Driver Manual

FS-8700-73 BACnet MS/TP

APPLICABILITY & EFFECTIVITY
Effective for all systems manufactured after February 2021.

Driver Revision: 2.21
Document Revision: 5.A

MSA Safety
1991 Tarob Court
Milpitas, CA 95035
Website: www.MSAsafety.com

U.S. Support Information:
+1 408 964-4443
+1 800 727-4377
Email: smc-support@msasafety.com

EMEA Support Information:
+31 33 808 0590
Email: smc-support.emea@msasafety.com

Contents

1 Description 5
1.1 BTL Mark – BACnet Testing Laboratory 5

2 Driver Scope of Supply 6
2.1 Supplied by MSA Safety 6
2.2 Provided by the Supplier of 3rd Party Equipment 6
2.3 Optional Items 6

3 Hardware Connections 7

4 Data Array Parameters 8

5 Client Side Configuration 9
5.1 Client Side Connection Parameters 9
5.2 Client Side Node Parameters 10
5.3 Client Side Map Descriptor Parameters 11
5.3.1 FieldServer Specific Map Descriptor Parameters 11
5.3.2 Driver Related Map Descriptor Parameters 11
5.3.3 Timing Parameters 13
5.4 Map Descriptor Examples 13
5.4.1 COV Specific 13

6 Configuring the FieldServer as a Server 14
6.1 Driver Specific FieldServer Parameters 14
6.2 Server Side Connection Parameters 15
6.3 Server Side Node Parameters 16
6.4 Server Side Map Descriptor Parameters 17
6.4.1 FieldServer Specific Map Descriptor Parameters 17
6.4.2 Driver Related Map Descriptor Parameters 17
6.5 Map Descriptor Examples 20

7 Useful Features 21
7.1 BACnet Properties 21
7.1.1 BACnet Object Names 21
7.1.2 Virtual Network Number 22
7.1.3 Accessing Data from BACnet Properties Comprising Arrays of Values 23
7.1.4 FieldServer Implementation of BACnet Priority Arrays 23
7.1.5 Relinquishing Control of a Point as a Client 24
7.1.6 BACnet State Text Preload 25
7.1.7 Factors Determining the Reliability Property 26
7.1.8 Update Property Function 26
7.1.9 BACnet State Text Preload 27
7.1.10 Factors Determining the Reliability Property 28
7.1.11 Update Property Function 28
7.1.12 Using a .ini File to Set Vendor_ID and Vendor_Name of the Device Object 29
7.1.13 Srv_Offline_Method Legal Values 29
7.2 BACnet Services 30
7.2.1 COV and Intrinsic Reporting 30
7.2.2 Specify Read/Write PropertyMultiple Transactions with Linked Map Descriptors 34
7.2.3 Disabling Selected BACnet Services 36
7.3 Virtual Router Configuration – Connect a Device to the Local BACnet Segment 37
7.4 Trending Using Trend Log Objects 37
7.4.1 Operating Statistics 38
7.4.2 Date and Time Synchronization 39
7.5 Writing to Custom Properties on Remote BACnet Devices 39
7.6 Intrinsic Reporting for Life Safety Point 40

8 Troubleshooting 41
8.1 Debugging a BACnet Connection 41
8.2 COV Configuration 41
8.3 BACnet Specific Statistics 41
8.4 BACnet Specific Error Messages 42
8.5 BACnet Error Response Decoding 42
8.5.1 BACnet Error Class 42
8.5.2 BACnet Error Codes for Error Class Object 43
8.6 Rediscovering Offline Devices 44
8.7 Timeout Errors 44
8.8 PDU Limit Reached Errors 44

9 Vendor Information 45
9.1 McQuay 45
9.2 Liebert 45
9.3 Automated Logic Corporation 45
9.4 Honeywell EBI 45
9.5 Using Cimetrics Explorer 45

10 Reference 47
10.1 FieldServer Vendor ID 47
10.2 Object_Type Legal Values – Abbreviation Descriptions 47
10.3 Configuring Binary Outputs 47
10.4 Property Legal Values 48
10.5 Supported BACnet Object Properties 51
10.6 Units 54

1 Description

The BACnet Master-Slave/Token-Passing (MS/TP) driver implements a data link protocol that uses the services of the RS-485 physical layer. See the FieldServer BACnet PIC statement for the level of conformance that this driver implements.

All information in a BACnet system is represented in terms of objects. The Object_Identifier is a 32-bit code that identifies the type of Object (also identified by the Object_Type Property) and its "Instance" number, which together uniquely identify the Object within its BACnet device. Theoretically, a BACnet device could have over four million Objects of a particular type. The Object_Name is a text string, which has a unique capability. BACnet devices may broadcast queries for devices that contain Objects with a specific Object_Name. This can greatly simplify project setup.

BACnet requires one Device Object to be present in every BACnet device. The Device Object makes information about the device and its capabilities available to other devices on the networks. Before one BACnet device starts control-related communications with another, it needs to obtain some of the information presented by the other device's Device Object. Unlike other Objects, the Device Object's Instance number must be unique across the entire BACnet internetwork because it is used to uniquely identify the BACnet devices. It may be used to conveniently identify the BACnet device from other devices during installation.

Standard object types are used to hold real time data and other information. Each Object Type is referenced by a number, for example 0 represents an Analog Input. See **Section 10.2** for abbreviation list.

Each Object consists of a number of prescribed properties, the main property being the Present_Value. Objects are monitored and controlled through their properties.

The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer.

1.1 BTL Mark – BACnet Testing Laboratory

The BTL Mark on ProtoNode is a symbol that indicates that a product has passed a series of rigorous tests conducted by an independent laboratory which verifies that the product correctly implements the BACnet features claimed in the listing. The mark is a symbol of a high-quality BACnet product.

Go to www.BACnetInternational.net for more information about the BACnet Testing Laboratory. Click here for the BACnet PIC Statement.

Max Nodes Supported

FieldServer Mode	Nodes	Comments
Client	1	
Server	32	

2 Driver Scope of Supply

2.1 Supplied by MSA Safety

Part #	Description
	Driver Manual

2.2 Provided by the Supplier of 3rd Party Equipment

Part #	Description
	BACnet Controller(s) on RS-485 Network

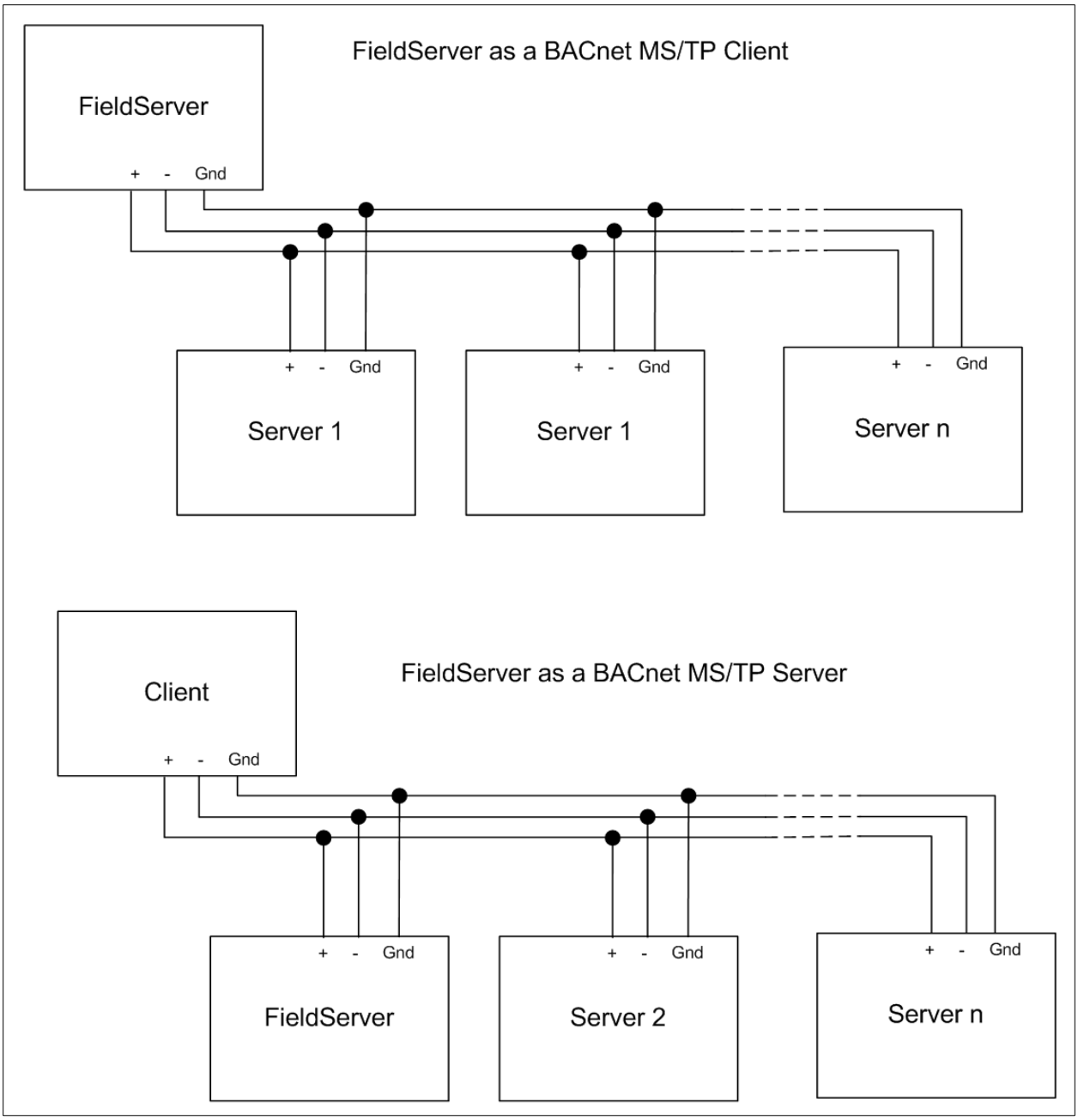
2.3 Optional Items

Part #	Vendor/Manufacturer	Description
FS-4301-00	MSA Safety	RS-232 to RS-485 converter

3 Hardware Connections

Configure the BACnet MS/TP according to manufacturer's instructions.

The FieldServer is connected to the BACnet MS/TP as shown in connection drawing below.



4 Data Array Parameters

Data Arrays are “protocol neutral” data buffers for storage of data to be passed between protocols. It is necessary to declare the data format of each of the Data Arrays to facilitate correct storage of the relevant data.

Section Title		
Data_Arrays		
Column Title	**Function**	**Legal Values**
Data_Array_Name	Provide name for Data Array.	Up to 15 alphanumeric characters
Data_Array_Format	Provide data format. Each Data Array can only take on one format.	Float, Bit, Uint16, Uint32, Sint16, Sint32, Byte
Data_Array_Length	Number of Data Objects. Must be larger than the data storage area required by the Map Descriptors for the data being placed in this array.	1-10000

Example

```
//    Data Arrays
Data_Arrays
Data_Array_Name    , Data_Array_Format    , Data_Array_Length
DA_AI_01           , UInt16,              , 200
DA_AO_01           , UInt16               , 200
DA_DI_01           , Bit                  , 200
DA_DO_01           , Bit                  , 200
```

5 Client Side Configuration

For detailed information on FieldServer configuration, refer to the FieldServer Configuration Manual. The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer (see ".csv" sample files provided with the FieldServer).

This section documents and describes the parameters necessary for configuring the FieldServer to communicate with a BACnet MS/TP Server.

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for BACnet MS/TP communications the following three actions must be taken. The driver independent FieldServer buffers need to be declared in the "Data Arrays" section. The destination device addresses need to be declared in the "Client Side Nodes" section. And the data required from the server(s) needs to be mapped in the "Client Side Map Descriptors" section. Details on how to perform these steps can be found in the following sections.

NOTE: In the following tables, * indicates an optional parameter and bold legal values are default.

5.1 Client Side Connection Parameters

Section Title		
Connections		
Column Title	**Function**	**Legal Values**
Port	Specify which port the device is connected to the FieldServer.	P1-P2[1], R1-R2[2]
Baud	Specify baud rate.	**9600**, 19200, 38400, 76800
Parity*	Specify parity.	Odd, Even, **None**
Data_Bits*	Specify data bits.	7**, 8**
Stop_Bits*	Specify stop bits.	**1**, 2
Protocol	Specify protocol used.	BACnet_MSTP
Poll Delay*	Time between internal polls.	0-32000s, **1s**
Max_Master*	Specify the highest MAC_Address for Master nodes. When designing a BACnet MS/TP network with multiple Master_Node devices, allocate the MAC_Address settings on the different devices starting at 1. This will allow specification of a small Max_Master parameter and hence will improve overall network performance.	1-**127**
Connection_Type*	Specify if the connection must be in Master_Mode or in Slave_Mode.	**MSTP_Master_Mode** MSTP_Slave_Mode
MAC_Address[3]	Specify the MS/TP MAC Address for this connection.	0-127 for MSTP_Master_Mode; 0-254 for MSTP_Slave_Mode

[1] P1 to P2 will require the optional RS-485 converter (Part # 4301-00).
[2] Not all ports shown are necessarily supported by the hardware. Consult the appropriate Instruction manual for details of the ports available on specific hardware.
[3] Previously the MAC_Address was set using the System_Node_ID parameter in the Common Information section.

Max_Info_Frames	Specify the maximum number of information frames the connection may send before it must pass the token.	1-65535
Max_Concurrent_Messages*	Specify at most how many messages driver can send out, before start waiting for response.	1 - 65534, **8** (1 means no concurrent messaging, i.e. single active message at any time)[4]

Example:

```
//    Client Side Connections

Connections
Port    , Baud     , Parity   , Data_Bits  , Stop_Bits  , Protocol      , Poll_Delay  , Timeout
R1      , 38400    , None     , 8          , 1          , BACnet_MSTP   , 0.100s      , 30s
```

5.2 Client Side Node Parameters

Section Title
Nodes

Column Title	Function	Legal Values
Node_Name	Provide name for Node.	Up to 32 alphanumeric characters
Node_ID	Specify the BACnet Device object instance.	0 – 4194303
Protocol	Specify protocol used.	BACnet_MSTP
Port	Specify port.	P1-P25, R1-R26
MAC_Address*[7]	Specify the BACnet MS/TP MAC address for this connection. Configuring the remote MAC address directly replaces the default operation (dynamic device binding) using Who-Is broadcasts. This is useful in situations where either the network or the target device do not support dynamic device binding (e.g. when talking to MS/TP slave devices). Note that the Device Instance / Node_ID should not be specified when the MAC Address is configured directly.[8]	0-127 for MSTP_Master_Mode; 0-254 for MSTP_Slave_Mode; **-**
Network_Number*	Use this parameter if the remote BACnet MS/TP server device is on another BACnet network connected via a router.	1-65536 bit positive numbers, **5**
Retries*	Number of timeouts before the Node goes offline.	Any positive integer, **3**

[4] Using Max_Concurrent_Messages value > 1 could improve communication performance depending upon server implementation. It is possible that the Server does not support multiple messaging. Try to match this number with the Server's capability.
[5] P1 to P2 will require the optional RS-485 converter (Part # 4301-00).
[6] Not all ports shown are necessarily supported by the hardware. Consult the appropriate Instruction manual for details of the ports available on specific hardware.
[7] Previously the MAC_Address was set using the System_Node_ID parameter in the Common Information section.
[8] Either Node_ID or MAC_Adddress must be specified. If the Server Node supports Who-Is and I-Am then it is better to use the Node_ID parameter.

APDU_Timeout*	The time in milliseconds between retransmissions of an APDU requiring acknowledgement for which no acknowledgment has been received.	**10000ms**[9]
APDU_Retries*	The maximum number of times that an APDU shall be retransmitted.	**3**[9]

Example

```
//    Client Side Nodes

Nodes
Node_Name    , Node_ID    , Protocol          , Port    , Retries
DEV_1        , 1          , BACnet_MSTP       , R1      , 2
```

5.3 Client Side Map Descriptor Parameters

5.3.1 FieldServer Specific Map Descriptor Parameters

Column Title	Function	Legal Values
Map_Descriptor_Name	Name of this Map Descriptor. This is used for Object_Name Property. Ensure that unique names are configured for each device. Refer to **Section 7.1**.	Up to 37 alphanumeric characters
Data_Array_Name	Name of Data Array where data is to be stored in the FieldServer.	One of the Data Array names from **Section 4**
Data_Array_Offset	Starting location in Data Array.	0 to (Data_Array_Length -1) as specified in **Section 4**
Function	Function of Client Map Descriptor.	WRBC, WRBX, ARCS, RDBC, AWS

5.3.2 Driver Related Map Descriptor Parameters

Column Title	Function	Legal Values
Node_Name	Name of Node to fetch data from.	One of the Node names specified in **Section 5.2**
Object_Type (Alias = Data_Type)	Type of object. Refer to **Section 10.2** for more information.	AI, AO, AV, BI, BO, BV, MI, MO, MV, NC, LSP, custom (**Section 7.6**)
Object_Instance (Alias = Address)	Instance of the object on the device.	0-4194303
Property	The BACnet property to be read.	Refer to **Section 10.4**
Data_Array_Low_Scale*	Scaling zero in Data Array.	-32767 to 32767, **0**
Data_Array_High_Scale*	Scaling max in Data Array.	-32767 to 32767, **100**
Node_Low_Scale*	Scaling zero in Connected Node.	-32767 to 32767, **0**
Node_High_Scale*	Scaling max in Connected Node.	-32767 to 32767, **100**

[9] Permitted Values (APDU_Timeout and APDU_Retries) – The total delay in seconds before giving up on a transmission (APDU_Timeout/1000 * (APDU_Retries/1000 + 1)) should not exceed 65 seconds.

Length*	Used to create an array of sequential Object_Instances on an Object_Type.	1 to max point count of the FieldServer, **1**
Linked_Map_Descriptors*	This parameter allows the linking of Map Descriptors to an active read or write Map Descriptor in order to construct a ReadPropertyMultiple or WritePropertyMultiple transaction. Refer to **Section 7.2.2**.	The name of a previously defined active Map Descriptor, **-**
Array_Index*	When referencing Multistate properties, allows the user to specify the index of the property to be read. If 1 is specified, the first one will be read, if 2 is specified, the second will be read, etc. If 0 is specified, the driver will return the total number (count) of array items linked to the property. If the parameter is not specified, a list of all items will be returned. Refer to **Section 7.1.3**.	0 to max number of array items in the BACnet Property, **-**
Write_Priority*	Allows the driver to specify the write priority used to write an output.	1-16, **16**
Length*	When the optional Length parameter is set to a value N, the Map Descriptor will include N consecutive instances of the specified Object_Type. For example, if a Map Descriptor specifies Object_Type, Address 4 and Length 3, this means that the AO instances 4, 5 and 6 will be included in the read or write transaction. The corresponding Data Array values are in consecutive positions in the Data Array, starting at the specified Data_Array_Offset.	Any positive integer that falls between the range of the Data_Array_Offset and the Data_Array_Length as specified in **Section 4**, **-**; see **Section 7.2.2**
COV Specific If the remote server supports COV-B it is possible to configure the FieldServer BACnet Client to subscribe to COV updates. This can be done instead of or in addition to periodic reads. The advantage of using COV is that changed values will be reported within a much shorter time, especially if a large number of points is being monitored. However, not all BACnet devices support COV, so exercise care when using this function.		
Service	BACnet Point service.	COVSubscribe
Confirmed	Used to select Confirmed or Unconfirmed COV Notifications.	Yes, No
Function	Set to ARS or AWS to subscribe once on startup, or to RDBC or WRBC to re-subscribe after each Scan_Interval.	ARS, AWS, RDBC, WRBC
COV_Lifetime	Specify the COV subscription lifetime in seconds. Use a large value to get the benefit of decreased network communications.	Value in seconds
Scan_Interval	If using RDBC in order to re-subscribe periodically, set the Scan_Interval to the desired re-subscription interval. This should be significantly shorter than COV_Lifetime.	Value in seconds

5.3.3 Timing Parameters

Column Title	Function	Legal Values
Scan_Interval*	Rate at which data is polled.	0-32000, **2s**

5.4 Map Descriptor Examples

All three examples below are addressing the same Modbus registers:

```
//   Client Side Map Descriptors

Map_Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Data_Type , Object_Instance , Property        , Scan_Interval
CMD_AI_01           , DA_AI_01        , 0                 , Rdbc     , DEV_1     , AI        , 1               , Present_Value , 20.000s
CMD_AI_02           , DA_AI_01        , 1                 , Rdbc     , DEV_1     , AI        , 2               , Present_Value , 20.000s
CMD_AI_03           , DA_AI_01        , 2                 , Rdbc     , DEV_1     , AI        , 3               , Present_Value , 20.000s
CMD_AO_01           , DA_AO_01        , 0                 , Rdbc     , DEV_1     , AO        , 1               , Present_Value , 30.000s
CMD_AO_02           , DA_AO_01        , 1                 , Rdbc     , DEV_1     , AO        , 2               , Present_Value , 30.000s
CMD_AO_03           , DA_AO_01        , 2                 , Rdbc     , DEV_1     , AO        , 3               , Present_Value , 30.000s

Map_Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Data_Type , Object_Instance , Property        , Scan_Interval
CMD_DI_01           , DA_DI_01        , 0                 , Rdbc     , DEV_1     , BI        , 1               , Present_Value , 15.000s
CMD_DI_02           , DA_DI_01        , 1                 , Rdbc     , DEV_1     , BI        , 2               , Present_Value , 15.000s
CMD_DI_03           , DA_DI_01        , 2                 , Rdbc     , DEV_1     , BI        , 3               , Present_Value , 15.000s
CMD_DO_01           , DA_DO_01        , 0                 , Rdbc     , DEV_1     , BO        , 1               , Present_Value , 30.000s
CMD_DO_02           , DA_DO_01        , 1                 , Rdbc     , DEV_1     , BO        , 2               , Present_Value , 30.000s
CMD_DO_03           , DA_DO_01        , 2                 , Rdbc     , DEV_1     , BO        , 3               , Present_Value , 30.000s
```

5.4.1 COV Specific

In this example, Map Descriptor CMD_AI_01_SUB creates and periodically renews the COV subscription. Map Descriptor CMD_AI_01_RD performs periodic reads of the same object and will also be updated by COV Notifications received.

```
Map Descriptor
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name , Data_Type
CMD_AI_01_SUB       , DA_AI           , 0                 , RDBC     , BCU_01    , AI
CMD_AI_01_RD        , DA_AI           , 1                 , RDBC     , BCU_01    , AI
```

```
, Object_ID , Service      , COV_Lifetime , Confirmed , Scan_Interval
, 0         , COVSubscribe , 600         , Yes       , 300
, 0         , -            , -           , -         , 30
```

6 Configuring the FieldServer as a Server

For detailed information on FieldServer configuration, refer to the FieldServer Configuration Manual. The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer (see ".csv" sample files provided with the FieldServer).

This section documents and describes the parameters necessary for configuring the FieldServer to communicate with a BACnet MS/TP Client.

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for BACnet MS/TP communications the following three actions must be taken. The driver independent FieldServer buffers need to be declared in the "Data Arrays" section. The FieldServer virtual node(s) need to be declared in the "Server Side Nodes" section. And the data to be provided to the client(s) needs to be mapped in the "Server Side Map Descriptors" section. Details on how to perform these steps can be found in the following sections.

NOTE: In the tables below, * indicates an optional parameter with the bold legal value as default.

6.1 Driver Specific FieldServer Parameters

Section Title
Bridge

Column Title	Function	Legal Values
Title	FieldServer name.	Text
System_Node_ID	If the FieldServer is acting as a BACnet Router by defining multiple server nodes, the system_node_id sets the BACnet Device instance of the Router. If the MAC Address is not defined in the connection section, the system_node_id also sets the MAC Address of the FieldServer if it is within the range of 0-127.	0-4194303

Example

```
//    FieldServer Driver specific parameters

Bridge
Title                                   , System_Node_ID
BacNet MSTP Server V1.02a               , 11
```

6.2 Server Side Connection Parameters

Section Title		
Connections		
Column Title	**Function**	**Legal Values**
Port	Specify which port the device is connected to the FieldServer.	P1-P2[10], R1-R2[11]
Baud*	Specify baud rate.	**9600**, 19200, 38400, 76800
Parity*	Specify parity.	**None,** Odd, Even
Data_Bits*	Specify data bits.	7**, 8**
Stop_Bits*	Specify stop bits.	**1**, 2
Protocol	Specify protocol used.	BACnet_MSTP
Connection_Type*	Specify if the connection must be in Master_Mode or in Slave_Mode.	**MSTP_Master_Mode** MSTP_Slave_Mode
Max_Master*	Specify the highest possible MAC_Address that exists on the MS/TP network. Only applicable for MSTP_Master Mode connections.[12]	1-**127**
Max_Info_Frames*	Specify the maximum number of information frames the connection may send before it must pass the token. Only applicable for MSTP_Master Mode connections. [12]	**1**-65535
MAC_Address[13]	Specify the MS/TP MAC address for this connection.	0-127 for MSTP_Master_Mode; 0-254 for MSTP_Slave_Mode
Reliability_Option*	The Reliability_Option parameter selects the rules by which the BACnet server determines the reliability property of a BACnet object whose present value is read from a remote server node. See **Section 7.1.7**.	Track_Object_Status, **Track_Node_Status,** Always_Reliable
Net_Addr_Len*	Set the address length of the virtual network used to connect multiple virtual BACnet devices to the external network. By default, this address is 6 bytes long; older FieldServer versions used 2 or 4 bytes, and if a legacy system depends on a length different to 6, this can be configured here. Note that since the virtual network address corresponds directly to the Node ID, the Node ID values should not exceed the maximum value that can be encoded in the number of address bytes specified.	1-**6**, **-**

[10] P1 to P2 will require the optional RS-485 converter (Part # 4301-00).
[11] Not all ports shown are necessarily supported by the hardware. Consult the appropriate Instruction manual for details of the ports available on specific hardware.
[12] For Server configurations, the Max_Master and Max_Info_Frames parameters (or properties) can be overridden from a BACnet Client by using the "Write Property" command. Note that when the FieldServer is restarted, the default values specified in the configuration file will be applied.
[13] Previously the MAC_Address was set using the System_Node_ID parameter in the Common Information section.

Virtual_Network_Number*	Specify a unique network number if there are multiple virtual Server Nodes. Refer to **Section 7.1.2**.	1 – 65535, **5**
Is_Router	Specifies whether the FieldServer acts as a router.	No, Yes (if there are multiple server nodes the default is **Yes**, if there is one then default is **No**)
Router_Network_Number*	Sets the network number for the external BACnet segment that the connection is physically connected.	1 – 65535, **5**

Example

```
//    Server Side Connections
Connections
Port     , Baud     , Parity     , Data_Bits    , Stop_Bits    , Protocol          , Timeout    , Connection_Type
R1       , 38400    , None       , 8            , 1            , BACnet_MSTP       , 30         , MSTP_Master_Mode
```

6.3 Server Side Node Parameters

Section Title
Nodes[14]

Column Title	Function	Legal Values
Node_Name	Provide name for Node.	Up to 31 alphanumeric characters
Node_ID	BACnet station address of physical Server node.	0 – 4194303
Protocol	Specify protocol used.	BACnet_MSTP
Node_Option*	Enable or disable COV for this Node. Refer to **Section 7.2.1** for further information.	COV_Enable, **COV_Disable**, **-**
Model_Name*	Sets the Model Name Property of the Device Object.	Up to 32 alphanumeric characters
Network_Location*	Specifies on which network the virtual node is located.	-, Local Segment; refer to **Section 7.3**
Device_Description	Sets the Description Property of the Device Object.	Any text string of length up to 40 characters, or a Data Array name and start offset that holds the text string in the following format: <Data Array name.offset> e.g. <DA_AI_01_Desc.0>
Device_Location	Sets the Location Property of the Device Object.	Up to 32 alphanumeric characters
Disabled_Services*	Certain BACnet Services can be disabled on a BACnet Server Node when specific requirements necessitate this. Refer to **Section 7.2.3** for more information.	**-**, WPM, COV, RPM

[14] Port is not declared in the Server Side Nodes.

APDU_Timeout*	The time in milliseconds between retransmissions of an APDU requiring acknowledgement for which no acknowledgment has been received.	**10000ms**[15]
APDU_Retries*	The maximum number of times that an APDU shall be retransmitted.	**3**[15]
Srv_Offline_Method*	Controls the system_status property of the device object.	**-**, Any_Offline, All_Offline; refer to **Section 7.1.13**

Example

```
//    Server Side Nodes
Nodes
Node_Name        , Node_ID   , Protocol        , Node_Option   , Model_Name
Virtual_DEV_11   , 11        , BACnet_MSTP     , COV_Enable    , WBA11M. MSTP
```

6.4 Server Side Map Descriptor Parameters

6.4.1 FieldServer Specific Map Descriptor Parameters

Column Title	Function	Legal Values
Map_Descriptor_Name	Name of this Map Descriptor. This is used for Object_Name Property. Refer to **Section 7.1**.	Up to 37 alphanumeric characters[16]
Data_Array_Name	Name of Data Array where data is to be stored in the FieldServer.	One of the Data Array names from **Section 4**
Data_Array_Offset	Starting location in Data Array.	0 to (Data_Array_Length-1) as specified in **Section 4**
Function	Function of Server Map Descriptor.	Passive

6.4.2 Driver Related Map Descriptor Parameters

Column Title	Function	Legal Values
Node_Name	Name of Node to fetch data from.	One of the Node names specified in **Section 6.3**
Object_Type (Alias = Data_Type)	Data type in Controller. Refer to **Section 10.2** for more information.	AI, AO, AV, BI, BO, BV, MI, MO, MV, NC, LSP (**Section 7.6**)
Object_Instance (Alias = Address)	Instance of the Object on the Device.	0-4194303
Data_Array_Low_Scale*	Scaling zero in Data Array.	-32767 to 32767, **0**
Data_Array_High_Scale*	Scaling max in Data Array.	-32767 to 32767, **100**
Node_Low_Scale*	Scaling zero in Connected Node.	-32767 to 32767, **0**
Node_High_Scale*	Scaling max in Connected Node.	-32767 to 32767, **100**
Length*	Used to create an array of sequential Object_Instances on an Object_Type.	1 to max point count of the FieldServer, **1**

[15] Permitted Values (APDU_Timeout and APDU_Retries) – The total delay in seconds before giving up on a transmission (APDU_Timeout/1000 * (APDU_Retries/1000 + 1)) should not exceed 65 seconds.
[16] Object_Name values of any length may be written via BACnet (subject only to memory and message length constraints).

Units*	The object units.	See **Section 10.6** or a Data Array name and offset can be used to dynamically change the units. The value stored in the offset should be a valid BACnet unit enumeration. It should be configured in the following format: <Data Array Name:offset> e.g. DA_Units:1; **No_Units**
Active_Text*[17]	Specify the Active Text property of the Object.	Any text string of length up to 40 characters, **Active**
Inactive_Text*[17]	Specify the Inactive Text property of the Object.	Any text string of length up to 40 characters, **Inactive**
Relinquish_Default[17]	Specify the value to be returned as Present_Value on startup or when control is relinquished. Must be specified for outputs of AO, BO or MO data types; see **Section 10.4**.	Any Float value
Notification_Class*	Specify the Mapdescriptor_Name of the Notification_Class Object that manages Intrinsic Reporting for this Map Descriptor; see **Section 7.2.1**.	A Mapdescriptor_Names of type NC (Notification Class). Objects are not available to NC if left out.
Ack_Required*	For a Notification_Class Object, specify whether EventNotifications require a user Acknowledgement; see **Section 7.2.1**.	Yes, **No**
COV_Increment*	For a Server Map Descriptor of type AO or AI, initialize the COV_Increment property; see **Section 7.2.1**.	Any Float value, **0**
Input_alarm_State*	This parameter is required when a **BI, BO or BV** Map Descriptor is configured for alarms, i.e. when a Notification_Class is specified for the Map Descriptor. It defines the value (0 or 1) that is to be treated as the alarm (i.e. off-normal) value.	0, 1 (there is no default)
Description*	Specify the object's description property (e.g. Room Temp).	Any text string up to 40 characters or a Data Array name and start offset that holds the text string in the following format: <Data Array name.offset> (e.g. <DA_AI_01_Desc.0>)[18]; if not configured, defaults to Object_Name

[17] See the BACnet DFS to determine if a particular object supports this property.
[18] Description values may be written via BACnet (subject only to memory and message length constraints).

Update_Property*	The Update_Property Parameter may be configured on a BACnet Server Map Descriptor to allow a property other than the Present_Value to be updated *in addition* to the Present_Value when the Data Array Value changes. **Warning:** This is a highly specialized function only to be used under very particular circumstances by users with a detailed understanding of BACnet. Refer to **Section 7.1.8**.	**Present Value**, Relinquish_Default
Process_ID*	The handle of a process within the recipient device that is to receive the event notification; see **Section 7.2.1**.	Any unsigned integer, **-**
Notification_Class_MD*	This is the Notification Class map descriptor governing the sending of Event Notifications (Intrinsic Alarming); see **Section 7.2.1**.	Map Descriptor defined per **Section 7.2.1**
Min_Pres_Value*	Specify the "Minimum Present Value" property for an AO Object. This setting can also be used on AV, even though the AV object itself does not support the property.	Any floating point value is legal. If not specified, default to +- Infinity[19]
Max_Pres_Value*	Specify the "Maximum Present Value" property for an AO Object. This setting can also be used on AV, even though the AV object itself does not support the property.	Any floating point value is legal. If not specified, default to +- Infinity[19]
Low_Alarm	For **AI, AO and AV** object types this sets the lower alarm limit.	Any floating point value
High_Alarm	For **AI, AO and AV** object types this sets the upper alarm limit.	Any floating point value
Alarm_Deadband*	For **AI, AO, and AV** types, this is the amount by which the present value must be greater than the Low_Alarm limit or less than the High_Alarm limit for the object to return to the Normal state and send a to-normal event.	Any floating point value, **0.0**

[19] In this implementation, the Min_Pres_Value & Max_Pres_Value properties can be read but not written via BACnet.

6.5 Map Descriptor Examples

```
//    Client Side Map Descriptors

Map_Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name     , Object_Type , Object_Instance , Units
SMD_AI_01           , DA_AI_01        , 0                 , Passive  , Virtual_DEV_11 , AI          , 1               , Degrees-Fahrenheit
SMD_AI_02           , DA_AI_01        , 1                 , Passive  , Virtual_DEV_11 , AI          , 2               , Degrees-Fahrenheit
SMD_AI_03           , DA_AI_01        , 2                 , Passive  , Virtual_DEV_11 , AI          , 3               , Degrees-Fahrenheit
SMD_AO_01           , DA_AO_01        , 0                 , Passive  , Virtual_DEV_11 , AO          , 1               , percent-relative-humidity
SMD_AO_02           , DA_AO_01        , 1                 , Passive  , Virtual_DEV_11 , AO          , 2               , percent-relative-humidity
SMD_AO_03           , DA_AO_01        , 2                 , Passive  , Virtual_DEV_11 , AO          , 3               , percent-relative-humidity
```

```
//    Client Side Map Descriptors

Map_Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name     , Object_Type , Object_Instance
SMD_DI_01           , DA_DI_01        , 0                 , Passive  , Virtual_DEV_11 , BI          , 1
SMD_DI_02           , DA_DI_01        , 1                 , Passive  , Virtual_DEV_11 , BI          , 2
SMD_DI_03           , DA_DI_01        , 2                 , Passive  , Virtual_DEV_11 , BI          , 3
SMD_DO_01           , DA_DO_01        , 0                 , Passive  , Virtual_DEV_11 , BO          , 1
SMD_DO_02           , DA_DO_01        , 1                 , Passive  , Virtual_DEV_11 , BO          , 2
SMD_DO_03           , DA_DO_01        , 2                 , Passive  , Virtual_DEV_11 , BO          , 3
```

```
Map_Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name     , Object_Type
SMD_11_AO_03        , DA_AO_01        , 0                 , Passive  , Virtual_Dev_11 , AO
```

```
, Object_Instance , Relinquish_Default , Min_Pres_Value , Max_Pres_Value
, 03              , 0                  , 10              , 20
```

7 Useful Features

7.1 BACnet Properties

7.1.1 BACnet Object Names

When an external BACnet Client builds a list of Object Names, the BACnet Server Map Descriptor names determine the BACnet Object Name. If the Map Descriptor length is greater than 1, the Object Name will be suffixed with the index into the Map Descriptor. For example, if the Map Descriptor name is SMD_AI_01 and the length 3, then the Object Names will be SMD_AI_01[0], SMD_AI_01[1] and SMD_AI_01[2]. The maximum length of a point Object Name is the same as the maximum Map_Descriptor_Name length. Refer to **Section 6.4**.

The Device Object Name is set from either the Node Name or the Model Name, and the maximum length is determined by the maximum number of characters defined in the legal values column. Refer to **Section 6.4**.

NOTE: It is important that unique Map Descriptor names are created for each device to ensure that the Object Names are unique.

Using Tags to Create Unique Device and Object Names

The <device_id> tag can be used to tag the actual device object instance to the end of the specified Object_Name.

```
Nodes
Node_Name                , Node_ID   , Protocol       , Node_Option
Dev_MSTP_<device_id>     , 11        , Bacnet_MSTP    , COV_Enable
```

In the example above the FieldServer will appear as Dev_MSTP_11 when it is discovered on BACnet. The Map_Descriptor reference to the BACnet node must also include the same <device_id> tag. If the Node_ID is now changed using the dip-switches, then the Device Object_Name will automatically change with the same value at the end.

```
Map_Descriptors
Map_Descriptor_Name      , Data_Array_Name   , Data_Array_Offset   , Function   , Node_Name
Dev_MSTP_<device_id>     , DA_DI_01          , 0                   , Server     , Virtual_BCU_<device_id>
```

```
, Data_Type   , Object_ID   , Relinquish_Default
, DI          , 1           , -
```

7.1.2 Virtual Network Number

Whenever there are multiple virtual nodes, the FieldServer interconnects these on a virtual internal network and acts as a virtual router between this and the external network. Each internal virtual network must be assigned a network number. The BACnet specification states that this network number must be unique across the entire BACnet internetwork. The following configuration error is generated when an attempt is made to reuse the same network number on the same device:

```
BACnet Router : Error: Network Number 5 has already been used. Configure a unique Virtual_Network_Number
on each BACnet _MSTP Connection
```

NOTE: Failing to use unique network numbers can lead to routing problems on a BACnet network.

The Virtual_Network_Number parameter can be used to assign network numbers to the internal virtual networks associated with each BACnet connection. The defaults will also be unique for the FieldServer itself, but it is up to the user to ensure that the network number is unique across the entire network. The error message shown above only arises when the same virtual network number is specified for more than one connection.

Section Title
Connections

Column Title	Function	Legal Values
Virtual_Network_Number*	Specify a unique network number if there are multiple virtual Server nodes.	1-65534, **5**

Example

```
//    Server Side Connections
Connections
Adapter         , Protocol          , Virtual_Network_Number
N1              , Bacnet_MSTP       , 5
```

NOTE: The BACnet driver is not limited to the number of device instances on the Server Side but caution that using over 200 Device Instances could result in issues with the Who-Is transaction.

NOTE: If an application demands multiple BACnet/IP Server nodes and multiple BACnet MS/TP Server Nodes simultaneously, they will at present all appear with the same virtual network number. This creates a potential conformance problem if there is an external BACnet router connecting the BACnet/IP and BACnet MS/TP networks to which the FS is also connected, as duplicate virtual networks with the same network number would then exist in the BACnet network, which is not permitted. Contact technical support for assistance.

7.1.3 Accessing Data from BACnet Properties Comprising Arrays of Values

Some BACnet Object properties (e.g. Priority_Array) are arrays of values (the Priority_Array property is an array of 16 values). In order to read a specific array entry, the Array_Index must be specified in the Map Descriptor. Array_Index is a Client Side Map Descriptor function.

The following example shows a configuration that will read the Priority_Array value at Array_Index 7, belonging to Analog Output 1.

```
Map_Descriptors
Map_Descriptor_Name  , Data_Array_Name  , Data_Array_Offset  , Function  , Node_Name  , Data_Type  , Object_ID  , Property        , Array_Index
CMD_AI_01            , DA_AI_01         , 0                  , Rdbc      , DEV_01      , AO         , 1          , Priority_Array  , 7
```

Details of the relevant BACnet properties and their associated arrays can be found in the BACnet Protocol Spec.

If no Array_Index is specified, then the entire property (i.e. the entire BACnet Array) is returned.

The Data_Index parameter functions as follows for ReadPropertyMultiple or WritePropertyMultiple requests when the Client Map Descriptor Length N is greater than 1:

- When an Array_Index is specified, the ReadPropertyMultiple or WritePropertyMultiple request will iterate through N successive Array_Index values for the given Object and Property (e.g. Priority Array Index 1 to 16 of Analog Output 1).
- When no Array_Index is specified, the ReadPropertyMultiple or WritePropertyMultiple request will iterate through N successive Object Instances, reading the specified Property from each successive object (e.g. Present Value of Analog Output 1 to Analog Output 16).

7.1.4 FieldServer Implementation of BACnet Priority Arrays

When BACnet Output objects are written to the Server side of the FieldServer, an associated write priority is given to each write value. When the FieldServer receives the write value, it stores it to the Map Descriptor Priority Array Table at the specified priority. The Priority Array Table is then scanned and the value with the highest priority is stored to the Data Array location specified by the Map Descriptor.

When a Write “Relinquished” command is received, the value is removed from the Priority Array Table and the next highest value from the Priority Array Table is stored to the Data Array.

If all values have been “Relinquished” from the Priority Array Table, then the Map Descriptors “Relinquish Default” value will be stored to the Data Array.

Accessing Priority Array Information

The Priority Array table and its “In_Use” (or Not Relinquished) state are stored internally to every Map Descriptor and cannot be accessed directly. The information can be accessed indirectly by specifying the following Data Arrays which will maintain an exact copy of the Priority Array Table for the Map Descriptor.

Section Title		
Map_Descriptors		
Column Title	**Function**	**Legal Values**
DA_Pri_Array	Name of Data Array where the Priority Array Table will be stored. Location 0 is the Relinquish Default value and locations 1 to 16 the different entries of the Priority Array Table.	Up to 16 alphanumeric characters
DA_Pri_Array_Offset*	Starting location in Data Array.	1-65535, **0**
DA_Pri_In_Use	Name of Data Array that indicates if a specific Priority Value is in use. Location 0 indicates whether the Relinquish Default has been set and locations 1 to 16 indicate whether the index is in use (1), or Relinquished (0).	Up to 16 alphanumeric characters
DA_Pri_In_Use_Offset*	Starting location in Data Array.	1-65535, **0**

```
//  Analog Output Map_Descriptor for testing Priority Arrays

Map_Descriptors
Map_Descriptor_Name  , Data_Type  , Object_ID  , Function  , Data_Array_Name  , Data_Array_Index  , Node_Name  , Length
CMD_AOP_1            , AO         , 1           , Passive   , DA_OUT           , 0                 , N1 11      , 1
```

```
, Relinquish_default  , DA_Pri_Array    , DA_Pri_Array_Offset  , DA_Pri_In_Use    , DA_Pri_In_Use_Offset
, 40.56               , DA_Pri_Array_1  , 0                    , DA_Pri_in_use_1   , 0
```

7.1.5 Relinquishing Control of a Point as a Client

It is possible to relinquish control of a point by writing a null to the correct priority level. The following example illustrates how this is done.

```
Map_Descriptors
Map_Descriptor_Name  , Data_Type  , Function  , Scan_Interval  , Data_Array_Name  , Data_Array_Index
CMD AO               , AO         , Rdbc      , 1.0s            , DA AO            , 2
CMD AO Rel           , AO         , Wrbx      , 1.0s            , DA AO            , 3
```

```
, Node_Name  , Address  , Length  , Write_Priority  , Service
, N1 1       , 1        , 1       , 7               , -
, N1 1       , 1        , 1       , 7               , Relinquish
```

In the above example:

- Map_Descriptor_Name CMD AO is a Read Map Descriptor that will write at priority 7 if a write-through occurs.
- Map_Descriptor_Name CMD AO Rel is a Write-on-Change Map Descriptor that will write a NULL at priority 7 (i.e. release Priority Array entry 7) when a write occurs.
- Address parameter length must be 1 as shown above.

7.1.6 BACnet State Text Preload

BACnet Multistate Objects have a State_Text property. This property is defined as an array of character strings representing descriptions of all possible states of the Present_Value. The number of descriptions matches the number of states defined in the Number_Of_States property. The Present_Value, interpreted as an integer, serves as an index into the array.

When Multistate Objects are configured on a BACnet server it is necessary to define the State_Text property. This section illustrates how to define the State_Text character strings and how to associate these definitions with Multistate Server Map Descriptors.

The maximum permitted length of any State_Text string is 50 characters.

Method 1 – Using an Offset/User Table

```
//set up a look up table

Offset_Table
Offset_Table_Name     , Table_String       , Table_Index_Value
FIRE_ALRM_TEXT        , SYSTEM READY       , 1
FIRE_ALRM_TEXT        , ALARM              , 2
FIRE_ALRM_TEXT        , MAINTENANCE        , 3
FIRE_ALRM_TEXT        , OFF-LINE           , 4
FIRE_ALRM_TEXT        , IN SERVICE         , 5
FIRE_ALRM_TEXT        , OTHER              , 6
```

NOTE: Valid entries for Table_Index_Value are integers of 1 to 100. Anything larger will be either truncated and/or omitted.

NOTE: The Offset_Table_Name can be up to 16 alphanumeric characters.

```
Data_Arrays
Data_Array_Name    , Data_Format    , Data_Array_Length
DA_MI_01           , UINT16         , 100
```

```
Map_Descriptors
Map_Descriptor_Name  , Data_Array_Name  , Data_Array_Offset  , Function  , Node_Name  , Data_Type  , Object_ID  , Length  , State_Text_Array
CMD_MI_01            , DA_MI_01         , 0                  , Passive   , N1 11      , MI         , 1          , 1       , Fire_Alrm_Text
```

Method 2 – Using a Single Data Array

```
Data_Arrays
Data_Array_Name    , Data_Format    , Data_Array_Length
DA_MI_01           , UINT16         , 100
DA_STATE_TXT       , BYTE           , 200
```

```
Preloads
Data_Array_Name  , Preload_Data_Value                                  , Preload_Data_Format  , Preload_Data_Index
DA_STATE_TXT     , MyState1 MyState2 MyState3 MyState4 MyState5 MyState6 , String               , 0
```

```
Map_Descriptors
Map_Descriptor_Name  , Data_Type  , Object_ID  , Function  , Data_Array_Name  , Node_Name  , Length  , State_Text_Array
CMD_MI_01            , MI         , 1          , Passive   , DA_MI_01          , N1 11      , 1       , Da_State_Txt
```

Using Intrinsic Reporting for a Multistate Value

To use Intrinsic Reporting for a Multistate value, it is necessary to classify each of the states as either Normal, Alarm or Trouble. This is done by adding another column to the Offset_Table, called Table_User_Value. Each state is then classified by inserting one of the following values in its row:

0 = normal
1 = alarm
2 = fault

```
//set up a look up table

Offset_Table
Offset_Table_Name   , Table_String        , Table_Index_Value   , Table_User_Value
FIRE_ALRM_TEXT      , SYSTEM READY         , 1                   , 0
FIRE_ALRM_TEXT      , ALARM                , 2                   , 1
FIRE_ALRM_TEXT      , MAINTENANCE          , 3                   , 2
FIRE_ALRM_TEXT      , OFF-LINE             , 4                   , 2
FIRE_ALRM_TEXT      , IN SERVICE           , 5                   , 0
FIRE_ALRM_TEXT      , OTHER                , 6                   , 2
```

NOTE: The state value (Table_Index_Value) is an enumerated value starting at 1. Zero is not a valid value, but since many client side configurations will wake up with values of zero, the BACnet server will treat a value of zero as a normal value (not as an alarm or fault value).

7.1.7 Factors Determining the Reliability Property

The Reliability Property of a BACnet object in a Server configuration is determined as follows:

- For Multistate Inputs, Outputs and Values, the Reliability property is set to MULTI_STATE_FAULT (9) when the Present_Value of the object corresponds to an entry in the Fault_Values property of the object, as configured by the State_Text table in the configuration file.
- For all object types, the Reliability property is set to COMMUNICATIONS_FAILURE (12) when the responsible Server Node is offline.
- When neither Condition 1 nor Condition 2 is met, the Reliability property is set to NO_FAULT_DETECTED (0)

7.1.8 Update Property Function

The Update_Property parameter may be configured on a BACnet Server Map Descriptor in order to allow a property other than the Present_Value to be updated in addition to the Present_Value when the Data Array value changes.

NOTE: This is a highly specialized function only to be used under very particular circumstances, by users with a detailed understanding of BACnet and of what is to be achieved.

Supported Values

- Present_Value (default operation)
- Relinquish_Default

7.1.9 BACnet State Text Preload

BACnet Multistate Objects have a State_Text property. This property is defined as an array of character strings representing descriptions of all possible states of the Present_Value. The number of descriptions matches the number of states defined in the Number_Of_States property. The Present_Value, interpreted as an integer, serves as an index into the array.

When Multistate Objects are configured on a BACnet server it is necessary to define the State_Text property. This section illustrates how to define the State_Text character strings and how to associate these definitions with Multistate Server Map Descriptors.

The maximum permitted length of any State_Text string is 50 characters.

Method 1 – Using an Offset/User Table

```
//set up a look up table

Offset_Table
Offset_Table_Name     , Table_String       , Table_Index_Value
FIRE_ALRM_TEXT        , SYSTEM READY       , 1
FIRE_ALRM_TEXT        , ALARM              , 2
FIRE_ALRM_TEXT        , MAINTENANCE        , 3
FIRE_ALRM_TEXT        , OFF-LINE           , 4
FIRE_ALRM_TEXT        , IN SERVICE         , 5
FIRE_ALRM_TEXT        , OTHER              , 6
```

NOTE: Valid entries for Table_Index_Value are integers of 1 to 100. Anything larger will be either truncated and/or omitted.

NOTE: The Offset_Table_Name can be up to 16 alphanumeric characters.

```
Data_Arrays
Data_Array_Name    , Data_Format    , Data_Array_Length
DA_MI_01           , UINT16         , 100
```

```
Map_Descriptors
Map_Descriptor_Name  , Data_Array_Name  , Data_Array_Offset  , Function  , Node_Name  , Data_Type  , Object_ID  , Length  , State_Text_Array
CMD_MI_01            , DA_MI_01         , 0                  , Passive   , N1 11      , MI         , 1          , 1       , Fire_Alrm_Text
```

Method 2 – Using a Single Data Array

```
Data_Arrays
Data_Array_Name    , Data_Format    , Data_Array_Length
DA_MI_01           , UINT16         , 100
DA_STATE_TXT       , BYTE           , 200
```

```
Preloads
Data_Array_Name  , Preload_Data_Value                                  , Preload_Data_Format  , Preload_Data_Index
DA_STATE_TXT     , MyState1 MyState2 MyState3 MyState4 MyState5 MyState6 , String               , 0
```

```
Map_Descriptors
Map_Descriptor_Name  , Data_Type  , Object_ID  , Function  , Data_Array_Name  , Node_Name  , Length  , State_Text_Array
CMD_MI_01            , MI         , 1          , Passive   , DA_MI_01          , N1 11      , 1       , Da_State_Txt
```

Using Intrinsic Reporting for a Multistate Value

To use Intrinsic Reporting for a Multistate value, it is necessary to classify each of the states as either Normal, Alarm or Trouble. This is done by adding another column to the Offset_Table, called Table_User_Value. Each state is then classified by inserting one of the following values in its row:

0 = normal
1 = alarm
2 = fault

```
//set up a look up table

Offset_Table
Offset_Table_Name   , Table_String        , Table_Index_Value   , Table_User_Value
FIRE_ALRM_TEXT      , SYSTEM READY         , 1                   , 0
FIRE_ALRM_TEXT      , ALARM                , 2                   , 1
FIRE_ALRM_TEXT      , MAINTENANCE          , 3                   , 2
FIRE_ALRM_TEXT      , OFF-LINE             , 4                   , 2
FIRE_ALRM_TEXT      , IN SERVICE           , 5                   , 0
FIRE_ALRM_TEXT      , OTHER                , 6                   , 2
```

NOTE: The state value (Table_Index_Value) is an enumerated value starting at 1. Zero is not a valid value, but since many client side configurations will wake up with values of zero, the BACnet server will treat a value of zero as a normal value (not as an alarm or fault value).

7.1.10 Factors Determining the Reliability Property

The Reliability Property of a BACnet object in a Server configuration is determined as follows:

- For Multistate Inputs, Outputs and Values, the Reliability property is set to MULTI_STATE_FAULT (9) when the Present_Value of the object corresponds to an entry in the Fault_Values property of the object, as configured by the State_Text table in the configuration file.
- For all object types, the Reliability property is set to COMMUNICATIONS_FAILURE (12) when the responsible Server Node is offline.
- When neither Condition 1 nor Condition 2 is met, the Reliability property is set to NO_FAULT_DETECTED (0)

7.1.11 Update Property Function

The Update_Property parameter may be configured on a BACnet Server Map Descriptor in order to allow a property other than the Present_Value to be updated in addition to the Present_Value when the Data Array value changes.

NOTE: This is a highly specialized function only to be used under very particular circumstances, by users with a detailed understanding of BACnet and of what is to be achieved.

Supported Values

- Present_Value (default operation)
- Relinquish_Default

Mode of Operation When Used for the Relinquish_Default Property

When the FieldServer starts up, the standard BACnet operation for AO, AV, BO and BV objects is that the Present_Value of the object is set to the configured Relinquish_Default value. This may not be desirable in Gateway situations where the BACnet communications module (i.e. the FieldServer) may restart independently of the controller with the physical outputs.

Instead, it is possible to use the Update_Property setting to initialize the Relinquish_Default property using the actual field Present_Value read from the Client Side device, so that a restart achieves two important outcomes:

- The Client side outputs are not modified
- The current state of the Client Side output can be read via the Relinquish_Default property. This is especially useful since the BACnet specification makes no provision for Feedback_Values on analog objects

Depending on the function of the Client Side Map Descriptor there are slightly differing behaviors.

- **RDBC** – If the Client Side Map Descriptor is configured with the RDBC function, then every read operation will cause the Relinquish_Default property on the BACnet Server side to be updated to match the new Present_Value.
- **ARS** (recommended) – If the Client Side Map Descriptor is configured with the ARS function, then only the startup read operation (which occurs on FieldServer restart, or if the Client Side Node has gone offline and is being recovered) will cause the Relinquish_Default property on the BACnet Server side to be updated to match the new Present_Value.

NOTE: This function cannot yet be used in conjunction with Complex data types.

Update Property Configuration Example

```
Map_Descriptors
Map_Descriptor_Name   , Function   , Data_Array_Name   , Data_Array_Index   , Node_Name   , Data_Type
AOP1                  , Passive    , DA AOP            , 0                  , NN 01       , AO
```

```
, Address   , Length   , Relinquish_Default   , Update_Property
, 1         , 1        , 10                   , Relinquish_Default
```

7.1.12 Using a .ini File to Set Vendor_ID and Vendor_Name of the Device Object

The vendor.ini file can be used to change the BACnet Vendor Name and Vendor ID if desired. A file with the following format must be created and downloaded to the FieldServer using the FS-GUI (Setup->File Transfer->General Tab).

```
vendor_name = Sierra Monitor Corporation
vendor_id = 37
```

NOTE: Changing the Vendor_ID and Vendor_Name on FieldServer BTL approved products may disqualify them from the BTL approval.

7.1.13 Srv_Offline_Method Legal Values

Legal Value	Description
-	Operational
All_Offline	'Non-Operational' if all of the linked Client Side nodes are offline, otherwise 'Operational'
Any_Offline	'Non-Operational' if any of the linked Client Side node is offline, otherwise 'Operational'

7.2 BACnet Services

7.2.1 COV and Intrinsic Reporting

The COV (Change of Value) and Intrinsic Reporting services are two distinct ways in which point values can be reported to a client workstation as they change in an event-driven opposed to a polling method. This can increase performance dramatically compared to using a polling method alone. This also reduces network traffic significantly.

For BACnet MS/TP, only MS/TP Master Nodes support COV and Intrinsic Reporting.

The services are suited to different purposes:

- COV is suited to value updates. On analog points the sensitivity can be set using the COV_Increment property. Only changes larger than the COV_Increment value will be reported. The value specified for the COV_Increment is not affected by Map Descriptor scaling parameters, ie. If the COV_Increment=1 the value in the Data Array must change by at least 1 for a COV notification to be sent, even if the Data Array Values are scaled.
- Intrinsic Reporting is used for alarming. It is implemented via Notification_Class objects, which can receive subscriptions from client workstations that add themselves to the RecipientList property of a Notification_Class object. Notifications are done using ConfirmedEventNotification or UnconfirmedEventNotification. Intrinsic Reporting also allows for alarms to be acknowledged (using the AcknowledgeAlarm service) and for all subscribed client workstations to be notified of alarm acknowledgements (using EventNotifications of type ACK_NOTIFICATION).

Notes on COV Configuration

- **COV functionality is not enabled by default for MS/TP.** The Node_Option parameter can be configured to enable or disable COV. An example configuration is presented in the Node Configuration example below. Systems using BACnet/IP or BACnet/Ethernet to BACnet/MS/TP routers may not function reliably if COV functionality is enabled as the router will make the MS/TP device appear as a BACnet/IP or BACnet/Ethernet device, which could lead the BACnet Workstation to expect unrealistic throughput performance when subscribing to multiple points. It is recommended only to enable COV functionality on MS/TP systems if the Client device communicates directly via MS/TP, or if the user knows that the throughput issue will not arise.
- **COV functionality (SubscribeCOV service) only applies to the Present_Value and Status_Flags properties of BACnet Objects - the SubscribeCOVProperty service is not supported by the FieldServer.** COV only works for BACnet Map Descriptors with length set to 1. If length is not specified, then it defaults to 1, so this is only a problem where length has been specified as greater than 1.
- **For analog Server Map Descriptors, the user may optionally configure a COV_Increment value to adjust the reporting threshold.** If it is not set the COV_Increment defaults to zero.
- **COV Notifications are generated for all data objects for which a remote client has issued a SubscribeCOV-Request.** The SubscribeCOV-Request regulates whether Notifications are Confirmed or Unconfirmed. The remote Client may also write the COV_Increment property in order to control the deadband for changes in analog values. The COV_Increment property can be initialized via the configuration file by setting the COV_Increment Map Descriptor Property. The value set by the configuration is an initial value that is loaded on startup. It would be replaced by any new value written by the Client.

Notes on Intrinsic Reporting Configuration

- Intrinsic Reporting is managed by Notification Class objects. At least one Notification Class object must be configured for Intrinsic Reporting to work.
- Each Data_Object that is to be monitored by Intrinsic Reporting must be linked to a Notification Class object via the Notification_Class Map Descriptor Property. For analog points alarm limits must be set up, and for binary points, the Input_Alarm_State (specifying which binary value (0 or 1) to regard as the Alarm state) must be set up.
- The Notification_Class object contains properties that allow a client workstation to modify rules governing event reporting, such as Event_Type, days of week, start and end times etc.
- The RecipientLists are not stored permanently but are kept in volatile memory, and subscriptions must be renewed on system restart.
- A permanent static RecipientList may be configured directly on the FieldServer, eliminating the need for a remote workstation to modify the RecipientList in order to receive Event Notifications. Using this option makes the RecipientList read-only, i.e. it can then only be modified via the FieldServer config.csv file. Refer to the Priority of Intrinsic Alarming section below.
- The Event_Enable property is required for any object supporting Intrinsic Alarming. This property is supported, but to save memory is only instantiated if a point is configured for alarming i.e. if it is set up with a reference to a Notification Class map descriptor.
- These parameters are not configurable but will default to the values listed by the customer:
  - Valid Days – Mon-Sun
  - Transactions – all
  - Notify_Type – confirmed
- The config can be verified by viewing the driver aspect of the BACnet connection. The following listing of Recipient_List entries should be viewable:

  Recipient List - Notification Class BAC_NC_P1 [Instance 1]
  Object Inst: 10
  Recipient Address Unknown
  From/To Time: 00:00.00 - 23:59.59
  Valid Days: Mon,Tue,Wed,Thu,Fri,Sat,Sun
  Process ID: 12345
  Confirmed: Yes
  Transitions: To-Off-Normal To-Fault To-Normal
- The Recipient Address will be shown as unknown until the FieldServer has discovered the device specified by the Object Instance using a Who-Is broadcast. Once discovered this will show the BACnet MAC Address of the recipient device.

Node Configuration Example – Enable or Disable COV

```
Nodes
Node_Name       , Node_ID   , Protocol          , Node_Option
Virtual_Dev_11  , 11        , BACnet_MSTP       , COV_Disable
```

Map Descriptor Example – COV

```
//    Change of Value

Map_Descriptors
Map_Descriptor_Name    , Data_Array_Name    , Data_Array_Offset    , Function    , Node_Name        , Data_Type    , Object_ID    , Ack_Required
SMD_NC_01              , DA_NC_01           , 0                    , Passive     , Virtual_Dev_11   , NC           , 01           , -
```

```
Map_Descriptors
Map_Descriptor_Name , Description  , Data_Array_Name , Data_Array_Offset , Function , Node_Name     , Data_Type , Object_ID , Relinquish_Default
SMD_AI_01           , Room Temp    , DA_AI_01        , 0                 , Passive  , Virtual_Dev_11 , AI        , 01        , -
```

```
, State_Text_Array    , Notification_Class    , High_Alarm    , Low_Alarm    , Input_Alarm_State    , Confirmed    , COV_Increment
, -                   , SMD_NC_01             , 100           , 10           , -                    , Yes          , 1.0
```

Map Descriptor Example – Intrinsic Reporting

```
//    Notification class Objects

Map_Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name      , Object_Type , Object_Instance , Ack_Required
SMC_NC_01           , DA_NC_01        , 0                 , Passive  , Virtual_DEV_11 , NC          , 1             , Yes
SMC_NC_02           , DA_NC_01        , 1                 , Passive  , Virtual_DEV_11 , NC          , 2             , No

Map_Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name      , Object_Type , Object_Instance , Notification_Class
SMD_AI_01           , DA_AI_01        , 0                 , Passive  , Virtual_DEV_11 , AI          , 1             , SMC_NC_01
SMD_AO_01           , DA_AO_01        , 0                 , Passive  , Virtual_DEV_11 , AO          , 1             , SMC_NC_02

Map_Descriptors
Map_Descriptor_Name , Data_Array_Name , Data_Array_Offset , Function , Node_Name      , Object_Type , Object_Instance , Notification_Class
SMD_DI_01           , DA_DI_01        , 0                 , Passive  , Virtual_DEV_11 , BI          , 1             , SMC_NC_01
SMD_DO_01           , DA_DO_01        , 0                 , Passive  , Virtual_DEV_11 , BO          , 1             , SMC_NC_02
```

Setting the Priority of Intrinsic Alarming

Section Title
Map_Descriptors

Column Title	Function	Legal Values
Pri_To_Off_Normal*	Priority for off normal events.	Refer to Priority Value table below
Pri_To_Normal*	Priority for normal events.	Refer to Priority Value table below
Pri_To_Fault*	Priority for faults.	Refer to Priority Value table below

A BACnet EventNotification message contains a Priority field indicating the priority of the event being reported. The Priority values to be used are configured via the Priority property of the Notification Class object, using the configuration file parameters Pri_to_offnormal, Pri_to_normal and Pri_to_fault, which determine the priorities to be used respectively for all to-off_normal, to-normal and to-fault transitions reported via the Notification Class object.

Choose priority values as shown below:

Alarm and Event Priority	Network Priority
00-63	Life Safety Message
64-127	Critical Equipment Message
128-191	Urgent Message
192-255	Normal Message

Map Descriptor Example – Set up a Permanent Static Recipient List

A permanent static RecipientList may be configured directly on the FieldServer, eliminating the need for a remote workstation to modify the RecipientList in order to receive Event Notifications. Using this option makes the RecipientList read-only, it can then only be modified via the FieldServer config.csv file.

This is done by adding an Event_Receiver section to the config.csv. This section must appear after the definition and all other references to the relevant Notification Class object.

In the Map Descriptor example below, the Notification Class map descriptor governs the sending of Event Notifications (Intrinsic Alarming).

```
Map_Descriptors
Map_Descriptor_Name  , Data_Array_Name  , Data_Array_Offset  , Function  , Node_Name       , Object_Type  , Object_Instance  , Ack_Required
BAC_NC_P1            , DA_NC_01         , 0                  , Passive   , Virtual_DEV_11  , NC           , 1               , No
```

Notification Class Objects are configured as Map Descriptors. Each Notification Class Object maintains a RecipientList Property which may contain multiple destinations. Each destination is configured as an Event_Receiver **after** the relevant Notification Class Object has been configured.

The example below has a BACnet Object that will report its alarms via the Notification Class map descriptor BAC_NC_P1.

```
Map_Descriptors
Map_Descriptor_Name   , Data_Array_Name   , Data_Array_Offset   , Function   , Node_Name       , Object_Type
CMD_AI_01             , DA_AI_01          , 10                  , Passive    , Virtual_DEV_11  , AI
```

```
, Object_Instance   , Notification_Class   , Low_Alarm   , High_Alarm
, 1                 , BAC_NC_P1            , 20          , 30
```

The Event_Receiver example below sets up RecipientList entries for the defined Notification Class map descriptor. It must appear at the end of the configuration, after all other references to the Notification Class map descriptor.

```
Event_Receiver
Node_Name    , Node_ID  , Protocol     , Process_ID  , Notification_Class_MD  , Adapter
OWS_01       , 10       , BACnet_MSTP  , 12345       , BAC_NC_P1             , N1
OWS_02       , 12       , BACnet_MSTP  , 56789       , BAC_NC_P1             , N1
```

NOTE: Each Event_Receiver node can only be associated with a single notification class map descriptor. To work around this, configure a different node name for each entry.

7.2.2 Specify Read/Write PropertyMultiple Transactions with Linked Map Descriptors

The ReadPropertyMultiple (RPM) and WritePropertyMultiple (WPM) BACnet services allow many objects and attributes to be read and written in a single transaction. Since FieldServer Map Descriptors can only refer to a single object type and address range, an RPM or WPM transaction can be constructed by linking multiple Map Descriptors. This is done using the following components:

- An active read or write Map Descriptor that defines the behavior of the RPM or WPM transaction via: Function (for example – Rdbc, Arcs, Wrbc, Wrbx), Scan_interval, Timeout and Length.
- Any number of Linked Map Descriptors specifying additional objects and data array locations to be included in the transaction. Each Linked Map Descriptor is specified with the following settings:
  - Linked_Map_Descriptor – This is a reference by name to the active Map Descriptor described above, which controls the transaction
  - Function – Must specify “Passive_Client” if linked
  - Length

The number of Linked Map Descriptors is limited by the maximum message length allowed for the BACnet driver in question. When too many Map Descriptors have been linked, a SEGMENTATION_NOT_SUPPORTED error message will be generated on the first poll attempt. This message will recommend splitting the RPM or WPM transaction into multiple transactions.

```
DRV->BACnet : Linked Map Desc. "CMD_WPM" is too long.
Message Segmentation not supported.
Please split the transaction into multiple Linked Map Descriptors.
```

Example

Map_Descriptors

Map_Descriptor_Name	Object_Type	Property	Function	Data_Array_Name	Data_Array_Offset	Node_Name
CMD WPM	AO	-	Wrbc	DA WPM	0	N1 1

Address	Length	Write_Priority	Linked_Map_Descriptor
1	2	7	-

For the example above, configure an active read or write Map Descriptor and give it a unique name. This Map Descriptor will be referred to by all other Map Descriptors forming part of the multiple property transaction.

Map_Descriptor_Name	Object_Type	Property	Function	Data_Array_Name	Data_Array_Offset	Node_Name
CMD WPM 1	AO	-	Passive_Client	DA WPM	3	N1 1
CMD WPM 2	AO	-	Passive_Client	DA WPM	7	N1 1
CMD WPM 3	Device	Max_Master	Passive_Client	DA WPM	12	N1 1
CMD WPM 4	Device	Max_Info_Frames	Passive_Client	DA WPM	13	N1 1
CMD WPM 5	BO	-	Passive_Client	DA BOP	0	N1 1

Address	Length	Write_Priority	Linked_Map_Descriptor
4	3	8	CMD WPM
8	4	10	CMD WPM
1	1	10	CMD WPM
1	1	10	CMD WPM
0	10	11	CMD WPM

For the example above:

- Configure a Map Descriptor for each Object and Property. These are component Map Descriptors forming part of the composite Map Descriptor CMD WPM shown in the first example.
- Set the function to Passive_Client.
- Set Linked_Map_Descriptor to the name of the active read or write Map Descriptor governing the transaction.

```
// ReadPropertyMultiple

Map_Descriptors
Map_Descriptor_Name   , Object_Type   , Property   , Function   , Data_Array_Name   , Data_Array_Index   , Node_Name
CMD RPM               , AO            , -          , Rdbc       , DA RPM            , 9                 , N1 1
```

```
, Address   , Length   , Write_Priority   , Linked_Map_Descriptor
, 1         , 2        , -                , -
```

```
Map_Descriptor_Name   , Object_Type   , Property          , Function          , Data_Array_Name   , Data_Array_Index   , Node_Name
CMD RPM 3             , Device        , Max_master        , Passive_Client    , DA RPM            , 0                 , N1 1
CMD RPM 4             , Device        , Max_Info_Frames   , Passive_Client    , DA RPM            , 1                 , N1 1
CMD RPM 2             , AO            , -                 , Passive_Client    , DA RPM            , 2                 , N1 1
CMD RPM 1             , AO            , -                 , Passive_Client    , DA RPM            , 6                 , N1 1
```

```
, Address   , Length   , Write_Priority   , Linked_Map_Descriptor
, 1         , 1        , -                , CMD RPM
, 1         , 1        , -                , CMD RPM
, 8         , 4        , -                , CMD RPM
, 4         , 3        , -                , CMD RPM
```

In the example above, these map descriptors form part of the read transaction defined by Map Descriptor CMD RPM above.

NOTES:

- For the Present_Value property it is permissible to set a Map Descriptor length >1. This will cause a read of the Present_Value property of consecutive BACnet objects of the type defined by this Map Descriptor.
- The number of properties that can be read or written at once is limited by the maximum APDU message length.
- Message segmentation is not supported.

7.2.3 Disabling Selected BACnet Services

Certain BACnet services can be disabled on a BACnet Server Node when specific requirements necessitate this:

COV – Certain BACnet MS/TP routers struggle to keep up with the bidirectional traffic that can result from many COVSubscribe requests in quick succession. In this case, better results might be obtained by turning COV off.

WritePropertyMultiple (WPM) – When the user requires writes to the BACnet Server to be acknowledged only once Client Side writes have succeeded, WPM is turned off.

ReadPropertyMultiple (RPM) – This disables read property multiple service. When the BACnet master sends an RPM request the FieldServer will respond with unsupported service.

These services can be disabled individually or together by listing them under the optional "Disabled_Services" parameter of the Node configuration section, separated by a space if more than one function is listed. When a service has been disabled, the BACnet Server Node will reject a corresponding service request with the reason “Unrecognized Service”.

The following values may be used under Disabled_Services:

- WPM (disable support for the WritePropertyMultiple service)
- RPM (disable support for the ReadPropertyMultiple service)
- COV (disable support for the SubscribeCOV service)

See below for examples:

Disable WPM only

```
Nodes
Node_Name   , Node_ID   , Protocol      , Disabled_Services
NN 01       , 1         , Bacnet_IP     , WPM
```

Disable COV only

```
Nodes
Node_Name   , Node_ID   , Protocol      , Disabled_Services
NN 02       , 2         , Bacnet_IP     , COV
```

Disable WPM and COV

```
Nodes
Node_Name   , Node_ID   , Protocol      , Disabled_Services
NN 03       , 3         , Bacnet_IP     , WPM COV
```

7.3 Virtual Router Configuration – Connect a Device to the Local BACnet Segment

When the FieldServer is configured with multiple BACnet Devices (Nodes) it creates an internal, virtual BACnet segment and acts as a virtual router in order to make all the Devices individually addressable via a single external MAC Address.

However, some 3rd party BACnet utilities depend on the presence of a BACnet Device directly on the local BACnet segment (i.e. not on the internal, virtual segment) to be able to discover the FieldServer. A local BACnet device can be created especially for this purpose as shown in the example below.

```
Nodes
Node_Name      , Node_ID   , Protocol      , Network_Location
Virtual_Dev_11 , 11        , Bacnet_MSTP   , Local_Segment
Virtual_Dev_12 , 12        , Bacnet_MSTP   , -
```

In the example, Virtual_Dev_11 will appear on the local BACnet segment, whereas Virtual_Dev_12 will appear on a remote BACnet segment identified by the Virtual_Network_Number assigned to the FieldServer. Only one Node may be configured to appear on the Local Segment.

7.4 Trending Using Trend Log Objects

The FieldServer BACnet driver has been updated to support trending using Trend Log objects. A Map Descriptor is required for each Trend log to be added to the device. The following parameters are specific to this application.

Section Title
Map Descriptors

Column Title	Function	Legal Values
Log_Data_Array	The Data Array containing the data which the Trend Log will log.	One of the specified Data Arrays
Log_Data_Array_Index	The position in the Data Array of the data value to be logged.	0 to (Data_Array_Length -1), -
Log_Buffer_Size*	The maximum number of records that will be kept in the log buffer. This should be limited due to the size constraints of the disk.	0 - 65535 log entries
Log_Interval*	How often a record will be stored, specified in seconds.	4 - 4294967295
Log_Enable*	This setting initializes the Enable property of the Trend Log at start-up. Logging only takes place while Enable is True (“Yes”). Note that this property is writable from BACnet, the user has the option to set it to “No” in the configuration file and allow the BACnet Client to set the Enable property as required.	**Yes**, No

NOTE:

- The Client will need to read the log every “Log Interval x Log Buffer Size” to prevent losing data (e.g. 500 records x 4 seconds / record = 2000 seconds = 33.3 Minutes).
- In this version, the Stop_When_Full property is not writable or configurable and is set to False.

```
//    Server Side Map Descriptors

Map_Descriptors
Map_Descriptor_Name   , Data_Array_Name   , Data_Array_Index   , Function   , Node_Name   , Data_Type   , Address
Trend Log 1           , DA_TREND          , 0                 , Passive    , BTU METER   , Trend_Log   , 0
Trend Log 2           , DA_TREND          , 1                 , Passive    , BTU METER   , Trend_Log   , 1
Trend Log 3           , DA_TREND          , 2                 , Passive    , BTU METER   , Trend_Log   , 2
Trend Log 4           , DA_TREND          , 3                 , Passive    , BTU METER   , Trend_Log   , 3
```

```
, Log_Data_Array   , Log_Data_Array_Index   , Log_Buffer_Size   , Log_Interval   , Log_Enable
, DA_AI_01         , 0                      , 500               , 4              , Yes
, DA_AI_01         , 1                      , 500               , 4              , Yes
, DA_AI_01         , 2                      , 500               , 4              , Yes
, DA_AI_01         , 3                      , 500               , 4              , Yes
```

7.4.1 Operating Statistics

The following stats were added for diagnosing the operation of the Trend Log, they can be viewed with FS-GUI.

BACnet Connection Stats	
Stat	**Description**
Pending Trend Log Writes	The number of writes waiting to be written to the disk.

BACnet Trend Log Map Descriptor Stats	
Stat	**Description**
Total Record Count	Number of entries recorded.
Record Count	Current number of entries within the log, this will stop at the value specified by the configuration.
Status Records Added	Number of status events.
Data Records Added	Number of data events.
Time Change Records Added	Number of time change events.

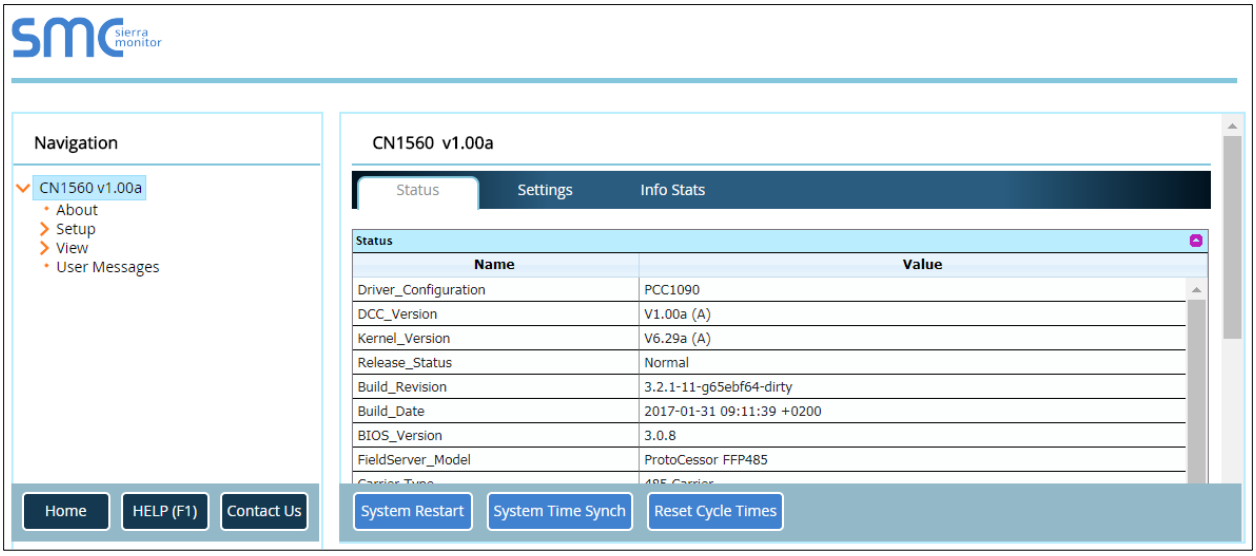
SPL LED
The blue LED will flash periodically when an event log is stored to the disk, this light should not be on all the time, otherwise we might be building up a back-log of outstanding writes.

7.4.2 Date and Time Synchronization

The FieldServer system time is not set on startup but depends on an external time synch to initialize its clock. This can be done via BACnet time synch and should be done whenever comms to the device have been down since this could indicate a restart. Since the time values are only correct after this synch it is best for the BACnet Client to perform a time synch before enabling trend logging.

If the device is configured to start logging by itself from restart the initial timestamps will start at Jan 1, 1970, and a time change record will be created on time synch.

NOTE: The FS-GUI can be used to set the system time if it connects to a FieldServer with an uninitialized system time. To synchronize time in FS-GUI, click the System Time Synch button on the bottom of the screen.



7.5 Writing to Custom Properties on Remote BACnet Devices

The Custom_Property and Tag_Type parameters allow the FieldServer to write to devices that implemented custom properties that are not part of the BACnet protocol specification.

The following example writes to custom property 650 with Tag_Type 1, 9 and 4 (which are examples vendor specific properties):

```
Map_Descriptors
Map_Descriptor_Name , Node_Name  , Function  , Data_Array_Name , Data_Array_Index , Data_Type
MultistateWR        , BCU_12     , WRBX      , DA_MV           , 1                , MV
BinaryWR            , BCU_12     , WRBX      , DA_BV           , 1                , BV
FloatWR             , BCU_12     , WRBX      , DA_AV           , 1                , AV
```

```
, Object_ID  , Custom_Property  , Tag_Type  , Length
, 1          , 650              , 1         , 1
, 1          , 650              , 9         , 1
, 1          , 650              , 4         , 1
```

7.6 Intrinsic Reporting for Life Safety Point

To use Intrinsic Reporting for a LifeSafetyPoint, it is necessary to classify each of the states as Normal, Alarm, Life-Safety Alarm or Trouble. This is done by adding another column to the Offset_Table, called Table_User_Value.

Each state is then classified by inserting one of the following values in its row:

0 = normal
1 = alarm
2 = fault
3 = life-safety alarm

Offset_Table

Offset_Table_Name	Table_String	Table_Index_Value	Table_User_Value
LSP_ALRM_TEXT	SYSTEM READY	0	0
LSP_ALRM_TEXT	PRE_ALARM	1	1
LSP_ALRM_TEXT	ALARM!!!	2	1
LSP_ALRM_TEXT	MAINTENANCE	3	2
LSP_ALRM_TEXT	LIFE-SAFETY-ALARM1	4	3
LSP_ALRM_TEXT	OFF-LINE	5	2
LSP_ALRM_TEXT	IN SERVICE	6	0
LSP_ALRM_TEXT	OTHER	7	1
LSP_ALRM_TEXT	LIFE-SAFETY-ALARM2	8	3
LSP_ALRM_TEXT	OFFLINE	9	2
LSP_ALRM_TEXT	LIFE-SAFETY-ALARM3	10	3

8 Troubleshooting

8.1 Debugging a BACnet Connection

- If duplicate Object_Instances are configured in the FieldServer, the second call of the Instance will overwrite the first one. This may cause a BACnet Object to be “lost.”
- If the Node Name configured on the BACnet Server Side of the configuration is not being indicated as the Device Name on the BACnet SCADA system, then the FieldServer is not communicating with the SCADA system. If the Device Object's name is being indicated, but the Present_Value shows question marks, then it is likely that the Client Side of the FieldServer is not communicating.
- Extra memory is required to store Map Descriptors that have the active/inactive text parameters specified. If the defaults are appropriate, do not specify these parameters. This will save memory and allow more Map Descriptors to be created.
- When a FieldServer/ProtoCessor is configured to support BACnet MS/TP virtual server nodes the firmware achieves this by implementing a virtual router and thus a network number is required so that it can expose its nodes to external devices as belonging to a specific network. (The default is network 5 if not specified). This network number is defined on the bridge’s connection section as Virtual_Network_Number. It must be unique for a site. Refer also to **Section 7.1.2**.

8.2 COV Configuration

- COV must be enabled if required for BACnet MS/TP. Refer to **Section 7.2.1** for more information.
- COV only works for BACnet Map Descriptors with length set to 1. If length is not specified, then it defaults to 1, so this is only a problem where length has been specified as greater than 1.

8.3 BACnet Specific Statistics

Stat	Description	Resolution
Link Control	A “who-is” link control message was sent or received.	It is normal to receive a few link control messages. If the number is higher than the transmit/receive messages; however, there may be a problem with lost communications.
Unsupported Properties	A request for an unsupported property was received.	This is not an error. BACnet clients often poll all properties of a particular object to determine which properties are supported.
Segmentation Not Supported	Data was requested but the response would have exceeded the maximum size of the APDU and could not be sent using an un-segmented message.	This is not an error - the BACnet client will use a different method to read data from the FieldServer.
Sequence Error	Invoke ID of a reply did not match the Invoke ID of the poll.	This message normally indicates a configuration error.
Write Access Denied	A write to an object was denied.	This typically happens when trying to write to an Input Object that is not Out-Of-Service. It is not possible to write to Input Objects.
Exception Errors	A BACnet Service was denied because it is not supported.	Consult the PIC statement to determine what services are supported.

8.4 BACnet Specific Error Messages

Message Description	Cause	Suggested Resolution
Bacnet DLL Temporarily out of receive buffers	The FieldServer was flooded with more BACnet packets than it could handle. This typically occurs when a workstation discovers a large network resulting in many Who-Is and I-Am broadcasts. The FieldServer will recover, but some timeout errors could result.	Take action only if the error is continuous, in which case the network load must be analyzed and corrected.

8.5 BACnet Error Response Decoding

BACnet reports errors in the following format:

```
T02> 10/22 02:57 HEXDUMP : ERROR_PDU
T02> 10/22 02:57 0x19d2d   50 97 0f 91 02 91 2a
T02> 10/22 02:57 BACnet -> Unexpected ERROR_PDU : err_class=2 err_code=42
```

These can be decoded using the tables below.

8.5.1 BACnet Error Class

Description	Value
Device	0
Object	1
Property	2
Resources	3
Security	4
Services	5

8.5.2 BACnet Error Codes for Error Class Object

Description	Value
Other	0
Authentication failed	1
Configuration in progress	2
Device busy	3
Dynamic creation not supported	4
File access denied	5
Incompatible security levels	6
Inconsistent parameters	7
Inconsistent selection criterion	8
Invalid data type	9
Invalid file access method	10
Invalid file start position	11
Invalid operator name	12
Invalid parameter data type	13
Invalid time stamp	14
Key generation error	15
Missing required parameter	16
No objects of specified type	17
No space for object	18
No space to add list element	19
No space to write property	20
Property is not a list	22
Object deletion not permitted	23
Object identifier already exists	24
Operational problem	25
Password failure	26
Read access denied	27
Security not supported	28
Service request denied	29
Timeout	30
Unknown object	31
Unknown property	32
Unknown vt class	34
Unknown vt session	35
Unsupported object type	36
Value out of range	37
Vt session already closed	38
Vt session termination failure	39
Write access denied	40
Character set not supported	41
Invalid array index	42
Invalid index	42

8.6 Rediscovering Offline Devices

The BACnet driver handles APDU retries internally and uses its own parameters, not the kernel parameters. The kernel timeout and retry values should be left to default.

The following parameters can be configured on the Client Node:

- **APDU_Timeout** – default value is 10s
- **APDU_Retries** – default value is 3

The FieldServer will only send requests to the remote device once it has discovered it using the Who-Is / I-Am process.

If a device does not respond and the APDU_Retries have been used up, the driver will revert to trying to discover the device using Who-Is requests.

NOTE: There is also a background process of rediscovering devices independently of the polling process. This occurs every 10 minutes.

8.7 Timeout Errors

NL TX Abort errors mean that the BACnet network layer was unable to transmit queued messages in time and gave up on them. Outgoing packets are programmed with a 10 second timeout. If packets have not been transmitted within this 10 second window, they are discarded and an error is recorded.

Timeout errors may be experienced on start-up if the network is not yet fully synchronized or if there are a high number of nodes on the network. The FieldServer can only transmit the I-Ams a few at a time, and only when it receives the MS/TP token.

It might help to give the MS/TP network some time to settle before attempting a discovery.

It may be advisable to discover the Nodes in a number of smaller Node ID ranges (say 10 at a time). The Node_ID's can be established from the configuration.

8.8 PDU Limit Reached Errors

A PDU limit reached error is returned when a device runs out of buffers for storing incoming or outgoing messages (PDU = protocol data unit). A possible scenario is when a high number of server nodes is configured and one or more global Who-Is requests is received. The FieldServer will then need to transmit a response for each one, but if the MS/TP network is busy it might not get enough airtime to send them all out. Another possibility is a buildup of unsent COV or Event Notifications.

9 Vendor Information

9.1 McQuay

McQuay Units are shipped with a default Device instance of the last 6 digits of the McQuay Serial number.

9.2 Liebert

Polling BACnet addresses that are not configured for Liebert systems may cause the connection to fail in older versions of Liebert. Contact Liebert supplier for more information.

9.3 Automated Logic Corporation

When an ALC module is powered up, and it does not detect valid BACnet MS/TP traffic on its MS/TP port, then the module goes into a terminal mode and MS/TP communications will not be initiated.

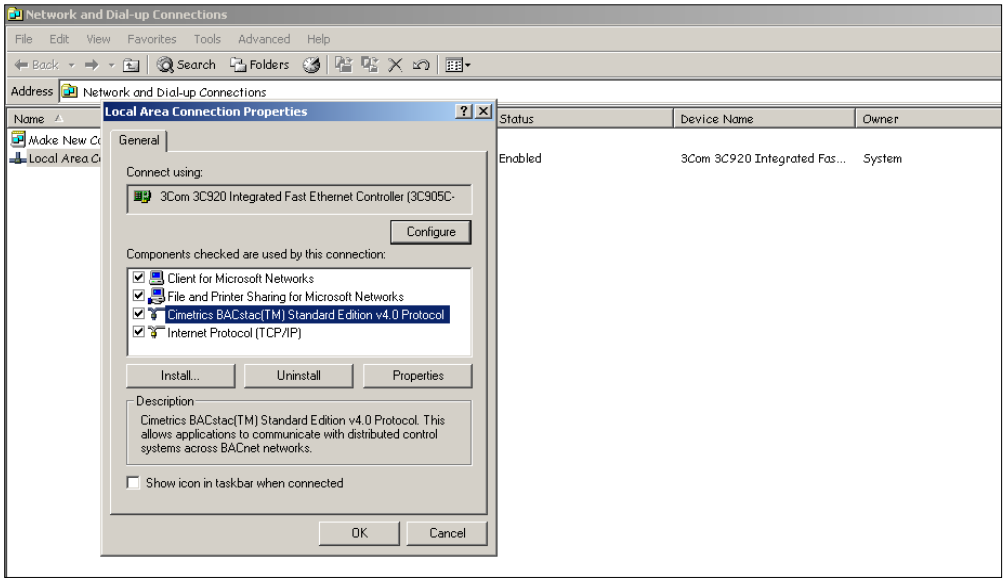
When connecting the FieldServer to an ALC BACnet MS/TP module, always start the FieldServer first. Wait until the RUN Led is flashing on the FieldServer before powering up the MS/TP module.

9.4 Honeywell EBI

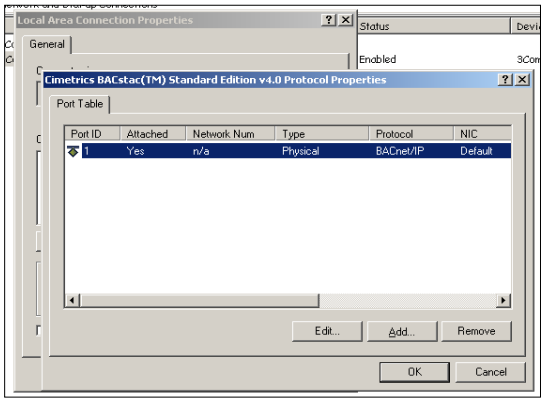
Honeywell EBI cannot process EventNotifications with ACK_Required set to 1. The ACK_Required property of Notification Class Map Descriptors configured for use with EBI must therefore be set to 0.

9.5 Using Cimetrics Explorer

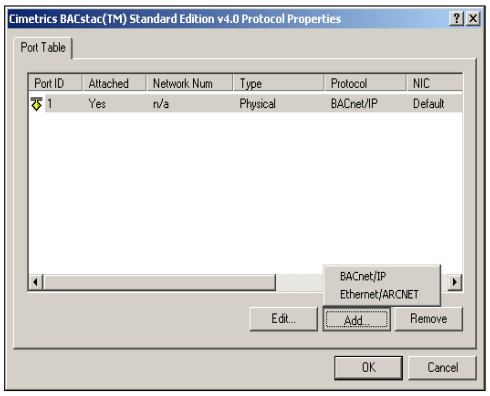
Cimetrics Explorer needs to be configured to use either BACnet Ethernet or BACnet/IP. This is not a setting in the actual Cimetrics BACnet Explorer, but is done in the Cimetrics BACstac Protocol that can be found in the Local Area Connection Properties as shown below:



Check the properties of this protocol to see what BACnet protocol is set on the Cimetrics Explorer.



To change the protocol, remove the current one and add a new one as shown below.



10 Reference

10.1 FieldServer Vendor ID

BACnet Vendor Name: Sierra Monitor Corporation
BACnet Vendor ID: 37

10.2 Object_Type Legal Values – Abbreviation Descriptions

Legal Value	Associated BACnet Number	Description
AI	0	ANALOG_INPUT
AO	1	ANALOG_OUTPUT
AV	2	ANALOG_VALUE
BI	3	BINARY_INPUT
BO	4	BINARY_OUTPUT
BV	5	BINARY_VALUE
MI	13	MULTI_STATE_INPUT
MO	14	MULTI_STATE_OUTPUT
MV	19	MULTI_STATE_VALUE
NC	15	NOTIFICATION_CLASS_OBJECT
LSP	21	LIFE_SAFETY_POINT
DEVICE	8	DEVICE

10.3 Configuring Binary Outputs

The BACnet specification defines the behavior of Binary Outputs such that the Present_Value property is treated as a set-point and is only expected to change as a result of BACnet write requests from upstream, and not as a result of values read from downstream. This may cause confusion in situations where a user wants to map BACnet Binary Output objects to corresponding Binary Output points on a downstream device, since the Present_Value property will not be updated to reflect the value of the downstream point. Instead, the Feedback_Value property may be monitored in order to know the state of the downstream point.

For alarms (EventNotifications) this implies that BACnet BO points will trigger a COMMAND_FAILURE alarm, which will trigger if the Feedback_Value (read from the downstream) differs from the Present_Value.

In order to use the Present_Value property to read and write to a downstream binary output object, configure a Binary Value object on the FieldServer.

10.4 Property Legal Values

Legal Value	Description
Object_Identifier	This property is a numeric code that is used to identify the object. It is unique within the BACnet Device that maintains it.
Object_List	Relevant to Device Object Type. This property is a BACnetARRAY of Object_Identifiers, one Object_Identifier for each object within the device that is accessible through BACnet services. An Object_Identifier is composed of Object Type and Object Instance and must be unique within a BACnet Device (for example: Object Type = Analog Input, Object Instance = 3).
Present_Value	This property contains the present value of the Input / Output / Value
Object_Name	Character string providing the name of a BACnet object. The set of characters used in the Object_Name is restricted to printable characters. The Object_Name is determined by the Map_Descriptor_Name.
Description	Character string describing a BACnet object. This can be defined by the user to give additional detail about the Object.
Out_of_Service	The Out_Of_Service property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the physical input that the object represents is not in service. This means that the Present_Value property is decoupled from the physical input and will not track changes to the physical input when the value of Out_Of_Service is TRUE. In addition, the Reliability property and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled from the physical input when Out_Of_Service is TRUE. While the Out_Of_Service property is TRUE, the Present_Value and Reliability properties may be changed to any value as a means of simulating specific fixed conditions or for testing purposes. Other functions that depend on the state of the Present_Value or Reliability properties shall respond to changes made to these properties while Out_Of_Service is TRUE, as if those changes had occurred in the physical input.
Event_State	The Event_State property, of type BACnetEventState, is included in order to provide a way to determine if this object has an active event state associated with it. If the object supports intrinsic reporting, then the Event_State property shall indicate the event state of the object. If the object does not support intrinsic reporting, then the value of this property shall be NORMAL. Other values: FAULT, OFF-NORMAL, HIGH-LIMIT, LOW-LIMIT, LIFE-SAFETY-ALARM.
Units	This property contains the units associated with the Present_Value property.
Reliability	The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the physical input in question is "reliable" as far as the BACnet Device or operator can determine and, if not, why. The following values are supported: NO_FAULT_DETECTED, UNRELIABLE_OTHER
Priority_Array	This property relates to Output and Value Object Types and is a read only array that contains prioritized commands or NULLs in the order of decreasing priority. The highest priority (lowest array index) with a non-NULL value is the active command.
State_Text	Relevant to Multistate Object Types: This property is a BACnetARRAY of character strings representing descriptions of all possible states of the Present_Value. The number of descriptions matches the number of states defined in the Number_Of_States property. The Present_Value, interpreted as an integer, serves as an index into the array.
Number_Of_States	Relevant to Multistate Object Types: this property sets the total number of states for which descriptions will be returned as defined under the State_Text property. The number of states will be determined automatically by the largest state number used when configuring the Offset Table. Refer to **Section 7.1.6**.

Legal Value	Description
Max_Master	Relevant to BACnet MS/TP Device Object Type: The Max_Master property, of type Unsigned, shall be present if the device is a master node on an MS/TP network. The value of Max_Master specifies the highest possible address for master nodes and shall be less than or equal to 127. If the Max_Master property is not writeable via BACnet services, its value shall be 127.
Max_Info_Frames	Relevant to BACnet MS/TP Device Object Type: The Max_Info_Frames property, of type Unsigned, shall be present if the device is a node on an MS/TP network. The value of Max_Info_Frames specifies the maximum number of information frames the node may send before it must pass the token. If Max_Info_Frames is not writable or otherwise user configurable, its value shall be 1.
Active_Text	Relevant to Binary Object Types: This property, of type CharacterString, characterizes the intended effect of the ACTIVE state of the Present_Value property from the human operator's viewpoint. The content of this string is a local matter, but it is intended to represent a human-readable description of the ACTIVE state. For example, if the physical input is a switch contact, then the Active_Text property might be assigned a value such as "Fan 1 On".
Inactive_Text	This property, of type CharacterString, characterizes the intended effect of the INACTIVE state of the Present_Value property from the human operator's viewpoint. The content of this string is a local matter, but it is intended to represent a human-readable description of the INACTIVE state. For example, if the physical input is connected to a switch contact, then the Inactive_Text property might be assigned a value such as "Fan 1 Off".
Description	A character string giving more information about the Object associated with the Present_Value property.
Firmware_revision	The firmware revision of the application.
Relinquish_Default	This property is the default value to be used for the Present_Value property when all command priority values in the Priority_Array property have a NULL value.
Event_Enable	The Event_Enable property is supported, but to save memory it is only instantiated it if a point is configured for alarming, i.e. if it is set up with a reference to a Notification Class Map Descriptor.
Min_Pres_Value	Specify the "Minimum Present Value" property.
Max_Pres_Value	Specify the "Maximum Present Value" property.
Mode	Operating Mode. Only 'ON' mode is supported.
Accepted_Modes	List of Operating Modes.
Silenced	Represents silenced state, but only "All Silenced" supported.
Operation_Expected	List of LifeSafety Operations, but only 'None' operation is supported.
Property_List	Returns the list of supported properties.

Legal Value	Description
Status_Flag	4 offsets of a data array will be used to contains four Boolean flags that indicate the general health of the object. Three of the four flags are associated with the values of other attributes within the same object. The four flags are as follows: • In Alarm – The In Alarm flag is False (0) if the Event State property value is Normal; otherwise, the In Alarm flag is True (1). • Fault – Fault flag is True (1) if the Reliability attribute is not reliable; otherwise, the Fault flag is False (0). • Overridden – Overridden flag is Logical 1 if the Present Value is decoupled from the hardware output because the Enabled attribute is False or the Out of Service attribute is True; otherwise, the Overridden flag is Logical 0. • Out of Service – Out of Service flag is True if the Out of Service attribute value is True; otherwise, the Out of Service flag is False. The following object types have a Status_Flags property: • Analog Input, Analog Output, Analog Value • Binary Input, Binary Output, Binary Value • Multistate Input, Multistate Output, Multistate Value • Trend Log

10.5 Supported BACnet Object Properties

Type	Supported Properties
Device Object	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Read	System_Status
Read	Vendor_Name
Read	Vendor_Identifier
Read	Model_Name
Read	Firmware_Revision
Read	Application_Software_Version
Optional	Location
Optional	Description
Read	Protocol_Version
Read	Protocol_Revision
Read	Protocol_Services_Supported
Read	Protocol_Object_Types_Supported
Read	Object_List
Read	Max_APDU_Length_Accepted
Read	Segmentation_Supported
Optional	Local_Time
Optional	Local_Date
Read	APDU_Timeout
Read	Number_Of_APDU_Retries
Optional	Max_Master
Optional	Max_Info_Frames
Read	Device_Address_Binding
Read	Database_Revision
Optional	Active_COV_Subscriptions
Read	Property_List
Analog Input	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Read	Present_Value
Optional	Description
Read	Status_Flags
Read	Event_State
Optional	Reliability
Read	Out_Of_Service
Read	Units
Optional	COV_Increment
Optional	Notification_Class
Optional	High_Limit
Optional	Low_Limit
Optional	Deadband
Optional	Limit_Enable
Optional	Event_Enable
Optional	Acked_Transitions
Optional	Notify_Type
Optional	Event_Time_Stamps
Read	Property_List
Analog Output	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Write	Present_Value
Optional	Description
Read	Status_Flags
Read	Event_State
Optional	Reliability
Read	Out_Of_Service

Type	Supported Properties
Read	Units
Optional	Min_Pres_Value
Optional	Max_Pres_Value
Read	Priority_Array
Read	Relinquish_Default
Optional	COV_Increment
Optional	Time_Delay
Optional	Notification_Class
Optional	High_Limit
Optional	Low_Limit
Optional	Deadband
Optional	Limit_Enable
Optional	Event_Enable
Optional	Acked_Transitions
Optional	Notify_Type
Optional	Event_Time_Stamps
Read	Property_List
Analog Value	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Read	Present_Value
Optional	Description
Read	Status_Flags
Read	Event_State
Optional	Reliability
Read	Out_Of_Service
Read	Units
Optional	Priority_Array
Optional	Relinquish_Default
Optional	COV_Increment
Optional	Time_Delay
Optional	Notification_Class
Optional	High_Limit
Optional	Low_Limit
Optional	Deadband
Optional	Limit_Enable
Optional	Event_Enable
Optional	Acked_Transitions
Optional	Notify_Type
Optional	Event_Time_Stamps
Optional	Min_Pres_Value
Optional	Max_Pres_Value
Read	Property_List
Binary Input	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Read	Present_Value
Optional	Description
Read	Status_Flags
Read	Event_State
Optional	Reliability
Read	Out_Of_Service
Read	Polarity
Optional	Inactive_Text
Optional	Active_Text
Optional	Time_Delay
Optional	Notification_Class
Optional	Alarm_Value
Optional	Event_Enable

Type	Supported Properties
Optional	Acked_Transitions
Optional	Notify_Type
Optional	Event_Time_Stamps
Read	Property_List
Binary Output	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Write	Present_Value
Optional	Description
Read	Status_Flags
Read	Event_State
Optional	Reliability
Read	Out_Of_Service
Read	Polarity
Optional	Inactive_Text
Optional	Active_Text
Read	Priority_Array
Read	Relinquish_Default
Optional	Time_Delay
Optional	Notification_Class
Optional	Feedback_Value
Optional	Event_Enable
Optional	Acked_Transitions
Optional	Notify_Type
Optional	Event_Time_Stamps
Read	Property_List
Binary Value	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Read	Present_Value
Optional	Description
Read	Status_Flags
Read	Event_State
Optional	Reliability
Read	Out_Of_Service
Optional	Inactive_Text
Optional	Active_Text
Optional	Priority_Array
Optional	Relinquish_Default
Optional	Time_Delay
Optional	Notification_Class
Optional	Alarm_Value
Optional	Event_Enable
Optional	Acked_Transitions
Optional	Notify_Type
Optional	Event_Time_Stamps
Read	Property_List
Multi State Input	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Read	Present_Value
Optional	Description
Read	Status_Flags
Read	Event_State
Optional	Reliability
Read	Out_Of_Service
Read	Number_Of_States
Optional	State_Text
Optional	Time_Delay
Optional	Notification_Class

Type	Supported Properties
Optional	Alarm_Values
Optional	Fault_Values
Optional	Event_Enable
Optional	Acked_Transitions
Optional	Notify_Type
Optional	Event_Time_Stamps
Read	Property_List
Multi State Output	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Write	Present_Value
Optional	Description
Optional	Device_Type
Read	Status_Flags
Read	Event_State
Optional	Reliability
Read	Out_Of_Service
Read	Number_Of_States
Optional	State_Text
Read	Priority_Array
Read	Relinquish_Default
Optional	Time_Delay
Optional	Notification_Class
Optional	Feedback_Value
Optional	Event_Enable
Optional	Acked_Transitions
Optional	Notify_Type
Optional	Event_Time_Stamps
Read	Property_List
Multi State Value	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Read	Present_Value
Optional	Description
Read	Status_Flags
Read	Event_State
Optional	Reliability
Read	Out_Of_Service
Read	Number_Of_States
Optional	State_Text
Optional	Priority_Array
Optional	Relinquish_Default
Optional	Time_Delay
Optional	Notification_Class
Optional	Alarm_Values
Optional	Fault_Values
Optional	Event_Enable
Optional	Acked_Transitions
Optional	Notify_Type
Optional	Event_Time_Stamps
Read	Property_List
Notification Class Object	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Read	Notification_Class
Read	Priority
Read	Ack_Required
Read	Recipient_List
Read	Property_List

Type	Supported Properties
Trend Log	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Optional	Description
Write	Enable
Optional	Log_Interval
Read	Stop_When_Full
Read	Buffer_Size
Read	Log_Buffer
Write	Record_Count
Read	Total_Record_Count
Read	Logging_Type
Read	Status_Flags
Optional	Notification_Threshold
Optional	Records_Since_Notification
Optional	Last_Notify_Record
Read	Event_State
Optional	Notification_Class
Optional	Event_Enable
Optional	Acked_Transitions
Optional	Notify_Type
Optional	Event_Time_Stamps
Read	Property_List

Type	Supported Properties
Life Safety Point	
Read	Object_Identifier
Write	Object_Name
Read	Object_Type
Read	Present_Value
Optional	Description
Read	Status_Flags
Read	Event_State
Optional	Reliability
Read	Out_Of_Service
Optional	Time_Delay
Optional	Notification_Class
Optional	Alarm_Values
Optional	Fault_Values
Optional	Event_Enable
Optional	Acked_Transitions
Optional	Notify_Type
Optional	Event_Time_Stamps
Write	Mode
Read	Accepted_Modes
Read	Silenced
Write	Operation_Expected
Read	Property_List

10.6 Units

Unit	Variation 1	Variation 2	Type
amperes	amps	A	Electrical
ampere-seconds			Energy
amperes-per-meter			Electrical
amperes-per-square-meter			Electrical
ampere-square-hours			Energy
ampere-square-meters			Electrical
bars			Pressure
becquerels			Other
btus			Energy
btus-per-hour			Power
btus-per-pound			Enthalpy
btus-per-pound-dry-air			Enthalpy
candelas			Light
candelas-per-square-meter			Light
centimeters			Length
centimeters-of-mercury			Pressure
centimeters-of-water			Pressure
cubic-feet			Volume
cubic-feet-per-day			Volumetric Flow
cubic-feet-per-hour			Volumetric Flow
cubic-feet-per-minute			Volumetric Flow
cubic-feet-per-second			Volumetric Flow
cubic-meters			Volume
cubic-meters-per-day			Volumetric Flow
cubic-meters-per-hour			Volumetric Flow
cubic-meters-per-minute			Volumetric Flow
cubic-meters-per-second			Volumetric Flow
currency1			Currency
currency10			Currency
currency2			Currency
currency3			Currency
currency4			Currency
currency5			Currency
currency6			Currency
currency7			Currency
currency8			Currency
currency9			Currency
cycles-per-hour			Frequency
cycles-per-minute			Frequency
days			Time
decibels			Electrical
decibels-a			Other
decibels-millivolt			Electrical
decibels-volt			Electrical
degree-days-Celsius			Temperature
degree-days-Fahrenheit			Temperature
degrees-angular			Other
degrees-Celsius	Deg-C	Deg_C	Temperature
degrees-Celsius-per-hour			Other
degrees-Celsius-per-minute			Other
degrees-Fahrenheit	Deg-F	Deg_F	Temperature
degrees-Fahrenheit-per-hour			Other
degrees-Fahrenheit-per-minute			Other
degrees-Kelvin	Deg-K	Deg_K	Temperature
degrees-Kelvin-per-hour			Temperature
degrees-Kelvin-per-minute			Temperature
degrees-phase			Electrical
delta-degrees-Fahrenheit			Temperature
delta-degrees-Kelvin			Temperature
farads			Electrical
feet			Length
feet-per-minute			Velocity
feet-per-second			Velocity
foot-candles			Light

Unit	Variation 1	Variation 2	Type
grams			Mass
grams-of-water-per-kilogram-dry-air			Humidity
grams-per-cubic-centimeter			Other
grams-per-cubic-meter			Other
grams-per-gram			Other
grams-per-kilogram			Other
grams-per-liter			Other
grams-per-milliliter			Other
grams-per-minute			Mass Flow
grams-per-second			Mass Flow
grams-per-square-meter			Other
gray			Other
hectopascals			Pressure
henrys			Electrical
hertz	Hz		Frequency
horsepower	HP		Power
hours			Time
hundredths-seconds			Time
imperial-gallons			Volume
imperial-gallons-per-minute			Volumetric Flow
inches			Length
inches-of-mercury			Pressure
inches-of-water			Pressure
joule-per-hours			Power
joules			Energy
joule-seconds			Other
joules-per-cubic-meter			Other
joules-per-degree-Kelvin			Entropy
joules-per-kilogram-degree-Kelvin			Entropy
joules-per-kilogram-dry-air			Enthalpy
kilobecquerels			Other
kilo-btus			Energy
kilo-btus-per-hour			Power
kilograms	kg		Mass
kilograms-per-cubic-meter			Other
kilograms-per-hour			Mass Flow
kilograms-per-kilogram			Other
kilograms-per-minute			Mass Flow
kilograms-per-second			Mass Flow
kilohertz	kHz		Frequency
kilohms			Electrical
kilojoules			Energy
kilojoules-per-degree-Kelvin			Entropy
kilojoules-per-kilogram			Energy
kilojoules-per-kilogram-dry-air			Enthalpy
kilometers			Length
kilometers-per-hour			Velocity
kilopascals	Kpa		Pressure
kilovolt-ampere-hours			Energy
kilovolt-ampere-hours-reactive			Energy
kilovolt-amperes	kilovolt-amps	KVA	Electrical
kilovolt-amperes-reactive	KVAR		Electrical
kilovolts			Electrical
kilowatt-hours	kWh		Energy
kilowatt-hours-per-square-foot			Other
kilowatt-hours-per-square-meter			Other
kilowatt-hours-reactive			Energy
kilowatts	kW		Power
liters			Volume
liters-per-hour			Volumetric Flow
liters-per-minute			Volumetric Flow
liters-per-second			Volumetric Flow
lumens			Light
luxes			Light
megabecquerels			Other

Unit	Variation 1	Variation 2	Type
mega-btus			Energy
megahertz	MHz		Frequency
megajoules			Energy
megajoules-per-degree-Kelvin			Entropy
megajoules-per-kilogram-dry-air			Enthalpy
megajoules-per-square-foot			Other
megajoules-per-square-meter			Other
megavolt-ampere-hours			Energy
megavolt-ampere-hours-reactive			Energy
megavolt-amperes	megavolt-amps		Electrical
megavolt-amperes-reactive	MVAR		Electrical
megavolts			Electrical
megawatt-hours	MWh		Energy
megawatt-hours-reactive			Energy
megawatts	MW		Power
megohms			Electrical
meters			Length
meters-per-hour			Velocity
meters-per-minute			Velocity
meters-per-second			Velocity
meters-per-second-per-second			Acceleration
micrograms-per-cubic-meter			Other
micrograms-per-liter			Other
microgray			Other
micrometers			Length
microsiemens			Electrical
microsieverts			Other
microsieverts-per-hour			Other
miles-per-hour			Velocity
milliamperes	milliamps		Electrical
millibars			Pressure
milligrams			Mass
milligrams-per-cubic-meter			Other
milligrams-per-gram			Other
milligrams-per-kilogram			Other
milligrams-per-liter			Other
milligray			Other
milliliters			Volume
milliliters-per-second			Volumetric Flow
millimeters			Length
millimeters-of-mercury			Pressure
millimeters-of-water			Pressure
millimeters-per-minute			Velocity
millimeters-per-second			Velocity
milliohms			Electrical
million-standard-cubic-feet-per-day			Volumetric Flow
million-standard-cubic-feet-per-minute			Volumetric Flow
millirems			Other
millirems-per-hour			Other
milliseconds			Time
millisiemens			Electrical
millisieverts			Other
millivolts			Electrical
milliwatts			Power
minutes			Time
minutes-per-degree-kelvin			Other
mole-percent			Other
months			Time
nanograms-per-cubic-meter			Other
nephelometric-turbidity-unit			Other
newton			Force
newton-meters			Torque
newton-seconds			Other
newtons-per-meter			Other
no-units	No_Units	None	Other

Unit	Variation 1	Variation 2	Type
ohm-meters			Electrical
ohm-meter-squared-per-meter			Electrical
ohms			Electrical
parts-per-billion			Other
parts-per-million			Other
pascals			Pressure
pascal-seconds			Other
percent			Other
percent-obscuration-per-foot			Other
percent-obscuration-per-meter			Other
percent-per-second			Other
percent-relative-humidity	% RH; %RH	Percent RH; Percent RH	Humidity
per-hour			Frequency
per-mille			Other
per-minute			Other
per-second			Other
pH			Other
pounds-force-per-square-inch	PSI	pounds-force-per-sq-inch	Pressure
pounds-mass			Mass
pounds-mass-per-day			Volumetric Flow
pounds-mass-per-hour			Mass Flow
pounds-mass-per-minute			Mass Flow
pounds-mass-per-second			Mass Flow
power-factor	PF		Electrical
psi-per-degree-Fahrenheit			Other
radians			Other
radians-per-second			Other
revolutions-per-minute			Other
seconds	Secs	S	Time
siemens			Electrical
siemens-per-meter			Electrical
sieverts			Other
square-centimeters			Area
square-feet			Area
square-inches			Area
square-meters			Area
square-meters-per-Newton			Other
standard-cubic-feet-per-day			Volumetric Flow
teslas			Electrical
therms			Energy
thousand-cubic-feet-per-day			Volumetric Flow
thousand-standard-cubic-feet-per-day			Volumetric Flow
ton-hours			Energy
tons			Mass
tons-per-hour			Mass Flow
tons-refrigeration			Power
us-gallons	Gallons		Volume
us-gallons-per-hour			Volumetric Flow
us-gallons-per-minute	GPM		Volumetric Flow
volt-ampere-hours			Energy
volt-ampere-hours-reactive			Energy
volt-amperes	Volt-Amps	VA	Electrical
volt-amperes-reactive	VAR		Electrical
volts	voltage		Electrical
volts-per-degree-Kelvin			Electrical
volts-per-meter			Electrical
volt-square-hours			Energy
watt-hours	wH		Energy
watt-hours-per-cubic-meter			Other
watt-hours-reactive			Energy
watts	W		Power
watts-per-meter-per-degree-Kelvin			Other
watts-per-square-foot			Light
watts-per-square-meter			Light
watts-per-square-meter-degree-kelvin			Other

Unit	Variation 1	Variation 2	Type
webers			Electrical
weeks			Time
years			Time