
X6	55350	Real	RW	NRAM 0	**Point 6's value in raw counts** specifies the x coordinate of point 6.
X7	55351	Real	RW	NRAM 0	**Point 7's value in raw counts** specifies the x coordinate of point 7.
X8	55352	Real	RW	NRAM 0	**Point 8's value in raw counts** specifies the x coordinate of point 8.
X9	55353	Real	RW	NRAM 0	**Point 9's value in raw counts** specifies the x coordinate of point 9.
XA	55361	Real	RW	NRAM 0	**Point 10's value in raw counts** specifies the x coordinate of point 10.
XB	55362	Real	RW	NRAM 0	**Point 11's value in raw counts** specifies the x coordinate of point 11.
Y1	55601	Real	RW	NRAM 0	**Point 1's value in engineering units** specifies the y coordinate of point 1.
Y2	55602	Real	RW	NRAM 0	**Point 2's value in engineering units** specifies the y coordinate of point 2.
Y3	55603	Real	RW	NRAM 0	**Point 3's value in engineering units** specifies the y coordinate of point 3.
Y4	55604	Real	RW	NRAM 0	**Point 4's value in engineering units** specifies the y coordinate of point 4.
Y5	55605	Real	RW	NRAM 0	**Point 5's value in engineering units** specifies the y coordinate of point 5.
Y6	55606	Real	RW	NRAM 0	**Point 6's value in engineering units** specifies the y coordinate of point 6.

Property	Identifier #	Data Type	Access	Storage & Default	Description
Y7	556017	Real	RW	NRAM 0	**Point 7's value in engineering units** specifies the y coordinate of point 7.
Y8	55608	Real	RW	NRAM 0	**Point 8's value in engineering units** specifies the y coordinate of point 8.
Y9	55609	Real	RW	NRAM 0	**Point 9's value in engineering units** specifies the y coordinate of point 9.
YA	55617	Real	RW	NRAM 0	**Point 10's value in engineering units** specifies the y coordinate of point 10.
YB	55618	Real	RW	NRAM 0	**Point 11's value in engineering units** specifies the y coordinate of point 11.

A.23 LOGIC 1-4

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (243), Instance 33-36	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Logic *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (243)	indicates membership in a particular object type class.
A1	49457	CharStr	RW	NRAM 0	**Input property 1** specifies the property associated with the first input object.
A2	49458	CharStr	RW	NRAM 0	**Input property 2** specifies the property associated with the second input object.
A3	49459	CharStr	RW	NRAM 0	**Input property 3** specifies the property associated with the third input object.
A4	49460	CharStr	RW	NRAM 0	**Input property 4** specifies the property associated with the fourth input object.
A5	49461	CharStr	RW	NRAM 0	**Input property 5** specifies the property associated with the fifth input object.
A6	49462	CharStr	RW	NRAM 0	**Input property 6** specifies the property associated with the sixth input object.
A7	49463	CharStr	RW	NRAM 0	**Input property 7** specifies the property associated with the seventh input object.
A8	49464	CharStr	RW	NRAM 0	**Input property 8** specifies the property associated with the eighth input object.
CV	50006	Unsigned	RO	RAM 0	**Current Value** indicates the result after the operand specified in **OP** has been applied to the inputs.
I1	51505	CharStr	RW	NRAM 0	**Input object 1** specifies the first input object.
I2	51506	CharStr	RW	NRAM 0	**Input object 2** specifies the second input object.
I3	51507	CharStr	RW	NRAM 0	**Input object 3** specifies the third input object.
I4	51508	CharStr	RW	NRAM 0	**Input object 4** specifies the fourth input object.
I5	51509	CharStr	RW	NRAM 0	**Input object 5** specifies the fifth input object.
I6	51510	CharStr	RW	NRAM 0	**Input object 6** specifies the sixth input object.
I7	51511	CharStr	RW	NRAM 0	**Input object 7** specifies the seventh input object.
I8	51512	CharStr	RW	NRAM 0	**Input object 8** specifies the eighth input object.

Property	Identifier #	Data Type	Access	Storage & Default	Description
OP	53072	Unsigned	RW	NRAM 0	**Operation** specifies the logic operation to be performed on the selected objects. 0=Diabled 1=OR 2=AND 3=NOT 4=XOR

A.24 MATH 1-2

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (243), Instance 17-18	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Math *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (243)	indicates membership in a particular object type class.
A1	49457	CharStr	RW	NRAM 0	**Input property 1** specifies the property associated with the first input object.
A2	49458	CharStr	RW	NRAM 0	**Input property 2** specifies the property associated with the second input object.
CV	50006	Real	RW	NRAM 0	**Current Value** indicates the numeric result of applying the operand specified in **OP** to the inputs.
I1	45361	CharStr	RW	NRAM 0	**Input object 1** specifies the first input object.
I2	45362	CharStr	RW	NRAM 0	**Input object 2** specifies the second input object.
OP	53072	Unsigned	RW	NRAM 0	**Operation** specifies the operation to be performed on the selected objects. 0=Disabled 1=Addition 2=Subtraction 3=Multiplication 4=Division

A.25 MIN/MAX/AVG 1-3

Property	Identifier #	Data Type	Access	Storage & Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (243), Instance 1-3	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Min/Max/Avg *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (243)	indicates membership in a particular object type class.
A1	49457	CharStr	RW	NRAM	**Input Property 1** specifies the property in the object selected in **I1** to be used as the first input min/max/avg calculations.
A2	49458	CharStr	RW	NRAM	**Input Property 2** specifies the property in the object selected in **I2** to be used as the second input min/max/avg calculations.
A3	49459	CharStr	RW	NRAM	**Input Property 3** specifies the property in the object selected in **I3** to be used as the third input min/max/avg calculations.
A4	49460	CharStr	RW	NRAM	**Input Property 4** specifies the property in the object selected in **I4** to be used as the fourth input min/max/avg calculations.
AV	49494	Real	RW	RAM 0	**Average Value** displays the arithmetic mean of the inputs selected.
HV	51286	Real	RW	NRAM 0	**High Value** displays the highest value of the inputs selected.
I1	51505	CharStr	RW	NRAM	**Input Object 1** specifies the object from which the first property to be used for min/max/avg calculations can be selected.
I2	51506	CharStr	RW	NRAM	**Input Object 2** specifies the object from which the second property to be used for min/max/avg calculations can be selected.
I3	51507	CharStr	RW	NRAM	**Input Object 3** specifies the object from which the third property to be used for min/max/avg calculations can be selected.
I4	51508	CharStr	RW	NRAM	**Input Object 4** specifies the object from which the fourth property to be used for min/max/avg calculations can be selected.
LV	52310	Real	RW	NRAM 0	**Low Value** displays the lowest value of the inputs selected.

A.26 INPUT SELECT 1-4

Property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (240), Instance 17-20	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Input Select *N*	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (240)	indicates membership in a particular object type class.
A1	49457	CharStr	RW	NRAM 0	**Input property 1** specifies the property associated with the first input object.
A2	49458	CharStr	RW	NRAM 0	**Input property 2** specifies the property associated with the second input object.
CV	50006	Real	RO	RAM 0	**Current Value** indicates the value of the property which has been selected.
I1	45361	CharStr	RW	NRAM	**Input object 1** specifies the object from which the first input property will be chosen.
I2	45362	CharStr	RW	NRAM	**Input object 2** specifies the object from which the second input property will be chosen.
SA	54081	CharStr	RW	NRAM	**Selection attribute** specifies the property to be used as the selection criteria. If the specified property has a value of 0, **CV** will take the value of the property specified in **I1** and **A1**. If the specified property has a value of 1, **CV** will take the value of the property specified in **I2** and **A2**.
SC	540834	CharStr	RW	NRAM 0	**Selection Channel** specifies the object from which the property used for selection will be chosen.

A.27 BROADCAST 0

Property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (240), Instance 0	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Broadcast Outside Air Temp	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (240)	indicates membership in a particular object type class.
BE	49733	Boolean	RW	NRAM 0	**Broadcast Enable** specifies whether the controller should broadcast the value of CV over the network. 0=No 1=Yes
BZ	49498	Boolean	RW	NRAM 0	**Broadcast Zone/Global** specifies whether the controller's broadcasts are sent to the zone or broadcast globally. 0=Zone broadcast 1=Global broadcast
CV	50006	Real	RW	RAM	**Current Value** indicates the current value of the referenced object specified in **IC** and **IA**.
IA	51521	CharStr	RW	NRAM 0	**Input property** specifies the property associated with the object specified in **IC** to be broadcast.
IC	51523	CharStr	RW	NRAM	**Input object** specifies the input object to be broadcast.
RB	53826	Boolean	RW	NRAM 0	**Receive Broadcast?** specifies that the controller should accept broadcasted values for **CV**.

A.28 BROADCAST 1

Property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (240), Instance 1	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Broadcast Outside Air Humidity	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (240)	indicates membership in a particular object type class.
BE	49733	Boolean	RW	NRAM 0	**Broadcast Enable** specifies whether the controller should broadcast the value of CV over the network. 0=No 1=Yes
BZ	49498	Boolean	RW	NRAM 0	**Broadcast Zone/Global** specifies whether the controller's broadcasts are sent to the zone or broadcast globally. 0=Zone broadcast 1=Global broadcast
CV	50006	Real	RW	RAM	**Current Value** indicates the current value of the referenced object specified in **IC** and **IA**.
IA	51521	CharStr	RW	NRAM 0	**Input property** specifies the property associated with the object specified in **IC** to be broadcast.
IC	51523	CharStr	RW	NRAM	**Input object** specifies the input object to be broadcast.
RB	53826	Boolean	RW	NRAM 0	**Receive Broadcast?** specifies that the controller should accept broadcasted values for **CV**.

A.29 BROADCAST 2

Property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (240), Instance 2	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Broadcast Enthalpy	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (240)	indicates membership in a particular object type class.
BE	49733	Boolean	RW	NRAM 0	**Broadcast Enable** specifies whether the controller should broadcast the value of CV over the network. 0=No 1=Yes
BZ	49498	Boolean	RW	NRAM 0	**Broadcast Zone/Global** specifies whether the controller's broadcasts are sent to the zone or broadcast globally. 0=Zone broadcast 1=Global broadcast
CE	49989	Boolean	RW	NRAM	**Enthalpy Calculation Enable** specifies whether Broadcast 2 should calculate the enthallpy based on the outside air temperature and humidity readings in Broadcast 0 and Broadcast 1.
CV	50006	Real	RW	RAM	**Current Value** indicates the current value of the referenced object specified in **IC** and **IA**.
IA	51521	CharStr	RW	NRAM 0	**Input property** specifies the property associated with the object specified in **IC** to be broadcast.
IC	51523	CharStr	RW	NRAM	**Input object** specifies the input object to be broadcast.
RB	53826	Boolean	RW	NRAM 0	**Receive Broadcast?** specifies that the controller should accept broadcasted values for **CV**.

A.30 BROADCAST 3

Property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (240), Instance 3	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Broadcast 3	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (240)	indicates membership in a particular object type class.
BE	49733	Boolean	RW	NRAM 0	**Broadcast Enable** specifies whether the controller should broadcast the value of CV over the network. 0=No 1=Yes
BZ	49498	Boolean	RW	NRAM 0	**Broadcast Zone/Global** specifies whether the controller's broadcasts are sent to the zone or broadcast globally. 0=Zone broadcast 1=Global broadcast
CV	50006	Real	RW	RAM	**Current Value** indicates the current value of the referenced object specified in **IC** and **IA**.
IA	51521	CharStr	RW	NRAM 0	**Input property** specifies the property associated with the object specified in **IC** to be broadcast.
IC	51523	CharStr	RW	NRAM	**Input object** specifies the input object to be broadcast.
RB	53826	Boolean	RW	NRAM 0	**Receive Broadcast?** specifies that the controller should accept broadcasted values for **CV**.

A.31 BROADCAST 4

Property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (240), Instance 4	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Broadcast 4	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (240)	indicates membership in a particular object type class.
BE	49733	Boolean	RW	NRAM 0	**Broadcast Enable** specifies whether the controller should broadcast the value of CV over the network. 0=No 1=Yes
BZ	49498	Boolean	RW	NRAM 0	**Broadcast Zone/Global** specifies whether the controller's broadcasts are sent to the zone or broadcast globally. 0=Zone broadcast 1=Global broadcast
CV	50006	Real	RW	RAM	**Current Value** indicates the current value of the referenced object specified in **IC** and **IA**.
IA	51521	CharStr	RW	NRAM 0	**Input property** specifies the property associated with the object specified in **IC** to be broadcast.
IC	51523	CharStr	RW	NRAM	**Input object** specifies the input object to be broadcast.
RB	53826	Boolean	RW	NRAM 0	**Receive Broadcast?** specifies that the controller should accept broadcasted values for **CV**.

A.32 BROADCAST 5

property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (240), Instance 5	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Broadcast Schedule	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (240)	indicates membership in a particular object type class.
BE	49733	Boolean	RW	NRAM 0	**Broadcast Enable** specifies whether the controller should broadcast the value of CV over the network. 0=No 1=Yes
BZ	49498	Boolean	RW	NRAM 0	**Broadcast Zone/Global** specifies whether the controller's broadcasts are sent to the zone or broadcast globally. 0=Zone broadcast 1=Global broadcast
CV	50006	Unsigned	RW	RAM	**Current Value** indicates the current value of the referenced object specified in **IC** and **IA**.
IA	51521	CharStr	RW	NRAM 0	**Input property** specifies the property associated with the object specified in **IC** to be broadcast.
IC	51523	CharStr	RW	NRAM	**Input object** specifies the input object to be broadcast.
RB	53826	Boolean	RW	NRAM 0	**Receive Broadcast?** specifies that the controller should accept broadcasted values for **CV**.

A.33 BROADCAST 6

Property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (240), Instance 6	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Broadcast 6	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (240)	indicates membership in a particular object type class.
BE	49733	Boolean	RW	NRAM 0	**Broadcast Enable** specifies whether the controller should broadcast the value of CV over the network. 0=No 1=Yes
BZ	49498	Boolean	RW	NRAM 0	**Broadcast Zone/Global** specifies whether the controller's broadcasts are sent to the zone or broadcast globally. 0=Zone broadcast 1=Global broadcast
CV	50006	Real	RW	RAM	**Current Value** indicates the current value of the referenced object specified in **IC** and **IA**.
IA	51521	CharStr	RW	NRAM 0	**Input property** specifies the property associated with the object specified in **IC** to be broadcast.
IC	51523	CharStr	RW	NRAM	**Input object** specifies the input object to be broadcast.
RB	53826	Boolean	RW	NRAM 0	**Receive Broadcast?** specifies that the controller should accept broadcasted values for **CV**.

A.34 BROADCAST 7

Property	Identifier #	Data Type	Access	Default	Description
object_identifier	75	BACnet ObjID	RO	- Proprietary (240), Instance 7	a numeric code that is used to identify the object.
object_name	77	CharStr	RW	NRAM Broadcast 7	represents a name for the object that is unique internetwork-wide.
object_type	79	BACnet ObjType	RO	- Proprietary (240)	indicates membership in a particular object type class.
BE	49733	Boolean	RW	NRAM 0	**Broadcast Enable** specifies whether the controller should broadcast the value of CV over the network. 0=No 1=Yes
BZ	49498	Boolean	RW	NRAM 0	**Broadcast Zone/Global** specifies whether the controller's broadcasts are sent to the zone or broadcast globally. 0=Zone broadcast 1=Global broadcast
CV	50006	Real	RW	RAM	**Current Value** indicates the current value of the referenced object specified in **IC** and **IA**.
IA	51521	CharStr	RW	NRAM 0	**Input property** specifies the property associated with the object specified in **IC** to be broadcast.
IC	51523	CharStr	RW	NRAM	**Input object** specifies the input object to be broadcast.
RB	53826	Boolean	RW	NRAM 0	**Receive Broadcast?** specifies that the controller should accept broadcasted values for **CV**.

APPENDIX B: SPL ERROR CODES

This section lists the possible error codes that can be generated by a running SPL program.

Code	Trappable	Meaning	Description
1	×	**Stack Overflow**	A value larger than the largest value supported by the program executor was encountered.
2	×	**Stack Underflow**	A value smaller than the smallest value supported by the program executor was encountered.
3	×	**Invalid Format String**	The PLB file is corrupted. The line of SPL is invalid.
4	×	**Invalid Coercion**	An attempt was made to coerce a variable to an incompatible datatype.
5	×	**Expression Stack Overflow**	Internal SPL Error.
6	×	**Expression Stack Underflow**	Internal SPL Error.
7	×	**Invalid Expression State**	Internal SPL Error.
8	×	**Invalid PCode**	The function or statement is invalid. Either the function is unsupported or the statement was used incorrectly.
9	×	**Invalid Term**	A reference or attribute has been misused.
10	×	**Not Implemented**	The function or statement is not implemented in the controller.
11	×	**On Goto**	The GOTO statement specified a target which is out of range.
12	×	**Bad Reference**	A reference in the source code is out of range.
13	×	**Invalid Datatype**	The referenced datatype does not match the datatype being written.
14	×	**Format Mismatch**	The PLB file is corrupted. The SPL expression does not match the required parameter format.
15	×	**Invalid Operator**	A unsupported operator was encountered during the execution of the program code.
16	×	**Table Read Only**	An attempt has been made to write to a table which is read only.
17	×	**Nesting Overflow**	The maximum number of nested expressions has been exceeded.

Code	Trappable	Meaning	Description
18	✕	**Queue Full**	The network transmission has been denied due to the transmission queue being full.