ALERTON®

Programmer's Guide and Reference

BACtalk Systems

 LTBT-TM-PRGRMR Rev. 0008

User agreement and limited warranty

IMPORTANT - PURCHASE OF ALERTON PRODUCTS OR USE OF SOFTWARE, FIRMWARE AND / OR ACCOMPANYING DOCUMENTATION (DEFINED BELOW) IS SUBJECT TO LICENSE RESTRICTIONS AND LIMITED WARRANTY. CAREFULLY READ THIS AGREEMENT BEFORE USING ALERTON PRODUCTS, SOFTWARE, FIRMWARE AND/OR DOCUMENTATION.

This is a legal "Agreement," concerning the purchase of Products and use of Software, Firmware and/or Documentation, between you, the "User" (either individually or as an authorized representative of the company that is purchasing, has purchased, or is using the Products, Software, Firmware or Documentation) and Alerton, a division of Novar Controls Corporation, 6670 - 185th Avenue NE, Redmond, Washington 98052 USA. ("Alerton").

PURCHASE OF ALERTON PRODUCTS OR USE OF SOFTWARE, FIRMWARE AND / OR ACCOMPANYING DOCUMENTATION INDICATES USER'S COMPLETE AND UNCONDITIONAL ACCEPTANCE OF THE TERMS AND CONDITIONS SET FORTH IN THIS AGREEMENT.

Alerton provides Alerton products ("Products"), software programs ("Software"), firmware, e.g., protocols, software program code, device drivers and related hardware ("Firmware") and accompanying documentation ("Documentation") and grants a non-exclusive and non-transferable license ("License") to User to use the Software and the Firmware only on the following terms and conditions. Taken together, Products, licensed Software, licensed Firmware and accompanying Documentation are collectively defined as "Alerton Product(s)" in this Agreement.

1. Copyright. The Software, Firmware and Documentation are copyrighted and protected by United States copyright laws and international treaty provisions and laws, contain valuable proprietary products, information and trade secrets, and shall remain the property of Alerton. User may not and shall not copy or otherwise reproduce or make available to any other party any part or all of the Software, Firmware or Documentation nor decompile, disassemble, reverse engineer, manufacture or modify any portion of the Products, Software, Firmware, Documentation or any portion of the same for any purpose or otherwise attempt to determine the underlying source code of the Software or Firmware or permit any such action; provided however, User may either (a) make one (1) copy of the Software solely for backup or archival purposes, or (b) transfer one (1) image of the Software to a single hard disk, CD or other comparable media, provided User keeps the original solely for backup or archival purposes.

2. License. User is hereby licensed to use one (1) copy of the Software for User's own use in operating the Products. User may not rent, lease or otherwise assign or transfer all or any part of the Software, Firmware or Documentation. In addition, User may not sublicense, assign or transfer this License or Agreement, or any part thereof. Any attempt to do so shall terminate this License and User's right to use the Software and Firmware and shall subject User to liability for damages to Alerton. LICENSING TO USER OF THE SOFTWARE AND FIRMWARE COMMENCES WHEN USER USES THE SOFTWARE, FIRMWARE AND / OR ACCOMPANYING DOCUMENTATION.

3. Copies, Modification or Merger. Except as specifically set forth in Paragraph 1, User may not copy, modify, transfer all or any portion of the Software, Firmware or Documentation or merge it or them into another program, unless expressly authorized in advance in writing by Alerton. User must, as a condition of this License, reproduce and include the identifying marks, copyright and proprietary notices on any permitted copy of the Software, Firmware and Documentation. "Copies" shall include, without limitation, any complete or partial duplication on any media, adaptations, translations, compilations, partial copies within modifications, mergers with other material from whatever source and updated works. User will use its best efforts to prevent any unauthorized copying or other activity with respect to the Software, Firmware and Documentation.

4. Third-Party Beneficiary. For any software or other technology under this Agreement licensed by Alerton from Microsoft(or other licensors, Microsoft or the applicable licensor is a third party beneficiary of this Agreement with the right to enforce the obligations set forth in this Agreement.

5. Warranty. Alerton warrants Alerton manufactured or produced Alerton Products to be materially free from defects and to substantially conform to Alerton's published specifications for a period of twenty-four (24) months from date of shipment from Alerton (the "Product Warranty Period"). This entire Section 5 is defined as the "Warranty."

Alerton also warrants Alerton Products that it has previously repaired or replaced for the greater of ninety (90) days from the date of their shipment from Alerton or the remainder of the Product Warranty Period of the originally shipped Alerton Product (the "Repair/Replacement Warranty Period").

During the Product Warranty or Repair/Replacement Warranty Period, Alerton will repair or replace the applicable Alerton Products without charge and will add applicable engineering changes and upgrades.

This Warranty only applies to defective materials and workmanship of Alerton Products and excludes defects that result from misuse, neglect, improper installation, unauthorized repair or alteration, damage during or after shipping, accident and/or misapplication of such products. This Warranty does not apply to parts, equipment, software, firmware, components, documentation or any other item that Alerton does not manufacture or produce. This Warranty is also voided by removal or alteration of Alerton Product identification labels.

Alerton's sole responsibility with respect to Alerton Products shall be, within the applicable Product Warranty Period, to furnish a replacement Alerton Product (FOB factory) or, at the option of Alerton, to repair and return (FOB Factory) the defective Alerton Product. ALERTON HEREBY EXCLUDES ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE AND ALL OTHER EXPRESS OR IMPLIED WARRANTIES WHATSOEVER WITH RESPECT TO ALERTON PRODUCTS. In no event shall Alerton be liable for personal injury, loss of profit, loss of production, loss of business or goodwill, business interruption, loss of business information or data, loss due to delays, any other pecuniary loss, any cost or liability of Users or any other parties, to themselves or to others, increased or uncovered operating or fixed costs, inefficiency, or any other special, exemplary, consequential, incidental, indirect or remote damages in any manner, directly or indirectly, related to design, manufacturing, supply, installation or use of, or inability to use, Alerton Products, or any other act or failure to act by Alerton or its agents or contractors.

ALERTON MAKES NO CLAIMS OR WARRANTIES WITH RESPECT TO THE SOFTWARE OR THE FIRMWARE AND SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE AND EXPRESS OR IMPLIED WARRANTIES THAT THE OPERATION OF THE SOFTWARE OR FIRMWARE OR ANY PORTION THEREOF WILL BE INTERRUPTION OR ERROR FREE. Notwithstanding anything to the contrary contained in this Warranty, Alerton shall not be liable to Users or any other parties for any damages, including, but not limited to consequential, incidental, indirect, special, exemplary remote or pecuniary damages and any stated or express warranties set forth in this warranty are in lieu of all obligations or liability for any damages arising out of or in connection with the use or performance of, or inability to use, Alerton Products and the licensed Software and Firmware.

User's exclusive remedy and Alerton's entire liability arising from or in connection with the Alerton Products, Software, Firmware, Documentation and/or this License and Agreement (including, without limitation, any breach of any warranty, express or implied) shall be, at Alerton's option, the repair or replacement of the Products or Software or Firmware as applicable, as stated above. ACCORDINGLY, ALERTON AND ITS DESIGNATED DEALERS AND THEIR DESIGNATED ASSOCIATE DEALERS HAVE EXCLUDED AND DISCLAIM ANY AND ALL IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE AND ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, WHATSOEVER, WITH RESPECT TO THE PRODUCTS, THE SOFTWARE, THE FIRMWARE, THE DOCUMENTATION AND/OR THE LICENSE. USER HEREBY ACKNOWLEDGES THE SAME.

6. Remedies of Alerton. IF USER BREACHES THIS AGREEMENT, USER'S LICENSE HEREUNDER SHALL BE AUTOMATICALLY TERMINATED. Upon termination, User shall return the Software, Firmware and all Documentation to Alerton and destroy any copies of the Software, Firmware and the Documentation or any portions thereof which have not been returned to Alerton, including copies resident on electronic or digital media. If User breaches this Agreement, Alerton shall be entitled to all damages suffered by Alerton resulting from such breach and Alerton shall be entitled to equitable and injunctive relief in addition to all other remedies at law. In this regard, User acknowledges that its breach of any provision of this Agreement will cause Alerton immediate and irreparable injury for which there are inadequate remedies at law. The prevailing party in any dispute concerning this Agreement shall be entitled to the costs of collection and enforcement, including but not limited to reasonable attorneys' fees, court costs and all necessary expenses, regardless of whether litigation is commenced.

7. Export. Alerton Products are subject to regulation by local laws and United States government agencies, which prohibit export or diversion of certain products, information about the products, and direct products of the products to certain countries and certain persons. User agrees that User will not export in any manner any Alerton Product or direct product of Alerton Product, without first obtaining all necessary approval from appropriate local and United States government agencies.

8. RESTRICTED RIGHTS NOTICE. Alerton Products, Software, Firmware and Documentation have been developed entirely at private expense and are commercially provided with RESTRICTED RIGHTS. Use, duplication or disclosure by the U.S. Government or a U.S. Government subcontractor is subject to the restrictions pursuant to DFARS 227.72013 (October 1988) and DFARS 52.227-19 (June 1987), as amended and as applicable. Manufacturer, licensor and publisher is Alerton, a division of Novar Controls Corporation, 6670 - 185th Avenue NE, Redmond, Washington 98052 USA.

9. Statute of Limitations. No action for any breach of a warranty, if any, deemed or actual, may be commenced more than one (1) year following the expiration of such warranty.

10. Other. User further agrees that this Agreement is the complete and exclusive statement of the agreement between User and Alerton and supersedes any proposal or prior agreement or any other communications between Alerton or any of its representatives and User relating to the use of the Software, Firmware, Documentation and purchase of the Products. This Agreement may only be modified by a physically signed writing between User and Alerton. Waiver of terms or excuse of breach must be in writing and shall not constitute subsequent consent, waiver or excuse. If any provision of this Agreement is finally determined to be unenforceable, the remaining provisions shall remain in effect. The laws of the State of Washington and the United States, including U.S. copyright laws, shall govern this Agreement. Venue in the event of any suit, proceeding or claim shall be in the courts located in King County, Washington, USA. If User has any questions regarding this Agreement, User may contact Alerton by writing Alerton at the above address.

This Agreement shall inure to the benefit of and be binding upon the parties and their successors, administrators, heirs and permitted assigns. Notwithstanding any termination of this Agreement and not in limitation of any other provision of this Agreement, User shall specifically continue to be fully obligated to comply with all of the requirements of paragraphs one (1) through four (4), as if the Agreement were not terminated and all remedy provisions hereunder shall apply to any breach of such obligations.

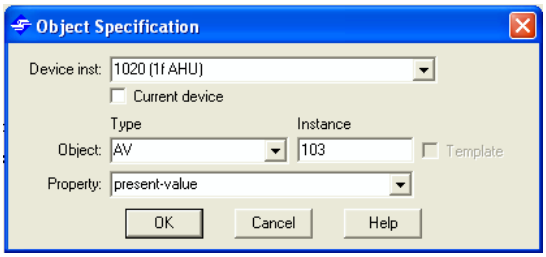
Contents

About BACnet, BACtalk, and DDC 1

This chapter describes BACnet, BACtalk and its components, and DDC programming and how it's implemented in BACtalk controllers.

About BACnet

BACnet identifies all information in terms of properties and objects. An object might represent a physical input or output, or it may represent something more abstract, such as a setpoint. And each property of the object provides data to describe something about the object. All data in a BACnet system is identified in this way. The most common property of almost all objects is the present-value. The property of an object is equivalent to what is traditionally known in control systems as a data point.



Three elements identify the source of a data point in a BACnet system:

- Device instance
- Object ID
- Property

Each object has an instance number that, along with its type, forms the object ID (for example, AI-1, represents Analog Input 1). This object ID allows the BACnet system to identify and use data.

For example, in a VAV–SD (a VAV box controller), AI-1 is a physical input. Its most important property is the room temperature, which is conveyed by its present-value property. Other properties of the object convey more information: the units property tells the system that the value is in degrees F and the description property that it is a space temperature.

All objects have some required properties and some that are optional. You can examine the device's protocol information conformance statement (PICS) to determine which objects a device supports. See the BACnet specification for more information about BACnet and PICS.

About BACtalk

BACtalk is Alerton's BACnet-compliant system. Operator workstations, global controllers, expandable controllers, and VLCs together make up a BACtalk system. All BACtalk components make their operational data available to other BACnet-compliant devices according to the BACnet standard. See

All BACnet-compliant devices in a BACnet system are identified by Device Instance, a unique numeric identifier that enables you to reference data in the device.

The BACtalk operator workstation communicates to other BACnet-compliant devices over an Ethernet local area network (LAN), WAN using BACnet/IP, point-to-point (PTP) modem, or serial connection using the BACnet protocol.

Envision for BACtalk<-------

Custom displays enable operators to command any BACnet-compliant device. Programming environments for BACtalk controllers enable developers to customize the sequence of operations for devices using DDC and VisualLogic.

BACtalk controllers

There are three classes of controllers, and each has different capabilities with respect to DDC and building automation features.

Global controllers Execute DDC and host building automation features such as schedules, trendlogs, and alarms. They orchestrate the operations of other controllers and have no direct input/output (I/O) capability associated with them. The BTI, BTI-100, and BACtalk Control Modules (BCM) are examples of global controllers. BACtalk ports, such as the BTP-MODBUS, are also global controllers.

Expandable controllers Execute DDC and host building automation features much like a global controller. An expandable controller is a hybrid of a global controller and VLC. The VLX is an example of an expandable controller.

automation features such as schedules, trendlogs,alarms, optimum start, demand limiting<----------

VisualLogic controllers (VLCs) Also known as unitary or field controller, VLCs execute DDC and support I/O. VLCs do not support locally stored automation features, relying on global controllers to supervise these functions.

About DDC

The programmable logic that controls the sequence of operations in BACtalk devices is called DDC (direct digital control). DDC sequences are stored and carried out in BACtalk controllers. Similarly, automation features—such as optimum start, demand limiting, trendlogs, schedule, and alarms—that you set up in Envision for BACtalk are downloaded and stored in host controllers (that is, a BTI,BTI-100, VLCP, VLX, or BCM).

Note For more information about how a DDC program loops and executes, see Figure 1 on page 18.

DDC programming environments

The Operate workstation software-Envision for BACtalk,include programing tool: VisualLogic(which requires Microsoft Visio)

DDC sequences are authored and downloaded to BACtalk controllers using Envision for BACtalk. BACnet objects and properties in these controllers are visible to any other BACnet-compliant devices. Envision for BACtalk has three environments for authoring DDC sequences and managing their execution in BACtalk field controllers: VisualLogic (which requires Microsoft Visio), Global/Building Controller DDC, and VLC DDC.

VLCs DDC files name:*.bd4; Global and Expandable controllers DDC files name: *.bd3

VisualLogic A graphical DDC programming environment you can use to manage and author DDC files for all BACtalk controllers that execute DDC. It requires Microsoft Visio. Use VisualLogic if you're new to DDC programming, if you need to create drawings concurrently with your DDC, or if you're familiar with Windows-based applications. Files authored for VLCs in VisualLogic are compatible with the VLC DDC programming environment. Likewise, DDC files authored for global controllers, expandable controllers, and BACtalk control modules are compatible with the Global/Building Controller DDC programming

environment. A DDC sequence saved or loaded in one environment can be opened and viewed in the other.

Global/Building Controller DDC Based on Alerton's long-standing DDC programming environment. It is used to program all global controllers (BCMs and BTIs) and expandable controllers (VLXs). Use Global/Building Controller DDC if you're familiar with DDC programming in the IBEX product line, or if you're more comfortable with a DOS-style, line-item environment. Working from an initial DDC diagram, experienced programmers can use the ten-key pad and F-key file management features to enter DDC with amazing quickness.

VLC DDC An environment similar to Global/Building Controller DDC, but used exclusively for VLCs.

Understanding VLC DDC in the BACnet environment

VLCs provide terminal unit control; they monitor inputs and command outputs directly through electrical connections to equipment.

Every VLC is fully programmable, although many VLCs are designed for specific applications and have downloadable, standard operating sequences created by Alerton (Alerton Standard applications). Values are stored differently in C3-series and Gen4 VLCs.

Storage of values in C3-series VLCs

C3-series VLCs have two types of memory: EEPROM and RAM. VLC EEPROM is capable of storing data indefinitely, even through a power outage. DDC programming is saved in EEPROM and executes in RAM. All BACnet setup data is also saved in EEPROM. This way, even if a power outage or other disaster occurs, the DDC programming is retained in EEPROM and begins execution on return to a normal state.

Additionally, the properties of certain objects (AVs, BVs, AO and BO priority arrays) are backed up in VLC EEPROM. Some object properties are stored in RAM and backed up in EEPROM on power-down. This ensures that after a power-down, these values will be reestablished on return to power.

Note Objects backed up in EEPROM should be used for critical values, such as setpoints and offsets that need to be reestablished after a power outage.

Other object properties are stored directly in EEPROM. Each has its advantages. The EEPROM-stored object properties should be used for critical setpoints for maximum reliability.

Only RAM-stored object properties can be written to using VLC DDC. RAM-stored object properties should also be used for values that will change frequently.

The present-value of AIs and BIs are not stored in EEPROM. Instead, these inputs are read when the VLC returns to normal operation.

EEPROM-stored values (AVs 50–89) The only points stored directly in EEPROM in the VLC are AVs 50–89. These EEPROM-stored values can't be written to in VLC DDC (that is, they can't appear on the output of a device). The one exception to this rule is Function 44: Runtime Accumulator.

This limitation is designed to protect VLC EEPROM, which is limited to a lifetime of 100,000 write cycles, from errant DDC. For example, if an errant DDC program caused the value of an EEPROM-stored point to change state every cycle of the DDC. The DDC would exceed the 100,000 write limitation in less than 3 hours.

Note You will start to have indeterminate problems with a device if you exceed the 100,000 write limitation.

Storage of values in Gen4 VLCs

Gen4 VLCs have two types of memory: RAM and flash. VLC flash memory is capable of storing data indefinitely, even through a power outage. DDC programming is saved in flash memory and executes in RAM. All BACnet setup data is also saved in flash. This way, even if a power outage or other disaster occurs, the DDC programming is retained and begins execution on return to a normal state.

Additionally, the properties of certain objects (AVs, BVs, AO and BO priority arrays) are backed up in VLC flash memory. Some object properties are stored in RAM and backed up in flash memory on power-down. This ensures that after a power-down, these values are reestablished on return to power.

The present-value of AIs and BIs are not stored in flash memory. Instead, these inputs are read when the VLC returns to normal operation.

RAM space limitations for certain VLC DDC functions

Some VLC DDC functions store data in VLC RAM between each pass of DDC. VLCs have a total RAM capacity of bytes and bits for this purpose.

- v3.01 (only used in the VAV-SDA), v1.15, and earlier versions have a 72 byte and a 39 bit limit
- v 2.04 has a 206 byte and a 143 bit limit
- v4.0 has a 248 byte and a 191 bit limit

Each time you use one of these functions in DDC, the function uses some of this RAM. If the sum of either bytes or bits used by your DDC exceeds the RAM capacity, no more functions can execute.

Limit usage of devices in Table 1 accordingly so that your VLC DDC program does not exceed the byte/bit limit.

Table 1 VLC DDC functions that use VLC RAM

Function	Bytes Used	Bits Used
15: One Shot	0	1
16: Delay on Break	2	0
17: Delay on Make	2	0
20: Flip Flop	0	1
21: Anti Short Cycle Relay	2	1
22: Analog Input Comparator	0	1
23: Change of State Detector	4	0
24: Restrictor Relay	4	0
44: Run-time Accumulator	2	0
47: Sample & Hold	4	0
48: Analog to Timed Binary Converter	2	0
51: Proportional Integral Controller	4	0
52: Proportional Integral Derivative Controller	16	0
54: Floating Motor Controller with No Time-out	2	0
55: Floating Motor Controller with Time-out	4	0

Identifying and using system data 2

When you program DDC, there are six primary objects you work with—AIs, AOs, AVs, BIs, BOs, and BVs. These are binary and analog inputs, outputs, and values. You typically only manipulate the present-value of these properties in DDC. This chapter explains the programmatic treatment of some of them.

Inputs and outputs (AIs, AOs, BIs, BOs)

Inputs (AIs and BIs) are directly associated with physical electrical input connections to a field-level controller. As such, they never have values written to them and appear only on the input side of a DDC function. Conversely, outputs (AOs and BOs) are directly associated with physical electrical output connections to a field-level controller. Outputs can appear on the input or output side of a DDC function.

All field-level controllers have the same number of logical outputs (BOs and AOs), but the configuration of the hardware determines the actual number of physical outputs.

Note When you write to an AO or BO in DDC, you actually write to the priority-array at priority index 14. When you read an AO or BO in DDC, you read the present-value. See “Priority arrays” on page 17 for more information.

Values (AVs and BVs)

Values (AVs and BVs) are objects in the field-level controller used for calculated values, setpoints, timers, and lockouts—virtually any value not directly associated with a physical input or output. In DDC, how you use an AV or BV depends on how it is stored internally in the field-level controller. See “Understanding VLC DDC in the BACnet environment” on page 9 for more information.

Multistate objects (MIs, MOs, and MVs)

Some non-Alerton devices—such as fans with high, medium, low, and off settings—use multistate objects. When you program DDC, you work with multistate objects in the same way you work with analog and binary objects.

However, there are a couple of things to consider when working with multistate objects:

- All DDC must be programmed using ordinal values. DDC editors cannot display the text corresponding to the values.
- Multistate inputs, outputs, and values are listed as either I/O point or BACnet Object options in VisualLogic and Global/Building Controller DDC.

Special point types available in DDC

In addition to the typical I/O points you work with in DDC (the present-value of AVs, BVs, AIs, AOs, BIs, and BOs), there are additional data points available. These are listed in Table 2.

Table 2 Special data points available in DDC

Point Type	Availability	Remarks
Branch	Global Controller (0-2047) VLC (0-7)	Provides storage for a data value or status.
Data	Global Controller VLC	Enables you to type a data value, either Boolean or a real number.
Initialize	Global Controller VLC	A flag that is set ON only during the first pass of DDC.
Comm Fail	VLC	A flag that is set ON only when the VLC loses communications with a global controller.
Current Time	Global Controller VLC	Provides the minutes elapsed since midnight.
MAC Address	Global Controller VLC	The decimal value of the MS/TP MAC address.
MS/TP Device Count	Global Controller	Provides a count of the devices currently communicating on the global controller's MS/TP LAN.

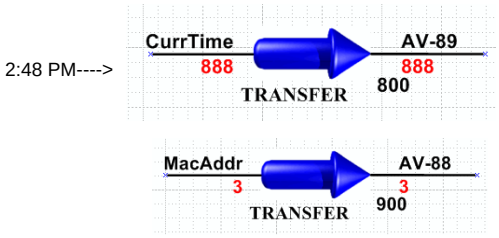


Table 2 Special data points available in DDC

Point Type	Availability	Remarks
Free Core Memory	Global Controller	Amount of core memory available, in bytes. With only ROC loaded, capacity is appx. 233908.
Free Object Memory	Global Controller	Amount of object memory available, in bytes. With only ROC loaded, capacity is appx. 2031432.
Free Paged Heap	Global Controller	Amount of free paged heap memory available, in bytes. With only ROC loaded, capacity is appx. 1048180.
Other	N/A	Not used.

Using NOT and REV on function inputs and outputs

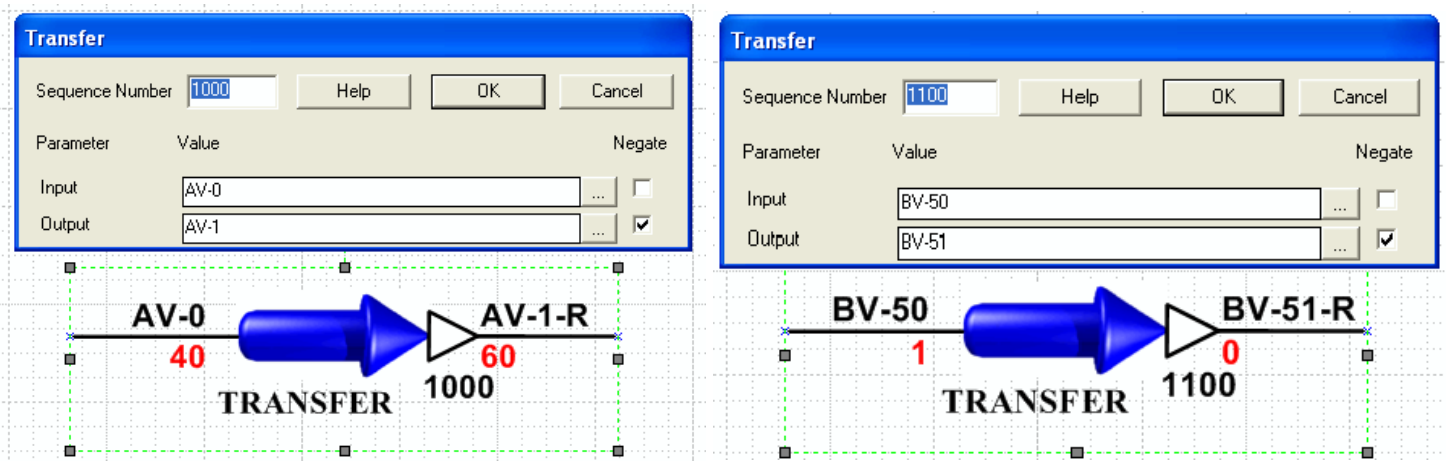
In DDC, you can choose to reverse or not an input or output. Reversing and notting are two distinct operations.

CAUTION When you view live data (F2 View Data) in Global/Building Controller DDC, values that have had a NOT or REV applied may not display correct values.

REV Reversing applies to analog-type inputs and outputs (real numbers). When an analog value is reversed, the result is a value equal to 100 minus the initial value. For example, if you reverse a function's input whose value is 20, the actual value passed to the function is 80. The same is true of outputs.

Note Binary values have analog equivalents of 1 (ON) or 0 (OFF). If you mistakenly reverse a binary value it results in values of 99 and 100, respectively.

NOT Notting applies to binary inputs and outputs. When you NOT an ON value, it results in an OFF, and vice versa.



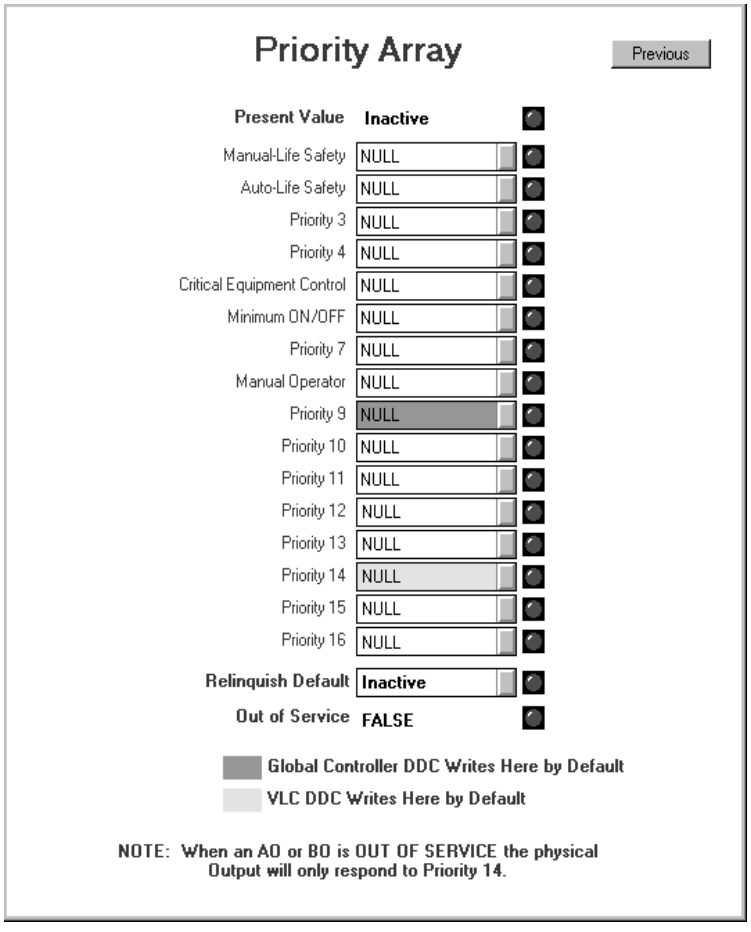
Priority arrays

BACnet uses a priority array to control the present-value property of certain objects. In BACtalk devices, only AOs, BOs, and BVs in VLCs and BVs in global controllers use a priority array. Because a number of commands may be issued simultaneously for a present-value property—for instance, an operator may command a fan ON while a schedule calls for it to be OFF—a scheme for prioritizing commands is necessary. This is achieved with the priority-array property. Other manufacturers may use a priority-array for other object types.

Every command for an AO or BO has a priority array index from 1 to 16 associated with it. Priority 1 is the highest, priority 16 is the lowest. Some priorities are designated by BACnet (priority 1, for example, is reserved for use by life/safety systems). When a command is issued for a present-value property of a BACtalk AO or BO, rather than directly affecting the present-value, the object stores the value in its priority-array property at the appropriate priority index. **The command with the highest priority drives the present-value.**

For a lower priority command to take effect, a NULL value—not an OFF value—must be written to all higher levels. For example, if a fire safety system (priority 1) writes an OFF value to a BO that controls a fan, the fan will remain OFF, regardless of commands written to that BO at priorities 2 through 16. The fire system, or some other system, must write a NULL value to the BO at priority 1 before any of the lower priority level commands will be effective.

A relinquish-default determines what status or value will take effect when all levels of the priority-array are NULL.



Subroutine DDC

You can use subroutine DDC to make your DDC program more efficient. You should program a subroutine any time a calculation or control sequence needs to be implemented repeatedly. For instance, you may need to convert fpm to cfm for a number of VAV boxes. Or you may need to control equipment in 200 hotel rooms in exactly the same way, but each must control equipment according to its own ambient conditions. These are both perfect opportunities for using subroutine DDC.

WARNING Only reference devices in subroutine DDC if they are connected to the MS/TP network on the host controller. Additionally, you should never reference devices that do not exist (for example, devices that you plan to add int eh future). Failure to follow these guidelines can cause significant degradation in network performance and temporary communication delays of up to several minutes.

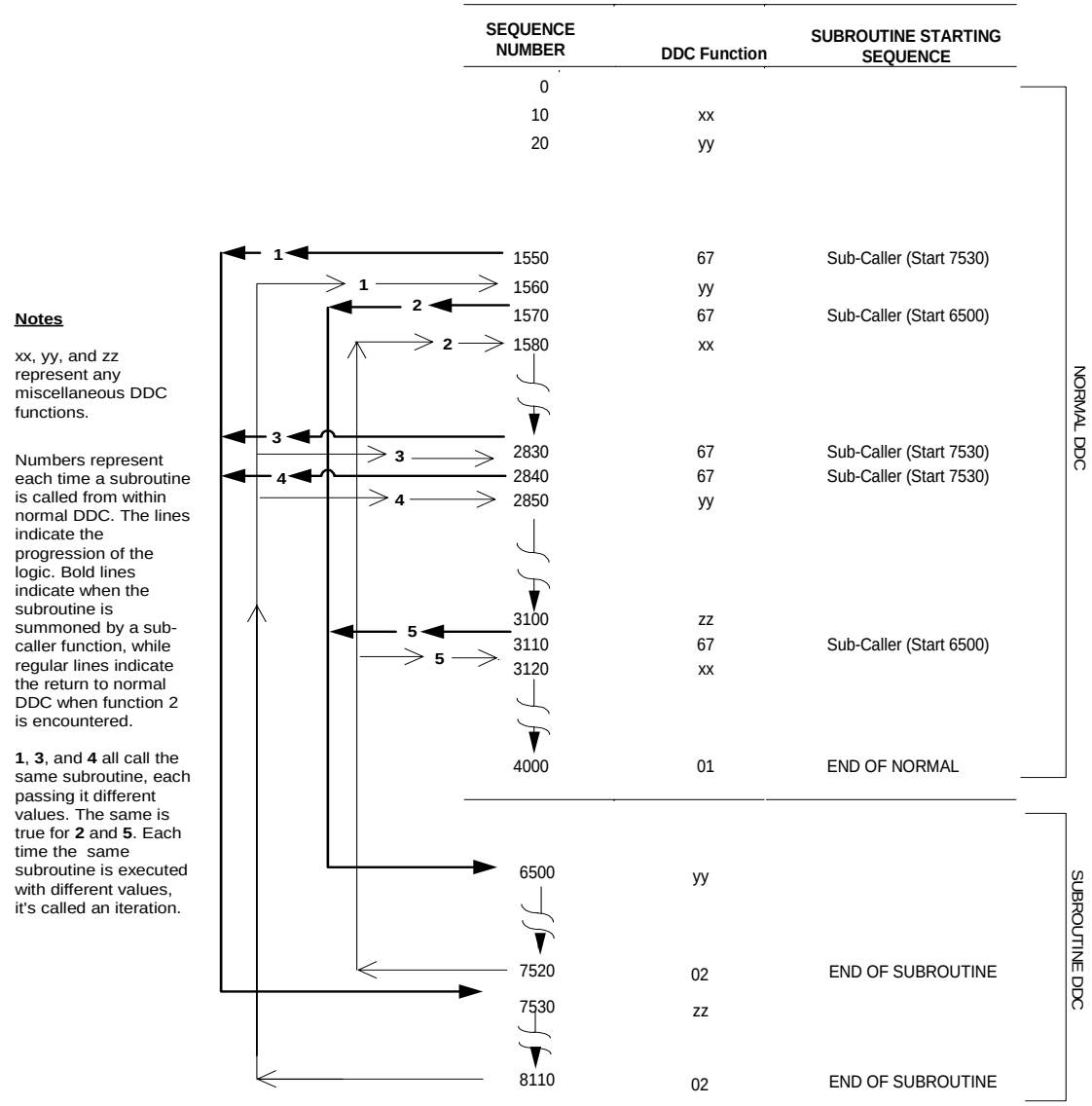
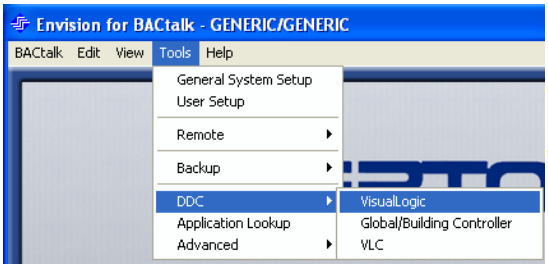


Figure 1 Graphical depiction of a DDC program execution with subroutine callers

The VisualLogic development environment

3



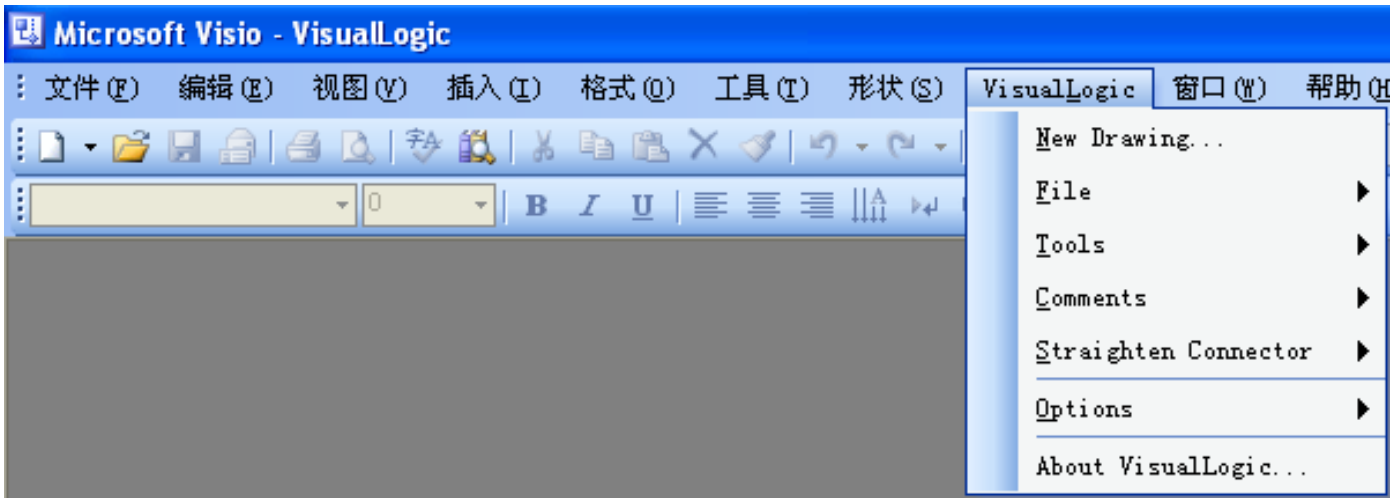
VisualLogic uses the Visio® drawing and design tool as its engine. When you start VisualLogic, custom menu items and functions in Visio enable you to author and manage DDC files in Alerton BACtalk controllers.

In VisualLogic, the DDC program and the drawing are one and the same. When you save your drawing, code for the VLC or global controller is saved along with it. This means you can generate a DDC program from a Visio drawing or convert DDC from a device into a Visio drawing.

You can use VisualLogic to:

- Author DDC in a totally graphical environment, creating documentation simultaneously as you program.
- View data in real-time from BACtalk controllers, monitoring DDC execution to test and verify operation.
- Manage DDC files on the operator workstation hard disk and in field controllers, reading and loading DDC even if the sequence was authored in BACtalk's other development environments (VLC and Global/Building Controller DDC).
- Set up the unit of measure and other object properties in global controllers and VLCs.

Note For instructions about using Visio toolbars and commands and general information about Visio, see Visio online Help or other user documentation.



The following graphic helps you identify the key elements in VisualLogic that you use to execute commands, author DDC, and set device and drawing properties.

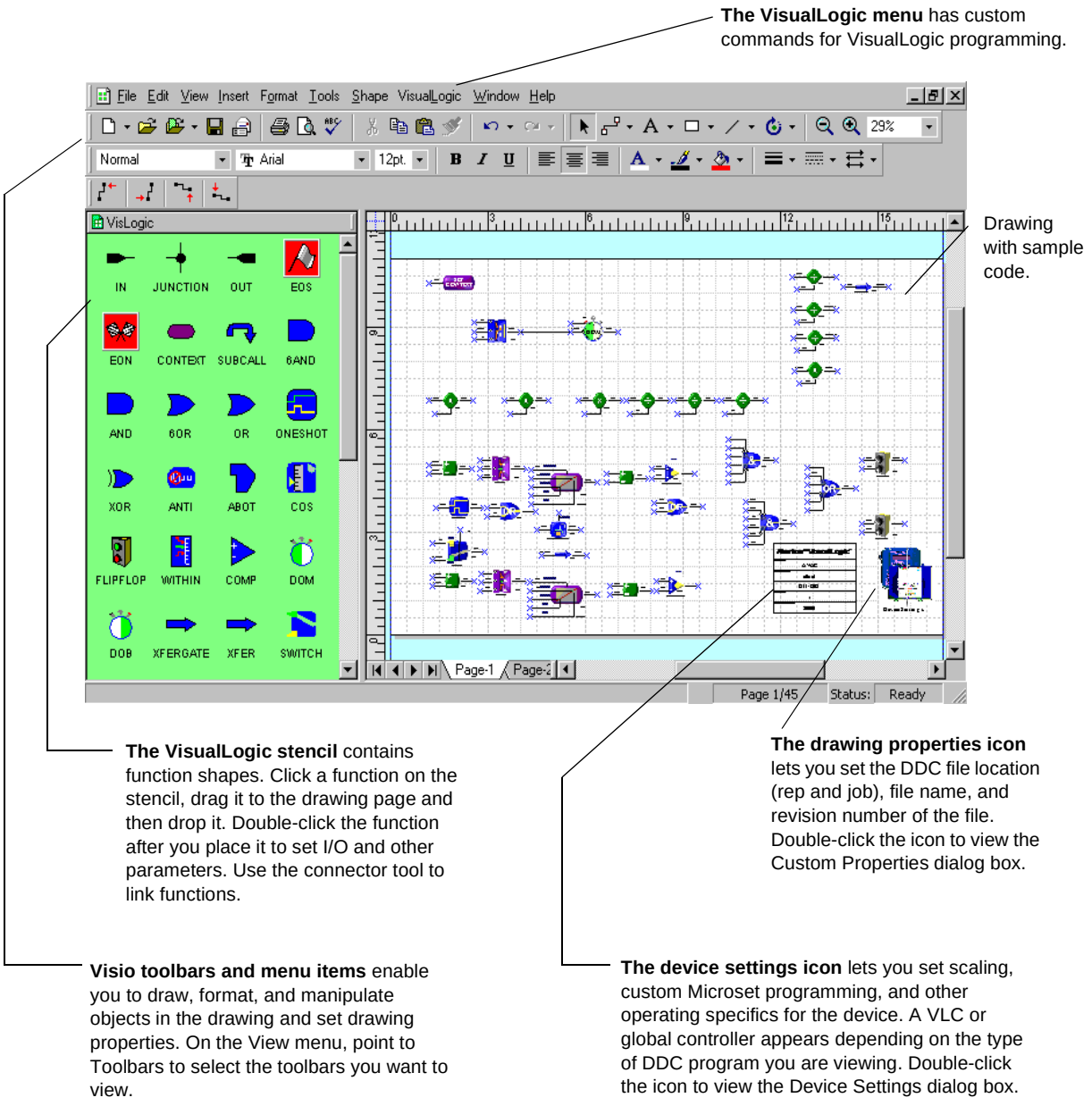
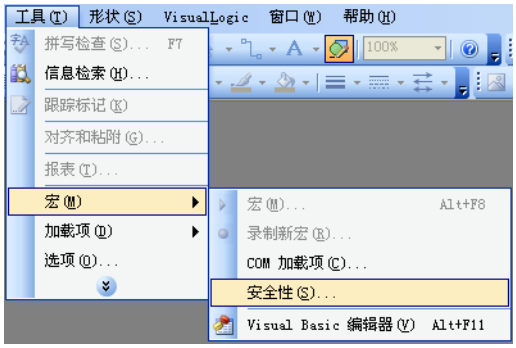


Figure 2 Key components of the VisualLogic development environment

Required Visio software and files for running VisualLogic

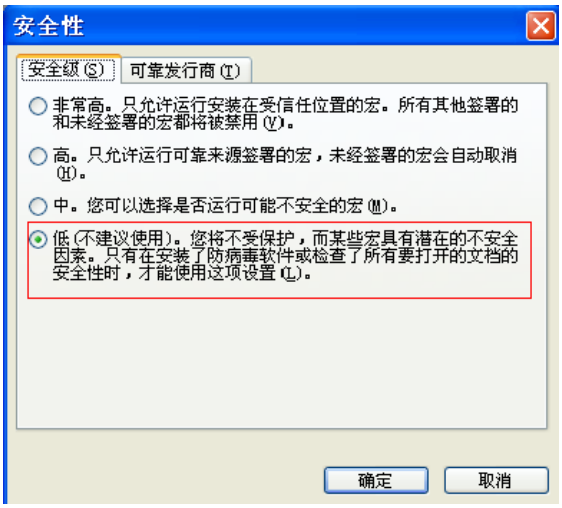


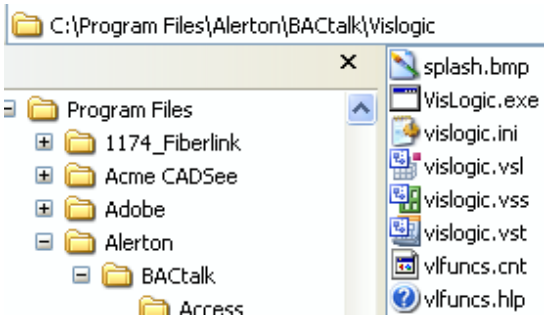
In addition to Envision for BACtalk, install one of the following Visio products to use the VisualLogic development environment:

- Visio Technical v2000 or 2002 (recommended, includes AutoCAD conversion tools and extensive engineering templates)
- Visio Professional v2000 or 2002
- Visio Standard v2000 or 2002

VisualLogic features are enabled through the Visio templates and stencils saved in the <bactalk root>\vislogic directory. These files are installed with Envision for BACtalk: vislogic.vss (stencil), vislogic.vst (template), and vislogic.vsl (program library)(See following picture).

Starting VisualLogic





When you start VisualLogic, the Visio development environment opens with a blank workspace. You then use the VisualLogic menu to either create a new VisualLogic program, load one from file, or load one from a BACtalk controller on the network.

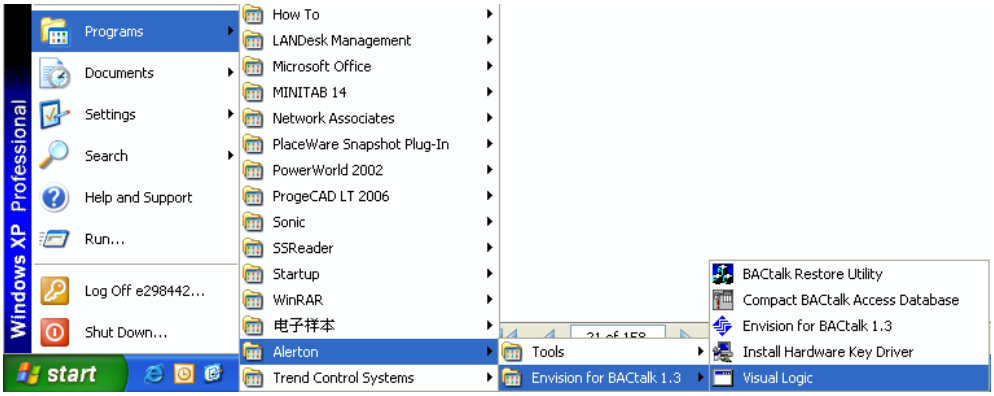
▶ **To start VisualLogic**

1. On the Tools menu in Envision for BACtalk, point to DDC, and then click VisualLogic.

   -OR-

   As long as Envision for BACtalk is running, you can start VisualLogic from the Envision for BACtalk program group.

2. Visio starts with a VisualLogic menu in the menu bar.



About DDC and drawing files

DDC files DDC files are identified by bd4 and bd3 file extensions. The bd4 extension identifies a DDC file for a VLC; the bd3 extension identifies DDC for a global or expandable controller. The two are not interchangeable; a file you create in one device format cannot be converted to the other. You can open these files from within VisualLogic or from within the appropriate DDC development environment in Envision for BACtalk—VLC DDC for bd4 files or Global/Building Controller DDC for bd3 files.

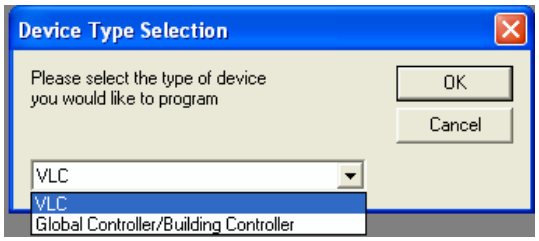
DDC files contain all DDC program information, but contain drawing information if they were created and saved in VisualLogic.

Visio drawing files To ensure that you save Visio drawing elements, in addition to saving DDC program information, you can save your work as a Visio drawing file (*.vsd) using Visio's Save command on the File menu. Visio drawing files are typically much larger than their corresponding DDC files and open more quickly. The Visio drawing file contains all drawing information as well as all DDC program information. You can't use VLC or Global/Building Controller DDC to open Visio files.

A Visio drawing file is automatically saved when you choose to send an open DDC file to a controller with the Save option selected.

Creating and opening VisualLogic files

As indicated in the previous section, you may work with a number of file formats in VisualLogic. As a general rule, to open DDC files, use commands on the VisualLogic menu; to open Visio drawing files, use commands on Visio's File menu.

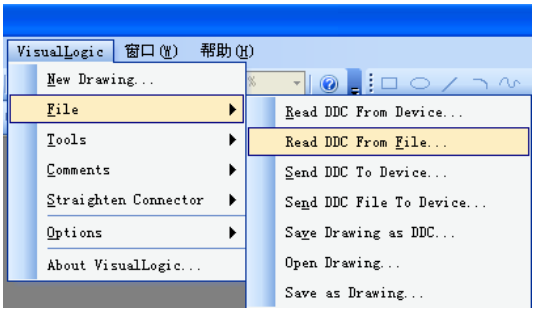


▶ To create a new DDC program in VisualLogic

1. On the VisualLogic menu, click New Drawing.

   The Device Type Selection dialog box appears, prompting you for the type of device you are programming (a VLC or global controller).

2. Select the type of device you are programming and then click OK. A new drawing opens for the type of device you selected.



▶ To open a Global/Building Controller DDC file (.bd3) or VLC DDC file (.bd4) in VisualLogic

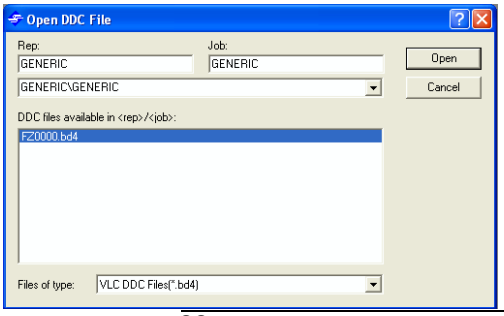
1. On the VisualLogic menu, point to File, and then click Read DDC from File.

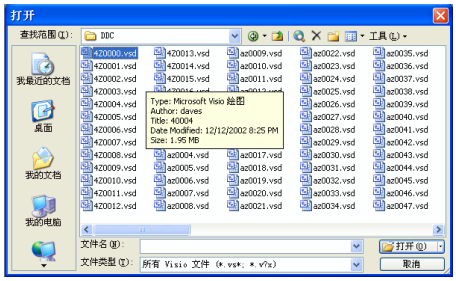
   The Create Drawing from DDC File dialog box opens to the DDC folder for the rep and job you are logged in to.

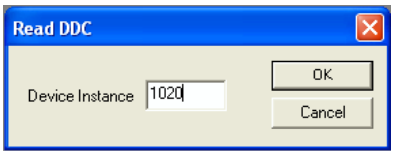
2. If necessary, select a different folder from the Look In list.

3. In the list of files, select the file name you want, and then click Open.

   VisualLogic generates a new drawing based on information in the DDC file. Conditions under which the file was created and saved dictate the rendering in VisualLogic.







▶ **To open a DDC file saved as a Visio drawing (.vsd)**

1. On the Visio File menu, click Open.
2. In the Open dialog box, select the folder and file you want, and then click Open.

▶ **To open a DDC file directly from a VLC or global controller**

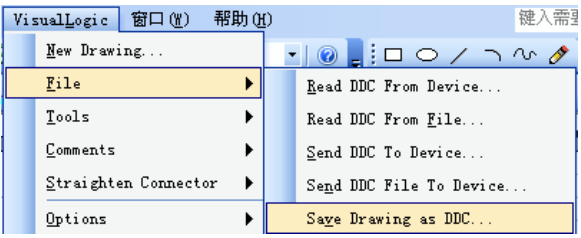
1. Make sure Envision for BACtalk is running and the controller you want to work with is online.
2. On the VisualLogic menu, point to File, click Read DDC from Device, and then type the device instance in the box provided.

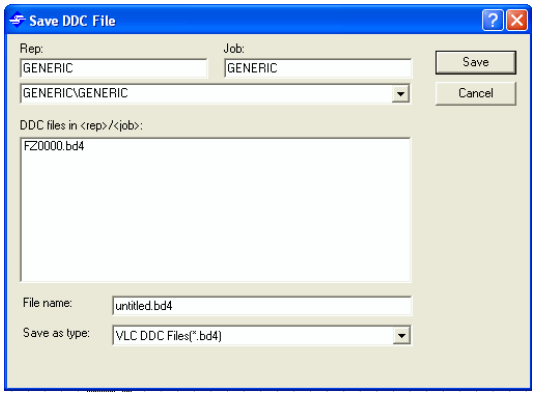
   VisualLogic constructs a drawing based on the DDC in the controller. Connectors and other geometry information may not be retrievable.

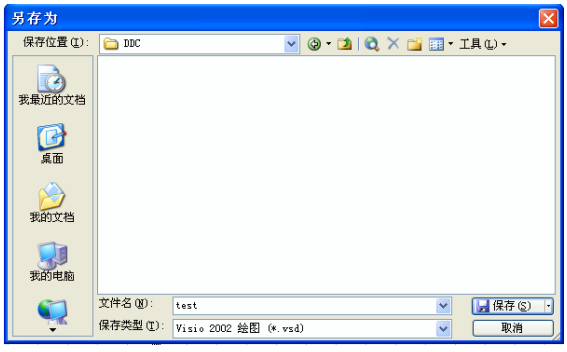
   Note It may take as long as 15 minutes to construct a more complex DDC program.

Saving your work

You have a couple of options when saving the DDC you create in VisualLogic. You can save it as a DDC file or save it as a Visio drawing file.







Saving your work as DDC

Saving your work as DDC creates Global/Building Controller DDC (.bd3) or VLC DDC (.bd4) files.

▶ **To save your work as a DDC file (.bd3 or .bd4)**

1. On the VisualLogic menu, point to File, and then click Save Drawing as DDC.

   The Save DDC as File dialog box opens to the DDC folder of the rep and job you are logged in to or as set in the Drawing Properties dialog box. The file name is based on the Program Name established in the Drawing Properties dialog box with the appropriate extension for the type of device you're programming.
2. If necessary, change the folder or file name.
3. Click Save.

Saving your work as a Visio drawing

Saving your work as a Visio drawing file (*.vsd) is the best way to ensure that shapes, connectors, and other drawing elements you added to your DDC file are saved. See “About DDC and drawing files” on page 22 for more information about Visio drawing files.

▶ **To save your work as a Visio drawing file (*.vsd)**

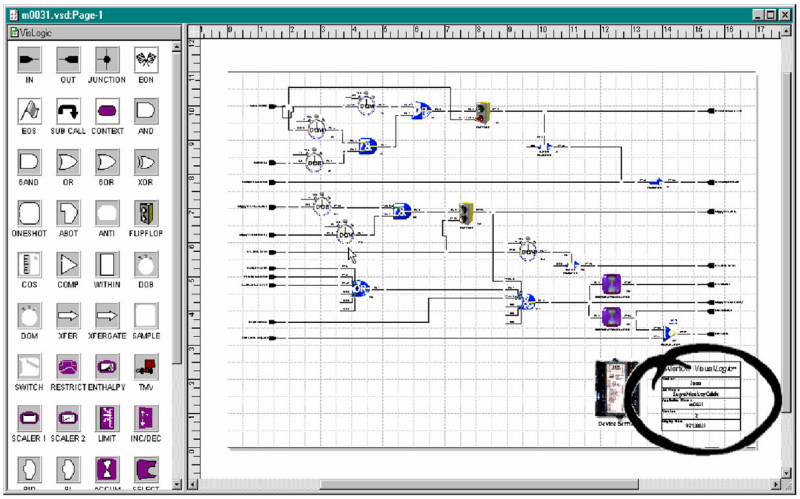
1. On the Visio File menu, click Save.
2. Select the folder you want and type a file name in the File Name box.

   Note If you don’t use the same filename and rep/job that are displayed in the title bar, it may appear not to save.

3. In the Save as Type box, make sure Drawing (.vsd) is selected.
4. Click Save.

Setting drawing properties

A title block labeled Drawing Properties appears on the first page of every DDC drawing. You can double-click the title block to view the Drawing Properties dialog box and change settings in it. You can set the rep, job, program name, and other DDC specifics. When you change drawing properties, the changes are reflected in the title block.



▶ **To set drawing properties**

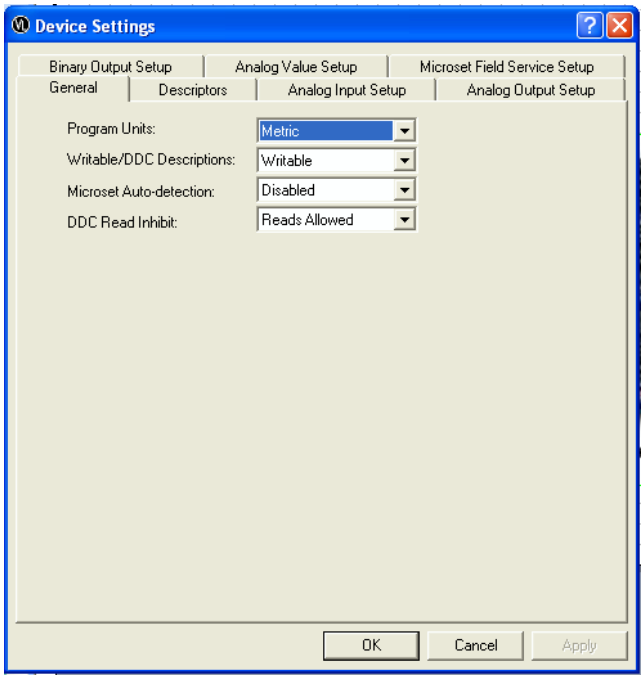
1. Double-click the title block on the first page of your DDC drawing to open the Drawing Properties dialog box.
2. Set drawing properties according to the following guidelines.

Item	Explanation
Alerton Representative and Job Name	The DDC file keeps a pointer to the rep\job folder in the BACtalk root directory, which establishes where this DDC file is saved. The default is the rep/job you are logged in to. Maximum of 8 characters.
Application Name	The file name used when you save a DDC file or Visio drawing file through the VisualLogic menu. Maximum of 8 characters.
Revision	A version number you can use to track revisions to the DDC.
Display Number	For future use.
Program Name	Not used.

3. Click OK.

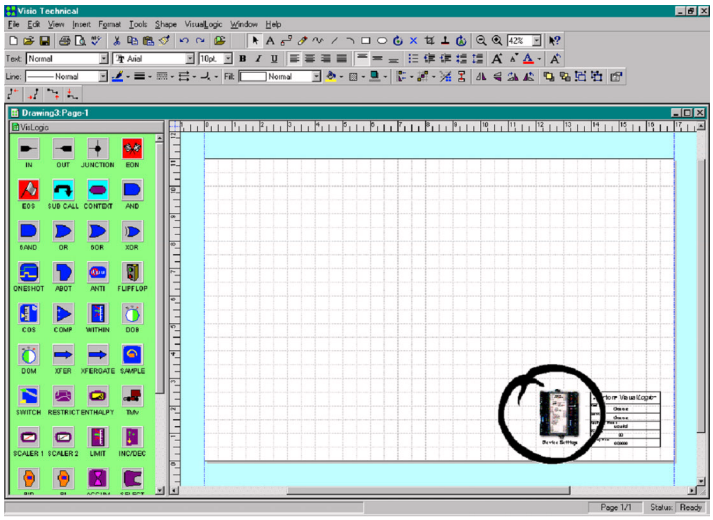
Using VisualLogic to set up VLCs

You can use the Device Settings dialog box to set up DDC header information in VLCs. You can set up input scaling, English or metric units, custom Microset codes, and more. Setup data is saved to the controller along with the DDC. If you modify any setup parameters, make sure you send the DDC to the controller.



▶ **To view and edit device settings**

- Double-click the device settings icon, which appears on the first page of every DDC diagram.



Setting object descriptors

You can assign descriptors to the AV, BV, AI, BI, AO, and BO objects in C3-series VLCs. It is a good practice to thoroughly document your programming; you and others can use the descriptors to identify and interpret how the program uses certain objects.

▶ **To assign object descriptors**

1. Click the Descriptors tab in the Device Settings dialog box.
2. In the list of descriptors, double-click the bracketed line to assign a new descriptor to an object, or click an existing item to change it.
3. In the Descriptor Entry dialog box, select the object type and object instance, and type a description in the boxes provided, and then click OK.

   The description appears in the list of descriptors.

▶ **To delete an object descriptor**

- Select the descriptor you want to delete in the list of descriptors and then press the Delete key.

Setting program units

In the Device Settings dialog box for VLCs, in addition to object descriptors, you can set up program units to be English or metric. This determines how the VLC interprets 10K ohm and 3K ohm thermistor inputs as well as Microset- and Microtouch-related objects. For VAV controllers, if program units are set to metric, enter box size in cm; the device then reports flow in liters per second (lps).

The selection of English versus metric units here can be read in DDC from BV-71, which is read-only. Most Alerton Standard applications use BV-71 to modify measurement related calculations according to the program units selection.

Table 3 BV-71 settings for English/metric

	BV-71
English	OFF
Metric	ON

▶ **To set program units to English or metric**

1. Open the Device Settings dialog box.
2. Click the General tab, and then type or select English or Metric in the Program Units box.
3. Send the DDC file to the controller or save it to file.

Analog input (AI) setup

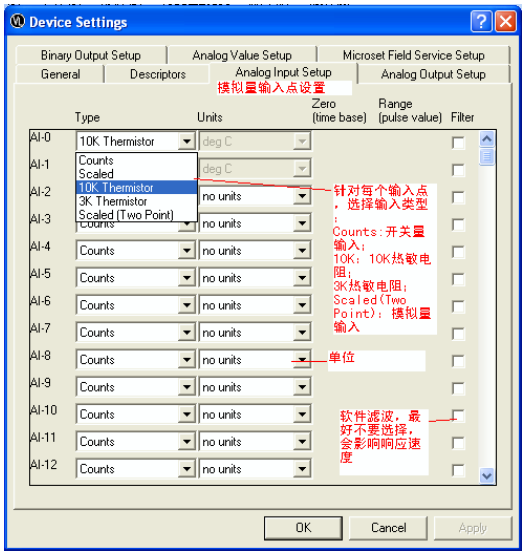
AI setup for VLCs is critical. When you set up an AI, you have several options for selecting the type of AI and its scaling. These options are applied to the AI before DDC processing in the VLC.

This topic briefly summarizes the options available for AI Setup in the Device Settings dialog box. For more detailed information about the appropriate setup for different types of hardware and notes about scaling and input options, see the bulletin TB-BTW-INCFG, Input Setup Reference for VLCs.

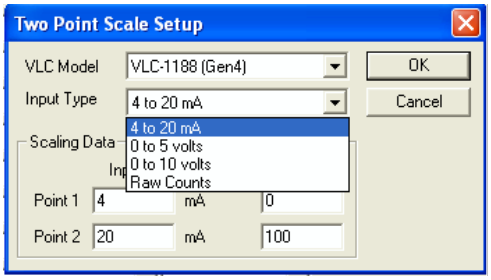
▶ **To set up AIs for a VLC**

1. Open the Device Settings dialog box.
2. Click the Analog Input Setup tab.

3. For each AI you want to work with, under Type, select the input type according to the following guidelines (see TB-BTW-INCFG for further details).



Type	Explanation
Counts	Input is read in raw counts from 0-4095.
Scaled	When you select a Scaled input type, the Zero (time base) and Range (pulse value) fields appear. AI = Zero + (Input * Range/ 4096) where “Input” is the pre-scaled input count.
10K Thermistor	Use this input type for Microset or Microtouch temperature inputs and for 10 k ohm thermistor inputs. The AI then reports temperature (in °F or °C according to the Program Units setting).
3K Thermistor	Use this input type for 3K ohm thermistor inputs. The AI then reports temperature (in °F or °C according to the Program Units setting).
Scaled (two point)	Enter two mA or voltage values and the desired AI range; VisualLogic then calculates and enters Zero (time base) and Range (pulse value) values for you.



4. Under Units, select the type of engineering units you want to assign to this input. This option is not available when 10K Thermistor or 3K Thermistor is selected as Type.

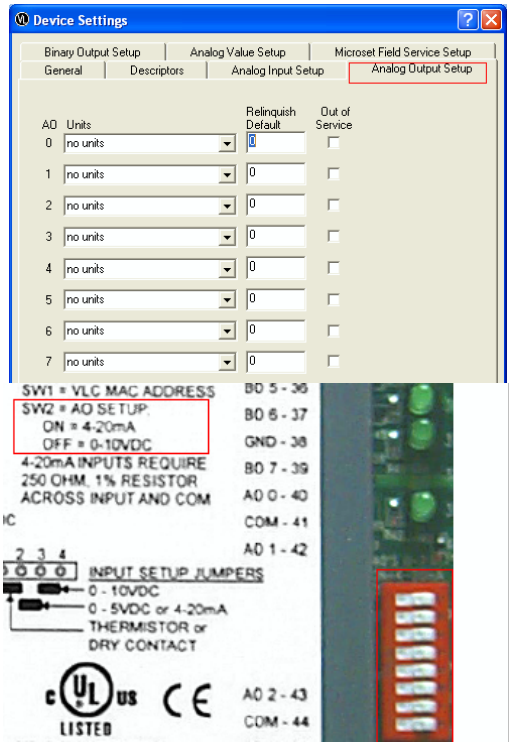
5. Type zero and range values according to the guidelines for Scaled in step 3 and the information in the *VLC Installation and Operations Guide* (LTBT-TM-GEN4VLC). These fields populate automatically if you select Scaled (two point).

6. Select the Filter check box if you want filtering applied to the input. The filter calculation is performed every 0.1 seconds and is expressed as:

$$FilteredCount = \frac{NewCount}{32} + \frac{31}{32}\langle PreviousCount\rangle$$

CAUTION Do not filter any Microset/Microtouch inputs.

Analog output (AO) setup



In VisualLogic, you can assign the unit of measure, the relinquish-default, and the out-of-service properties for each AO in a VLC. The relinquish-default and out-of-service properties are critical for priority-array operation.

NOTE: When a VLC AO is OUT OF SERVICE the physical Output will only respond to Priority 14.

▶ **To set up AOs for a VLC**

1. Open the Device Settings dialog box.
2. Click the Analog Output Setup tab.
3. Select options according to the following guidelines.

Item	Explanation
Type	Not used.
Units	Type a BACnet engineering unit ID in this box. The unit of measure indicates the unit of measure for the AO. This doesn't affect calculations; it is for display and reference purposes only.
Relinquish Default	Type a default value for the AO, which can be any real number. The relinquish default determines the value of the AO when all elements in the AO's priority-array are NULL–essentially the default value of the AO.
Out of Service	Select this check box to set the AO's out-of-service property = TRUE. The out-of-service property controls the relationship of the physical AO to its present-value. When out-of-service = TRUE, the AO is decoupled from its present-value, and the AO value is the result only of DDC execution in the VLC.

Binary output (BO) setup

In VisualLogic, you assign a relinquish-default and set the out-of-service flag for binary outputs (BOs) in a VLC. These settings affect how the BO responds to its present-value property. The relinquish-default is the status of the BO when all priority-array indexes are NULL. The out-of-service flag controls the relationship of the physical BO to its present-value. When out-of-service = TRUE, the BO is decoupled from its present-value, and the status is the result only of DDC execution in the VLC.

NOTE: When a VLC AO is OUT OF SERVICE the physical Output will only respond to Priority 14.

▶ **To set up BOs for a VLC**

1. Open the Device Settings dialog box.
2. Click the Binary Output Setup tab.
3. For each BO, if you want the BO to default to Active (ON) if all priority-array indexes are NULL, select the Active check box under Relinquish Default.
4. For each BO, if you want to decouple the BO from its present-value and consign control exclusively to the VLC's DDC, select the out-of-service check box.

Analog value (AV) setup

For AVs, the extent of setup is assigning a BACnet engineering unit code to the AV, which is optional.

▶ **To set up AVs for a VLC**

1. Open the Device Settings dialog box.
2. Click the Analog Value Setup tab.
3. For each AV, type a BACnet engineering unit code under Code.

Microset field service code setup

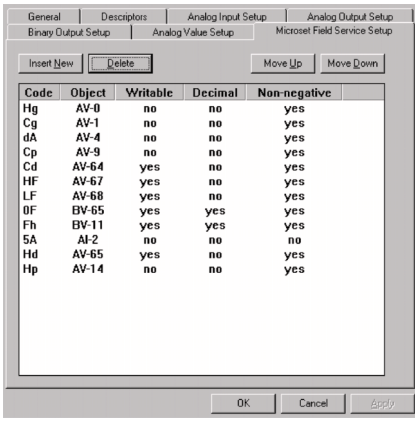
The Microset's field service mode enables technicians to query and change the value of objects in a VLC. You can customize the values that a technician can change from a Microset and choose the two-digit code that identifies data. You can also select whether a technician can change the value at the Microset or only view it, whether the value appears with a decimal, and whether negative values are acceptable.

As a technician scrolls through the list of Microset field service codes, the codes at the top of the list appear first. You can arrange codes in VisualLogic so the more frequently queried values appear first. For a list of the standard Microsoft codes, see “Microset Field Service mode custom codes” on page 155.

▶ **To set up microset field service codes**

1. Open the Device Settings dialog box, and then click the Microset Field Service Setup tab.

   A list shows the Microset codes set up for the current DDC program.



2. To add a new field service code, click the Insert New button, and then double-click the blank code entry in the list.

   —or—

   To edit an existing code, double-click the code entry on the list. The Edit Microset Entry dialog box opens.

3. In the Display Digits boxes, type or select the first and second digits you want displayed at the Microset..

Table 4 Acceptable Microset Field Service mode characters

0	1	2
3	4	5
6	7	8
9[a]	A	b
c	C	d
E	F	g
h	H	i
J	L	n
o	p	r
u	U	y
-	_	

a.The characters 9 and g are virtually indistinguishable on the Microset display.

4. In the Object boxes, type the object type and instance whose present-value you want associated with the Display Digits.

5. Select check boxes according to the following guidelines

Check Box	Explanation
Writable	Select this check box to enable a technician to change the value of object using the keys on the Microset.
Decimal	Select this check box to have decimals (one significant digit) displayed at the Microset.
Unsigned	Select this check box if the value is a positive number and will never be negative.

6. Click OK.

Authoring DDC in VisualLogic

To author DDC in VisualLogic, you drag functions from the stencil and drop them onto your drawing. Once a function is placed, you double-click it to assign inputs, outputs, sequence numbers, and other parameters.

VisualLogic doesn't require that the DDC functions appear in any order (they always execute in order of sequence number) or that they be linked with connectors. However, arranging functions in logical order makes the DDC much easier to read, and using connectors enables you to use the Check Drawing and Propagate Parameter commands.

Adding functions to your drawing from the stencil

The stencil appears on the left side of the Visio workspace. When you open VisualLogic from Envision for BACtalk, the stencil appears.

Note If you need to open the VisualLogic stencil manually, click Stencils on the File menu, and then select the stencil, vislogic.vss, in the <bactalk root>\vislogic folder.

▶ **To add a function from the stencil to your DDC drawing**

- In the stencil, click the function you want to use, drag it to where you want it to appear in your drawing, and drop it.

Setting inputs, outputs, and other function parameters

After you place a function on your drawing, double-click the function to view the setup dialog box and assign inputs, outputs, and other parameters. For more information about the function and its parameters, click the Help button in the dialog box.

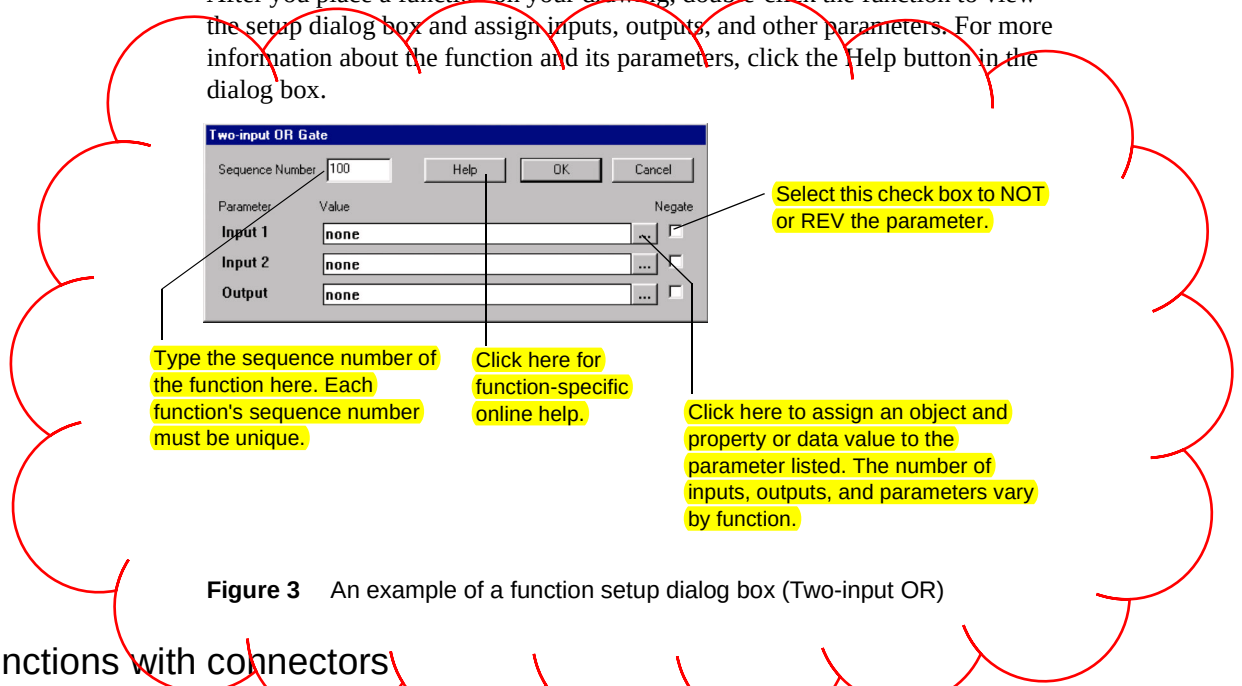


Figure 3 An example of a function setup dialog box (Two-input OR)

Linking functions with connectors

Connectors make your diagram easier to read. They also enable you to use the Propagate Parameters command. Functions do not require connectors to link; the sequence numbers and input/output assignments in the functions' setup dialog boxes are sufficient for VisualLogic to determine the order of execution and processing of values.

When you link functions with a connector, the output of one function isn't necessarily linked to the input of another. You still assign inputs and outputs in the functions' setup dialog box. The exception is when you use the Propagate Parameters command.

▶ **To link functions with connectors**

1. Click the Connector button  on the Visio toolbar.
2. Position the mouse pointer over a connection point on the source function.

   The mouse pointer appears as a rectangle.
3. Click the connection point, drag the mouse pointer to the connection point on the target function, and then release the mouse button.

   A connector appears between the functions.

Tips for working with connectors

- Use the Junction shape to create branching connections. Place the Junction shape, and then use the connector tool to connect to one of the four connection points on the shape. Zoom in so you don't accidentally use the wrong connection point on the Junction shape.
- Use the Straighten Connector tools to straighten connectors left, right, up, or down. Select the connectors you want to straighten, and then click one of the Straighten tools in the toolbar or point to Connector on the VisualLogic menu and then choose a Straighten command. Straighten connectors as a last step.

Creating program comments and generating a sequence of operations

VisualLogic has a program comment feature that you can use to document your DDC operation. You can author any number of topics and associate them with any range of functions. The comments are saved to disk (not in the controller) along with the DDC file and the Visio drawing. A user or developer can select a function and view comments about its operation.

You can save all comments for a drawing to a rich text file (*.rtf) and use it as a source for creating a sequence of operations for your job. Topics are saved to file in the order that they appear in the Program Comments dialog box. You can adjust the order of topics.

▶ **To view comments**

1. Select the function whose comments you want to view.
2. On the VisualLogic menu, point to Comments, and then click View.
   -or-
   Right-click the function whose comments you want to view, and then click View Comments.

▶ **To create comments for a function or range of functions**

1. On the VisualLogic menu, point to Comments, and then click Edit.

   The Program Comments dialog box appears.

2. Click the Add button.

   A new topic with the title "New Comment" appears and is added to the bottom of the list of topics.

3. Replace the "New Comment" title with your own text (the title automatically appears in the list of topic titles), and then type comment text.

4. In the Range box, type the sequence number of the function or the range of functions (separated by a dash, for example 10-100) you want this comment associated with.

5. Repeat steps 2-4 for each comment topic you want to add.

6. When you finish, click OK.

▶ **To change the topic sequence**

1. On the VisualLogic menu, point to Comments, and then click Edit.

2. In the Program Comments dialog box, select the topic title you want to rearrange, and then click the Up button to move it toward the top of the list or click the Down button to move it toward the bottom of the list.

   Note Topics are saved to the *.rtf file in the order of appearance in the list of topics. This order is independent of their assignment to function sequence numbers.

▶ **To save all comment topics to a rich text format file**

1. On the VisualLogic menu, point to Comments, and then click Edit.

2. In the Program Comments dialog box, click Save.

3. Select a folder and file name to save to, and then click Save.

▶ **To change the font of a comment**

1. On the VisualLogic menu, point to Comments, and then click Edit.

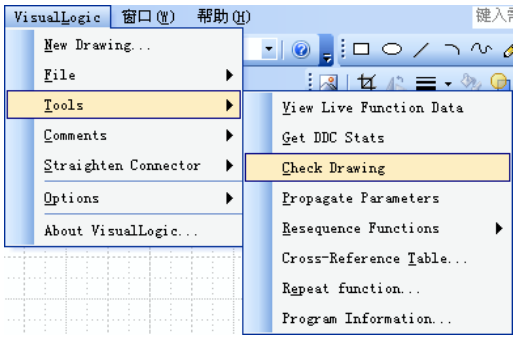
2. In the Program Comments dialog box, select the comment topic you want to work with from the list.

3. Select the text whose font you want to change.

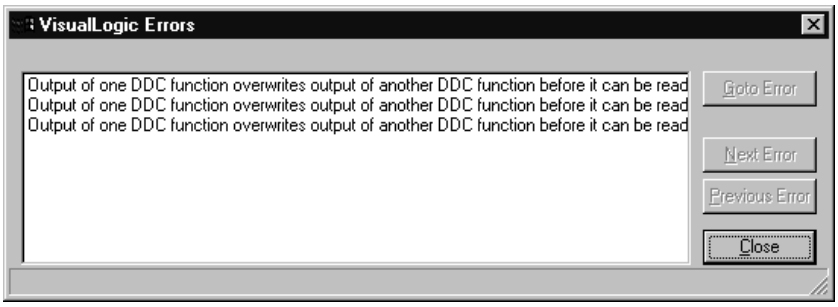
4. Click the Font button, and then choose font details in the box provided.

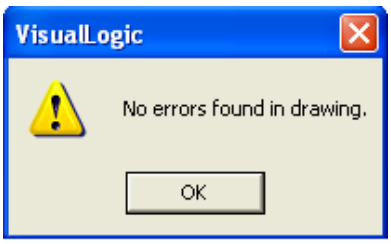
5. Click OK to apply the changes to the selected text.

Checking your drawing



The Check Drawing command searches your drawing for some common DDC programming errors: faulty I/O assignments, duplicate sequence numbers, and the lack of the proper termination functions (End of Normal, End of Subroutine), among others. It's a good idea to check your drawing before sending it to a controller.





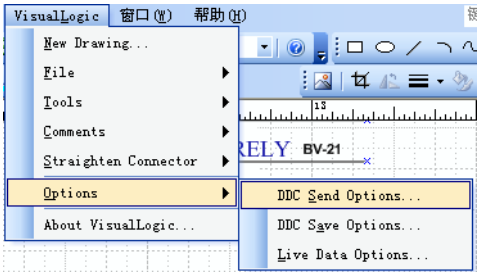
恭喜你，你的程序编译通过！

▶ **To check your drawing for errors**

- On the VisualLogic menu, point to Tools, and then click Check Drawing.

  If errors are found, select one from the list, and then click Goto Error to jump to the area of the drawing where the error occurs.

Sending the DDC file to a device



Send DDC Options SetUP

If your DDC files is too big to download to VLC,don't select them. So you must save DDC files to disk frequently!

When you have a DDC drawing open, you can send DDC to any BACtalk controller on the network. The BACtalk controller saves the DDC in its memory and executes it locally.

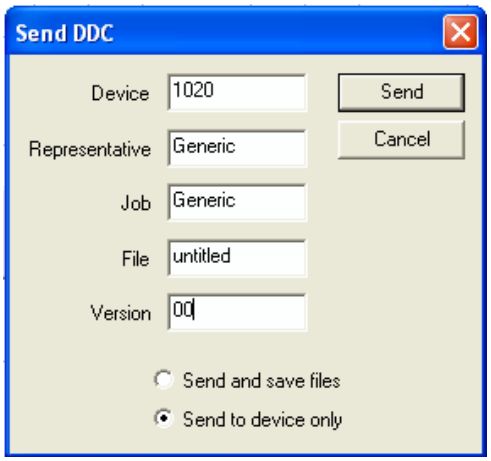
VisualLogic prompts you for information about the device and the DDC file (the rep, job, file name, and version). You can save the file to the rep and job directory in conjunction with the Send command, or you can send files without saving them (not recommended). VisualLogic requires DDC file information even if you don't choose to save. This is because the VLC retains the rep/job information along with the DDC.

CAUTION VLCs and global controllers have a limited amount of memory for DDC. Occasionally, the controller won't have enough memory to save the geometric information—the spatial relationship of functions and connectors—in the controller. When this occurs, the message appears: **There is not enough space for DDC on this device. Clear the Geometric Info option (VisualLogic > Options > DDC Send Options) and then try again**. Always save a copy of such files to disk. If edits are necessary, open the Visio drawing file from disk, edit it, and resend DDC to the controller.

▶ **To send the active DDC file to a device**

1. On the VisualLogic menu, point to File, and then click Send DDC to Device.

   The Send DDC dialog box appears. You can enter information about the DDC file being sent and indicate whether you want to save the file to disk at the same time.



2. Type information and select options according to the following guidelines. This information is duplicated in the Drawing Properties dialog box.

Item	Explanation
Device	The device instance of the controller you want to receive the DDC.
Representative and Job	The Alerton representative and job name. The DDC file keeps a pointer to the rep\job folders in the BACtalk root directory. If you choose to save files, this is the rep\job they are saved in.
File	The file name of the DDC, which is stored as a pointer within the program and used if you save the file.
Version	A version number you can use to track revisions to the DDC.
Send and save files	Click this option if you want a DDC file and Visio drawing file saved to disk at the same time you send the DDC to a device. Files are saved to <bactalk root>\<rep>\<job> with the name <file>.bd3|bd4 and <file>.vsd.
Send to device only	No DDC or Visio drawing files are created concurrently with the send.

3. Click Send.

Sending a DDC File from disk to device

You can use VisualLogic to send a saved DDC file to a controller without loading the DDC for editing.

▶ To send a DDC file from disk to a device

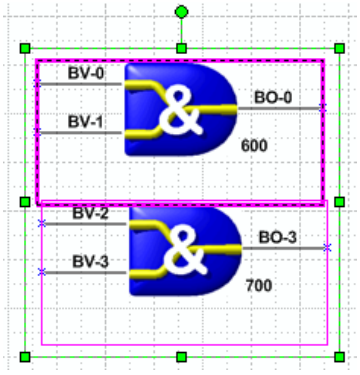
1. On the VisualLogic menu, point to File, and then click Send DDC File to Device.

   The DDC File to Send dialog box opens to the current rep and job folder.

2. Select the folder and file you want to send, and then click Open.

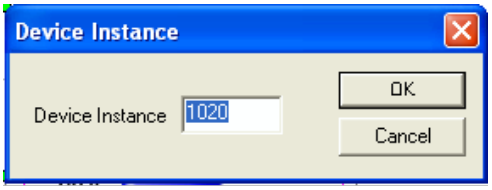
3. In the Device Instance dialog box, type the device instance of the VLC or global controller you want to receive the DDC, and then click OK.

Viewing real-time DDC data

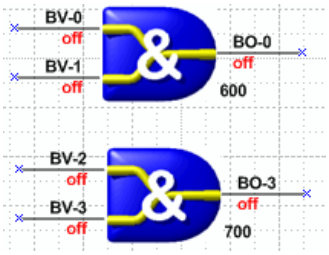


Step 1:Select the module you want to view.

Step 2:



Step 3: Type the device instance of the device you want to monitor, then click OK button.



Step 4: The red value is the real-time data

You can use VisualLogic to view values as they are processed in DDC. This is a useful tool for verifying your DDC. VisualLogic traps data and displays values beside the inputs and outputs on your DDC drawing.

When you load the drawing file, you can either read DDC from the device or from a file on disk. You can view real-time data for a single function or any number of functions. The speed of data updates depends on many factors: network traffic and the number of functions you choose to view, for example. As a general rule, choosing to view data for fewer functions results in quicker data updates.

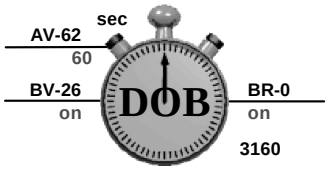
▶ **To view real-time data in VisualLogic**

1. Make sure the device you want to monitor is connected to the BACtalk network.
2. On the VisualLogic menu, point to File and then click Read DDC from Device or Read DDC from File. Specify the device instance or the file you want to read from.

   CAUTION If you read DDC from a file, make sure that it is the same file loaded in your controller; otherwise, erroneous data appears.

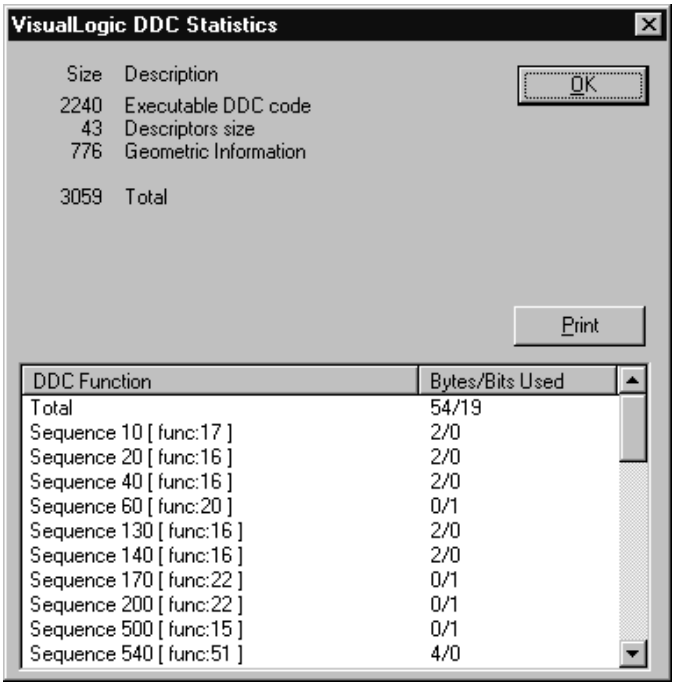
   Additionally, if you read DDC from a device, and geometric information was not saved to the device, DDC functions will appear on numerous pages in no particular order.
3. Select the functions for which you want to view live data values.
4. On the VisualLogic menu, point to Tools, and then click View Live Function Data.
5. In the Device Instance dialog box, type the device instance of the device (VLC or global controller) you want to monitor, and then click OK.

   Data from the controller appears in red beside its associated I/O connector.



Viewing DDC statistics

You can get DDC statistics to see how much memory, in bytes, VisualLogic uses for different elements of your drawing. You can also use this information to help you modify your DDC if you go over your byte/bit count for a VLC. See Table 1 on page 11 for byte/bit counts for VLCs.



▶ **To view DDC statistics**

- On the VisualLogic menu, point to Tools, and then click Get DDC Statistics. VisualLogic processes the drawing and provides a DDC statistics list.

Modifying DDC

When you edit DDC in VisualLogic, the DDC program you work on is stored in the memory of the computer running VisualLogic. No changes are made to the DDC programming in the controller until you save the new DDC program to the controller.

You can adjust the DDC program at the computer without affecting the controller's operation. The controller continues to execute the most recently downloaded DDC program until a new program replaces the old one.

Propagating function parameters

You can use VisualLogic's Propagate Parameters command to automatically assign the output of one function as input to another function. Use this command after you use Visio connectors to link the I/Os graphically. For example, if you assign BV-3 as the output of a function at Sequence 100 and then use a Visio connector to link it to an input of a function at Sequence 110, the Propagate Parameters command automatically assigns BV-3 as the input at Sequence 110. If you have assigned an input that doesn't match its corresponding output, the command overwrites the input with the output assignment.

CAUTION If you have multiple I/O assignments using the Junction shape, the Propagate Parameters command will automatically propagate the assignment of the output with the lowest sequence number to all the outputs, overwriting previous assignments, if any.

▶ **To automatically copy output assignments to input assignments**

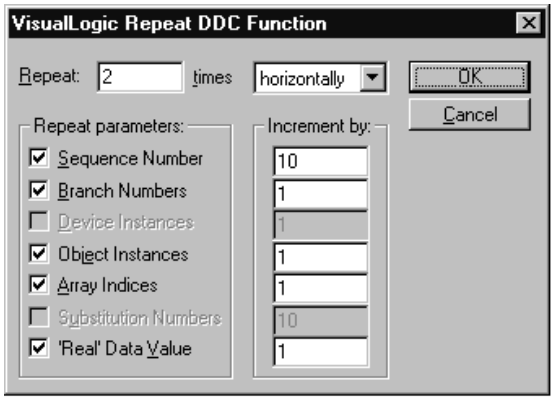
- On the VisualLogic menu, point to Tools, and then click Propagate Parameters.

Repeating functions

This feature provides a way to create several DDC functions of any kind at one time. You can increment DDC function parameters with input specific to each DDC function.

You can also increment the following optional parameters: branch numbers, device instances, object instances, array indexes, substitution numbers, and 'Real' data values. Only some of these parameters are applied to particular DDC functions. If the increment value is zero, the correspondent parameter does not increment.

To use this feature, only one function can be selected.



▶ **To repeat functions**

1. Select the function you want to repeat.
2. On the VisualLogic menu, point to Tools, and then click Repeat Function.
3. Select whether you want to place the functions horizontally or vertically on the drawing (in relation to the selected function).
4. Select the parameters you want to repeat and how you want to increment them.
5. Click OK.

Resequencing functions

Often, a programmer needs to change the sequence of a range of functions while programming, usually because functions were added or omitted. VisualLogic enables you to select a range of functions and renumber them as a whole. Rather than changing sequence numbers one-by-one, you assign a starting sequence number and specify the increment of sequential function numbers—10, 20, 30, or 5, 10, 15, for example.

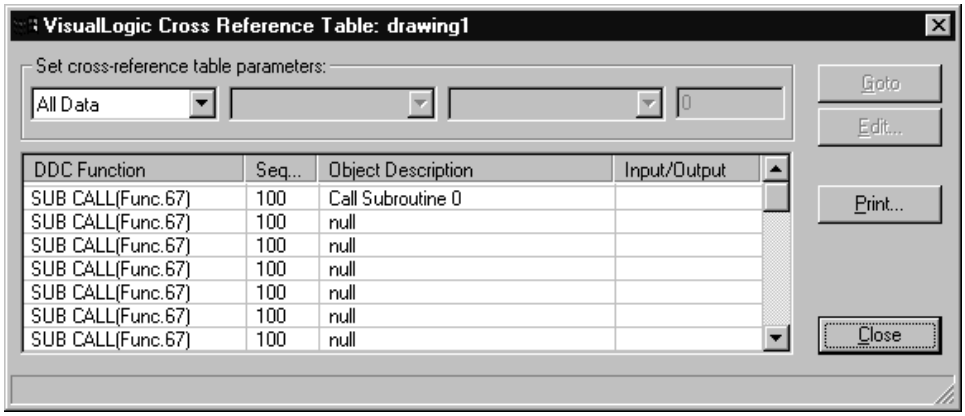
▶ **To resequence functions**

1. Select the functions you want to resequence. Use Shift+click to add functions to an initial selection.
2. On the VisualLogic menu, point to Tools, and then click Resequence Functions.
3. In the Start box, type the lowest sequence number you want assigned to the selected functions.
4. In the Step box, type the increment by which you want sequence numbers to increase.
5. Click OK.

   The function with the lowest sequence number is renumbered with the Start sequence number. Other functions are numbered in increasing increments of the Step value.

Cross-referencing functions

The DDC cross-reference table displays all DDC functions matching the input criteria. Users can select any DDC function in the output cross-reference table and go to the drawing shape associated with the DDC function.



▶ **To cross-reference functions**

1. On the VisualLogic menu, point to Tools, and then click Cross-Reference Table.
2. By default, All Data is shown. Select how you want to filter functions in the table. For example, you can choose to view only sub caller data.
3. Select a function and click Goto to jump to that function in your drawing.
4. Click Edit to modify a cross-reference.

Programming VLC DDC for the BACtalk Microset and Microtouch

4

VLCs include built-in programming designed to work with either a Microset or a Microtouch. Current heating and cooling setpoints are calculated automatically based on occupied status, day and night setpoints, and heating, cooling, and demand offsets. Management of the after-hours timer is also handled automatically.

In addition, a reserved range of AVs and BVs in the Microset enable customized control of the unit. These features greatly simplify DDC programming for zone temperature control.

About the Microset

The BACtalk Microset is a microprocessor-controlled room sensor designed to work with VLCs. It connects to the terminal labeled MSET on the VLC. The Microset enables an occupant to adjust the occupied setpoint, activate the system after-hours, and view both the room temperature and the outside air temperature. The Microset also includes a Field Service mode, which enables authorized service personnel to view and change system operating parameters. The VLC automatically detects when a Microset is connected to the appropriate terminal.

About the Microtouch

The Microtouch includes a 10K thermistor and a setpoint adjustment potentiometer. The Microtouch must be connected to IN-0 and IN-1 (the first two inputs) to function. The Microtouch allows the user to adjust the occupied setpoint by moving a lever on the side to either increase (up) or decrease (down) the setpoint. Pressing the center of the Microtouch during unoccupied hours activates the after-hours timer. A jack is provided to allow connection of a field service tool.

Analog and binary values assigned to Microset operation

Every VLC has a range of data points (AVs and BVs) reserved for Microset operation. Each data point has a particular function with respect to Microset operations. Some provide feedback information (such as room temperature); others enable control and interface with the unit. Table 5 summarizes these data points.

Table 5 VLC data points reserved for Microset and Microtouch operation

Object Instance	Function	Remarks
AV-90	Setpoint	Displayed at the Microset and adjusted up or down when the WARMER or COOLER button is pressed.
AV-91	Setpoint High Limit	The maximum value of AV-90.
AV-92	Setpoint Low Limit	The minimum value that AV-90.
AV-93	Cooling Setpoint Offset	Used in the calculation of the Occupied Cooling Setpoint (AV-99).
AV-94	Heating Setpoint Offset	Used in the calculation of Occupied Heating Setpoint (AV-100).
AV-95	Unoccupied Cooling Setpoint	The cooling setpoint used when BV-67 is OFF.
AV-96	Unoccupied Heating Setpoint	The heating setpoint used when BV-67 is OFF.
AV-97	After-hours Timer Limit (Hours)	The adjustable override limit. Maximum is 9.5 hours.
AV-98	After-hours Timer (Hours) Value	The current value of the after-hours timer. Automatically counts down when after-hours operation is enabled (BV-66).
AV-99	Current Cooling Setpoint	Read only. Internally calculated based on occupied mode, setpoints, and offsets. (see "Setpoint calculation" on page 43).
AV-100	Current Heating Setpoint	Read only. Internally calculated based on occupied mode, setpoints, and offsets. (see "Setpoint calculation" on page 43).
AV-101	ROOM Button Display Value	The value displayed at the Microset when the ROOM button is pressed. Typically, this is used to display the room temperature. AI-0 is the space temperature input read from the Microset, and DDC is written to transfer the value of AI-0 to AV-101 for display.
AV-102	Space humidity (%RH). Optional.	Read only. If the Microset is equipped with the humidity sensor option, this AV provides the humidity sensor reading in %RH.
AV-103	OUTSIDE Button Display Value	The value displayed at the Microset when the OUTSIDE button is pressed. Typically, this is the outside air temperature value, which is transferred from elsewhere in the system to AV-103 in the VLC.
AV-104	Microtouch Lever Offset	Automatically calculated from current lever position.

Table 5 VLC data points reserved for Microset and Microtouch operation

Object Instance	Function	Remarks
AV-105	Microtouch Lever Value	Maximum offset when lever is either all the way up or down.
AV-106	Demand Offset	Provides an interface for demand limiting programs or other system setpoint adjustment. Used in conjunction with the setpoint (AV-90) and heating and cooling offsets (AV-93, AV-94) as appropriate to calculate current heating and cooling setpoints (AV-99, AV-100).
BV-64	Scheduled Occupied Command	Provides a system schedule interface for setting occupied, unoccupied operation. Write an ON value to BV-64 to set occupied operation, OFF for unoccupied operation. (See “Occupied and unoccupied modes” on page 44).
BV-65	Microset Occupied/ Unoccupied Enable Command	Enables occupant to set unoccupied operation using the Microset OFF button as long as BV-64 is ON. (See “Occupied and unoccupied modes” on page 44.)
BV-66	After-hours status	Read only. ON when after-hours operation is in effect.
BV-67	Occupied/Unoccupied Status	Read only. ON = Occupied mode. OFF = Unoccupied mode. Result of BV-64, BV-65, BV-66.
BV-68	Microset Field Service Lockout	Removes the ability to start Field Service Mode from the Microset. ON = Field Service Mode disabled.
BV-69	Swap English/Metric Display	Affects the Microset display only. ON = Microset display is opposite of English/Metric mode selection in DDC header (see “AI setup” on page 152).
BV-70	Microset Detected Flag	Read only. ON when Microset is detected on AI-0.
BV-71	English/Metric Status	Read only. Reflects English or Metric selection in DDC header. ON = English. OFF = Metric.

Setpoint calculation

Throughout your DDC, use AV-99 and AV-100 for the actual current cooling and heating setpoints, respectively. The VLC automatically includes offsets and switches between occupied and unoccupied setpoints as appropriate. The Microtouch lever offset is also automatically considered if a Microset is not detected.

When the VLC is in unoccupied mode, it sets the current cooling (AV-99) and heating (AV-100) setpoints to the unoccupied cooling (AV-95) and unoccupied heating (AV-96) setpoints, respectively. When the VLC is in occupied mode, the current cooling setpoint is set to the occupied setpoint (AV-90) plus the

Microtouch lever offset (AV-104, only included if Microset is not detected), plus the cooling offset (AV-93), plus the demand offset (AV-106).

Current cooling setpoint calculation

AV-99 = (AV-90) + (AV-104) + (AV-93) + (AV-106)

The current heating setpoint is set to the occupied setpoint (AV-90), plus the Microtouch lever offset (AV-104, only included if Microset is not detected), minus the heating offset (AV-94), minus the demand offset (AV-106).

Current heating setpoint calculation

AV-100 = (AV-90) + (AV-104) - (AV-94) - (AV-106)

Microtouch offsets

If a Microset is not detected, the VLC assumes a Microtouch is connected, and input 1 is read as a Microtouch bias. The VLC calculates the Microtouch offset (AV-104) based on the position of the lever and the Microtouch lever value (AV-105). Typically, when a Microtouch is used, the Microtouch lever value (AV-105) is placed on a display to provide an easy way to adjust the amount of influence the Microtouch lever has on the occupied setpoint. If the lever is in the full up position, the Microtouch offset will be a positive value equal to the Microtouch lever value. Conversely, if the lever is in the full down position, the Microtouch offset will be a negative value equal to the Microtouch lever value. As the lever is moved between the two stops, the Microtouch offset will vary between these two values, with a value of zero at the center position.

IMPORTANT If neither a Microset nor a Microtouch is connected to the VLC, you should set the Microtouch lever value (AV-105) to zero. If any other value is used, a Microtouch offset will be applied to the Occupied setpoint, depending on what, if anything is connected to input 1 on the VLC.

Occupied and unoccupied modes

The occupied mode of the VLC is controlled by BV-64, BV-65, and BV-66. When BV-64 is activated, the VLC goes into occupied mode. The VLC remains in occupied mode as long as BV-64 is ON, unless the OFF button feature is activated (BV-65 is ON). When the OFF button feature is activated, the user can press the OFF button to set the VLC to unoccupied mode. As long as BV-64 is ON, the user can then toggle manually between occupied and unoccupied modes using the Microset ON and OFF buttons.

After-hours operation

When BV-64 is OFF, the VLC remains in unoccupied mode unless the after-hours timer (BV-66) is activated.

The value of the after-hours timer (AV-98) automatically counts down at a rate of 0.5 every 30 minutes until it reaches zero, at which time the after-hours timer status (BV-66) turns OFF. The after-hours timer automatically resets to zero when BV-64 turns ON. As long as BV-64 is OFF, the after-hours timer (AV-98) can be manually adjusted in increments of 0.5 hours from a BACtalk data display.

Microset

When a Microset is connected, the user can activate the after-hours timer by pressing the ON button. For each press of the ON button, the after-hours timer (AV-98) increments 0.5 hours, up to the after-hours limit (AV-97). Similarly, the occupant can press the OFF button to decrement the after-hours timer in 0.5 hour increments.

Microtouch

When a Microtouch is used, pressing the Microtouch center will set the after-hours timer to the value of the after-hours duration (AV-97). The after-hours timer status is ON whenever the after-hours timer is above zero.

Typical DDC for a Microset

Typically, a schedule in a global controller controls a BV object in the global controller, which is then transferred to the VLC using global controller DDC. The BV can be either directly transferred to BV-64 in the VLC to provide scheduled zone operation or transferred to an intermediate BV in the VLC.

The Alerton Standard applications use an intermediate BV (typically BV-40) so that this BV, or the warmup BV or the cooldown BV, can be used to activate BV-64, thus putting the VLC into occupied mode. This programming technique can be used with the optimum start feature. This programming method also allows you to independently adjust ventilation parameters depending on whether the zone was put into occupied mode with a warmup, cooldown, scheduled occupancy, or after-hours override command.

Heating and cooling control sequences can use AV-99 and AV-100 as the current cooling and heating setpoints. These setpoints are automatically calculated as described above depending on occupied mode status, offsets, and other factors. BV-67 can be used as an indication of when the VLC transitions between occupied and unoccupied mode. This can be useful for resetting Proportional Integral (PI) functions when the setpoint changes.

Typically, a single VLC in a given installation has an outside air temperature sensor. Global controller DDC is then used to transfer this value to AV-103 for all other VLCs.

VLC DDC must be used to transfer the space temperature to AV-101. Typically, the space temperature will be AI-0, but in some applications you may wish to average two or more sensors or use a different sensor. We recommend that you use AV-101 as the space temperature throughout your programming, then if you want to change your sensor configuration you only have to modify your DDC in one place.

The demand offset is included in the setpoint calculation to allow for the future implementation of a demand limiting program. The demand offset will be sent from a global controller to AV-106.

The Microset present flag (BV-70) allows you to create display items that will change depending on whether or not a Microset is installed. The English/metric flag (BV-71) allows you to write DDC programming that will automatically adjust tuning parameters and limits depending upon the mode to which the VLC is set.

Field Service Mode

The Microset may be used to access important control points within a VLC. If BV-68 (Field Service Lockout) is OFF, it will be possible to enter Field Service mode by pressing the buttons on the Microset in a special sequence. The BACtalk Microset differs from the IBEX Microset in that two Field Service modes are available. Also note that the decimal point to the right of the Field Service codes displayed at the Microset denote a default fixed code. Any custom codes that you set up will not have a decimal point.

Pressing the button sequence BLANK-BLANK-ROOM-ON-WARMER allows access to the normal Field Service mode. The Hot and Cold Flow CFM settings are read-only. The default fixed codes listed in Table 6 are available in addition to any custom codes set up in the DDC header (see "Microset Field Service mode custom codes" on page 155).

Table 6 Default fixed codes for Microset Field Service Mode

Code	Meaning	Associated Data Point
UC.	Unoccupied Cooling Setpoint	AV-95
UH.	Unoccupied Heating Setpoint	AV-96
CO.	Cooling Offset	AV-93
HO.	Heating Offset	AV-94
CS.	Current Cooling Setpoint	AV-99
HS.	Current Heating Setpoint	AV-100
AL.	After-hours Limit	AV-97
HI.	Setpoint High Limit	AV-91
LO.	Setpoint Low Limit.	AV-92
SP.	Current Setpoint	AV-90
CF.	Cold Deck cfm (read only)	
HF.	Hot Deck cfm (read only). Dual duct only.	

Custom codes can be entered in the DDC device settings header. You are allowed to assign a custom code for up to 26 BI, BV, AV, BO, or AO objects. For each entry, you can specify whether the number will appear with or without one decimal place, and if the item is to be read-only. Items can also be restricted to positive numbers only. Note that if a BI, BV, or BO is specified, an ON value is represented on the Microset as .1 and an OFF value is 0. You must therefore activate the decimal point for binary items.

VAV Box Field Service Mode

The button sequence BLANK-BLANK-ROOM-ON-COOLER enables a special VAV Box Field Service Mode, which is used for balancing. This mode allows access to the codes listed in Table 7 as well as entries 0–3 (for Microset v1.15) in the Microset Field Service Mode Setup screen of VLC DDC (entries 0-4 apply for Microset II v1.17). See "Microset Field Service mode custom codes" on page 155, for more information.

Table 7 Additional default fixed codes for VAV Box Field Service Mode

Code	Meaning	Associated Data Point
HI.	Setpoint High Limit	AV-91
LO.	Setpoint Low Limit	AV-92
SP.	Current Setpoint	AV-90
SC.	Cold Deck Box Size	
SH	Hot Deck Box Size (Dual Duct only)	
CF.	Cold Deck cfm	Range is 0-32767
HF.	Hot Deck cfm (Dual Duct Only)	Range is 0-32767

CF and HF (if applicable) show the current cfm, which can be adjusted by modifying the calibration factor (k) during balancing. While in VAV Box Field Service Mode, the WARMER and COOLER buttons change the k factor in increments of ± 0.01, while the ROOM and OUTSIDE buttons change the k factor in increments of ± 0.1. Using the VAV Box Field Service Mode, you can adjust the k factor until the CF and HF readings match those from a balancing hood.

Use VLC DDC to view the value of the k factor (see “Setting parameters for a VAV airflow sensor” on page 156).

Programming techniques and strategies

5

Direct digital control (DDC) programming underlies all operations performed at an operator workstation. DDC ultimately drives all actions carried out by a user at a data display or template.

Programming DDC is a challenging undertaking and should not be attempted without a thorough knowledge of Alerton systems and the equipment being controlled. This chapter describes some common programming techniques and strategies. Carefully plan and test your DDC program before implementing it.

Include Function 1: END OF NORMAL SEQUENCE

This is a common oversight. If you use subroutine DDC, make sure that you remove the default function 1 at sequence 9999 and replace it at a sequence number that marks the end of normal DDC. Otherwise, error messages appear.

Write separate DDC programs for devices on different networks

Even though BCMs are connected together in an array, you should always write separate DDC programs to read/write data to/from VLCs connected to the MS/TP network of each BCM (or to read/write data to/from TUXs connected to the TUX trunk of each BCM-TUX). Basically, you should treat each BCM as a global controller. Failure to do separate DDC can result in a serious degradation in network communications and temporary communication delays of several minutes.

Don't write to a data point more than once

DDC won't tolerate multiple writes to data points. Doing so results in an error message and DDC fails to execute. To make sure you aren't writing to these data points more than once, use the cross-reference utility. Remember that if you write to a data point in a subroutine, that data point is written to several times (once for each sub-caller).

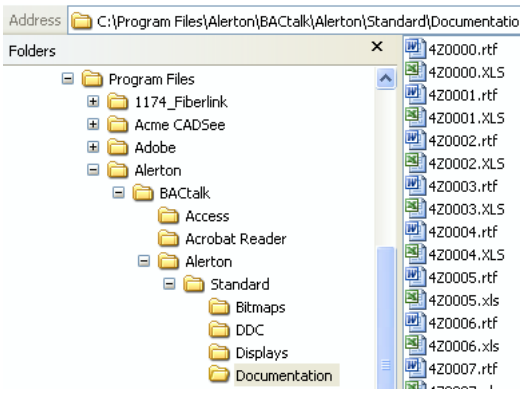
Note This only applies to objects without a priority array.

Leave room to grow

Space your sequences by at least 10—the first function at sequence 10, the second at sequence 20, the third at sequence 30, and so on. The DDC Editor is set up to facilitate sequencing by tens. Even though functions can be entered at every sequence number, this enables you to add functions if you make a mistake.

Plan before you program

Study the desired sequences of operations for the equipment to be controlled and then create a DDC diagram for each sequence. Make sequences modular for documentation purposes and ease of entry.



rtf XLS

Document meticulously

As you develop the program, document the point assignments, starting with all the physical control points at the VLC level. The control point list should include all logical and physical points for VLC DDC. The control point list for Global/ Building Controller DDC should include all points. Be sure to note the point type (AI, BI, AO, BO, AV, BV) and whether any additional automated control features are required (alarms, schedules, optimum start, for example).

Leave a written history of the assignment of inputs and outputs to equipment and how control sequences were implemented. Document in such a way that another engineer or programmer can decipher your work.

Control the start loop with the Initialize Flag

On the first pass of DDC after it's restarted (called the startup loop), no feedback information is available from the VLCs. This can have unpredictable results. Use the Initialization Flag to control DDC during the startup loop.

Save your DDC to disk frequently

Get in the habit of saving your DDC to disk frequently. Also, make sure that the DDC you want your controller to load automatically is saved to the correct filename.

Test before equipment startup

As much as possible, test the control sequences you program before you actually hook up equipment. Alerton advises that you use a test bench to simulate equipment inputs and monitor outputs.

Integrating with other applications using automation

Automation is a feature of ActiveX, which is Microsoft technology, and an evolution of their component object model (COM) interface. For those who may be familiar with dynamic data exchange (DDE) in IBEX systems, ActiveX in BACtalk works much the same way. Other applications that support ActiveX can use BACtalk's ActiveX Interface to fetch data from BACtalk while BACtalk is running. The ActiveX Interface is, in fact, an Automation Object.

The BACtalk Automation Object essentially exposes BACtalk data so that it can be used by scripts, programs, and applications that support Automation. This enables you to read property values, write property values, and send a time sync from another application.

Using the BACtalk Automation Object requires knowledge of object-oriented programming techniques. See the Microsoft web site at http://www.microsoft.com/com/tech/activex.asp to learn more.

Using DDC to detect VLC communications failure

This section presents two methods of using Global/Building Controller DDC to confirm communications between a VLC and a global controller and then generating an alarm or another action in DDC if VLC communications is lost. DDC is presented along with the advantages and disadvantages of each approach.

Method 1

The Delay on Make and Delay on Break Functions (DOM and DOB) in VLC DDC combine to produce a pulse (BV-63) every 240 seconds. The Exclusive OR (XOR) Function in the Global Controller DDC reacts to the pulse generated by the VLC. As BV-63 switches from ON to OFF and back to ON, the XOR resets the DOB, turning the VLC Comm Fail (BV-201) OFF. As long as the Global Controller can read the pulse (BV-63) from the VLC, the XOR will continually reset the DOB. If communications between the Global Controller and the VLC fails, a Comm Failure condition will be reported through BV-201 after 900 seconds (15 minutes) has elapsed. This elapsed time is set using the delay input to the DOB. The sequence numbering of the DDC functions in the global controller is very important. Note that the XOR, at sequence 4500, executes before the Transfer (XFR), at sequence 4510. This is the trick in getting the XOR to pulse ON to OFF and repeat. Also, the branch point between the XFR and the XOR (BR-401) must be a reserved branch point. This means that BR-401 cannot be referenced anywhere else in Global Controller DDC.

Set up the VLC Comm Fail Alarm (BV-201) as you would any other binary point alarm using the Alarm Wizard in BACtalk. The delay setting in the alarm point setup should be set to zero so as not to conflict with the DOB in Global Controller DDC.

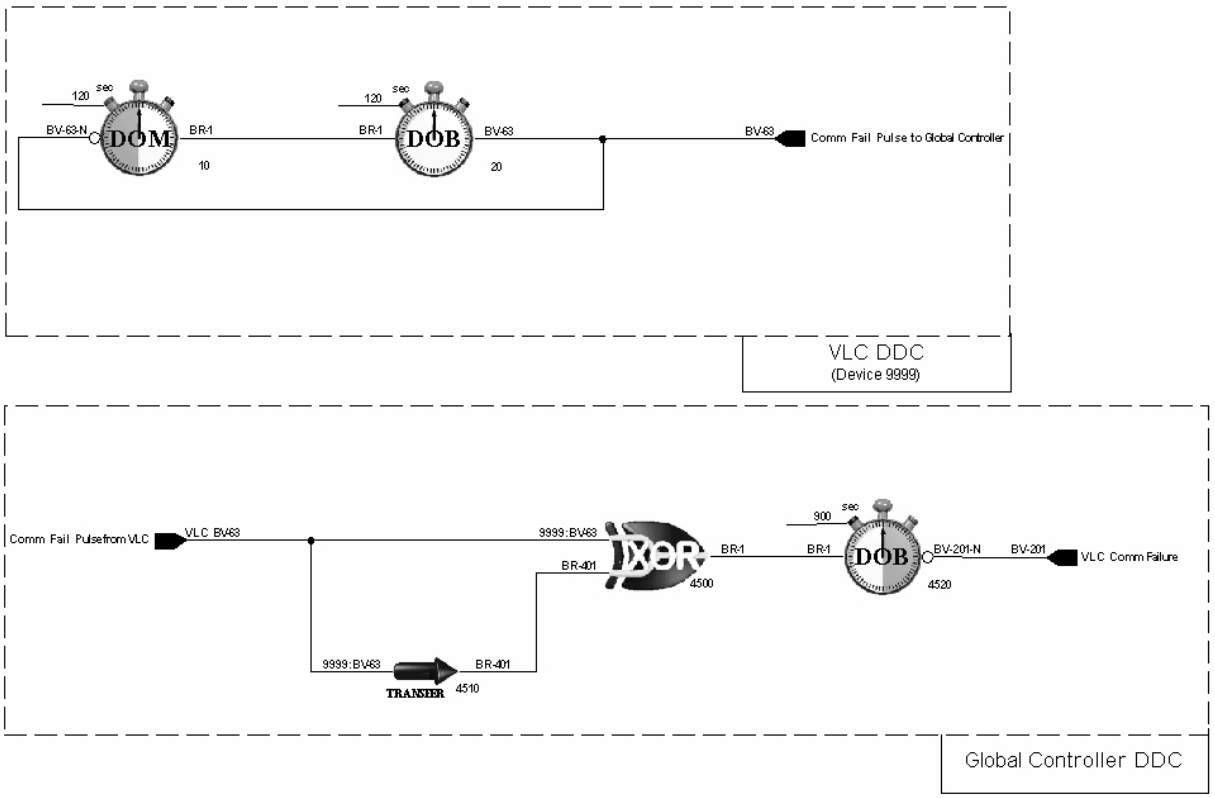


Figure 4 Method 1: VLC and global controller DDC sequences for detecting VLC communications failure at the global controller

Advantages and disadvantages

The advantage of using this method over Method 2 is that only one data point is being transferred between the VLC and the global controller. The disadvantage is that the two timing functions in VLC DDC consume RAM space, which may be needed elsewhere in the program depending on the application.

Method 2

The Two-Input AND Function (&) in VLC DDC generates a pulse (BV-63) to the global controller. BV-63 pulses between ON and OFF every time the global controller transfers BV-63 to BV-62. As long as communication is established between the VLC and the global controller, BV-63 will continually pulse between ON and OFF. The Change of State Detector (COS) Function in Global Controller DDC reacts to the pulse generated by the VLC. As BV-63 switches from OFF to ON, the COS resets the Delay on Break (DOB), turning the VLC Comm Fail (BV-201) OFF. As long as the global controller can read the pulse (BV-63) from the VLC, the COS will continually reset the DOB. If communications between the global controller and the VLC fail, a Comm Failure condition will be reported through BV-201 after 300 seconds (5 minutes) have elapsed. This elapsed time is set through the delay input to the DOB.

Set up the VLC Comm Fail Alarm (BV-201) as you would any other binary point alarm using the Alarm Wizard in BACtalk. The delay setting in the alarm point setup should be set to zero so as not to conflict with the DOB in Global Controller DDC.

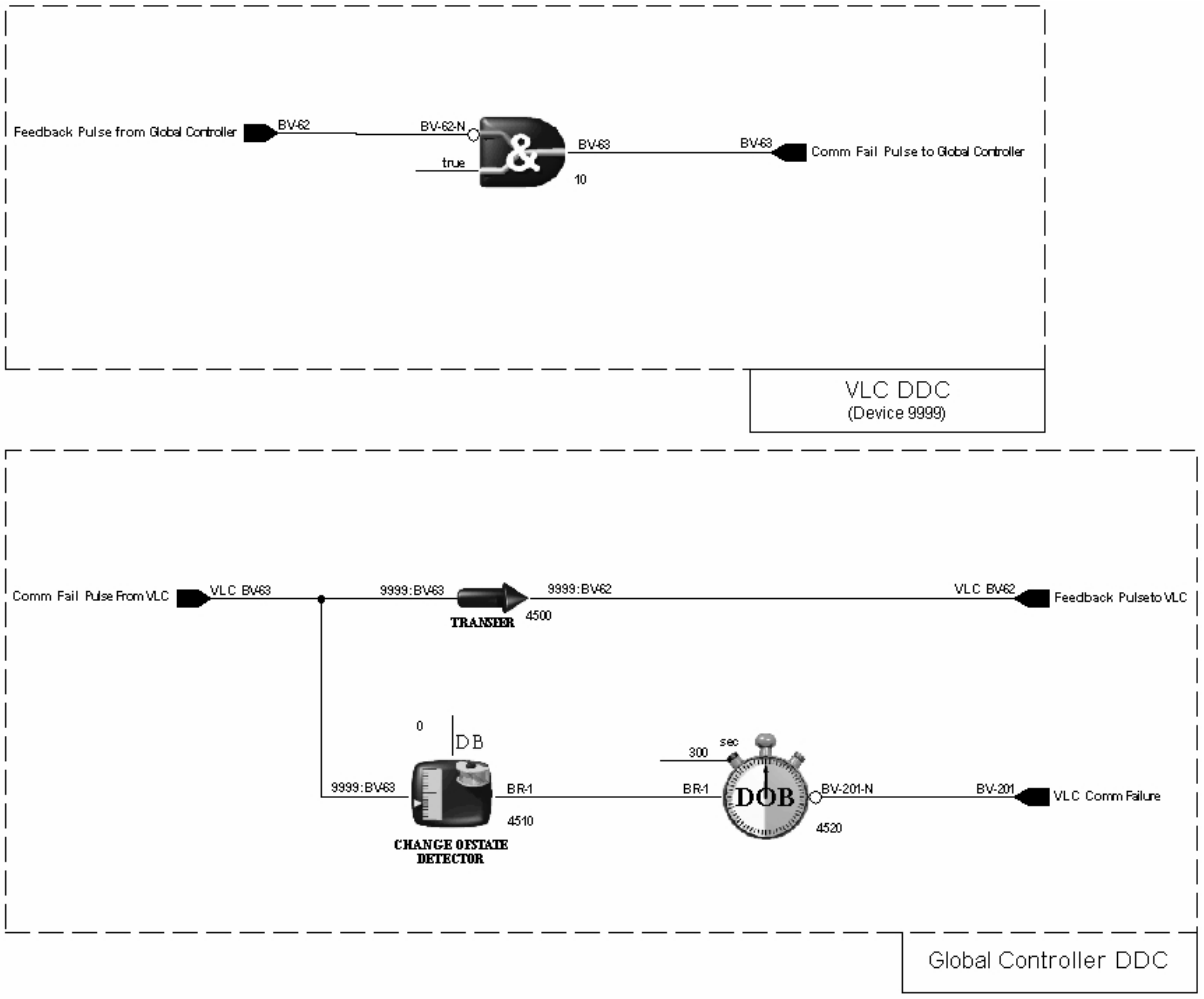


Figure 5 Method 2: VLC and global controller DDC sequences for detecting VLC communications failure at the global controller

Advantages and disadvantages

The advantage in using this method over Method 1 is the Two Input AND function in VLC DDC does not consume RAM space, which may be needed elsewhere in the program depending on the application. The disadvantage is that two data points are being transferred between the VLC and its host global controller.

Using DDC to detect communications failure in a global controller

The Comm Fail flag in a VLC does not transition ON in a VLC when communications with the VLC's global controller is lost. In fact, the Comm Fail flag monitors whether the VLC passes the token on the MS/TP LAN. Two VLCs—or any number of VLCs—without a global controller online will pass the token to one another as long as they can communicate. Therefore, the Comm Fail flag does not transition ON until the VLC loses communication with all other devices on the MS/TP LAN, not just the global controller.

As a result, the Comm Fail flag in the VLC is useful in DDC only insofar as it enables VLC DDC to revert to a stand-alone mode of operation (for example, causing setpoints to revert to stand-alone, default settings) in the absence of all other devices, not just the global controller. It is not useful for determining whether a VLC is online with a global controller, or whether global controller DDC has halted.

To determine the status of global controller communications and DDC execution reliably, a "heartbeat" DDC sequence in the global controller is necessary, combined with a "pickup" DDC sequence in the VLC. The pickup DDC effectively listens for the heartbeat and generates an output if it ceases. This section presents DDC sequences for the global controller and VLC that generate this effect. This is only one solution among many. You may find a more appropriate or efficient DDC sequence for your specific application.

Explanation of DDC

The DDC sequences shown are for an global controller with Device Instance 211 and a VLC with Device Instance 702. It shows only the heartbeat and pickup sequences.

Global Controller DDC sequence

The DDC sequence in the global controller uses Function 21: Anti Short Cycle Relay to generate the heartbeat, which is written to the present-value of BV-10 in VLC 702. The choice of BV-10 is arbitrary and could just as well be any other property appropriate for your application. The output is written back to the input and is negated. The Min. ON and Min. OFF values are set to 60 seconds. These values determine how fast the heartbeat occurs. Without these values, BV-10 would toggle ON and OFF with each pass of DDC and quickly occupy bandwidth on the MS/TP LAN. A minimum time of 60 seconds for these values is recommended to keep MS/TP communications traffic down. Longer Min. ON and Min. OFF values may be desirable.

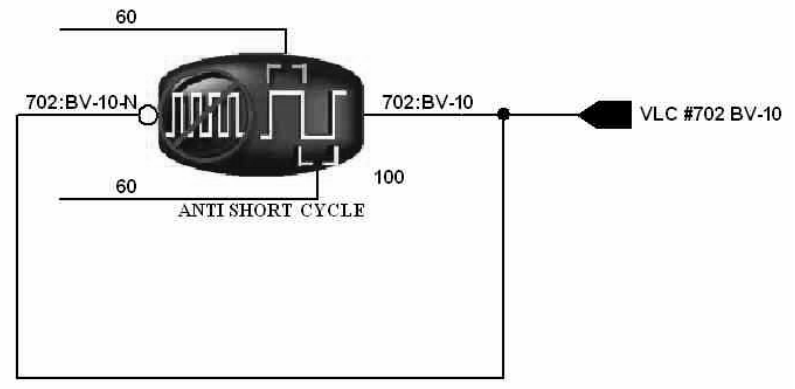


Figure 6 Heartbeat generation sequence in the global controller using Function 21: Anti Short Cycle Relay

VLC DDC sequence

In VLC DDC, the pickup sequence consists of Function 23: Change of State Detector and Function 16: Delay on Make (DOM). Essentially, Function 23 picks up the global controller heartbeat signal (written to VLC BV-10) and feeds it through a branch point (selected arbitrarily) into Function 16 (DOM). The output of Function 16 remains OFF until communications is lost for 300 seconds, the Function 16 delay time. The output of Function 16, in this case BV-11, is used in VLC DDC as a global controller communications failure flag, which transitions ON if the global controller heartbeat isn't detected. Use the Function 16 delay time to control the sensitivity of the sequence. This delay is how long communications must be lost before the failure flag goes ON.

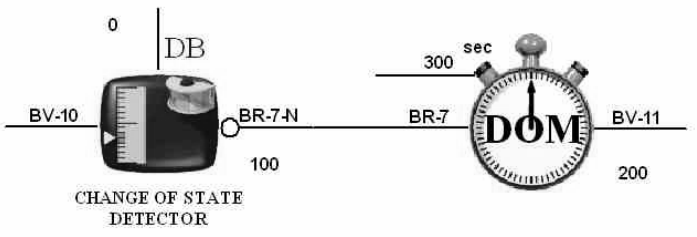


Figure 7 Pickup sequence in the VLC, which uses BV-11 (selected arbitrarily) as the global controller communications failure flag

Resolution of Microset-related AVs and use in DDC

Data resolution of Microset-related AV present-values is limited to increments of 0.5. As a result, using these values in DDC can have undesirable results. Microset-related AVs are in the range AV-90 to AV-107.

Actual results depend on how Microset AVs are used in the control sequence. In certain applications, AV-101 has been used as a space temperature input to the proportional component of a proportional integral (PI) routine. AV-101 varied by 0.5 when the actual space temperature change was much smaller. This caused a disproportional response and frequent equipment response, causing over cycling of the controlled variable and possible premature wear on controlled equipment.

Do not use AVs 90-107 in control sequences where their limited resolution may be an issue. Instead, use actual input values. For example, use AI-0 for space temperature.

Understanding BACtalk PI and PID functions

Proportional Integral (PI) functions and Proportional Integral Derivative (PID) functions in DDC (Functions 51 and 52) help control the balance between environmental conditions and feedback in BACtalk systems to maintain zone setpoints.

What is PID control?

The proportional, integral, and derivative components, when added together, generate an output value in response to a feedback input and setpoint. The output of the function varies in an attempt to get the feedback input to match the setpoint. In HVAC applications, the feedback input is often a temperature, such as a room temperature, and the setpoint is the room setpoint. The output is then used to control a damper, valve, or cycling time of heat stages. The proportional component responds instantly to changes in the feedback input or setpoint, while the integral component responds gradually over time. The derivative component is based on the rate of change of the feedback input relative to the setpoint.

PID logic has been used with industrial controls for a long time. The Alerton BACtalk PI and PID controllers use industry-standard terminology wherever possible.

PI vs. PID

For virtually all HVAC applications, it is generally wiser to use PI control rather than PID control. The Derivative component generally does not add much responsiveness, and it can be difficult to tune. Erratic response often results from using Derivative control in HVAC applications. Although PID control is often included as a specification requirement, the Derivative component should probably not be used unless you are very confident in your understanding of the complexities of tuning PID controllers.

How is the output of the PI function calculated?

The output of the BACtalk PI controller is the sum of three factors: proportional component (P), integral component (I), plus a constant of 50. The output is limited to a range of 0 to 100.

Output = 50 + P + I

The primary inputs are the setpoint (SP) and feedback input (FB). The other inputs are tuning parameters, and they are generally set to fixed values as needed to achieve responsive yet stable control. Both the P and I components are calculated from the difference between the setpoint and the feedback input. This difference is called the *error* (E), and it represents how far away from setpoint the feedback input is.

E = FB - SP

The goal of the control function is to get this error value to go to zero (that is, feedback input = setpoint.)

The proportional component responds instantly and is simply proportional to the value of the error. The only tuning parameter that affects the proportional

component is the *proportional constant* (K_p). The proportional component is equal to the error times the proportional constant.

$$\mathbf{P = E \times K_p}$$

The integral component is calculated as a running total over time. It is used to make incremental adjustments over time to get the feedback input to match the setpoint. When the function is first started, the integral component is set to the value of integral startup (STUP). From then on, the integral component is a running total, with an integral adjustment value added every second. The integral adjustment value that is added each second is the error times the *integral constant* (K_i) divided by 60. The integral constant is divided by 60 since it represents the change per minute, and the calculation is performed every second.

$$\mathbf{I = I_{previous} + (E \times K_i / 60)}$$

The integral adjustment value is limited by the maximum integral change (I_{max}), which is the maximum amount the integral component is allowed to change per minute. The maximum allowed integral adjustment (which is added once per second) is then $I_{max}/60$.

Finally, the value of the integral component is limited by the integral limit (I_{limit}). I is not allowed to be greater than I_{limit} or less than $-I_{limit}$.

$$\mathbf{|I| \leq I_{limit}}$$

Reversing the output for reverse acting applications

The default output action of the proportional integral functions is direct acting (DA). As the feedback input increases, the output increases. Likewise, when the feedback input decreases, the output decreases. Some applications require the PID controller output to be reverse acting (RA); as the feedback increases, the output decreases. Likewise, when the feedback decreases, the output decreases.

For RA applications, use the REV feature of DDC on the output of the function. When this is done, the Reversed Output = (100 - Output).

Direct acting (DA) output is appropriate for the following applications:

- Normally open (NO) heating valve
- Normally closed (NC) cooling valve
- Building static pressure control (relief damper or variable frequency drive)
- Economizer dampers where the outside air (OA) damper is NC

Reverse acting (RA) output is appropriate for the following applications:

- Normally closed heating valve (steam for instance)
- Normally open cooling valve
- Supply duct static pressure control (inlet vanes or variable frequency drive)
- NC steam humidifier valve
- NO relay contacts serving electric heating coils

Setting the tuning parameters

Tuning a controller has always been a bit of an art. Experimentation is usually necessary to achieve optimum performance. Increasing the values for K_p and K_i increases the responsiveness of the BACtalk PI controller, but reduces the stability. The goal is to get the controller to provide the most responsive control possible without hunting.

One of the challenges in tuning PI loops in HVAC control is that system response functions vary considerably depending on circumstances. For example, when controlling an economizer damper to maintain a desired supply temperature, the effect of modulating the damper will be very different when the outside air temperature is 0°F versus 50°F. For this reason, it is generally best to tune loops conservatively. This slows response but helps ensure stable control over all operating conditions. Be sure to consider the effect of the current and possible operating conditions when tuning a loop.

Proportional constant (K_p)

This constant adjusts how responsive the proportional component will be to differences between the feedback input and the setpoint. A larger value for K_p increases the influence of the proportional component.

Proportional constant is the amount the output will change in response to a change in the error value equal to one.

A typical value for K_p for room temperature control applications would be 12.0 for English unit applications. This means that if the room temperature (feedback input) is 76° and the Setpoint is 74°, the proportional component would be 2 x 12.0 = 24. If the integral component was zero, the output would then be 50 + 24 = 74.

When it takes a long time for the feedback input to change once the controlled item is adjusted (for example, when modulating a VAV cooling damper to achieve a desired room temperature), it generally reduces the value of K_i relative to $K_{p.}$ The proportional constant often helps prevent overshooting in these cases.

When the feedback input responds quickly to changes in the controlled item (for example, when modulating a damper to achieve a desired airflow), the proportional constant should generally be set fairly low, perhaps even to zero, leaving most of the control to be performed by the integral component.

To determine an appropriate initial value for K_p, divide 3 by the smallest amount you would typically adjust the Setpoint. For example, for controlling a supply

fan speed to maintain duct pressure, the smallest setpoint adjustment would probably be 0.1. Thus, the starting value might be 3/0.1, or 30. You will of course have to use your judgment as to whether to adjust this value up or down depending on the stability and responsiveness of what you are controlling as well as the consequences of overshooting the value.

One disadvantage of the proportional component is that it instantly responds to any change in the feedback value, which can result in subtle hunting or excessive minor adjustments to a damper or valve, even when the feedback is near setpoint. In some cases, it is better to use only the integral component ($K_p = 0$) to prevent this problem. Another option for solving this problem is to set the feedback input of the PI function to the setpoint when the actual feedback value is acceptably close to the setpoint. For an example of this, refer to the damper control logic used in the Alerton Standard VAV applications.

Integral constant (K_i)

This constant adjusts how responsive the integral component will be to differences between the feedback input and the setpoint. A larger value for K_i increases the influence of the Integral component.

Integral Constant is the amount the Output will change over the course of one minute, in response to an error value equal to one.

Typically, the initial value calculated above for K_p represents a reasonable starting point for K_i, Divide 3 by the smallest amount you would typically adjust the setpoint. When the feedback input responds quickly to changes in the controlled item, the integral constant should be set much higher than for slower response applications like room temperature control.

A typical value for K_i for room temperature control applications would be 1.0. This means that if the room temperature (feedback input) is 76° and the setpoint is 74°, the integral component would increase by 2.0 every minute (provided I_{max} is greater than 2.0).

When modulating a damper to obtain a desired airflow, a good starting point is K_i = 100/FBrange where FBrange, is the full range you would expect for the feedback input as the control output varies between 0 and 100. This is a reasonable starting point for applications where the feedback responds almost immediately to changes in the actuator position.

Maximum integral change (I_{max})

This constant limits the rate of change of the integral component. The integral component is not allowed to change faster than I_{max} per minute, or I_{max} /60 per second.

I_{max} represents the maximum amount of change allowed for the integral component in one minute.

Typically, I_{max} should be set to match the speed of the actuator being controlled. It should equal the percentage of full stroke that can be achieved in one minute. For example, if a VAV damper is being controlled, either to achieve a desired room temperature or to achieve a desired airflow, and the full stroke damper time is 5 minutes, then I_{max} should be set to 20. This is because the damper can only

stroke 20% of its full stroke in one minute. As another example, if a valve can be actuated in 30 seconds, then I_{max} should be set to 200, since the valve can stroke 200% in one minute.

Integral limit (I_{limit})

This constant limits the value of the integral component to ± I_{limit}. It is used to limit the integral contribution allowed in the overall control signal. It is typically set to 50 which allows the integral component to bias the output to anywhere between 0 and 100 when the feedback input equals the setpoint (which by definition makes the proportional component equal to zero).

If you use a number greater than 50, the output could potentially stay fixed at 0 or 100 for a while, even when the error indicates a need to adjust the output in the opposite direction. The industry term for this phenomenon is "integral windup." It can be useful for certain applications when you want the integral component to be able to build up. Building up the integral component will delay any change from 0 or 100 (depending on whether the integral component is negative or positive).

Integral startup (STUP)

This constant is used to initialize the value of the integral component upon startup. Also, the integral component will be set to this value whenever K_i is set to zero. This is useful for resetting the integral component upon a change in setpoint or when starting equipment such as a fan. It is generally good practice to momentarily switch K_i to zero upon a change in setpoint or when starting the controlled equipment to reset the integral component.

Upon initialization (or when K_i is set to zero), the Integral component will equal STUP. For applications where the K_p is equal to zero or when the Feedback Input happens to match the setpoint, the output will be equal to 50 plus STUP upon initialization. You can calculate a good value for STUP by taking the output you wish to see upon startup, and then subtracting 50. For example, if you want the output to start at 20, the STUP should be 20 - 50 = -30.

Be careful in applications where the output is reversed. For example, consider an application where you are controlling fan speed to achieve a desired duct pressure. The output would be reversed and, upon startup, you may want the output to be about 20%. This means that you want the normal (not reversed) output to be 80. If K_p is equal to zero (not a bad idea for fan control), the normal output upon initialization will equal 50 plus STUP. If you want an initial value of 80, then STUP should be set to 30. The reversed output will then be set to 20 upon initialization.

Proportional constant vs. throttling range

Many control engineers who are used to working with pneumatic controls or the Alerton receiver controller are used to setting the proportional response by adjusting a throttling range. Increasing the throttling range actually *decreases* the responsiveness of the controller, while increasing the proportional constant (K_p) *increases* the responsiveness of the controller.

For example, consider an application where you are controlling a VAV damper to achieve a desired room temperature. If the throttling range is set to 5, the damper will be adjusted from fully closed to fully open as the input varies from 2.5 degrees below the setpoint to 2.5 degrees above the setpoint. To determine the corresponding value of K_p, you simply look at how much you want the output to change for every degree of change in the Input. In this case, it would be 20. A simple way to convert throttling range to K_p is to use the formula: $K_p = 100/$ Throttling Range.

Bit-packer and bit-unpacker DDC

Envision for BACtalk does not have bit-packer and bit-unpacker DDC functions. For this reason, Alerton developed the bit-packer and bit-unpacker DDC routines discussed here. The bit packer and unpacker DDC is contained in a VisualLogic file. Page 1 is the bit-packer DDC. Page 2 is the bit-unpacker DDC. This file is located on the Alerton Support Network under Applications > BACtalk > Miscellaneous.

Primarily, these DDC routines enable integration of a BACtalk system and third-party devices that write or read multiple bits in a single control point. Some manufacturers use a single 8-bit or 16-bit number to issue up to 16 different digital commands. The number of bits used is dependent on the manufacturer. Each bit in the number represents a different command. For example, you can use this DDC to integrate your BACtalk system with a Modbus VFD, which may use a single register/coil address to represent multiple points, each point represented by a bit in the register/coil address number.

Once you understand the DDC routines, you can scale them to manage only the number of bits you need to work with.

Bit-packer DDC routine

The bit-packer DDC routine encodes an 8-bit number as 8 BVs (as written, BV-0 through BV-7).

The DDC assigns the appropriate value to each BV and writes the output value as AV-0, including DDC to write only on change-of-state. For example, turning BV-4 ON activates a Function 40: Switch to pass the value of 16 (BV-4 represents the 16's position) to a Function 33: 6-input Addition. The 16 is then added to any other bit values that have been set to ON. As written, any number in the range 0 (all BVs OFF) to 255 (all BVs ON) is possible.

Bit-unpacker DDC routine

The bit-unpacker DDC routine does just the opposite of the bit-packer routine. It reads an analog value (as written, AV-0) and unpacks that value into eight binary values, each representing a bit (as written, BV-0 through BV-7).

The bit-unpacker DDC has a three-function, bit-checker DDC sequence for each possible bit in the value. Each sequence consists of a Function 39: Within, Function 30: Subtraction, and Function 40: Switch. Beginning with the largest possible bit-value, the bit-checker sequence reads the value from the register coil address and checks to see if it is within a specific bit range. If the bit-encoded value is detected within the analog value, the bit-checker sequence sets the BV representing the bit to ON, subtracts the bit value from the total value, and then passes the reduced value to the next bit-checking sequence. The process repeats for all bit-encoded values to be checked.

DDC function reference

6

This section contains references for DDC functions available in Global/Building Controller DDC and VLC DDC. Some operational differences exist between functions in a global or expandable controller and functions in a VLC. Mostly, these relate to timing issues. Furthermore, some devices may exist only in Global/Building Controller or VLC DDC. These are indicated.

CAUTION In VisualLogic, it is possible to program Global/Building Controller DDC or VLC DDC with functions that cannot be executed in the device. Make sure that you program only appropriate functions for the global controller or VLC as appropriate.

Function 1: End of Normal Sequence

Description Denotes the end of normal DDC and, if applicable, the beginning of subroutine DDC space.

Remarks Function 1 must be included in every VLC or global controller DDC program. Only one Function 1 can be programmed per VLC or global controller DDC program.

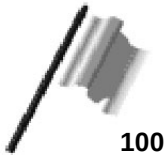
100

END OF NORMAL

Function 2: End of Subroutine (global controller only)

Description Denotes the end of subroutine DDC.

Remarks Function 2 must be included if subroutine DDC is used.



100
END OF
SUBROUTINE

Function 3: Set Context (global controller only)

Description Defines the set context device and is a critical component of reusable subroutine DDC (see the example), providing a device instance context for each iteration of the subroutine. The input may be either a data value equal to a device instance, or a data point ID that identifies a particular BACnet device. The set context device remains set until another Set Context Function executes.

Remarks When you enter a data point in Global/Building Controller DDC, you have three options: you can specify the device where the data point originates, you can specify the local device (the global controller in which the DDC executes), or you can choose a set context device. If you choose the set context device, the function references the data point in the device instance of the last Set Context Function to execute.

Example A subroutine transfers data to and from multiple VAV-SD controllers. For each VAV-SD, a Function 67: Subroutine Caller calls the same subroutine DDC. Substitution Point 0 in each Subroutine Caller is the device instance of the associated VAV-SD. The first function in the subroutine is Function 3: Set Context, with Substitution Point 0 entered as the context device instance. All subsequent data points in the subroutine DDC that must reference the associated VAV-SD are entered with the Set Context Device check box selected.

Function 6: Velocity Pressure to fpm Converter

Description Performs square root extraction to convert an analog input that represents velocity pressure (vp) to an analog signal that represents velocity in feet per minute (fpm).

Remarks The inputs are input, zero, and k factor. Input should be a signal representing the velocity pressure (vp) of the measured airflow. The k factor is used as a multiplier. The zero is used to compensate for variations in transducer readings at zero airflow; set the zero input equal to the value of the vp input when there is no airflow. The output of Function 6 can be expressed as:

Output = $k\sqrt{vp - z}$

where

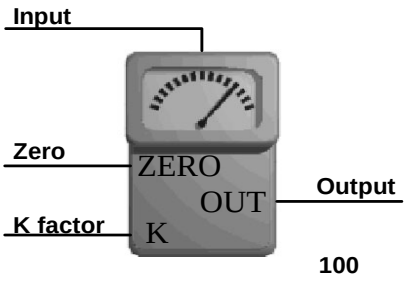
k = k factor

vp = velocity pressure input

z = zero

The k factor can be used to correct for the pickup multiplier and any other factors necessary to convert the input signal value to actual velocity pressure. Most vp pickups (except true pitot tubes) produce a pressure differential that is greater than the actual vp. The conversion factor is typically referred to as a pickup multiplier. Use the following equation to calculate the k factor using the pickup multiplier (PM) and the signal range (SR) as the pressure varies over the pressure range (PR) of the sensor.

$$kfactor = 4005\sqrt{\frac{PR}{SR \times PM}}$$



Function 8: Enthalpy Calculator

Description Calculates enthalpy from temperature and relative humidity.

Remarks Function 8 uses two input values—temperature in degrees F and relative humidity (RH) in %—to calculate enthalpy in Btu/lb. Minimum temperature used is 0 degrees F. Maximum temperature used is 102 degrees F.

Maximum enthalpy that can be calculated is 68 BTU/lb. at 102 degrees Fahrenheit and 100% relative humidity. The accuracy of this calculation is best between 55 and 80 degrees Fahrenheit.

ENTHALPY CALCULATOR

Function 10: Two-Input AND Gate

Description Performs the logic AND function of two binary-type inputs and sets a binary output accordingly.

Remarks The output will be set to ON only if both inputs are ON. If either input is OFF, the output is OFF.

In Global/Building Controller DDC, NULL inputs are acceptable; in VLC DDC, they are not. NULL values are considered OFF.

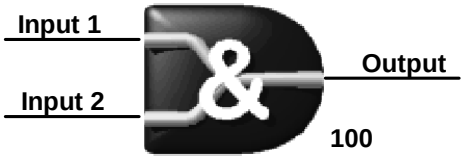


Table 8 Function output logic

Input 1	Input 2	Output
OFF	OFF	OFF
ON	OFF	OFF
OFF	ON	OFF
ON	ON	ON

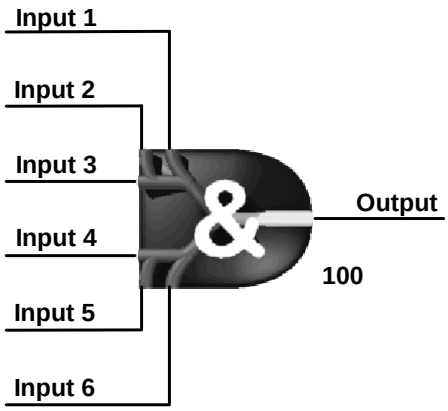
Function 11: Six-Input AND Gate

Description Performs the logical AND function of six binary inputs and sets the binary output accordingly.

Remarks Function 11 is similar to Function 10: Two-Input AND Gate, except that six binary inputs are logically compared to generate one binary output. Function 11 is used when there are more than two inputs. All six inputs must be assigned. If there are fewer than six inputs, the unused inputs should be set to any of the used input values or to a data value of (TRUE) but cannot be left blank.

The function uses all six inputs to set the output ON or OFF. If any of the six inputs is OFF, the output is set to OFF. The output is set to ON only if all of the six inputs are ON.

In Global/Building Controller DDC, NULL inputs are acceptable; in VLC DDC, they are not. NULL values are considered OFF.



Function 12: Two-Input OR Gate

Description Performs the logical OR comparison of two binary inputs and sets the binary output accordingly.

Remarks The output is ON if either or both inputs is ON. The output is OFF only if both inputs are OFF.

In Global/Building Controller DDC, NULL inputs are acceptable; in VLC DDC, they are not. NULL values are considered OFF.

Table 9 Function output logic

Input 1	Input 2	Output
OFF	OFF	OFF
ON	OFF	ON
OFF	ON	ON
ON	ON	ON

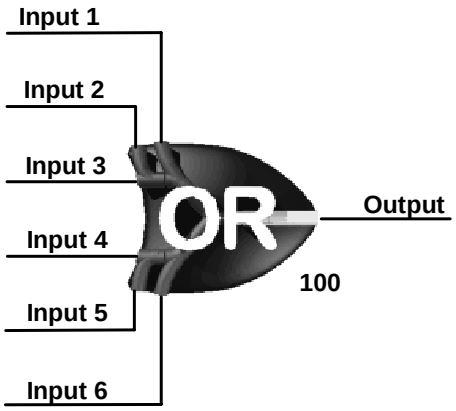
Function 13: Six-Input OR Gate

Description Performs the logical OR of six binary inputs and sets the binary output accordingly.

Remarks Function 13 is similar to Function 12: Two-Input OR Gate, except that it works with six binary inputs to generate one binary output. Function 13 is used when there are more than two inputs. All six inputs must be assigned. If there are fewer than six inputs, the unused inputs should be set to any of the used input values or to a data value of (False) but cannot be left blank.

The function uses all six inputs to set the output as ON or OFF. If any of the six inputs is ON, the output will be ON. The output will be OFF only if all six inputs are OFF.

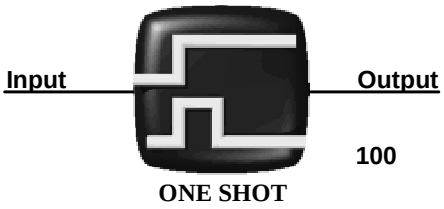
In Global/Building Controller DDC, NULL inputs are acceptable; in VLC DDC, they are not. NULL values are considered OFF.



Function 15: One Shot

Description Sets the output ON for one pass of DDC whenever the input transitions from OFF to ON.

Remarks The output remains ON only for a single pass of DDC, even if the input stays ON for a longer or period.

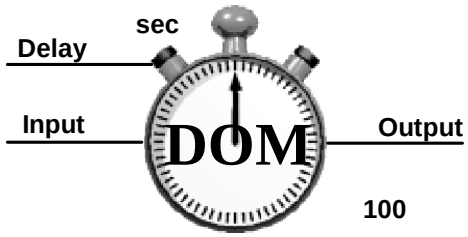


Function 16: Delay on Make (seconds)

Description Delays a binary transition from OFF to ON.

Remarks When the input transitions from OFF to ON, the function or output transitions ON only after the specified delay time (t). If the input value transitions OFF at any time during the delay period, the timer is reset. The delay time (t) resolution is one tenth of a second. The output transitions OFF immediately when the input transitions OFF.

Note There is a resolution of 0.1 seconds in VLCs and 1 second in global controllers.

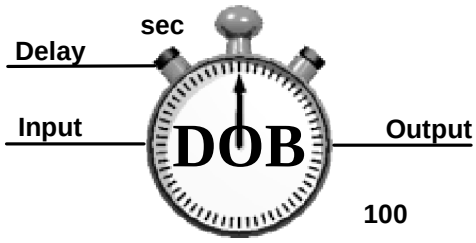


Function 17: Delay on Break

Description Delays a transition from ON to OFF.

Remarks When the input changes from ON to OFF, the output changes to OFF after a delay time (t). If the input value changes to ON at any time during delay time (t), the timer is reset. The output transitions ON immediately when the input transitions ON.

Note The delay time (t) resolution is 0.1 second in VLCs, 1 second in global controllers.



Function 18: Two-Input Exclusive OR

Description Sets the binary output OFF if both inputs match (both ON or both OFF), and sets the output ON if only one of the inputs is ON.

Remarks Function 18 is similar to Function 12: Two-Input OR Gate, with one exception: the output is OFF if both inputs are ON. Other values remain consistent with the OR function: the output is OFF if both inputs are OFF, and the output is ON only if one input is ON.

In Global/Building Controller DDC, NULL inputs are acceptable; in VLC DDC, they are not. NULL values are considered OFF.

Table 10 Function output logic

Input 1	Input 2	Output
OFF	OFF	OFF
ON	OFF	ON
OFF	ON	ON
ON	ON	OFF

Function 20: Flip Flop Gate

Description Two binary inputs, set (S) and reset (R), are used to switch the binary output between ON and OFF, respectively.

Remarks Function 20 has two binary inputs, set (S) and reset (R), which determine how the output is set. A momentary ON of the set (S) input turns the output ON if the reset (R) input is OFF. The output stays ON once it is set, even if the set (S) input transitions OFF. If the reset (R) input turns ON, the output transitions OFF. The reset (R) input has priority over the set (S) input, so the output is OFF if both inputs are ON.

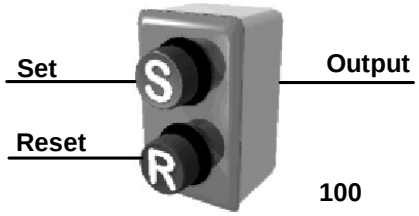


Table 11 Logic table

Set input	Reset input	Output
Momentarily ON	OFF	ON - Stays ON
OFF	Momentarily ON	OFF - Stays OFF
ON	ON	OFF

Function 21: Anti Short Cycle Relay

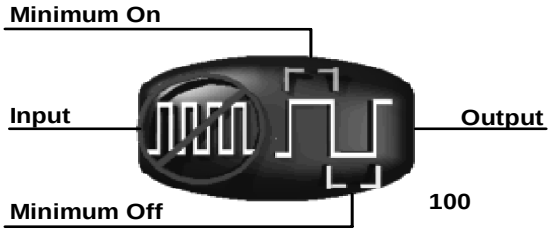
Description Prevents an output from changing state repeatedly, or "short cycling," by setting minimum ON and OFF times.

Remarks Once the output turns ON, it will stay ON for the minimum ON time, even if the input goes OFF.

When the output goes OFF, it will stay OFF for the minimum OFF time, even if the input goes ON.

Minimum ON and OFF times can be set to different values.

For VLCs, the time resolution is 0.1 second. For global controllers, the time resolution is 1 second and must be entered in whole seconds; the decimal value is ignored. For example, 308.7 = 308.

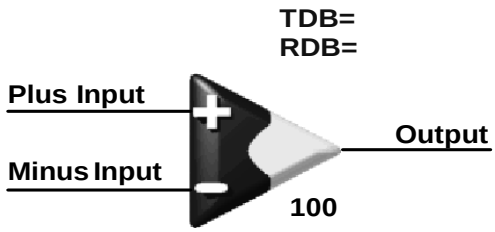


ANTI SHORT CYCLE

Function 22: Analog Input Comparator

Description Compares two analog inputs and produces a binary signal as a result of the comparison.

Remarks The output will be ON when the plus input is greater than or equal to the minus input plus the trigger deadband (TDB). The output will go OFF when the plus input is less than or equal to the minus input minus the restore deadband (RDB).



COMPARATOR

Table 12 Function 22 logic

Condition	Output
Plus Input ≥ (Minus Input + TDB)	ON
Plus Input ≤ (Minus Input + RDB)	OFF
(Minus Input - RDB) < Plus Input < (Minus Input + TDB)	No change

Function 23: Change of State (COS) Detector

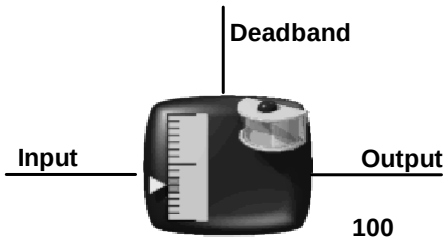
Description Turns a binary output ON momentarily whenever the analog input changes by more than the value entered for deadband (DB).

Remarks Function 23 compares the input value to a stored value. The stored value is set to the input value each time the output is ON and does not change while the output is OFF.

The output is ON when the input is greater than the stored value plus the DB, or the input is less than the stored value minus the DB. Otherwise, the output is OFF.

Example If DB is set to 1.0, and the input when the DDC is first initiated is 13.2 (which then becomes the stored value), then the output turns ON for one pass of the DDC the first time the input reaches 14.3 or greater or 12.1 or less. If the input varies between 12.2 and 14.2, the stored value remains the same.

If the input were to suddenly change to 14.6, for example, the output would turn ON and 14.6 would become the new stored value.



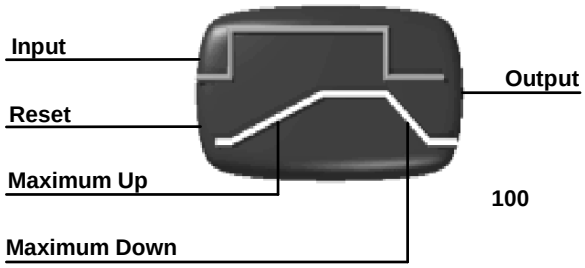
CHANGE OF STATE DETECTOR

Function 24: Restrictor

Description Restricts the rate of change of an analog value.

Remarks The inputs to Function 24 are made up of an analog data input (IN), binary reset input (RST), maximum up count (MUP), and maximum down count (MDN). As long as the binary reset input is ON, the output attempts to match the analog data input; however, the rate of change of the output is limited by the maximum up and maximum down inputs. The maximum up input regulates the maximum increase allowed in the output per second, while the maximum down input regulates the maximum decrease allowed in the output per second. The maximum up count and maximum down count are independently adjustable.

The output is set immediately to zero when the reset input turns OFF.



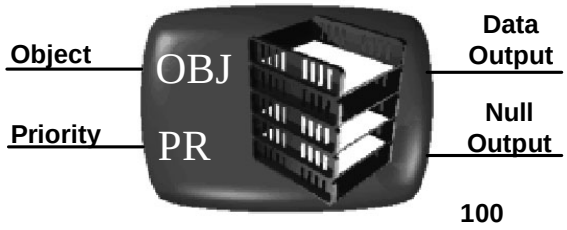
RESTRICTOR

Function 26: Priority Array Read (VLC only)

Description Outputs the value of a specified element of an object's BACnet priority-array index and indicates with a separate output whether the specified element is NULL.

Remarks The BACnet object (OBJ) and priority-array index (PR) are inputs to this function. The data output equals the current value of the specified element of the priority array, except when the element is NULL, in which case the data output is 0 (or OFF when used as a binary value).

The function's NULL output is binary. It is OFF if the specified element is NULL and ON if other than NULL.



PRIORITY READ

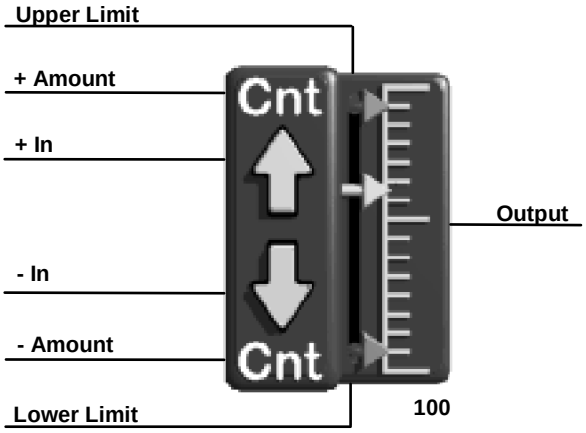
Function 27: Increment/Decrement

Description Increases the output value (IO) by an Up/Pass value (UPCNT) or decrease it by a Down/Pass value (DNCNT) only if an ON value is passed to the Up Input (UP) or Down Input (DN), respectively, for each pass of the DDC.

Remarks For each pass of the DDC, the value of the UPCNT value is added to IO whenever UP is ON.

Similarly, the value of the DNCNT is subtracted from IO whenever DN is ON. The value of IO is limited to the range defined by the Upper Limit and Lower Limit inputs.

Note that this function only adds to IO or subtracts from IO when either UP or DN is ON.

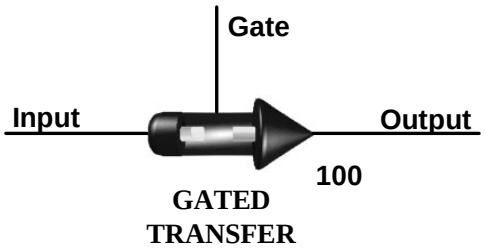


INCREMENT/DECREMENT

Function 28: Gated Transfer

Description Write the input value to the output value only when the Gate input is ON.

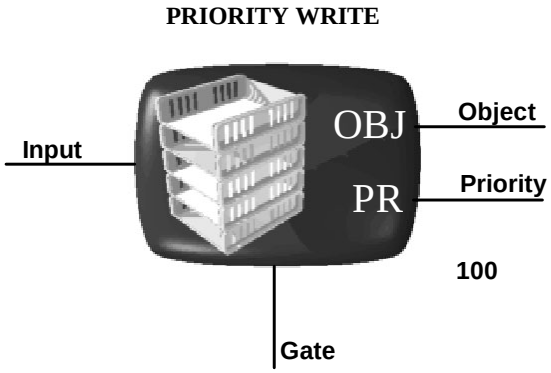
Remarks This function will not write to the output if the Gate input is OFF. Note that, unlike Function 47: Sample and Hold, this function does not store the value of the output, meaning the output will not necessarily remain constant when the Gate input is OFF.



Function 29: Gated Priority Transfer (VLC only)

Description Write the input value to a BACnet object with the specified priority other than the default priority. When the binary gate control is OFF, a NULL value is written to the specified output.

Remarks When the binary gate control is ON, the value of the input is written to the output (which should be a BACnet object), with the priority specified by the priority input. The output must be a BACnet object that has a priority array.



Function 30: Subtraction

Description Subtracts one input value from another.

Remarks Function 30 subtracts the value of analog input 2 (-) from the value of analog input 1 (+). The output is then set to the result.

Function 31: Addition

Description Adds two input values.

Remarks Function 31 adds analog input 1 and analog input 2. The output is then set to the result.

Function 32: Transfer Data

Description Copies a value from one property to another.

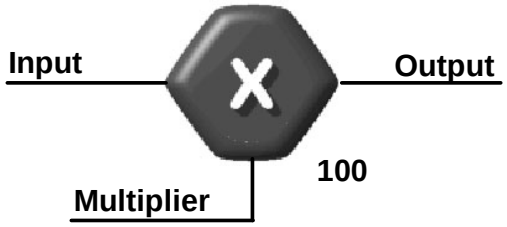
Remarks Function 32 is used to transfer analog or binary data from one property to another without changing the data content.

TRANSFER

Function 35: Multiplication

Description Multiplies (*) one value by another.

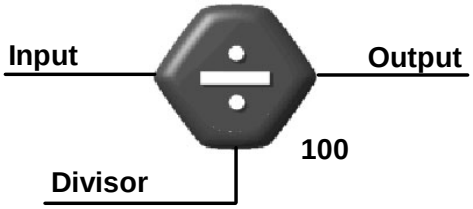
Remarks Input (multiplicand) is multiplied by multiplier, and the product is written to the output.



Function 36: Division

Description Divides (/) one value by another.

Remarks Input (dividend) is divided by divisor, and the resulting quotient is written to the output.

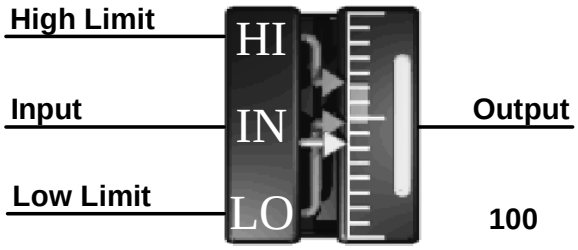


Function 39: Within a Range

Description Turns the output from OFF to ON whenever the input value is within the range defined by the Low 1 and Limit 2 values.

Remarks Function 39 has three analog inputs and one binary output. The output is ON whenever the input value is ≤ the Low Limit value and the input value is ≥ the High Limit value. Otherwise, the output is OFF.

Note that the output is ON if the input value is equal to either of the limits, and one does not necessarily have to be less than the other.

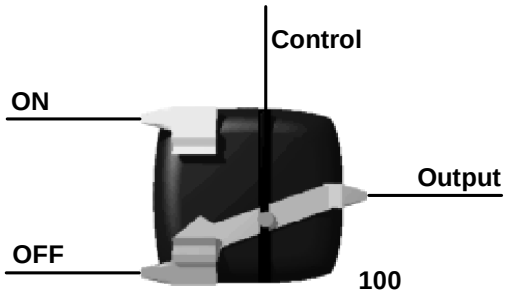


WITHIN

Function 40: Switch

Description Selects an output value from two input values, depending on the value of a binary input.

Remarks Function 40 has two analog inputs, one binary input, and an analog output. The output equals the ON analog input if the binary control input is ON, and the output equals the OFF analog input if the binary control input is OFF.



SWITCH

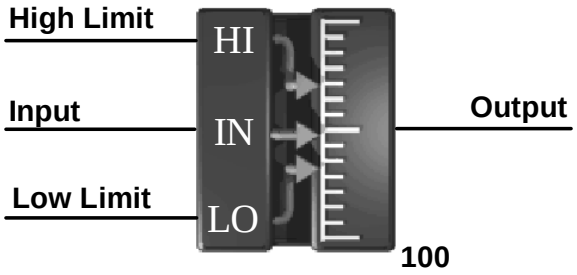
Function 41: High/Low Limiter

Description Restricts an analog value to a range defined by two limits.

Remarks Function 41 has three analog inputs and one analog output. The output will equal the analog input as long as it is within the range defined by the high limit and low limit.

If the analog input exceeds the high limit, the output is set to the value of the high limit. If the analog input is less than the low limit, the output is set to the low limit.

If the high limit value is less than the low limit value, the high limit has priority (that is, the output is set to the high limit, regardless of the analog input value).



LIMITER

Function 44: Run-Time Accumulator

Description Outputs a cumulative analog run time value for a binary input.

Remarks The output (run time) equals the total accumulated time, in hours, that the binary input has been ON. Run time increases by 1 for each hour that the input has been ON. The run-time output can be assigned to any AV in the VLC, including those stored directly in EEPROM. The AV is written to only when an additional hour of run time has accumulated.

This is the only DDC function that can write to an EEPROM-stored AV.

RUNTIME ACCUMULATOR

Function 45: Two-Point Linear Converter

Description Performs a linear conversion on the input to produce an output. Two points (IN1, OUT1) and (IN2, OUT2) are used to define the line relating input to output.

Remarks A straight line relationship (of the form y = a + bx) determines the output as a function of the input.

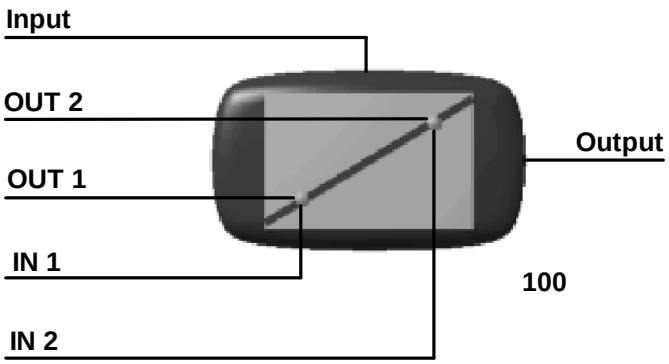
By inputting two points on this line, the line is then defined.

Note VisualLogic refers to inputs differently than DDC.

Use the following to translate: X1 = IN 1, X2 = IN 2, Y1 = OUT 1, Y2 = OUT 2.

When input = IN1, output = OUT1, and when input = IN2, output = OUT2. You can use this function to convert degrees Fahrenheit to degrees Celsius, using the following values for IN1, OUT1, IN2 and OUT2:

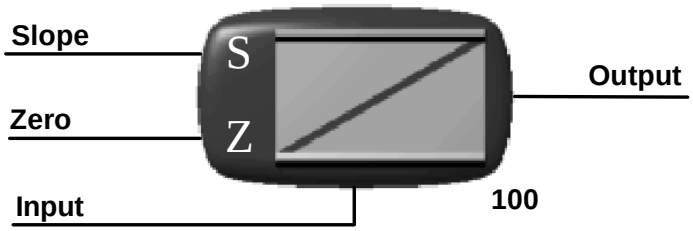
- IN1=32
- OUT1=0
- IN2=212
- OUT2=100



2 INPUT SCALER

Function 46: Linear Converter

Description Performs a linear conversion on the input and produce an output using zero and range values as follows: output = Zero + (Input X Slope).

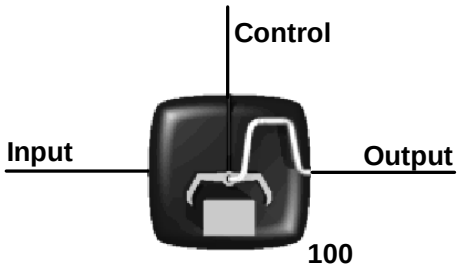


LINEAR SCALER

Function 47: Sample and Hold

Description Records and stores an analog value as prompted by a binary input.

Remarks Function 47 has one analog input, one binary sample control (CTRL) input, and an analog output. When the CTRL input is ON, the output and the stored value are set equal to the input. When the CTRL input is OFF, the output is set to the last stored value.



SAMPLE & HOLD

Function 48: Analog to Timed Binary Converter

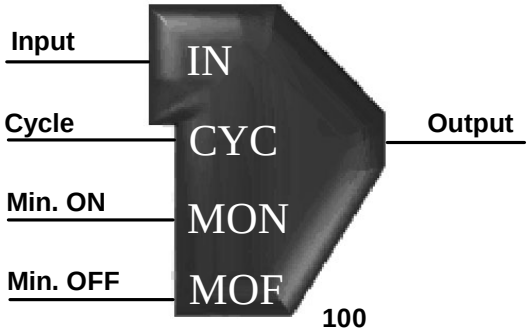
Description Cycles the output ON for a portion of each cycle time (CYC), which is adjustable, based on a 0.0 to 100.0 analog input control signal. A minimum ON time (MON) and minimum OFF time (MOF) prevent short cycling.

Remarks The output turns ON only if the calculated ON time is greater than the MON. If the output is ON, it remains ON until it has been ON for the calculated ON time and the MON has expired. The output remains ON continuously if the calculated OFF time is less than the MOF.

Time resolution is 1 second for VLC DDC and Global/Building controller DDC.

$$CalculatedOnTime = CycleTime\langle\frac{Input}{100.0}\rangle$$

$$CalculatedOffTime = CycleTime - CalculatedOnTime$$



Analog to Binary Output Timer

Function 49: Thermal Valve, Modulating Output (VLC only)

Description Pulses the binary output ON every 2.55 seconds, varying the ON time of the output (pulse width) from 0 to 2.55 seconds as the analog input varies from 0–100.0. If analog input is 0, the output remains OFF.

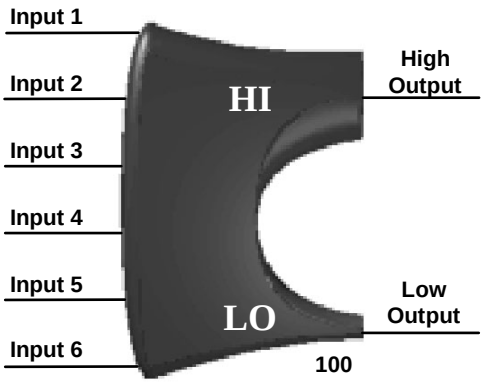
Remarks The pulse width is calculated using a nonlinear conversion to better match the thermal modulating valve (TMV). Do not use a NOT on the output of this device. To reverse a valve, reverse the signal by subtracting it from 100 before inputting it to this function.

Function 50: High/Low Selector

Description Selects the highest and lowest values from among six inputs.

Remarks Function 50 has six analog inputs and two analog outputs. The high output will equal the value of the highest input. The low output will equal the value of the lowest input. All inputs must be assigned.

If fewer than six inputs are needed, repeat one or more of the input assignments to fill the remaining inputs.



HIGH/LOW SELECTOR

Function 51: Proportional Integral (PI) Controller

Description Provides proportional integral (PI) control. Output is adjusted in an attempt to get the feedback input (FB) to match the setpoint (SP).

Remarks Inputs are FB (typically a space temperature), setpoint (SP), proportional constant (Kp), integral constant (Ki), maximum integral change (I_{max}), integral startup (STUP), and integral limit (I_{lim}). The output of this function can be expressed as P+I+50, where P is the proportional component and I is the integral component. Each of these is calculated as indicated below. Note that Error (E) is an intermediate variable equal to FB - SP.

Output = P + I + 50

where

$P = K_p(E)$

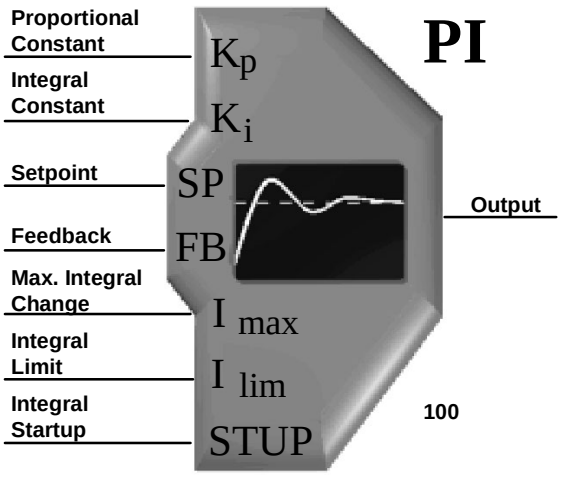
$I = I_{prev} + I_{inc}$ (I is limited to I_{lim})

$I_{inc} = E\left(\frac{K_i}{60}\right)$, which is calculated each DDC pass. (I_{inc} is limited to a maximum of $\frac{IMX}{60}$.)

I_{prev} is I from the most recent calculation. When DDC initializes, I_{prev} is set to STUP for the first DDC loop.

Also, when $K_i = 0$, the value of I = STUP.

See “Understanding BACtalk PI and PID functions” on page 58 for more information.



Function 52: Proportional Integral Derivative (PID) Controller

Description Provides proportional integral derivative (PID) control. Function 52 uses a PID algorithm to adjust the output in an attempt to get the input to match the setpoint.

Remarks This function is similar to Function 51, except that a derivative component (D) is included in the output calculation. D is the rate of change in E per second times the constant K_d, which is an input to the device.

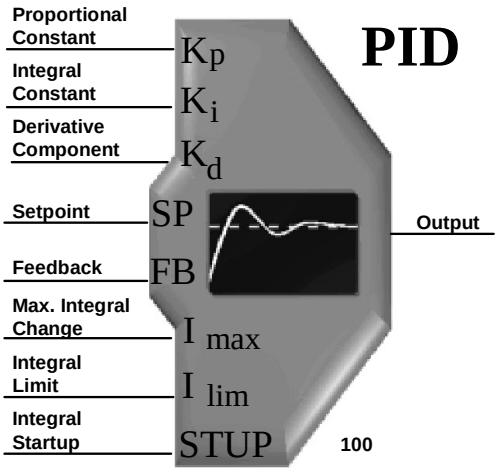
Note that Error (E) is an intermediate variable equal to FB – SP.

Output = P + I + D + 50

where

P and I are as calculated for Function 51.

$D = K_d(E - E_{prev})$. E_{prev} represents the value of E from the previous pass of DDC. D is calculated every 0.1 second in VLCs and every 1 second in global controllers and expandable controllers.



Function 54: Floating Motor Controller with No Time-out

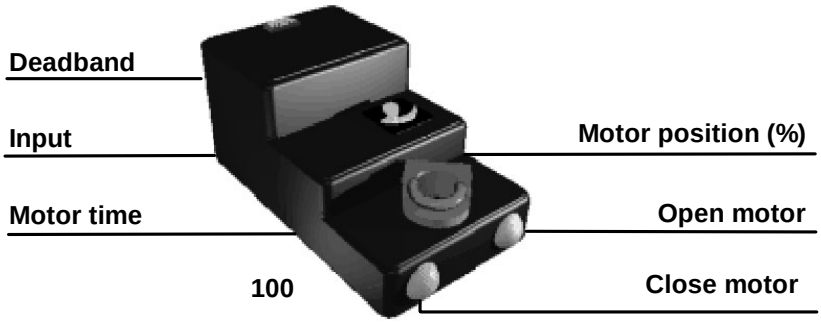
Description Provides floating point control of a motor based on a 0–100.0 control signal.

Remarks Function 54 has three analog inputs, two binary outputs, and one analog output. The input control signal (0.0–100.0) is compared to the current motor position (as estimated internally by the function). If the desired position is greater than the current position by more than deadband (DB), the open motor (OP) output will be ON. If the desired position is less than the current position by more than DB, the close motor (CL) output will be ON. If the current position is within DB of the input, both binary outputs will remain OFF.

The motor time (MT) input represents the time required (in seconds) for the motor to go from fully closed to fully open. The function estimates the current motor position (%) output based on the motor time and the cumulative ON times of the open motor and close motor outputs.

As the motor modulates open and closed, the function-estimated motor position will typically deviate further from the actual position. Also, the VLC assumes on power up that the motor is fully closed and will pulse the motor open to the currently desired position.

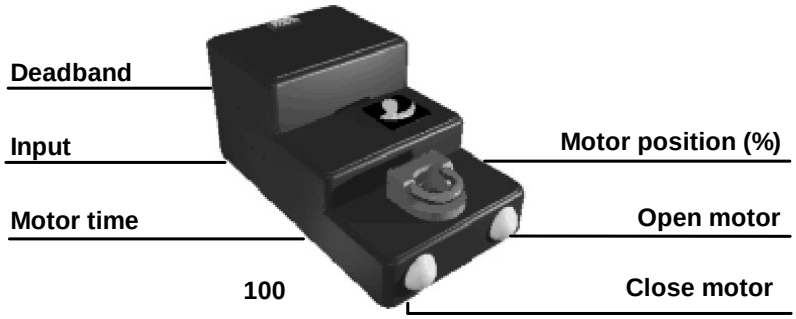
This means that the motor can also get out of phase with the function-estimated position if it is driven open and a power interrupt occurs. Use the initialization flag, which is ON only during the first DDC loop, and additional DDC to eliminate this out-of-phase condition on power-up. When the control signal reaches 0.0 or 100.0 and the motor is driven fully closed or open, the estimated position is automatically recalibrated.



Function 55: Floating Motor Controller with Time-out

Description Provides floating point control of a motor (or any device driven open or closed by a BO) based on a 0–100.0 control signal.

Remarks This function is the same as Function 54, with the addition of a time-out feature. When the input remains at 0.0 for an extended period, the close command (CL) output turns OFF for motor time (MT) seconds after the estimated damper position is fully closed. When the input remains at 100.0 for an extended period, the open command (OP) output turns OFF for MT seconds after the estimated damper position is fully open.

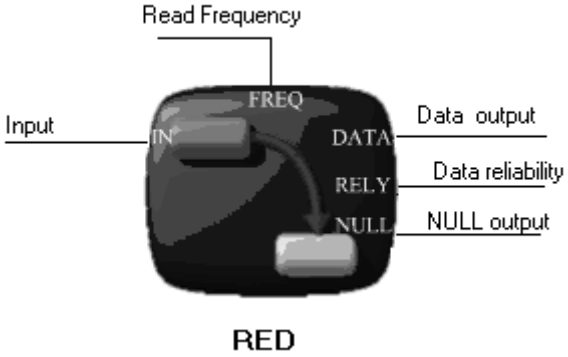


FMC w/TIMEOUT

Function 60: Read External Device

Description Reads data from an external BACnet device object at a specified rate (READ FREQUENCY).

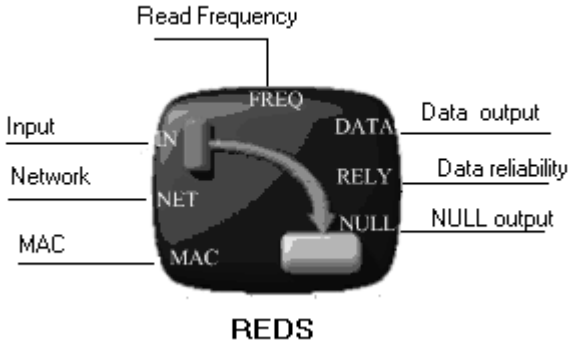
Remarks The INPUT is the device instance, object, and property that will be transferred to the present-value of the OUTPUT object. If the VLC can successfully read the INPUT, the DATA RELIABILITY is 1 (ON), otherwise it is 0 (OFF). NULL OUTPUT is normally 0 (OFF). If the device object being read contains a BACnet value of NULL (empty), the NULL OUTPUT is set to 1 (ON) to indicate the DATA OUTPUT is invalid.



Function 61: Read External Slave Device

Description Reads data from an external BACnet slave device object at a specified rate (READ FREQUENCY).

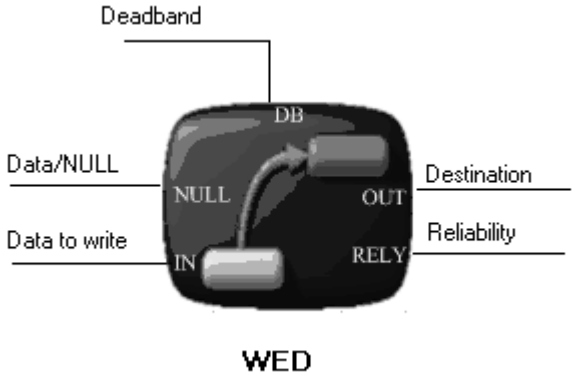
Remarks The NETWORK and MS/TP MAC specify the network and MAC address of the external object. The INPUT is the device instance, object, and property that will be transferred to the present-value of the OUTPUT object. If the VLC can successfully read the INPUT, the DATA RELIABILITY is 1 (ON), otherwise it is 0 (OFF). NULL OUTPUT is normally 0 (OFF). If the device object being read contains a BACnet value of NULL (empty), the NULL OUTPUT is set to 1 (ON) to indicate the DATA OUTPUT is invalid. Otherwise, NULL OUTPUT is set to 0 (OFF).



Function 62: Write External Device

Description Writes data (DATA TO WRITE) to an external BACnet device object.

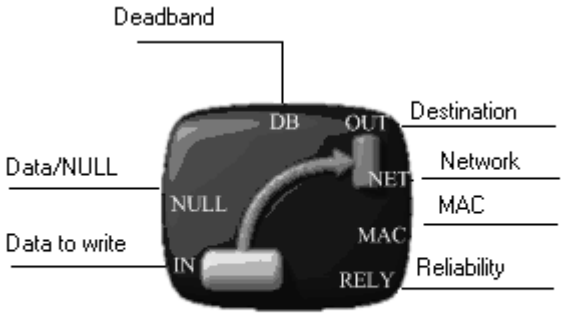
Remarks The DESTINATION specifies the object, property, and index to be written to. A write is attempted whenever the DATA TO WRITE changes by the amount of DEADBAND. DATA/NULL is set to 1 (DATA) to send the value in DATA TO WRITE, and 0 (NULL) if the special BACnet NULL (empty) value is to be written. The WRITE RELIABILITY is set to 1 (ON) when the external device acknowledges the write. It is set to 0 (OFF) whenever the external device does not respond.



Function 63: Write External Slave Device

Description Writes data (DATA TO WRITE) to an external BACnet slave device object.

Remarks The NETWORK and MS/TP MAC specify the BACnet network and MAC where the slave device resides. The DESTINATION specifies the object, property, and index to be written to. A write is attempted whenever the DATA TO WRITE changes by the amount of DEADBAND. DATA/NULL is set to 1 (DATA) to send the value in DATA TO WRITE, and 0 (NULL) if the special BACnet NULL (empty) value is to be written. The WRITE RELIABILITY is set to 1 (ON) when the external device acknowledges the write. It is set to 0 (OFF) whenever the external device does not respond.



WEDS

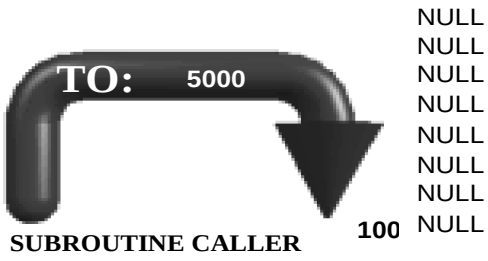
Function 67: Subroutine Caller (global controller only)

Description Calls a subroutine and sets values of the substitution points to be used for the subroutine.

Remarks When Function 67: Subroutine Caller executes, the next function executed is the sequence number specified as the Subroutine's Starting Sequence in the Subroutine Caller setup. The Subroutine Starting Sequence must be programmed at a sequence number higher than Function 1: End of Normal and lower than Function 2: End of Subroutine (that is, it must be outside normal DDC space and within subroutine DDC space). Functions execute in order until End of Subroutine is encountered, at which point program execution returns to the normal DDC space, beginning with the function immediately after the Subroutine Caller.

Parameters are equivalent to substitution points. Parameter 0, for example, is equivalent to Substitution Point 0. When subroutine DDC encounters an input or output defined as a substitution point, the value or data point entered at the Subroutine Caller is used.

Each Subroutine Caller can have different data points assigned. In this way, a single DDC subroutine can execute using different data points and values, as long as the subroutine is called with different Subroutine Callers.



Object and property reference 7

This chapter lists the objects in BACtalk unitary, global, and expandable controllers followed by a reference to the properties of those objects. Use this list to interpret the source and nature of system data.

BACtalk expandable controller

This section lists the objects in a BACtalk expandable controller, followed by a reference to the properties of those objects. Use this list to interpret the source and nature of system data.

Notes

- The **W** column indicates whether the property is writable. Properties without a check mark in this column are read-only. Some items can only be written to through special setup. These are checked as writable and noted under Remarks.
- In the **Example** column, items in Boldface always appear as listed for that item. For example, the object-type property of a device object will always return the word "Device" to the Envision for BACtalk display.
- The **Type** column indicates a BACnet data type. Unsigned and Signed indicate integer values; enumerated indicates an enumerated value table; other data types may exist.

Objects in the VLX controller

Table 13 VLX objects

Object (instance range)	Remarks
AI (0-7999)	Analog input objects associated with physical, universal input terminals on EXPs. AIs are identified as AI-e0nn, where e is the EXP address (0-7) and nn is the input terminal number.
AO (0-7999)	Analog output objects associated with physical output terminals on EXPs. AOs are identified as AO-e0nn, where e is the EXP address (0-7) and nn is the output terminal number.
AV (0–7999)	**RESERVED AVs** for EXP configuration, status, and backup values. Do not use these AVs for general programming or automation. These AVs do not support the priority-array property.
AV (8000–8499)	**General use AVs.** These AVs support the priority-array property.
BI (0-7999)	Binary input objects associated with physical, universal input terminals on EXPs. BIs are identified as BI-e0nn, where e is the EXP address (0-7) and nn is the input terminal number.
BO (0-7999)	Binary output objects associated with physical output terminals on EXPs. BOs are identified as BO-e0nn, where e is the EXP address (0-7) and nn is the output terminal number.
BV (8000-8499)	General use BVs. These BVs support the priority-array property.
Calendar	Describes a list of calendar dates, special event dates, holiday dates, and date ranges.
Device	Provides general information about a device.
Event Enrollment	Defines an event and connects the occurrence of the event to the transmission of an event notification. Used in BACtalk primarily for alarms.
File (0)	Provides information about the ROC file.
File (1024)	Provides information about the current DDC file.
File (2048)	Provides information about DDC trap file.
Notification Class	Stores a list of available recipients for the distribution of event notifications (alarms, trendlog gathering, and so on).
Program 0	Stores information about the ROC/controller program.
Program 1024	Stores program status information about the current DDC program.
Schedule	Controls designated properties by periodic schedule that may recur during a range of dates.

Properties of VLX AI objects

Table 14 Properties of the VLX AI object

Property	W	Type	Example	Remarks
cov-increment	✓	Real		If the present-value changes by this amount or greater, a change-of-value notification is sent to subscribed devices. Not used at present.
description	✓	Character string	Return Air Temp	An editable description of the object's location or function.
event-state		Enumerated	NORMAL	
object-identifier		BACnet_ Object_ Identifier	**AI** 5	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.

Table 14 Properties of the VLX AI object (Continued)

Property	W	Type	Example	Remarks
object-name		Character string	**EXP** 7 **AI** 05	
object-type		Enumerated	**AI**	Indicates an analog input (AI) object.
out-of-service	✓	Boolean	FALSE	TRUE decouples the present-value property from the physical input, and the present-value does not track further physical input changes. While TRUE, the present-value can be changed to any value to simulate conditions for testing. FALSE indicates that the present-value is tracking changes to the physical input.
present-value	✓	Real	72.3	Writable only when out-of-service = TRUE (see herein). Range of present-value depends on input setup. See "Setting inputs, outputs, and other function parameters" on page 31.
reliability		BACnet_ Reliability	NO FAULT DETECTED	Other possibility is UNRELIABLE_OTHER.
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.
units	✓	Enumerated	Deg F	Indicates the unit of measure for the AI, in BACnet engineering units.

Properties of VLX AO objects

Table 15 Properties of VLX AO objects

Property	W	Type	Example	Remarks
description	✓	Character string	Economizer Damper	An editable description of the object's location or function.
event-state		Enumerated	NORMAL	
object-identifier		BACnet_ Object_ Identifier	**AO** 5	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**EXP** 0 **AO** 00	
object-type		Enumerated	**AO**	
out-of-service	✓	Boolean	FALSE	TRUE decouples the present-value property from the physical output. While TRUE, the present-value can be changed to any value to simulate conditions for testing without affecting the actual physical output.
present-value	✓	REAL	75	Valid values are real numbers in the range 0–100. Values greater than 100 are interpreted as 100. When commanded, values are written to the present-value with a priority for writing, which corresponds to a priority-array index (see "priority-array" herein).
priority-array		BACnet Priority Array	<Array of BACnet Priority Value>	An array of prioritized values (indexes 1-16) controlling the present-value, index 1 having the highest priority. The value with the highest priority for writing controls the present-value. Possible values for priority-array indexes are real values or NULL. A NULL value indicates no command is issued at that priority level.

Table 15 Properties of VLX AO objects (Continued)

Property	W	Type	Example	Remarks
reliability		BACnet_ Reliability	NO FAULT DETECTED	Other possibilities are OVER RANGE, UNDER RANGE, UNRELIABLE_OTHER.
relinquish-default	✓	REAL	0	Default value to be used for present-value property when all priority-array indexes are NULL.
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.
units	✓	Enumerated	%	Indicates the AO's unit of measure in BACnet engineering units.

Properties of VLX AV objects

Table 16 Properties of VLX AV objects

Property	W	Type	Example	Remarks
cov-increment	✓	Real		If the present-value changes by this amount or greater, a change-of-value notification is sent to subscribed devices. Not used at present.
description	✓	Character string	Occupied Setpoint	A description assigned to describe the object's function.
event-state		Enumerated	Normal	
object-identifier		BACnet_ Object_ Identifier	**AV** 1	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string		For reserved AVs, shows the EXP and AV of interest. For example, **EXP 0 AV 01**. Otherwise shows AV <instance>.
object-type		Enumerated	**AV**	
out-of-service		Boolean	FALSE	
present-value	✓	Real	76.4	Range is $+/-3 \times 10^{38}$ (six significant digits of resolution).
status-flags		Bit string	<Bit string>	A four-position bit string that indicates the status of the AV. If a status bit =1, that status is TRUE.
units	✓	Enumerated	Deg F	Indicates the unit of measure, in BACnet engineering units, for the AV present-value.
priority-array		BACnet PriorityArray	<Array of BACnet PriorityValue>	NOT AVAILABLE IN AVs (0-7999). GENERAL PURPOSE AVs ONLY. An array of prioritized values (indexes 1-16) controlling the present-value, index 1 having the highest priority. The value with the highest priority controls the present-value. Possible values for priority-array indexes are real values or NULL. A NULL value indicates no command is issued at that priority index.
relinquish-default	✓	REAL	0	NOT AVAILABLE IN AVs (0-7999). GENERAL PURPOSE AVs ONLY. Default value to be used for present-value property when all priority-array indexes are NULL.

Properties of VLX BI objects

Table 17 Properties of the VLX BI object

Property	W	Type	Example	Remarks
description	✓	Character string	Fan Status	An editable description of the object's location or function.
event-state		Enumerated	NORMAL	
object-identifier		BACnet_ Object_ Identifier	**BI** 10	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**BI** 10	
object-type		Enumerated	**BI**	Indicates a binary input (BI) object.
out-of-service	✓	Boolean	FALSE	TRUE decouples the present-value property from the physical input, and the present-value does not track further physical input changes. While TRUE, the present-value can be changed to any value to simulate conditions for testing. FALSE indicates that the present-value is tracking changes to the physical input.
polarity			NORMAL	
present-value	✓	Logical state	ACTIVE	ACTIVE or INACTIVE. Writable only when out-of-service = TRUE (see herein).
reliability			NO FAULT DETECTED	Other possibility is UNRELIABLE_OTHER.
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.

Properties of VLX BO objects

Table 18 Properties of the VLX BO object

Property	W	Type	Example	Remarks
description	✓	Character string	Fan Start/ Stop	An editable description of the object's location or function.
event-state		Enumerated	Normal	If the object does not support intrinsic reporting, the value will be NORMAL.
object-identifier		BACnet_ Object_ Identifier	**BO** 1	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**BO** 01	
object-type		Enumerated	**BO**	
out-of-service	✓	Boolean	FALSE	TRUE decouples the present-value property from the physical output. While TRUE, the present-value can be changed to any value to simulate conditions for testing without affecting the actual physical output.
present-value	✓	Enumerated	INACTIVE	Either ACTIVE or INACTIVE. Note that a NULL value can be written to the present-value on data displays, but the value is actually written to a priority-array property. The present-value is the result of the priority-array.
priority-array		BACnet Priority Array	<Array of BACnet Priority Value>	A read-only array of prioritized values (1-16) controlling the present-value, priority 1 having the highest priority. The value with the highest priority controls the present-value. Possible values for priority-array indexes are ACTIVE, INACTIVE, or NULL. A NULL value indicates no command is issued at that priority level.
reliability		BACnet_ Reliability	NO FAULT DETECTED	Other possibility is UNRELIABLE_OTHER.
relinquish-default	✓	Enumerated	INACTIVE	Default value used for present-value property when all priority-array values are NULL.
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.

Properties of VLX BV objects

Table 19 Properties of VLX BV objects

Property	W	Type	Example	Remarks
active-text			**ON**	
description	✓	Character string	Occupied Setpoint	A description assigned for the object's function.
event-state		Enumerated	NORMAL	
inactive-text			**OFF**	
object-identifier		BACnet_ Object_ Identifier	**BV** 8413	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**BV** 8413	
object-type		Enumerated	**BV**	Indicates a binary value (BV).

Table 19 Properties of VLX BV objects (Continued)

Property	W	Type	Example	Remarks
out-of-service		Boolean	FALSE	
present-value	✓	Enumerated	INACTIVE	Either ACTIVE (ON) or INACTIVE (OFF).
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.
priority-array		BACnet PriorityArray	<Array of BACnet PriorityValue>	An array of prioritized values (indexes 1-16) controlling the present-value, index 1 having the highest priority. The value with the highest priority controls the present-value. Possible values for priority-array indexes are ACTIVE, INACTIVE, or NULL. A NULL value indicates no command is issued at that priority-array index.
relinquish-default	✓	REAL	0	Default value to be used for present-value property when all priority-array indexes are NULL.

Properties of the VLX device object

Table 20 Properties of the VLX device object

Property	W	Type	Example	Remarks
apdu-segment- timeout	✓	Unsigned	6000	The time after transmission of a "segment" until the lack of a reply means it was assumed to be lost (in milliseconds, 1000 = 1 sec). Default = 6000.
apdu-timeout	✓	Unsigned	6000	The time after transmission of an APDU until the lack of a reply means it was assumed to be lost. The APDU time-out value for this device in milliseconds (1000 = 1 sec). Default = 6000.
application-software-version		Character string	VLX V1.0	Indicates the ROC file version.
daylight-savings-status	✓	Boolean	FALSE	Indicates whether daylight savings is in effect (TRUE) or not (FALSE). Not used at present.
description	✓	Character string	Second floor controller	Assigned by the user to describe the device's function.
device-address-binding		List		Empty.
firmware-revision		Character string	BACtalk VLX v1.1 02/02/ 2002	Indicates the VLX boot code version.
local-date	✓	Date	Sunday, 02/ 24/2002	Indicates date: day of the week, month/day/year. Writable through Time Sync.
local-time	✓	Time	10:15:56.00 am	Indicates the time stored in the device. Writable through Time Sync.
location	✓	Character string	East Wing	Indicates the physical location of the device.
max-apdu-length-accepted		Unsigned	1476	The maximum message packet size that the device can handle.
max-info-frames	✓	Unsigned	60	Number of MS/TP messages the device will send per token hold. Default = 60. Max. = 200.
max-master		Unsigned	127	Highest MAC address (above this unit's) that another MS/TP master should be set to.

Table 20 Properties of the VLX device object (Continued)

Property	W	Type	Example	Remarks
model-name		Character string	VLX Controller	Assigned by the vendor to indicate the device model.
number-of-apdu-retries	✓	Unsigned	3	The number of times a message will be resent after it is assumed to be lost.
object-identifier		BACnet_ Object_ Identifier	**Device** 200	This property consists of the object-type property and the device instance, which is a numeric code that identifies the device of interest.
object-list		Array		An array whose elements list the object-identifier properties of all objects the device supports.
object-name		Character string	Controller 200	No two devices are permitted to have the same object name.
object-type		Enumerated	**Device**	
protocol-conformance-class		Unsigned	3	Integer from 1–6 indicating the conformance class of the device. A device must support a standardized set of services and object types to claim a particular class conformance.
protocol-object-types-supported		Bit string	<Bit string>	An internally used bit string. Indicates which BACnet object types reside in the device.
protocol-services-supported		Bit string	<Bit string>	An internally used bit string. Indicates which BACnet services the device can process.
protocol-version		Unsigned	1	Indicates the version of the BACnet protocol supported by the device.
segmentation-supported		Enumerated	segmented both	Device is capable of segmenting both transmission and reply messages.
system-status		Enumerated	Operational	Other possible values are operational-read-only, download-required, download-in-progress, non-operational.
utc-offset	✓	Signed	0	Coordinated Universal Time offset, in minutes. Not used at present.
vendor-identifier		Unsigned	18	A unique code assigned by ASHRAE to the manufacturer, in this case, Alerton.
vendor-name		Character string	Alerton	Indicates the device manufacturer.

Properties of VLX event-enrollment objects

Table 21 Properties of VLX event-enrollment objects

Property	W	Type	Example	Remarks
acked-transitions	✓	bit string	To-offnormal =1 , To-fault = 1, To-normal = 1	Indicates whether the corresponding transitions have been acknowledged. A 1 indicates that the transition was acknowledged.
description	✓	Character string	Event enrollment 0	A description assigned to describe the object's function.
event-enable	✓	bit string	To-offnormal =1 , To-fault = 1, To-normal = 1	Indicates whether notifications are enabled for these event transition types. A 1 indicates that the transition will be reported. Set in the Event Enrollment Editor at the operator workstation.
event-parameters		BACnetEvent Parameter	change_of_ bitstring	

Table 21 Properties of VLX event-enrollment objects (Continued)

Property	W	Type	Example	Remarks
event-state		Enumerated	NORMAL	Indicates the current state of the event.
event-type	✓	Enumerated	CHANGE_OF _BITSTRING	Indicates the type of event algorithm to be used to detect events.
issue-confirmed-notifications		Boolean	TRUE	Determines whether confirmed or unconfirmed notifications are used when a notification-class object isn't used (that is, a recipient is specified). Set in the Event Enrollment Editor at the operator workstation.
notification-class		Enumerated	1	Indicates the notification class to be used for event transitions. Set in the Event Enrollment Editor at the operator workstation.
notify-type		Unsigned	alarm	Indicates whether the object is set up for alarms or events.
object-identifier		BACnet_ Object_ Identifier	**Event-enrollment** 0	This property consists of the object-type property and the Object Instance, which is a numeric code that identifies the object of interest.
object-name		Character string	Alarm	Assigned at the operator workstation.
object-property-reference	✓	Boolean	FALSE	Indicates whether the file has been saved for backup.
object-type			Event-enrollment	
priority	✓	Unsigned	9	Priority for issuing event notifications.
process-identifier		Unsigned	3	A numeric identifier for a handling process in the recipient device. Set in the Event Enrollment Editor at the operator workstation.
recipient		Enumerated	<>	Unused.

Properties of VLX file objects

Table 22 Properties of VLX file objects

Property	W	Type	Example	Remarks
archive	✓	Boolean	FALSE	Indicates whether the file has been saved for backup.
description	✓	Character string	VLX ROC File	A description assigned to describe the object's function.
file-access-method		Enumerated	stream access	
file-size		Unsigned	983040	The size of the file, in bytes.
file-type		Character string	ROC	Also DDC or TRAP.
modification-date		Time	4/29/1997 10:22:20:00a	The date and time the file was last modified.
object-identifier		BACnet_ Object_ Identifier	**file** 0	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**File** 0	
object-type		Enumerated	file	
read-only		Boolean	TRUE	Indicates whether the file can be written to by BACnet services.

Properties of VLX notification-class objects

Table 23 Properties of VLX notification-class objects

Property	W	Type	Example	Remarks
ack-required	✓	Bit string	To offnormal = 1, to fault = 1, to normal = 1	Indicates whether an acknowledgment is required for event transitions. A 1 indicates that acknowledgment is required. Set up at the operator workstation.
description	✓	Character string	Alarm Handler	An editable description of the object's location or function.
notification-class		Unsigned	1	Echoes the object instance.
object-identifier		BACnet_ Object_ Identifier	**Notification -class** 1	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name	✓	Character string	Alarm Handler 1	
object-type		Enumerated	Notification-class	
recipient-list	✓	List	<List of BACnet Destination>	Lists the devices that will receive notification when the notification class transitions. Set up at the operator workstation.
priority	✓	Array of Unsigned		Indicates the priority to be used for event notifications for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively.

Properties of VLX program objects

Table 24 Properties of VLX program objects

Property	W	Type	Example	Remarks
description	✓	Character string	Occupied Setpoint	A description assigned to describe the object's function.
description-of-halt		Character string	Program halted by request	
instance-of		Character string	alerton hq alerVLX 0*00000000*	Header information for the file. Program 0 does not support this property.
object-identifier		BACnet_ Object_ Identifier	**program** 1024	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**Program Object** 1024	
object-type		Enumerated	Program	
out-of-service		Boolean	FALSE	
program-change	✓	Enumerated	READY	Used to command the program state. A program can be stopped using the HALT command, for example, and started again with RESTART.
program-location		Character string	DDC Sequence = 60	Set when program stops.

Table 24 Properties of VLX program objects (Continued)

Property	W	Type	Example	Remarks
program-state		Enumerated	RUNNING	Possible states include RUNNING, IDLE, HALTED.
reason-for-halt		Enumerated	PROGRAM	
reliability		Enumerated	NO FAULT DETECTED	
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.

Properties of VLX schedule objects

Table 25 Properties of VLX schedule objects

Property	W	Type	Example	Remarks
description	✓	Character string	Weekend Gym	A description assigned to describe the object's function.
effective-period	✓	Sequence	<BACnet DateRange>	Assigned in schedule setup at the operator workstation.
exception-schedule	✓	Sequence	<Array of BACnet Special Event>	Assigned in schedule setup at the operator workstation.
list-of-object-property-references	✓	List	<List of BACnet Object Property Reference>	The list of objects that this schedule commands.
object-identifier		BACnet_ Object_ Identifier	**schedule** 0	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name	✓	Character string	schedule 000	Assigned in schedule setup at the operator workstation.
object-type		Enumerated	schedule	
present-value	✓		ACTIVE	Indicates the value most recently written to a referenced object property. May be analog, binary, or other, depending on the controlled property.
priority-for-writing	✓	Unsigned	16	Assigned in schedule setup at the operator workstation.
weekly-schedule	✓	Sequence	<Array of BACnetDaily Schedule>	Assigned in schedule setup at the operator workstation.

BACtalk global controller

This section lists the objects in a BACtalk global controller (this includes the BACtalk control modules (BCMs) which act as global controllers in the BACtalk system), followed by a reference to the properties of those objects. Use this list to interpret the source and nature of system data. See the *BCM Installation and Operations Guide* (LTBT-TM-BCMIOG) for information about virtual objects and properties that appear in the BCM-TUX module.

Notes

- The **W** column indicates whether the property is writable. Properties without a check mark in this column are read-only. Some items can only be written to through special setup. These are checked as writable and noted under Remarks.
- In the **Example** column, items in boldface always appear as listed for that item. For example, the object-type property of a device object will always return the word “Device” to the Envision for BACtalk display.
- The **Type** column indicates a BACnet data type. Unsigned and Signed indicate integer values; enumerated indicates an enumerated value table; other data types may exist.

Objects in global controllers

Table 26 Objects in the global controller

Object	Function	Range of Present Value
AV (0-999)	REAL.	$\pm$ 3.40282 x 10^{38} (six significant digits of resolution)
BV (0–999)	Binary value.	ACTIVE or INACTIVE
Calendar	Describes a list of calendar dates, special event dates, holiday dates, and date ranges.	N/A
Demand Limiter	Proprietary Alerton object for demand limiting function	0-100 (ramp value)
Device	Provides general information about a device.	
Event Enrollment	Defines an event and connects the occurrence of the event to the transmission of an event notification. Used in BACtalk primarily for alarms.	
File 0	Stores information about the ROC file in a controller.	
File 1024	Stores file information about the current DDC program.	
File 2048	DDC trap file.	
Notification Class	Stores a list of available recipients for the distribution of event notifications (alarms, trendlog gathering, etc.).	
Program 0	Stores information about the ROC/Controller program.	
Program 1024	Stores program status information about the current DDC program.	
Schedule	Controls designated properties by periodic schedule that may recur during a range of dates.	
Zones	Contains the individual properties and references required to support the optimum start and tenant activity features of Envision for BACtalk.	

Properties of the AV object

Table 27 Properties of a global controller AV object

Property	W	Type	Example	Remarks
cov-increment	✓	Real		If the present-value changes by this amount or greater, a change-of-value notification is sent to subscribed devices.
description	✓	Character string	Occupied Setpoint	A description assigned to describe the object's function.
event-state		Enumerated	Normal	
object-identifier		BACnet_ Object_ Identifier	**AV** 1	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**AV** 001	
object-type		Enumerated	AV	
out-of-service		Boolean	False	
present-value	✓	Real	76.4	Range is 3×10^{38} (six significant digits of resolution).
status-flags		Bit string	<Bit string>	A four-position bit string that indicates the status of the AV. If a status bit =1, that status is TRUE.
units		Enumerated	Deg F	Indicates the unit of measure, in BACnet engineering units, that the AV is expressed in.

Properties of the BV object

Table 28 Properties of a global controller BV object

Property	W	Type	Example	Remarks
description	✓	Character string	Occupied Setpoint	A description assigned for the object's function.
event-state		Enumerated	NORMAL	
object-identifier		BACnet_ Object_ Identifier	**BV** 1	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**BV** 001	
object-type		Enumerated	BV	Indicates a binary value (BV).
out-of-service		Boolean	FALSE	
present-value	✓	Enumerated	INACTIVE	Either ACTIVE (ON) or INACTIVE (OFF).
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.

Properties of the calendar object

Table 29 Properties of a global controller calendar object

Property	W	Type	Example	Remarks
date-list	✓	List	<List of BACnet Calendar Entry>	List of calendar dates.
description	✓	Character string	Holidays 1997	A description assigned to describe the object's function.
object-identifier		BACnet_Object_ Identifier	**calendar** 1	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name	✓	Character string	calendar 001	
object-type		Enumerated	calendar	
present-value	✓	Boolean	FALSE	TRUE if the current date is in the date list; FALSE if it is not.

Properties of the demand limiter object

Table 30 Properties of the demand limiter object

Property	W	Type	Example	Remarks
object-identifier		BACnet_Object_ Identifier	demand limiter 1	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name	✓	Character String		
object-type		BACnet_Object_ Type		
aler-enabled	✓	BOOLEAN		True if the demand limiter is enabled; False if it's disabled.
aler-max-enable-disable-ramp-step	✓	Real		Valid range is 1 to 100%.
description	✓	Character String		
aler-demand-window	✓	Unsigned		The total number of minutes in the demand window. Range is 1 to 60 minutes.
aler-anticipation-factor	✓	Real	3	Valid range is 1.0 to 10.0.
aler-meter-inputs	✓	BACnet Array 1 of BACtalk Meter Input	<List of meter inputs>	The list of meter inputs to the demand limiter. Currently limited to one meter input.
units	✓	Character String		Indicates the measurement units of aler_instantaneous_demand.
aler-instantaneous-demand		Real		Reflects the current reading from the meter (after applying conversion factors).
aler-average-demand		Real		The larger of two calculated averages: Average Demand Window and Average Anticipation Interval (if not set to 1.0).
aler-ramp-control	✓	BACtalk Device Object Property Reference or Value		The input that identifies the ramp parameters.

Table 30 Properties of the demand limiter object (Continued)

Property	W	Type	Example	Remarks
aler-ramp-parameters	✓	BACtalk Ramp Parameters		The shed ramp parameters.
aler-ramp-value		Real		The current ramp value (0-100%). This number is calculated whether the demand limiter object is enabled or not.
aler-active-ramp		Real		The current ramp value (0-100%) being used to control loads. This value will be between zero and the Aler_Ramp_Value. The exact value depends on whether the demand limiter object is enabled or disabled.
aler-binary-loads	✓	BACnet Array 5 of BACtalk Binary Shed Level		The list of binary loads.
aler-custom-binary-loads	✓	List of BACtalk Binary Loads		The list of custom binary loads.
aler-binary-load-status		List of BACtalk Demand Load Status		The status of each binary and custom binary load.
aler-total-binary-loads		Unsigned		The total number of binary and custom binary loads.
aler-binary-loads-shed		Unsigned		The total number of binary and custom binary loads that have been shed.
aler-total-analog-loads		Unsigned		The number of defined analog loads.
aler-analog-loads-in-shed		Unsigned		The total number of analog loads that have been shed.
aler-analog-loads		List of BACtalk Analog Loads		The list of analog loads.
aler-analog-load-status		List of BACtalk Demand Load Status		The status of each analog load.
aler-recent-history		List of BACtalk Demand History Sample		A list of historical data with the most recent data first.
out-of-service		BOOLEAN	FALSE	Not currently used so always set to False.

Properties of the device object

Table 31 Properties of a global controller device object

Property	W	Type	Example	Remarks
APDU-segment-timeout	✓	Unsigned	2000	The time after transmission of a "segment" until the lack of a reply means it was assumed to be lost (in milliseconds, 1000 = 1 sec).
APDU-timeout	✓	Unsigned	3000	The time after transmission of an APDU until the lack of a reply means it was assumed to be lost. The APDU timeout value for this device in milliseconds (1000 = 1 sec).
application-software-version		Character string	BTI Controller V3.20	Indicates the ROC version.
daylight-savings-status	✓	Boolean	FALSE	Indicates whether daylight savings is in effect (TRUE) or not (FALSE).
description	✓	Character string	Second floor controller	Assigned by the user to describe the device's function.
device-address-binding		List		Inaccessible.
firmware-revision		Character string	BACtalk BTI v3.20 11/17/ 2002	Indicates the firmware version.
local-date		Octet String	Wednesday, 5/14/1997	Indicates date: day of the week, month/day/year.
local-time		Time	10:15:56.00a m	Indicates the time stored in the device.
location	✓	Character string	East Wing	Indicates the physical location of the device.
max-APDU-length-accepted		Unsigned	1476	The maximum message packet size that the device can handle.
max-info-frames	✓	Unsigned	200 (maximum number)	Number of MS/TP messages the BTI will send per token hold.
max-master		Unsigned	127	The highest MS/TP MAC address the BTI will attempt to pass the token to.
model-name		Character string	BTI Controller	Assigned by the vendor to indicate the device model.
number-of-APDU-retries	✓	Unsigned	3	The number of times a message will be resent after it is assumed to be lost.
object-identifier		BACnet_ Object_ Identifier	**Device** 200	This property consists of the object-type property and the device instance (a numeric code that identifies the device) of the device of interest.
object-list		Array		An array whose elements list the object-identifier properties of all objects the device supports.
object-name		Character string	Controller 200	No two devices are permitted to have the same object name.
object-type		Enumerated		Device.
protocol-conformance-class		Unsigned	3	Integer from 1–6 indicating the conformance class of the device. A device must support a standardized set of services and object types to claim a particular class conformance.
protocol-object-types-supported		Bit string	<Bit string>	An internally used bit string. Indicates which BACnet object types reside in the device.

Table 31 Properties of a global controller device object (Continued)

Property	W	Type	Example	Remarks
protocol-services-supported		Bit string	<Bit string>	An internally used bit string. Indicates which BACnet services the device can process.
protocol-version		Unsigned	1	Indicates the version of the BACnet protocol supported by the device.
segmentation-supported		Enumerated	segmented both	Device is capable of segmenting both transmission and reply messages.
system-status		Enumerated	operational	Other possible values are operational-read-only, download-required, download-in-progress, non-operational.
utc-offset	✓	Signed	0	Coordinated Universal Time offset, in minutes.
vendor-identifier		Unsigned	18	A unique code assigned by ASHRAE to the manufacturer.
vendor-name		Character string	Alerton	Indicates the device manufacturer.

Properties of the event enrollment object

Table 32 Properties of a global controller event enrollment object

Property	W	Type	Example	Remarks
acked-transitions	✓	Bitstream	To-offnormal =1 , To-fault = 1, To-normal = 1	Indicates whether the corresponding transitions have been acknowledged. A 1 indicates that the transition was acknowledged. Set in the Event Enrollment Editor at the operator workstation.
description	✓	Character string	Event enrollment 0	A description assigned to describe the object's function.
event-enable	✓	Bitstream	To-offnormal =1 , To-fault = 1, To-normal = 1	Indicates whether notifications are enabled for these event transition types. A 1 indicates that the transition will be reported. Set in the Event Enrollment Editor at the operator workstation.
event-parameters		Time	<BACnet Event Parameter>	Determines the method used to monitor the referenced object.
event-state		Boolean	TRUE	Indicates whether the file can be written to by BACnet services.
event-type	✓		change of state	Indicates the type of event to be monitored and reported.
issue-confirmed-notifications		Boolean	TRUE	Determines whether confirmed or unconfirmed notifications are used when a notification class object isn't used (that is, a recipient is specified). Set in the Event Enrollment Editor at the operator workstation.
notification-class		Enumerated	1	Indicates the notification class to be used for event transitions. Set in the Event Enrollment Editor at the operator workstation.
notify-type		Unsigned	alarm	Indicates whether the object is set up for alarms or events.
object-identifier		BACnet_ Object_ Identifier	Event-enrollment 0	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	Alarm	Assigned at the operator workstation.
object-property-reference	✓	Boolean	FALSE	Indicates whether the file has been saved for backup.
object-type			event-enrollment	
priority		Unsigned	9	Priority for issuing event notifications. Set in the Event Enrollment Editor at the operator workstation.
process-identifier		Unsigned	3	A numeric identifier for a handling process in the recipient device. Set in the Event Enrollment Editor at the operator workstation.
recipient		Enumerated	<>	

Properties of the file object

Table 33 Properties of a global controller file object

Property	W	Type	Example	Remarks
archive	✓	Boolean	FALSE	Indicates whether the file has been saved for backup.
description	✓	Character string	BTI ROC File	A description assigned to describe the object's function.
file-access-method		Enumerated	stream access	
file-size		Unsigned	983040	The size of the file, in bytes.
file-type		Character string	ROC	Also DDC or TRAP.
modification-date		Time	4/29/1997 10:22:20:00 a	The date and time the file was last modified.
object-identifier		BACnet_ Object_ Identifier	**file** 0	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**File** 0	
object-type		Enumerated	file	
read-only		Boolean	TRUE	Indicates whether the file can be written to by BACnet services.

Properties of the notification class object

Table 34 Properties of a global controller notification class object

Property	W	Type	Example	Remarks
ack-required	✓	Bit string	To offnormal = 1, to fault = 1, to normal = 1	Indicates whether an acknowledgment is required for event transitions. A 1 indicates that acknowledgement is required. Set up at the operator workstation.
description	✓	Character string	Alarm Handler	An editable description of the object's location or function.
notification-class		Unsigned	1	Echoes the object instance.
object-identifier		BACnet_ Object_ Identifier	Notification-class 1	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name	✓	Character string	Alarm Handler 1	
object-type		Enumerated	Notification-class	
recipient-list	✓	List	<List of BACnetDestination>	Lists the devices that will receive notification when the notification class transitions. Set up at the operator workstation.
to-fault-priority		Unsigned		Indicates the priority used for event notifications, 0–255. Lower numbers indicate a higher priority.
to-normal-priority		Unsigned		Indicates the priority used for event notifications, 0–255. Lower numbers indicate a higher priority.
to-offnormal-priority		Unsigned		Indicates the priority used for event notifications, 0–255. Lower numbers indicate a higher priority.

Properties of the program object

Table 35 Properties of a global controller program object

Property	W	Type	Example	Remarks
description	✓	Character string	Occupied Setpoint	A description assigned to describe the object's function.
description-of-halt		Character string	Program halted by request	
instance-of		Character string	alerton hq alerbti 0*00000000 *	Header information for the file. Program 0 does not support this property.
object-identifier		BACnet_ Object_ Identifier	program 1024	This property consists of the Object_Type property and the Object Instance, which is a numeric code that identifies the object of interest.
object-name		Character string	Program Object 1024	
object-type		Enumerated	Program	
out-of-service		Boolean	FALSE	
program-change	✓	Enumerated	READY	Used to command the program state. A program can be stopped using the HALT command, for example, and started again with RESTART.
program-location		Character string	DDC Sequence = 60	Set when program stops.
program-state		Enumerated	RUNNING	Possible states include RUNNING, IDLE, HALTED.
reason-for-halt		Enumerated	PROGRAM	
reliability		Enumerated	NO FAULT DETECTED	
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.

Properties of the schedule object

Table 36 Properties of a global controller schedule object

Property	W	Type	Example	Remarks
description	✓	Character string	Weekend Gym	A description assigned to describe the object's function.
effective-period	✓	Sequence	<BACnet DateRange>	Assigned in schedule setup at the operator workstation.
exception-schedule	✓	Sequence	<Array of BACnet Special Event>	Assigned in schedule setup at the operator workstation.
list-of-object-property-references	✓	List	<List of BACnet Object Property Reference>	The list of objects that this schedule commands.
object-identifier		BACnet_ Object_ Identifier	**schedule** 0	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name	✓	Character string	schedule 000	Assigned in schedule setup at the operator workstation.
object-type		Enumerated	schedule	
present-value	✓		ACTIVE	Indicates the value most recently written to a referenced object property. May be analog, binary, or other, depending on the controlled property.
priority-for-writing	✓	Unsigned	16	Assigned in schedule setup at the operator workstation.
weekly-schedule	✓	Sequence	<Array of BACnetDaily Schedule>	Assigned in schedule setup at the operator workstation.

Properties of the global controller zones objects

Table 37 Properties of the global controller zones objects

Property	W	Type	Example	Remarks
object-identifier		BACnetObject Identifier		Uniquely identifies the object within the BACnet device.
object-name	✓	Character string		Assigned during zone setup at the operator workstation.
object-type		BACnetObjectType		A BACnetObjectType of the value BT_Zone.
aler-reference-device	✓	BACnetObject Identifier		A BACnetObjectIdentifier indicating the primary BACnet device referenced through the zone object.
aler-weekly-schedule-inputs		Array [n] of BACnetPriorityValue		This array works exactly like a priority array and allows up to [n] schedule objects to write weekly schedule commands to the zone.
aler-weekly-schedule-object	✓	Array [n] of BACtalkOptionalDevObjRef		An array that points to the schedule objects used for weekly schedules.
aler-holiday-schedule-input		BACnetPriorityValue		The value last written by the holiday schedule. If Aler_Holiday_Schedule_Object contains a NULL value, this property is also set to NULL.

Table 37 Properties of the global controller zones objects (Continued)

Property	W	Type	Example	Remarks
aler-holiday-schedule-object	✓	BACtalkOptionalDevObjRef		A reference to the schedule object that writes to Aler_Holiday_Schedule_Input.
aler-event-schedule-inputs		Array [n] of BACnetPriorityValue		This array works exactly like a priority array and allows up to [n] schedule objects to write weekly schedule commands to the zone.
aler-event-schedule-objects	✓	Array [n] of BACtalkOptionalDevObjRef		An array that points to the schedule objects used for event schedules.
priority-for-writing	✓	Unsigned	13	Contains the priority at which commands are written to the referenced properties: Day_Night_Command_Reference, Aler_Warmup_Command_Reference, and Aler_Cooldown_Command_Reference. The default is 13; acceptable range is 1-16.
present-value	✓	BACtalkZoneState	Occupied	Alerton Enumerated type that reflects the current state of the zone: Occupied, Unoccupied, Warmup, Cooldown, or Tenant Override.
units	✓	BACnetEngineeringUnits		A value that indicates the unit of measure for all temperatures in the zone object.
aler-persistence-rate	✓	Unsigned	300	A value indicating the number of seconds between refresh times for controlled command points in the zone: occupied, warmup, and cooldown command points.
aler-refresh	✓	BACnetPriorityValue		Writes to this value refreshes all referenced properties. This includes input and output values.
aler-refresh-rate	✓	Unsigned	60	A value indicating when referenced data is old and should be refreshed. If the data is accessed and it hasn't been updated in the time (number of seconds) indicated here, it is immediately refreshed. Values are limited in the range of 10-900 seconds. **NOTE:** Data is refreshed every 15 minutes minimum.
aler-optimum-start-mode	✓	BACtalkOptimumStartMode		An enumerated type indicating the algorithm used by optimum start: None, Manual, or Automatic.
aler-maximum-advance-time		Unsigned	240	A value representing the maximum number of minutes to begin optimum start operations in advance of the next scheduled occupied time. The default value is 240; valid range is 1-1440.
aler-oa-temp-reference		BACtalkDevObjPropRefOrValue		The reference that indicates the outside air temperature to use in optimum start calculations.
aler-oa-temp-value		Real		A real value as read from the aler-oa-temp-reference.
aler-humidity-reference		BACtalkDevObjPropRefOrValue		A value or reference that indicates the humidity value used in optimum start calculations.
aler-humidity-value		BACnetPriorityValue		A value which is read from the aler-humidity-reference.
aler-oa-limit	✓	Real	65.0	A real value used in optimum start calculations for heating operations. The default value is 65.0.
aler-building-mass	✓	Real	4.0	A real value that indicates the amount of desired temperature change. Values are limited between 0-10.
aler-warmup-factor		Real	1.0	A real value used in warmup calculations for a zone. This value is adjusted each time warmup mode is initiated. Valid range is 0-10. A 0 (zero) disables warmup calculations.

Table 37 Properties of the global controller zones objects (Continued)

Property	W	Type	Example	Remarks
aler-cooldown-factor		Real	1.0	A real value used in cooldown calculations for a zone. This value is adjusted each time cooldown mode is initiated. Valid range is 0-10. A 0 (zero) disables cooldown calculations.
aler-alt-warmup-factor		Real		An alternate value used for the warmup factor when the zone was not occupied during the previous 24 hours. Set to NULL to disable alternate warmup.
aler-alt-cooldown-factor		Real		An alternate value used for the cooldown factor when the zone was not occupied during the previous 24 hours. Set to NULL to disable alternate cooldown.
aler-tuning-factor		Real		A real value that determines how aggressive the system should self-tune the warmup and cooldown factors.
aler-cooling-temperature-rate		Real		A real value that indicates the rate (in degrees per hour) that the cooldown mode is expected to change the temperature of the zone when in manual optimum start mode.
aler-heating-temperature-rate		Real		A real value that indicates the rate (in degrees per hour) that the warmup mode is expected to change the temperature of the zone when in manual optimum start mode.
aler-occupied-command-value		BACnetBinaryPV	active	Active is written to this value when the zone is occupied. If the zone is unoccupied or in warmup or cooldown mode, Inactive is written to this value.
aler-occupied-command-reference		BACtalkDevObjPropRefOrValue		
aler-warmup-command-value		BACnetBinaryPV		Active is written to this value when the zone is in warmup mode. Otherwise, Inactive is written.
aler-warmup-command-reference		BACtalkDevObjPropRefOrValue		
aler-cooldown-command-value		BACnetBinaryPV		Active is written to this value when the zone is in cooldown mode. Otherwise, Inactive is written.
aler-cooldown-command-reference		BACtalkDevObjPropRefOrValue		
aler-zone-temp-value		Real		Indicates the current zone temperature. The default is a reference to AV101.
aler-zone-temp-reference		BACtalkDevObjPropRefOrValue		
aler-occupied-heating-setpoint-value		Real		The heating setpoint to use in optimum start calculations.
aler-occupied-heating-setpoint-reference		BACtalkDevObjPropRefOrValue		
aler-occupied-cooling-setpoint-value		Real		The cooling setpoint to use in optimum start calculations.
aler-occupied-cooling-setpoint-reference		BACtalkDevObjPropRefOrValue		
aler-tenant-override-reference		BACtalkDevObjPropRefOrValue		Indicates that the zone is in tenant override mode.
aler-tenant-override-value		BACnetBinaryPV		
aler-tenant-act-recipient		BACnetRecipient		Indicates the device to which tenant override events are sent. A NULL value prevents tenant override events from being sent.

Table 37 Properties of the global controller zones objects (Continued)

Property	W	Type	Example	Remarks
aler-diagnostics		CharacterString		A formatted character string indicating the current status of the zone object and its operation.
description		CharacterString		A string of characters describing the zone.
zone-command-mode		BACtalkZoneCommandMode		Indicates whether it's a binary or multistate object. 0=binary and 1=multistate.
zone-main-truth-table		Array [3] of BACtalkZoneTruthTableEntry		Indicates the value to be written to each of the three main output points for each of the four possible modes of operation. The array elements are mapped as follows. Element 1= Defines the values written to aler-occupied-command-reference. Element 2= Defines the values written to aler-warmup-command-reference. Element 3= Defines the values written to aler-cooldown-command-reference. Even though each entry is the table is a BACnetPriorityValue type, only Unsigned values may be written to the table elements. In all cases, the zone object writes values of the correct data type (unsigned for multistate mode and BinaryPV in all other cases) when sending commands. If a property reference is NONE, then the corresponding array element is ignored. It is good practice to write a zero into the array element in this case.

BACtalk VLCs

This section lists the objects in a BACtalk VLC, followed by a reference to the properties of those objects. Use this list to interpret the source and nature of system data.

Notes

- The **W** column indicates whether the property is writable. Properties without a check mark in this column are read-only. Some items can only be written to through special setup. These are checked as writable and noted under Remarks.
- In the **Example** column, items in boldface always appear as listed for that item. For example, the object-type property of a device object will always return the word “Device” to the Envision for BACtalk display.
- The **Type** column indicates a BACnet data type. Unsigned and Signed indicate integer values; enumerated indicates an enumerated value table; other data types may exist.

Objects in VLCs

Table 38 Objects in BACtalk VLCs

Object	Function	Range of Present Value	Remarks
AI (0-15)	Associated with physical input terminals. Number of inputs varies with VLC model.	0–4095 in counts	Input setup and scaling done in VLC DDC or VisualLogic. AI-1 corresponds to input terminal IN-1, and so on.
AO (0-7)	Physical analog output.	0–100	AO present-value property is the result of the priority array.
AV (0-107)	Real numbers not directly associated with physical input terminals. Typically used for setpoints and intermediate calculations.	$\pm 3 \times 10^{38}$	Six significant digits of resolution. All AVs are used in RAM and backed up in Flash memory on loss of power. **IMPORTANT NOTE:** In C3 VLCs, AVs 0–49 stored in VLC RAM, 50–89 stored in VLC EEPROM, 90–106 reserved for Microset use.
BI (0-15)	Associated with physical input terminals. Number of inputs varies with VLC model.	ACTIVE or INACTIVE	BIs correspond to the same input terminals as AIs. BI-1 corresponds to input terminal IN-1, and so ON. The BI turns ON when the associated AI drops to a value in counts less than or equal to 448. It reads OFF when the AI rises to a value in counts greater than or equal to 512. The BI does not change state while the value is in the range 449–512.
BO (0-15)	Physical binary output.	ACTIVE or INACTIVE	BO present-value property is the result of the priority array.
BV (0–84)	Binary value.	ACTIVE or INACTIVE	BV 0-63 are for general use. BV 64-84 are reserved for Microset control.
Device	Provides general information about device.	N/A	
File 0	Provides information about the DDC program file.	N/A	
File 1	Provides information about the DDC trap file.	N/A	
Program 0	Provides information about DDC program execution.	N/A	

Properties of the AI object

Table 39 Properties of the VLC AI object

Property	W	Type	Example	Remarks
description	✓	Character string	Return Air Temp	An editable description of the object's location or function.
event-state		Enumerated		Normal.
object-identifier		BACnet_ Object_ Identifier	**AI** 5	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**AI** 05	
object-type		Enumerated	AI	Indicates an analog input (AI) object.
out-of-service		Boolean	FALSE	

Table 39 Properties of the VLC AI object (Continued)

Property	W	Type	Example	Remarks
present-value	✓	Real	72.3	Range of present-value goes from 0–4000 as input voltage goes from 0–5.12VDC.
reliability		BACnet_ Reliability	NO FAULT DETECTED	Other possibilities are OVER RANGE, UNDER RANGE, UNRELIABLE_OTHER.
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.
units		Enumerated	Deg F	Indicates the unit of measure, in BACnet engineering units, that the AI is expressed in.

Properties of the AO object

Table 40 Properties of the VLC AO object

Property	W	Type	Example	Remarks
description	✓	Character string	Economizer Damper	An editable description of the object's location or function.
event-state		Enumerated	NORMAL	
object-identifier		BACnet_ Object_ Identifier	**AO** 1	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**AO** 01	
object-type		Enumerated	AO	
out-of-service		Boolean	FALSE	Options are TRUE or FALSE. If TRUE, the physical output is de coupled from the AO and its present-value, and internal DDC execution alone determines the status of the physical output.
present-value	✓	REAL	75	Valid values are real numbers in the range 0–100. Values greater than 100 are interpreted as 100. When commanded from a data display, values are actually written to the priority array (see priority-array entry herein) and read back from the present-value for display.
priority-array		BACnet PriorityArray	<Array of BACnet Priority Value>	A read-only array of prioritized values (1-16) controlling the present-value, priority 1 having the highest priority. The value with the highest priority controls the present value. Possible values for priority array indexes are real values or NULL. A NULL value indicates no command is issued at that priority level.
reliability		BACnet_ Reliability	NO FAULT DETECTED	Other possibilities are OVER RANGE, UNDER RANGE, UNRELIABLE_OTHER.
relinquish-default	✓	REAL	0	Default value to be used for present-value property when all priority-array indexes are NULL. Set up in DDC or VisualLogic.
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.
units	✓	Enumerated	%	Indicates the unit of measure, in BACnet engineering units, that the AO is expressed in. Set in VLC DDC or in VisualLogic.

Properties of the AV object

Table 41 Properties of the VLC AV object

Property	W	Type	Example	Remarks
description	✓	Character string	Static Pressure Setpoint	An editable description of the object's location or function.
event-state		Enumerated	NORMAL	
object-identifier		BACnet_Object_ Identifier	**AV** 15	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**AV** 15	
object-type		Enumerated	AV	Indicates an analog value (AV) object.
out-of-service		Boolean	FALSE	
present-value	✓	Real	0.02	Real number in the range $\pm$ 3 x 10^{38} with six significant digits of resolution.
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.
units	✓	Enumerated	%	Indicates the unit of measure, in BACnet engineering units, that the AV is expressed in. Set in VLC DDC or in VisualLogic.

Properties of the BI object

Table 42 Properties of the VLC BI object

Property	W	Type	Example	Remarks
description	✓	Character string	Fan Status	An editable description of the object's location or function.
event-state		Enumerated	NORMAL	
object-identifier		BACnet_ Object_ Identifier	**BI** 10	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**BI** 10	
object-type		Enumerated	BI	Indicates a binary input (BI) object.
out-of-service		Boolean	FALSE	
polarity			NORMAL	
present-value	✓	Logical state	ACTIVE	ACTIVE or INACTIVE.
reliability			NO FAULT DETECTED	Not currently used in VLCs.
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.

Properties of the BO object

Table 43 Properties of the VLC BO object

Property	W	Type	Example	Remarks
description	✓	Character string	Fan Start/ Stop	An editable description of the object's location or function.
event-state		Enumerated	Normal	If the object does not support intrinsic reporting, the value shall be NORMAL.
object-identifier		BACnet_ Object_ Identifier	**BO** 1	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**BO** 01	
object-type		Enumerated	BO	
out-of-service		Boolean	FALSE	Options are TRUE or FALSE. If TRUE, the physical output is de coupled from the BO and its present-value, and internal DDC execution alone determines the status of the physical output.
present-value	✓	Enumerated	INACTIVE	Either ACTIVE or INACTIVE. Note that a NULL value can be written to the present-value on data displays, but the value is actually written to a priority-array property. The present-value is the result of the priority-array.
priority-array		BACnet PriorityArray	<Array of BACnet Priority Value>	A read-only array of prioritized values (1-16) controlling the present-value, priority 1 having the highest priority. The value with the highest priority controls the present value. Possible values for priority-array indexes are ACTIVE, INACTIVE, or NULL. A NULL value indicates no command is issued at that priority level.
reliability		BACnet_ Reliability	NO FAULT DETECTED	Other possibilities are OVER RANGE, UNDER RANGE, UNRELIABLE_OTHER.
relinquish-default	✓	Enumerated	INACTIVE	Default value used for present-value property when all priority-array values are NULL. Set up in DDC or VisualLogic.
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.

Properties of the BV object

Table 44 Properties of the VLC BV object

Property	W	Type	Example	Remarks
description	✓	Character string	Test Mode	An editable description of the object's location or function.
event-state		Enumerated	Normal	
object-identifier		BACnet_ Object_ Identifier	**BV** 20	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	**BV** 20	

Table 44 Properties of the VLC BV object (Continued)

Property	W	Type	Example	Remarks
object-type		Enumerated	BV	
out-of-service		Boolean	FALSE	
present-value	✓	Enumerated	INACTIVE	Either ACTIVE or INACTIVE.
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.

Properties of the file object

Table 45 Properties of the VLC file object

Property	W	Type	Example	Remarks
archive	✓	Boolean	FALSE	Indicates whether some sort of backup file has been created (TRUE) or not (FALSE). Set manually.
description		Character string	Generic Generic v0011	Data from program information in the DDC header.
file-access-method		Enumerated	stream access	
file-size		Unsigned	6885	Size of the file, in bytes.
file-type		Character string	VLC DDC Program File	.
modification-date		Time	Wednesday, 5/29/1997 10:22:20:00 a	Date and time that the DDC file was downloaded to the VLC.
object-identifier		BACnet_ Object_ Identifier	file 0	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	VLC DDC Program File	
object-type		Enumerated	file	
read-only		Boolean	TRUE	

Properties of the device object

Table 46 Properties of the VLC device object

Property	W	Type	Example	Remarks
APDU-timeout		Unsigned	0	The time after transmission of an APDU until the lack of a reply means it was assumed to be lost. The APDU timeout value for this device in milliseconds (1000 = 1 sec).
application-software-version		Character string	BTI Controller V3.20	Indicates the ROC version.
description	✓	Character string	Mechanical Room BTI	An editable description that identifies the device's location or function.
device-address-binding		List		Inaccessible.
firmware-revision		Character string	BACtalk BTI v3.2 11/17/2002	Indicates the firmware version.
local-date	✓	Octet String	Wednesday, 5/14/ 2002	Indicates date: day of the week, month/day/year.
local-time	✓	Time	10:15:56.00am	Indicates the time stored in the device.
location	✓	Character string	East Wing	Indicates the physical location of the device.
max-APDU-length-accepted		Unsigned	206	The maximum message packet size that the device can handle.
max-info-frames		Unsigned	1	Number of MS/TP messages the VLC will send per token hold.
max-master		Unsigned	127	The highest MS/TP MAC address the VLC will attempt to pass the token to.
model-name		Character string	VLC Controller	Assigned by the vendor to indicate the device model.
number-of-APDU-retries		Unsigned	0	The number of times a message will be resent after it is assumed to be lost.
object-identifier		BACnet_ Object_ Identifier	**Device** 200	This property consists of the object-type property and the device instance of the device of interest.
object-list		Array		An array whose elements list the object-identifier properties of all objects the device supports.
object-name	✓	Character string	**Device** 200	No two devices are permitted to have the same object name.
object-type		Enumerated	Device	
protocol-conformance-class		Unsigned	3	Integer from 1–6 indicating the conformance class of the device. A device must support a standardized set of services and object types to claim a particular class conformance.
protocol-object-types-supported		Bit string	<Bit string>	A bit string that indicates the BACnet object types that reside in the device. A 1 indicates the device is present.
protocol-services-supported		Bit string	<Bit string>	A bit string that indicates the BACnet services the device can process. A 1 indicates that service is supported.
protocol-version		Unsigned	1	Indicates the version of the BACnet protocol supported by the device.

Table 46 Properties of the VLC device object (Continued)

Property	W	Type	Example	Remarks
segmentation-supported		Enumerated	No segmentation	VLCs do not support segmentation.
system-status		Enumerated	Operational	Other possible values are operational-read-only, download-required, download-in-progress, non-operational.
vendor-identifier		Unsigned	18	A unique code assigned by ASHRAE to the manufacturer.
vendor-name		Character string	Alerton	Indicates the device manufacturer.

Properties of the program object

Table 47 Properties of the VLC program object

Property	W	Type	Example	Remarks
description		Character string	Generic Generic V0011 P8223001	Data from program information in DDC or in VisualLogic. Format is <rep> <job> <program name> <revision> <displaynum>.
instance-of		Character string	Generic Generic V0011 P8223001	Echoes the description property.
object-identifier		BACnet_ Object_ Identifier	**program** 0	This property consists of the object-type property and the object instance, which is a numeric code that identifies the object of interest.
object-name		Character string	VLC DDC program	
object-type		Enumerated	program	
out-of-service		Boolean	FALSE	
program-change	✓	Enumerated	READY	One can use this property to manipulate the Program State property. Doing this is not recommended.
program-state		Enumerated	RUNNING	Possible states include RUNNING, IDLE, HALTED.
reliability		BACnet_ Reliability	no fault detected	Other possibilities are OVER RANGE, UNDER RANGE, UNRELIABLE_OTHER.
status-flags		Bit string	In alarm = 0, fault = 0, overridden = 0, out of service = 0	A four-position bit string that indicates the status of the object. If a status bit =1, that status is TRUE.

Scaling factors

8

Input scaling refers to the conversion of an electrical signal from an input device to some useful range, whatever that range is for your application: 0-100 %RH, 50-100 deg. F or 0-10 deg. C, -0.1 to 0.1 inches WC for static air pressure, and so on.

There are three key elements at work during this conversion: the electrical signal, the raw counts, and the scaled input value.

Analog input signals to a VLC are fed into an analog-to-digital (A/D) converter in the VLC. The A/D converts this into a binary number, which is the foundation for the raw counts. This is necessary so the microprocessor and other components on the VLC can interpret the signal. Software in the VLC then converts the raw counts to a usable range, what you see in Envision for BACtalk as the present-value of an analog input (AI).

When setting up inputs from the Analog Input Setup tab of the Device Settings dialog box in VisualLogic, choose "Scaled (Two Point)" from the Type drop-down menu. The Two Point Scale Setup dialog box opens, enabling you to enter two mA or voltage values along with the desired AI values.

Select "5.12V w/jumper or switch" for all Gen4 VLCs that include an input jumper or switch and select "5.12V no jumper or switch" for all Gen4 VLCs that do not have an input jumper or switch. The input scaling utility automatically calculates the correct values for zero and range.

The following Gen4 VLCs do not include an input jumper or switch:

VAV-SD	VAVi-SD
VAV-DD	VAV-DD7
VAV-SD2A	VLC-660R
VLC-651R	VLC-16160

The following Gen4 VLCs have an input jumper or switch:

VLC-1188	VLC-853
VLC-550	VLC-1600

DDC header file setups in VLC DDC 9

Each VLC has a header file with information about the VLC's configuration and addressing. The header is part of the DDC program.

Whenever a DDC sequence is saved, downloaded, or read from a VLC, information in the DDC header is packaged with it. The DDC header contains information about DDC files on the operator workstation, as well as AI, AV, AO, BO, and Microset Field Service mode configurations. You set up this information through the DDC Header menu in VLC DDC.

▶ **To view the DDC Header menu**

1. Make sure the device instance of the VLC you want to work with is specified. Also confirm that you have read the DDC from the VLC so you do not inadvertently overwrite an existing setup.

2. From the VLC DDC Main menu, choose F2 DDC, and then choose F2 Edit DDC Header.

Program Information screen

Note Data in the Program Information screen also appears and can be edited in the Disk Files Screen.

The Program Information screen (F1 Program Information from the DDC Header menu) gives you data about the currently viewed DDC file. This file information helps you identify the origin of the DDC file for troubleshooting. Use the mouse to position the cursor in the field you want to change, press Enter, and then type the settings you want according to the following guidelines.

Representative and Job The representative and job name under which the file was saved on the operator workstation hard disk. You must have security to modify, save, or send DDC files with a given representative and job name.

Program The file name (minus the file extension) of the DDC program when it was saved to disk.

Revision A revision number for the DDC program, which is typically important only for Alerton Standard DDC files. This revision number can also be used by any DDC author to manage versions of custom DDC.

Display The display number associated with this DDC file.

AI setup

The Analog Input Setup screen (F2 AI from the DDC Header menu) is where you perform scaling options for AIs. Use this screen to designate the type of scaling you want to use for an input or the type of thermistor connected to the input.

Universal inputs on VLCs can be configured to accept a variety of input types. You perform this configuration in the AI Setup screen of VLC DDC. Depending on the application and VLC, the setup for an input will probably include the configuration of DIP switches or jumpers on the VLC itself.

These setup factors affect the software *count* that results from a given electrical input.

Setting the AI type and scaling factors In the Type field of the AI Setup screen, you can select one of the following:

- **Counts** The raw count is used (infinitely great resistance, an open, results in a count of 4095; no resistance, a short, results in a count of 0).
- **Scaled** The raw input count is affected by the value of zero and range as indicated on-screen.

$$ScaledCount = Zero + \frac{RawCount \times Range}{(4096)}$$

- **10K Thermistor or 3K Thermistor** Raw counts are automatically converted to temperature.

Note See the *VLC Installation and Operations Guide (LTBT-TM-GEN4VLC)* for more detailed information and input scaling tables. Guidelines for setting up pulse-type inputs is also provided.

Keep the following in mind:

- The zero and range options apply only to scaled input types. No additional scaling is required for 3K ohm and 10k ohm thermistor AIs.
- For any AI set up as a thermistor, the input will be configured automatically to degrees F or degrees C as appropriate for the selection of English or Metric units.
- Use F10 to view setup parameters for additional inputs.
- The scaling options you select are saved with the DDC as part of the DDC header.

Scaling example 1 A 0-10VDC pressure transducer with a range of 0"-4" water column (WC) is used to measure duct static pressure. The value for range is calculated as the value of AI with an input of 10VDC (input count = 4095) minus the value of AI with an input of 0VDC (input count =0). Range = 4". The value for zero is the value of AI when the input is 0. Zero = 0".

Scaling Example 2 A 4-20mA pressure transducer with a range of –0.25" to 0.25" WC is used to measure building pressure. The range is calculated as the variation in AI as the input count goes from 800-4000 or 880-4000 depending on the VLC type. See the *VLCs Installation and Operations Guide* (LTBT-TM-GEN4VLC) for more information.

In this example, however, the transducer output only goes down to 4mA, not 0mA. The challenge is to figure out what the pressure range would be if the sensor output went all the way from 0mA to 20mA. Since the 16mA change from 4mA to 20mA corresponds to pressure range of 0.5" (0.25" to -0.25" =0.5"), a 20mA would theoretically correspond to a pressure range of 0.625" (20/16 x 0.5). The value for range in this example is then equal to 0.625". To calculate and enter values for range, AI, and input at 20mA, 0.25" = zero + (4000 x 0.625)/4096. Zero is then equal to -.375".

Using input filters VLCs use a 10-bit A/D converter, which is very responsive to changes in electrical signals. This makes the inputs of the VLCs extremely sensitive to minute changes, which may or may not be desirable.

You can turn filtering ON and OFF by placing the cursor in the Filter field for an AI and pressing Enter to toggle the value.

The filter calculation is performed every 0.1 seconds and is expressed as:

$$FilteredCount = \frac{NewCount}{32} + \frac{31}{32}\langle PreviousCount\rangle$$

CAUTION Do not filter any Microset/Microtouch inputs.

English/Metric Mode setup VLCs support English or metric units. Once set up from the AI Setup screen in the DDC header, the VLC makes all of its calculations and writes to the Envision for BACtalk displays and Microset display.

Having the unit of measure specified in the DDC header enables DDC programs to individually accommodate different measurement systems.

▶ **To toggle between English and metric mode**

1. Choose F2 AI from the DDC Header menu.
2. Position the cursor in the English/Metric Mode field.
3. Press Enter to change the mode.

Note For VAV controllers, the setting of the English/Metric Mode field determines whether airflow is reported in cfm or lps.

Units setup Units do not affect calculations. They are included for reference only, and to populate the Units property for the AI. Position the cursor in the field, press Enter, and type the numerical ID of the units you want to use. Before you press Enter, press F4 to view a description of the current units. Press F5 to view a list of all units and corresponding IDs.

AV setup

In the AV Setup screen (F3 AV from the DDC Header menu), you determine the units property of all the AV objects present in the VLC. The units property has no effect on the present-value of the AV. For most applications, it is unnecessary to set up units unless you use the property on a display.

Note You can also press Enter in the Units field and type the engineering unit code directly.

▶ **To set up the units property of an AV**

1. From the VLC DDC main menu, choose F2 DDC, and then choose F2 Edit DDC Header from the DDC menu.
2. Choose F3-AV-Analog Values.

   A list of AVs appears, each AV with a corresponding Units field. In the Units field, the unit ID is listed.
3. Press F9 or F10 to view additional AVs.
4. Position the cursor in the Units field for the AV you want to configure.
5. Press F5 to view a list of engineering unit codes and descriptions.
6. Press F9 or F10 to view additional unit codes and descriptions.
7. Position the mouse pointer on the unit code you want to use and click once.

   On the AV Setup screen, the unit type you selected appears in the Units field.
8. Repeat steps 4 thru 7 for each AV you want to configure.

BO setup

In the BO Setup screen (F4 BO from the DDC Header menu), you configure two important features for each BO—the relinquish-default and the out-of-service property. The relinquish-default and out-of-service properties relate to the priority-array.

Relinquish default The relinquish-default is the value sent to the BO when commands for every level of its priority-array are NULL—essentially, its normal state when no command has been issued for the BO.

Options are ACTIVE (ON) and INACTIVE (OFF). Position the cursor in this field for the BO you want to configure, and then press Enter to toggle between ACTIVE and INACTIVE.

CAUTION Present–value will continue to represent the result of the priority-array index when out-of-service is set to true.

Out-of-service The out-of-service flag determines whether the BO can be commanded from an external device. If out-of-service is TRUE, only the device's internal DDC controls the status of the BO, and the BO is decoupled from its priority-array. If out-of-service is FALSE, the BO can be commanded from throughout the BACnet system and its status is determined by the priority-

array. Position the cursor in the Out of Service field for the BO you want to configure, and then press Enter to toggle between TRUE and FALSE.

AO setup

The AO Setup screen (F5 AO from the DDC Header menu) allows you to configure the units, out-of-service property, and relinquish-default for AOs. The relinquish-default and out-of-service properties relate to the priority-array.

Note Set the units property of the AO as you do for AVs. See “AV setup” on page 154 for details.

Relinquish-default and out-of-service These have the same function as for BOs. However, for the relinquish-default, you specify a number from 0–100.

Microset Field Service mode custom codes

VLCs enable you to set up custom field service codes for the Microset. Using this feature, you can display and change the value of AI (read-only), BI (read-only), AV, BV, AO, and BO objects from the BACtalk Microset. You can set up to 26 field service mode codes.

In Field Service mode, a series of two-character codes correspond to each object. You select the objects to be displayed and choose the codes that identify them using the Microset Field Service Mode Setup screen (F6 Microset Field Service Mode Control Table from the DDC Header menu).

Entry Identifies the entry and can’t be edited. In Field Service mode, entry 0 appears first, entry 25 appears last.

Display Code The two-digit code to be displayed at the Microset. Limited to acceptable characters and must be two characters. Press F5 to view the list of acceptable characters or use the following reference. Position the cursor in this field, press Enter, and then type acceptable characters.

Table 48 Acceptable Microset Field Service mode characters

0	1	2
3	4	5
6	7	8
9[a]	A	b
c	C	d
E	F	g
h	H	i
J	L	n
o	p	r
u	U	y
-	_	

a.The characters 9 and g are virtually indistinguishable on the Microset display.

Point ID Defines the VLC data point whose value will be associated with the display code. AI, BI, AO, BO, AV, and BV are acceptable point types. Position the cursor in this field, press Enter, and then type an acceptable point type. Press Enter again and type the point’s instance (for AI 1, type 1, for example). Make sure the point you reference actually exists in the VLC.

Writable Determines whether the value will be writable from the Microset (YES) or read only (NO). Position the cursor in the field and press Enter to toggle the value. AIs and BIs are not writable.

Decimal Determines whether the Microset will show the value with a decimal (YES) or without one (NO). Position the cursor in this field and press Enter to toggle the value.

CAUTION The BACtalk Microset displays OFF values as 0 and ON values as .1. You must set the Decimal field to YES to view or change binary values at the Microset.

Positive only Determines whether only positive values can be displayed (YES) or negative values as well (NO). Position the cursor in this field and press Enter to toggle the value.

Setting parameters for a VAV airflow sensor

VAV controllers have a special screen in the VLC DDC program for setting up airflow sensor parameters. In this configuration screen, you specify a box size, a calibration factor, the point below which airflow should read zero (zero cutoff), and the values of AI-8 and AI-10 (for dual duct VLCs), or AI-10 (for single duct VLCs).

Note You can also adjust the values for the hot duct and cold duct calibration factors from the BACtalk Microset using the Microset Airflow Calibration mode.

This is a one-time setup you perform for each VAV VLC; the setup is stored in the VLC independently of DDC or header information. This configuration affects the values that represent airflow for VAV VLCs.

Note The program units setting (English or metric) in the Analog Input Setup screen (see "AI setup" on page 152) determines whether AIs report airflow in cfm or lps, and whether temperatures are reported in degrees F or C.

▶ **To set up airflow parameters for a VAV VLC**

1. Make sure you select the Device ID of the VLC you want to work with in the Configuration screen. Also, choose F2 Read File to retrieve the existing setup data from the VLC so you do not inadvertently overwrite an existing setup.
2. From the VLC DDC Main menu, choose F1 Configure.
3. From the Configuration menu, choose F3 VAV Flow Sensor.

   The VAV Flow Sensor Setup screen appears with fields for entering a box size, calibration factor, and velocity zero cutoff for both the cold duct and the hot duct. For single-duct VAV controllers (SD), only the cold duct information is used.
4. Position the cursor in the field you want to edit, press Enter, and then set parameters according to the following guidelines.

Box diameter The box diameter of the VAV box to be controlled by this VLC. Enter this value in inches or centimeters as appropriate for English or metric setup in the AI Setup screen (see “AI setup” on page 152). Obtain box diameter data from the VAV box manufacturer. The box diameter represents the round duct equivalent diameter of the duct where the pressure pickup is located.

Calibration factor Adjusts the VLC airflow readings to match the specific VAV box, airflow pickup, and conditions of an installation. Increasing the value of the calibration factor will increase the value of the corresponding airflow (AI-8 for hot duct flow, if applicable; AI-10 for cold duct flow).

The calibration factor for each duct should be adjusted until the corresponding airflow matches the airflow measured using a balancing hood. The default value for the calibration factor, 1.00, accounts for no pressure loss in the tubes. Increase the calibration factor to compensate for tube loss.

Velocity zero cutoff This value is entered in percent of full airflow (0-100). Airflows lower than the cutoff are reported as zero. This feature eliminates non-zero airflow readings due to ambient fluctuations when the main fan is off.

▶ **To send data to the VLC**

- When you finish, press F1 to send the VAV airflow setup information to the VLC.

