This document shows examples of BACnet messages as they might appear on an ARCNET or BACnet Point-to-Point (PTP) link. The examples describe the derivation of the encoded values from the service definitions and encoding rules contained in ANSI/ASHRAE 135-1995, the BACnet standard (referred to herein as “BACnet” or “ANSI BACnet”.)

References to page numbers and to numbered “clauses” are to the BACnet standard.

References to numbered “sections” are to sections of this document.

To construct a BACnet request, proceed as follows
1. Find the appropriate service description in clauses 13 to 17. This will describe in English the parameters to be used.
2. Find the ASN.1 definition of the service and its parameters in clause 21. This is the formal definition of the service request.
3. Use clause 20.1 to encode the APDU header.
4. Use clause 20.2 to encode the tagged portion of the APDU.
5. Use clause 6 to define and encode the network layer header.
6. Use clause 7 to 11 to define and encode the datalink and physical layer information.

The examples which follow show this process.

1.0 A Quick Course in ASN.1

This section is intended to assist readers unfamiliar with the ASN.1 notation in reading BACnet clause 21. The reader is also referred to ISO/IEC 8824, which formally defines ASN.1, and to BACnet clause 20.2 which defines BACnet’s encoding of ASN.1. Experts may quibble with the descriptions given here. The intention of this section is merely tutorial.

name ::= expression
    defines “name” in terms of the expression.

SEQUENCE {} means that one of each item in the bracketed construct is included, unless it is marked with the keyword OPTIONAL, in which case it may or may not be included.

SEQUENCE OF item means a list of zero or more items. The item may be another ASN.1 definition, or it may be a SEQUENCE {}. Note the difference between the one-shot SEQUENCE and the list defined by SEQUENCE OF.

CHOICE {} means a selection of one item of the bracketed construct is included.
ENUMERATED {name (value),...}
defines an enumeration, where a list of "names" are assigned values shown in parenthesis.

[n]
A number in square brackets denotes a context tag, which will generally appear in the encoding
(the exception being tags defined in the APDU header productions)

ABSTRACT-SYNTAX.&Type
Indicates that a CHOICE of any tagged datatype is legal.

2.0 General Notes

BACnet Point to Point (PTP), as described in BACnet clause 10, is a full duplex Datalink protocol. Thus, frames may be sent and received simultaneously. Requester implementations must be prepared to receive Data and Heartbeat frames at any time. In particular, Data frames must be acknowledged via Datalink acknowledgment in a timely fashion, even if the content of the Data frame is not expected by the upper layers. If this is not done, the peer device will be unable to send subsequent Data frames which may contain expected data. Frames containing PDUs not expected by the upper layers should be dealt with as described in BACnet clause 5. Appropriate actions will generally be either to discard the PDU, or to respond to it with a BACnet-Abort-PDU.

ARCNET, as described in BACnet clause 8, is a peer to peer Datalink protocol. Requester implementations must be prepared to receive frames at any time. Frames containing PDUs not expected by the upper layers should be dealt with as described in BACnet clause 5. Appropriate actions will generally be either to discard the PDU, or to respond to it with a BACnet-Abort-PDU.

Attempts to implement a requester as the master of a master/slave pair will not be successful unless the above concerns are addressed. In particular, implementations must be prepared to receive various broadcast messages from time to time. For example, the Tracer Summit Workstation will broadcast Who-Is messages on a periodic basis. Summit BCUs may issue such messages as well.

3.0 Establishing a PTP Connection

This section does not apply to ARCNET interfaces.

To establish a PTP connection with the BCU, a PTP REQUESTER sends "BACnet"

0x42 0x41 0x43 0x6E 0x65 0x74 0x0D

The PTP BCU responds with Connect Request

0x55 0xFF 0x0C 0x00 0x0D
The PTP REQUESTER responds with Connect Response

```
0x55  Preamble
0xFF  
0x0D  Frame Type 13: Connect Response
0x00  Length = 0
0x00  
0xDC  Header CRC
```

The PTP REQUESTER sends Heartbeat XON

```
0x55  Preamble
0xFF  
0x01  Frame Type 1: Heartbeat XON
0x00  Length = 0
0x00  
0x76  Header CRC
```

The PTP BCU sends Heartbeat XON

```
0x55  Preamble
0xFF  
0x01  Frame Type 1: Heartbeat XON
0x00  Length = 0
0x00  
0x76  Header CRC
```

If the BCU does not transmit any other frames for 15 seconds, it will send a Heartbeat frame. Requester implementations should be prepared to receive Heartbeat frames at any time.

The PTP REQUESTER sends I-Am-Router-To-Network (required by BACnet, but optional as far as the BCU is concerned.) This message is described in clauses 6.4.2, 6.6.3.3, and 6.7. The network layer header is described in clause 6.2.

**Note:** the correct contents of the network header for this message are currently under discussion. The issue is whether or not a DNET and DADR (and possible SNET and SADR) should be present. Implementations should be prepared for all combinations of network header.

```
0x55  Preamble
0xFF  
0x02  Frame Type 2: Data 0
0x00  Length = 9
0x09  
0xDA  header CRC
NPDU follows
0x01  Network version 1
0x88  Network control octet
      Bit7  = 1 "network message"
      Bit6  = 0 not used
      Bit5  = 0 DNET absent
      Bit4  = 0 not used
      Bit3  = 1 SNET present
      Bit2  = 0 no reply expected
      Bit1,0 = 00 normal priority
```
4.0 Unconfirmed Services (Who-Is and I-Am)

Who-Is and I-Am may be used to dynamically bind Device IDs (Device Object Identifiers) to addresses. Tracer Summit uses Who-Is and I-Am in this fashion. Since these messages are globally broadcast, they will be routed between ARCNET and PTP. Thus, requester implementations must be prepared to see these messages at any time.

4.1 Who-Is

The Who-Is and I-Am services are described in clause 16.9. The ASN.1 definition of these services is in clause 21. These are Unconfirmed services. On page 348, BACnet-Unconfirmed-Request-PDU is defined to be a service-choice followed by a service-request. The comment states that tags are not used in the header encoding. Header encoding for BACnet-Unconfirmed-Request-PDU is shown in clause 20.1.3.3.

The service-choice is defined to be of type BACnetUnconfirmedServiceChoice, which is defined on page 359 as an enumeration. The value for who-Is is 8 decimal.

The service-request is defined to be of type BACnetUnconfirmedServiceRequest, which is defined on page 359. The CHOICE for who-Is is the production Who-Is-Request, which is defined on page 361, as a SEQUENCE consisting of an OPTIONAL pair of unsigned integers which specify the range of devices which should respond. The encoding of an unsigned integer is defined in clause 20.2.4. In this example, the integers are absent, meaning (according to clause 16.9.2) that all devices should respond with I-Am.

4.1.1 Example of Who-Is on PTP

In the example which follows, the network header specifies a global broadcast (DNET 0xffff, DADR of length 0). The network header specifies the "return address" of the requester as SNET and SADR. Neither SNET nor SADR may be "broadcast", and SNET may not be the same as DNET.
PTP REQUESTER sends

<table>
<thead>
<tr>
<th>Hex</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x55</td>
<td>Preamble</td>
</tr>
<tr>
<td>0xFF</td>
<td></td>
</tr>
<tr>
<td>0x03</td>
<td>Frame Type 3: Data 1</td>
</tr>
<tr>
<td>0x00</td>
<td>Length = 12</td>
</tr>
<tr>
<td>0x0C</td>
<td></td>
</tr>
<tr>
<td>0x41</td>
<td>Header CRC</td>
</tr>
<tr>
<td></td>
<td>NPDU follows</td>
</tr>
<tr>
<td>0x01</td>
<td>Network version 1</td>
</tr>
<tr>
<td>0x28</td>
<td>Network control octet</td>
</tr>
<tr>
<td></td>
<td>Bit7 = 0  &quot;BACnet APDU&quot;</td>
</tr>
<tr>
<td></td>
<td>Bit6 = 0  not used</td>
</tr>
<tr>
<td></td>
<td>Bit5 = 1  DNET present</td>
</tr>
<tr>
<td></td>
<td>Bit4 = 0  not used</td>
</tr>
<tr>
<td></td>
<td>Bit3 = 1  SNET present</td>
</tr>
<tr>
<td></td>
<td>Bit2 = 0  no reply expected</td>
</tr>
<tr>
<td></td>
<td>Bit1,0 = 00 normal priority</td>
</tr>
<tr>
<td>0xFF</td>
<td>DNET 0xFFFF (broadcast)</td>
</tr>
<tr>
<td>0xFF</td>
<td></td>
</tr>
<tr>
<td>0x00</td>
<td>DLEN 0 (denotes broadcast)</td>
</tr>
<tr>
<td>0x00</td>
<td>SNET 0x0002</td>
</tr>
<tr>
<td>0x02</td>
<td></td>
</tr>
<tr>
<td>0x01</td>
<td>Source address length 1 octet</td>
</tr>
<tr>
<td>0x15</td>
<td>SADR 0x15</td>
</tr>
<tr>
<td>0xFF</td>
<td>Hop Count 255</td>
</tr>
<tr>
<td></td>
<td>APDU follows</td>
</tr>
<tr>
<td>0x10</td>
<td>APDU type 1: Unconfirmed Request (with DLE escape)</td>
</tr>
<tr>
<td>0x90</td>
<td></td>
</tr>
<tr>
<td>0x08</td>
<td>Service Choice 8: Who Is</td>
</tr>
<tr>
<td></td>
<td>No Tagged Service parameters for this example</td>
</tr>
<tr>
<td>0xE2</td>
<td>Data CRC</td>
</tr>
<tr>
<td>0x56</td>
<td></td>
</tr>
</tbody>
</table>

PTP BCU acknowledges

<table>
<thead>
<tr>
<th>Hex</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x55</td>
<td>Preamble</td>
</tr>
<tr>
<td>0xFF</td>
<td></td>
</tr>
<tr>
<td>0x07</td>
<td>Frame Type 7: Data Ack 1 XON</td>
</tr>
<tr>
<td>0x00</td>
<td>Length = 0</td>
</tr>
<tr>
<td>0x00</td>
<td></td>
</tr>
<tr>
<td>0x23</td>
<td>Header CRC</td>
</tr>
</tbody>
</table>

4.1.2 Example of Who-Is on ARCNET

In the example which follows, the network header specifies a global broadcast (DNET 0xffff, DADR of length 0). The network header contains no source network or address for the requester, as the requester is assumed in this example to reside on the same ARCNET network where the request is observed. The CRC is not shown, as it is inserted and removed by the ARCNET interface hardware.

ARCNET REQUESTER sends

<table>
<thead>
<tr>
<th>Hex</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x15</td>
<td>ARCNET Source ID (assume hex 15)</td>
</tr>
<tr>
<td>0x00</td>
<td>ARCNET Destination ID (broadcast)</td>
</tr>
</tbody>
</table>
0x0C  ARCNET Frame Length  =  12
0xCD  ARCNET System Code for BACnet
0x82  8802-2 SSAP (reserved by the IEEE for BACnet)
0x82  8802-2 DSAP (reserved by the IEEE for BACnet)
0x03  8802-2 caution: UI
                    NPDU follows
0x01  Network version 1
0x20  Network control octet
        Bit7  =  0  "BACnet APDU"
        Bit6  =  0  not used
        Bit5  =  1  DNET present
        Bit4  =  0  not used
        Bit3  =  0  SNET not present
        Bit2  =  0  no reply expected
        Bit1,0 = 00  normal priority
0xFF  DNET 0xFFFF (broadcast)
0xFF
0x00  DLEN 0 (denotes broadcast)
0xFF  Hop Count 255
                    APDU follows
0x10  APDU type 1: Unconfirmed Request
0x08  Service Choice 8: Who Is
                    No Tagged Service parameters for this example

### 4.2 I-Am

When the BCU receives a Who-Is whose device instance range limits include the BCU’s Device Identifier, or a Who-Is which does not contain device instance range limits, the BCU will respond by broadcasting an I-Am.

The I-Am service is described in clause 16.9. The ASN.1 definition is in clause 21. This is an Unconfirmed service. On page 348, BACnet-Unconfirmed-Request-PDU is defined to be a service-choice followed by a service-request. The comment states that tags are not used in the header encoding. Header encoding for BACnet-Unconfirmed-Request-PDU is shown in clause 20.1.3.3.

The service-choice is defined to be of type BACnetUnconfirmedServiceChoice, which is defined on page 359 as an enumeration. The value for I-Am is 0.

The service-request is defined to be of type BACnetUnconfirmedServiceRequest, which is defined on page 359. The CHOICE for I-Am is the production I-Am-Request, which is defined on page 360, as a SEQUENCE consisting of a BACnetObjectIdentifier containing the device's identifier, an unsigned integer specifying the maximum APDU length receivable by the device, an enumeration specifying the device's segmentation abilities, and an unsigned integer specifying the device's vendor. The encoding of a BACnetObjectIdentifier is defined in clause 20.2.14. In this example, the BCU's Device Object has Object Identifier objectType = 8 (Device), instance = 1. The encoding of an unsigned integer is defined in clause 20.2.4. In this example, the BCU's maximum APDU length is 480. The encoding of an enumeration is defined in clause 20.2.11. In this example, the BCU can both send and receive segmented messages, so the value is "segmented-both". The value of the BCU's vendor identifier is 02 (Trane).

Note that the network header for both ARCNET and PTP specifies a global broadcast destination.
4.2.1 Example of I-Am on PTP

PTP BCU sends

0x55  Preamble
0xFF
0x02  Frame Type 2: Data 0
0x00  Length = 24
0x18
0xD5  Header CRC
      NPDU follows
0x01  Network version 1
0x28  Network control octet
      Bit7  = 0  "BACnet APDU"
      Bit6  = 0  not used
      Bit5  = 1  DNET present
      Bit4  = 0  not used
      Bit3  = 1  SNET present
      Bit2  = 0  no reply expected
      Bit1,0 = 00 normal priority
0xFF  DNET 0xFFFF (broadcast)
0xFF
0x00  DLEN 0 (denotes broadcast)
0x00  SNET 0x0001
0x01
0x01  Source address length 1 octet
0x01  SADR 0x01
0xFF  Hop Count 255
      APDU follows
0x10  APDU type Unconfirmed Request (with DLE escape)
0x90
0x00  Service Choice 0: I Am
      Tagged Service parameters follow
0xC4  Application Tag 12 (Object Identifier), length 4
0x02  Object type 8 (Device), Object Instance 1
0x00
0x00
0x01
0x22  Application Tag 2 (Unsigned Integer), length 2
0x01  Maximum APDU size: 480
0xE0
0x91  Application Tag 9 (Enumerated), length 1
      segmentation: segmented-both
0x00
0x21  Application Tag 2 (Unsigned Integer), length 1
      value 2 = vendorID Trane
0xCB  Data CRC
0x1A

PTP REQUESTER acknowledges

0x55  Preamble
0xFF
0x06  Frame Type 6: Data Ack XON
0x00  Length = 0
0x00
0xBB  Header CRC
4.2.2 Example of I-Am on ARCNET

**ARCNET BCU sends**

0x01  ARCNET Source ID (assume hex 01)
0x00  ARCNET Destination ID (broadcast)
0x18  ARCNET Frame Length = 24
0xCD  ARCNET System Code for BACnet
0x82  8802-2 SSAP (value assigned by IEEE to BACnet)
0x82  8802-2 DSAP
0x83  8802-2 function: UI
     NPDU follows
0x01  Network version 1
0x20  Network control octet
     Bit7 = 0  "BACnet APDU"
     Bit6 = 0  not used
     Bit5 = 1  DNET present
     Bit4 = 0  not used
     Bit3 = 0  SNET not present
     Bit2 = 0  no reply expected
     Bit1,0 = 00  normal priority
0xFF  DNET 0xFFFF (broadcast)
0xFF
0x00  DLEN 0 (denotes broadcast)
0xFF  Hop Count 255
     APDU follows
0x10  APDU type Unconfirmed Request
0x00  Service Choice 0:  I Am
     Tagged Service parameters follow
0xC4  Application Tag 12 (Object Identifier), length 4
0x02  Object type 8 (Device), Object Instance 1
0x00
0x00
0x01
0x22  Application Tag 2 (Unsigned Integer), length 2
0x01  Maximum APDU size: 480
0xE0
0x91  Application Tag 9 (Enumerated), length 1
     segmentation: segmented-both
0x00
0x21  Application Tag 2 (Unsigned Integer), length 1
0x02  value 2 = vendorID Trane

5.0 ReadProperty - Single Property of Analog Input

Example of ReadProperty for the present-value property of Analog Input Object number 1.
5.1 ReadProperty - Request

The Analog Input Object and its properties are described in clause 12.1. The ReadProperty service is described in clause 15.5. The ASN.1 definition of ReadProperty is in clause 21. This is a Confirmed service. On page 348, BACnet-Confirmed-Request-PDU is defined to be a SEQUENCE of items. The comment states that tags are not used in the header encoding.

Header encoding for BACnet-Confirmed-Request-PDU is shown in clause 20.1.2. Values may be determined as follows:

- **segmented-message**
  BOOLEAN, TRUE if this is a segmented message, FALSE otherwise. FALSE in this example.

- **more-follows**
  BOOLEAN, TRUE if this is a segmented message, and this is not the final segment. FALSE otherwise. FALSE in this example.

- **segmented-message-accepted**
  BOOLEAN, TRUE if the requester will accept a segmented BACnet-Complex-ACK-PDU as a response, FALSE otherwise. TRUE in this example.

  The BCU will accept segmented BACnet-Confirmed-Request-PDUs, and is capable of returning segmented BACnet-ComplexACK-PDUs. Support of segmentation by requesters is optional.

- **maximum-APDU-size-accepted**
  Specifies the maximum size of a single PDU or PDU segment. In this example, an enumerated value of 3, denoting a maximum of 480 octets. This is the maximum value allowed for BACnet PDUs on ARCNET and PTP, and is the maximum value accepted by the BCU.

- **invokeID**
  "Serial number" used by the requester to associate the response with the request. All segments of a segmented request must have the same invokeID. The invokeID should be incremented for each new BACnet-Confirmed-Request-PDU. This facilitates debugging, as it allows responses to be correlated with the requests which caused them. The invokeID will increment in each example which follows.

- **sequence-number**
  Identifies the segments of a segmented BACnet-Confirmed-Request-PDU. Not present in un-segmented requests, such as this example.

- **proposed-window-size**
  Specifies the maximum segmentation window size acceptable to a requester sending a segmented BACnet-Confirmed-Request-PDU. Not present in un-segmented requests, such as this example.

- **service-choice**
  The service-choice is defined to be of type BACnetConfirmedServiceChoice, which is defined on page 350 as an enumeration. The value for readProperty is 12 decimal.

- **service-request**
The service-request is defined to be of type BACnet-Confirmed-Service-Request, which is defined on page 351. The CHOICE for readProperty is the production ReadProperty-Request, which is defined on page 355 as a SEQUENCE consisting of "objectIdentifier", tagged with context tag 0 and of type BACnetObjectIdentifier; "propertyIdentifier", tagged with context tag 1 and of type BACnetPropertyIdentifier; and "propertyArrayIndex", an OPTIONAL unsigned integer tagged with context tag 2. Since the present-value of an Analog Input Object is not an array, the index will not be present in this example.

The BACnetPropertyIdentifier is an enumeration defined on page 375. present-value has an enumerated value of 85 decimal. Enumerations are primitive, and their encoding is described in clause 20.2.11. In this case, the ASN.1 "[0]" specifies that encoding should contain context tag 0, rather than the application tag 12 which would be used if the ASN.1 did not contain the "[0]". Clause 20.2.15 subitem "a)" and "b)" specify the encoding of such a value to be the context tag, followed by the data octets encoded as specified in clause 20.1.11.

5.1.1 ReadProperty - Request on PTP

PTP REQUESTER sends

0x55  Preamble
0xFF  
0x02  Frame Type 2: Data 0
0x00  Length = 23
0x16  
0x2E  Header CRC
NPDU follows
0x01  Network version 1
0x2C  Network control octet
   Bit7 = 0  "BACnet APDU"
   Bit6 = 0  not used
   Bit5 = 1  DNET present
   Bit4 = 0  not used
   Bit3 = 1  SNET present
   Bit2 = 1  reply expected (confirmed service request)
   Bit1,0 = 00  normal priority
0x00  DNET 0x0001
0x01  
0x01  Destination address length 1 octet
0x01  DADR 0x01
0x00  SNET 0x0002
0x02  
0x01  Source address length 1 octet
0x15  SADR 0x15
0xFF  Hop Count 255
APDU follows
0x02  APDU type 0: Confirmed Request; Bit1: Segmented Response Accepted
0x03  Max response 3: up to 480 octets
0x00  Invoke ID 0
0x0C  Service Choice 12:  Read Property
Tagged Service parameters follow
0x0C  Context Tag 0, length 4:  Object Identifier
0x00  Data: Object type 0 (Analog Input), Object Instance 1
0x00  
0x01
BACnet Samples

0x19 Context Tag 1, length 1: BACnetPropertyIdentifier
0x55 Data: Property 85: present-value
0xD8 Data CRC (second octet is 0x13 with DLE escape)
0x10
0x93

PTP BCU acknowledges

0x55 Preamble
0xFF Frame Type 6: Data Ack 0 XON
0x00 Length = 0
0x00
0xBB Header CRC

5.1.2 ReadProperty - Request on ARCNET

ARCNET REQUESTER sends

0x15 ARCNET Source ID
0x01 ARCNET Destination ID
0x11 ARCNET Frame Length = 17
0xCD ARCNET System Code for BACnet
0x82 8802-2 SSAP (value assigned by IEEE to BACnet)
0x82 8802-2 DSAP
0x83 8802-2 function: UI
  NPDU follows
0x01 Network version 1
0x04 Network control octet
  Bit7 = 0 "BACnet APDU"
  Bit6 = 0 not used
  Bit5 = 0 DNET not present
  Bit4 = 0 not used
  Bit3 = 0 SNET not present
  Bit2 = 1 reply expected (confirmed service request)
  Bit1,0 = 00 normal priority
APDU follows
0x02 APDU type 0: Confirmed Request; Bit1: Segmented Response Accepted
0x03 Max response 3: up to 480 octets
0x00 Invoke ID 0
0x0C Service Choice 12: Read Property
  Tagged Service parameters follow
  0xC Context Tag 0, length 4: Object Identifier
  0x00 Data: Object type 0 (Analog Input), Object Instance 1
  0x00
  0x00
  0x01
  0x19 Context Tag 1, length 1: BACnetPropertyIdentifier
  0x55 Data: Property 85: present-value
5.2 ReadProperty - Response

When the BCU receives a ReadProperty-Request, it services the request as described in clause 15.5.2. The normal response is a Result(+), which is defined in Table 15-5.

Since the response includes data, it must be returned as a BACnet-ComplexACK-PDU. On page 349, BACnet-ComplexACK-PDU is defined to be a SEQUENCE of items. The comment states that tags are not used in the header encoding.

Header encoding for BACnet-ComplexACK-PDU is shown in clause 20.1.5. Values may be determined as follows:

- segmented-message
  BOOLEAN, TRUE if this is a segmented response, FALSE otherwise. FALSE in this example.

- more-follows
  BOOLEAN, TRUE if this is a segmented response, and this is not the final segment. FALSE otherwise. FALSE in this example.

- originalInvokeID
  Will contain the invokeID of the BACnet-Confirmed-Request-PDU which is being responded to.

- sequence-number
  Identifies the segments of a segmented BACnet-Confirmed-Request-PDU. Not present in un-segmented requests, such as this example.

- proposed-window-size
  Specifies the maximum segmentation window size acceptable to a responder sending a segmented BACnet-ComplexACK-PDU. Not present in un-segmented ComplexACKs, such as this example.

- service-ACK-choice
  The service-ACK-choice is defined to be of type BACnetConfirmedServiceChoice, which is defined on page 350 as an enumeration. The value for readProperty is 12 decimal.

- service-ACK
  The service-ACK is defined to be of type BACnet-Confirmed-Service-ACK, which is defined on page 351. The CHOICE for readProperty is the production ReadProperty-ACK.

  ReadProperty-ACK is defined on page 355 as a SEQUENCE consisting of "objectIdentifier", tagged with context tag 0 and of type Object Identifier; "propertyIdentifier", tagged with context tag 1 and of type BACnetPropertyIdentifier; "propertyArrayIndex", an OPTIONAL unsigned integer tagged with context tag 2; and "propertyValue" tagged with context tag 3 and of type ABSTRACT-SYNTAX.&Type.

  In this example, a propertyValue will be returned. The designation "ABSTRACT-SYNTAX.&Type" means that the encoding (defined in clause 20.2.19) is the complete encoding including tags of the data type defined for the property value being returned. Since such an encoding includes tags, the ABSTRACT-SYNTAX.&Type is "constructed" according to the definition of 20.2.1.3.2, the context tag 3 must be encoded as an opening tag/closing tag pair surrounding the encoding of the ABSTRACT-SYNTAX.&Type. In this example, the value is a real number.
5.2.1  ReadProperty - Response on PTP

PTP BCU sends ComplexACK

0x55  Preamble
0xFF
0x03  Frame Type 3: Data 1
0x00  Length = 28.
0x1C
0xB0  Header CRC
NPDU follows
0x01  Network version 1
0x28  Network control octet
    Bit7  = 0   "BACnet APDU"
    Bit6  = 0   not used
    Bit5  = 1   DNET present
    Bit4  = 0   not used
    Bit3  = 1   SNET present
    Bit2  = 0   no reply expected
    Bit1,0 = 00  normal priority
0x00  DNET 0x0002
0x02
0x01  Destination address length 1 octet
0x15  DADR 0x15
0x00  SNET 0x0001
0x01
0x01  Source address length 1 octet
0x01  SADR 0x01
0xFF  Hop Count 255
APDU follows
0x30  APDU type 3: Complex ACK
0x00  Invoke ID 0
0x0C  Service Choice 12:  Read Property ACK
Tagged Service parameters follow
0x0C  Context Tag 0, length 4:  Object Identifier
    Data: Object type 0 (Analog Input), Object Instance 1
0x00
0x00
0x00
0x01
0x19  Context Tag 1, length 1:  BACnetPropertyIdentifier
0x55  Data: Property 85:  present-value
0x3E  Context Tag 3, opening tag: ABSTRACT-SYNTAX.&Type
0x44  Application Tag 4 (Real), length 4
0x42  Data: value = 46.4
0x39
0x99
0x9A
0x3F  Context Tag 3, closing tag
0x1F  Data CRC
0x6D

PTP REQUESTER acknowledges

0x55  Preamble
0xFF
0x07  Frame Type 7: Data Ack 1 XON
5.2.2 ReadProperty - Response on ARCNET

ARCNET BCU sends ComplexACK

0x01 ARCNET Source ID
0x15 ARCNET Destination ID
0xC0 ARCNET Frame Length = 23.
0xC2 8802-2 SSAP (value assigned by IEEE to BACnet)
0xC2 8802-2 DSAP
0xCE 8802-2 function: UI
NPDU follows
0x01 Network version 1
0x00 Network control octet
   Bit7  = 0 "BACnet APDU"
   Bit6  = 0 not used
   Bit5  = 0 DNET not present
   Bit4  = 0 not used
   Bit3  = 0 SNET not present
   Bit2  = 0 no reply expected
   Bit1,0 = 00 normal priority
NPDU follows
0x30 APDU type 3: Complex ACK
0x00 Invoke ID 0
0x0C Service Choice 12: Read Property ACK
Tagged Service parameters follow
0x0C Context Tag 0, length 4: Object Identifier
   Data: Object type 0 (Analog Input), Object Instance 1
0x00
0x00
0x00
0x01
0x19 Context Tag 1, length 1: BACnetPropertyIdentifier
0x55 Data: Property 85: present-value
0x3E Context Tag 3, opening tag: ABSTRACT-SYNTAX.&Type
0x44 Application Tag 4 (Real), length 4
0x42 Data: value = 46.4
0x39
0x99
0x9A
0x3F Context Tag 3, closing tag

6.0 ReadPropertyMultiple - One Property of Analog Input

The previous example used the ReadProperty service to read a property of an object. This example uses the ReadPropertyMultiple service to read the same property. Either service may be used. However, ReadPropertyMultiple has additional capabilities which will be shown in later examples.
6.1 ReadPropertyMultiple - Request

APDU header encoding is as in the example of 5.1, except for service-choice and service-request:

service-choice
The service-choice is defined to be of type BACnetConfirmedServiceChoice, which is defined on page 350 as an enumeration. The value for readPropertyMultiple is 14 decimal.

service-request
The service-request is defined to be of type BACnet-Confirmed-Service-Request, which is defined on page 351. The CHOICE for readPropertyMultiple is the production ReadPropertyMultiple-Request, which is defined on page 356 as a SEQUENCE consisting of a SEQUENCE OF ReadAccessSpecifications.

ReadAccessSpecification is defined on page 382 as a SEQUENCE consisting of an "objectIdentifier", tagged with context tag 0 and of type BACnetObjectIdentifier; and a "listOfPropertyReferences", tagged with context tag 1 and of type SEQUENCE OF BACnetPropertyReference.

BACnetPropertyReference is defined on page 378 as "propertyIdentifier", tagged with context tag 0 and of type BACnetPropertyIdentifier; and "propertyArrayIndex", an OPTIONAL unsigned integer tagged with context tag 1. Since the present-value of an Analog Input Object is not an array, the index will not be present in this example.

In this example, the request contains a single ReadAccessSpecification, which in turn specifies a single BACnetPropertyReference for the property to be read.

6.1.1 ReadPropertyMultiple - Request on PTP

PTP REQUESTER sends

0x55 Preamble
0xFF Preamble
0x03 Frame Type 3: Data 1
0x00 Length = 24
0x18
0x4D Header CRC
NPDU follows
0x01 Network version 1
0x2C Network control octet
   Bit7 = 0 "BACnet APDU"
   Bit6 = 0 not used
   Bit5 = 1 DNET present
   Bit4 = 0 not used
   Bit3 = 1 SNET present
   Bit2 = 1 reply expected (confirmed service request)
   Bit1,0 = 00 normal priority
0x00 DNET 0x0001
0x01
0x01 Destination address length 1 octet
0x01 DADR 0x01
0x00 SNET 0x0002
0x02
0x01 Source address length 1 octet
6.1.2 ReadPropertyMultiple - Request on ARCNET

ARCNET REQUESTER sends

0x15   ARCNET Source ID
0x01   ARCNET Destination ID
0x13   ARCNET Frame Length = 19.
0xCD   ARCNET System Code for BACnet
0x82   8802-2 SSAP (value assigned by IEEE to BACnet)
0x82   8802-2 DSAP
0x83   8802-2 function: UI
NPDU follows
0x01   Network version 1
0x04   Network control octet
  Bit7 = 0   "BACnet APDU"
  Bit6 = 0   not used
  Bit5 = 0   DNET not present
  Bit4 = 0   not used
  Bit3 = 0   SNET not present
  Bit2 = 1   reply expected (confirmed service request)
  Bit1,0 = 00 normal priority
APDU follows
0x02   APDU type 0: Confirmed Request; Bit1: Segmented Response Accepted
0x03   Max response 3: up to 480 octets
0x02   Invoke ID 2
0x0E   Service Choice 14: Read Property Multiple
Tagged Service parameters follow

0x0C  Context Tag 0, length 4:  Object Identifier
0x00  Object type 0 (Analog Input), Object Instance 1
0x00
0x00
0x01
0x1E  Context Tag 1, opening tag:  SEQUENCE OF BACnetPropertyReference
0x09  Context Tag 0, length 1:  BACnetPropertyIdentifier
0x55  Property 85:  present-value
0x1F  Context Tag 1, closing tag

6.2  ReadPropertyMultiple - Response

When the BCU receives a ReadPropertyMultiple-Request, it services the request as described in clause 15.7.2  The normal response is a Result(+), which is defined in Table 15-11.

Since the response includes data, it must be returned as a BACnet-ComplexACK-PDU. On page 349, BACnet-ComplexACK-PDU is defined to be a SEQUENCE of items. The comment states that tags are not used in the header encoding.

Header encoding is as for the example of 5.2 except for service-ACK-choice and service-ACK:

service-ACK-choice
The service-ACK-choice is defined to be of type BACnetConfirmedServiceChoice, which is defined on page 350 as an enumeration. The value for readPropertyMultiple is 14 decimal.

service-ACK
The service-ACK is defined to be of type BACnet-Confirmed-Service-ACK, which is defined on page 351. The CHOICE for readPropertyMultiple is the production ReadPropertyMultiple-ACK, which is defined on page 356 as a SEQUENCE OF ReadAccessResults.

The ReadAccessResult is defined on page 381 as a SEQUENCE consisting of "objectIdentifier", tagged with context tag 0 and of type Object Identifier; and "listOfResults", tagged with context tag 1 and of type SEQUENCE OF SEQUENCE.

Each element of the listOfResults SEQUENCE OF consists of a "propertyIdentifier", tagged with context tag 2 and of type BACnetPropertyIdentifier; a "propertyArrayIndex", an OPTIONAL unsigned integer tagged with context tag 3; and a CHOICE of either "propertyValue" tagged with context tag 4 and of type ABSTRACT-SYNTAX.&Type, or "propertyAccessError", tagged with context tag 5 and of type Error.

In this example, the response contains a single ReadAccessResult, which specifies a single element listOfResults.

In this example, a propertyValue will be returned. The designation "ABSTRACT-SYNTAX.&Type" means that the encoding (defined in clause 20.2.19) is the complete encoding including tags of the data type defined for the property value being returned. Since such an encoding includes tags, the ABSTRACT-SYNTAX.&Type is "constructed" according to the definition of 20.2.1.3.2, the context tag 4 must be encoded as an opening tag/closing tag pair surrounding the encoding of the ABSTRACT-SYNTAX.&Type. In this example, the value is a real number.
6.2.1 ReadPropertyMultiple - Response on PTP

PTP BCU sends ComplexACK

0x55   Preamble
0xFF
0x02   Frame Type 2: Data 0
0x00   Length = 30
0x1E
0xD7   Header CRC
NPDU follows
0x01   Network version 1
0x28   Network control octet
    Bit7  = 0  "BACnet APDU"
    Bit6  = 0  not used
    Bit5  = 1  DNET present
    Bit4  = 0  not used
    Bit3  = 1  SNET present
    Bit2  = 0  no reply expected
    Bit1,0 = 00  normal priority
0x00   DNET 0x0002
0x02
0x01   Destination address length 1 octet
0x15   DADR 0x15
0x00   SNET 0x0001
0x01
0x01   Source address length 1 octet
0x01   SADR 0x01
0xFF   Hop Count 255
APDU follows
0x30   APDU type 3: Complex ACK
0x02   Invoke ID 2
0x0E   Service Choice 14: Read Property Multiple ACK
Tagged Service parameters follow
0x0C   Context Tag 0, length 4: Object Identifier
Data: Object type 0 (Analog Input), Object Instance 1
0x00
0x00
0x01
0x1E   Context Tag 1, opening tag: SEQUENCE OF SEQUENCE
0x29   Context Tag 2, length 1: BACnetPropertyIdentifier
0x55   Data: Property 85: present-value
0x4E   Context Tag 4, opening tag: ABSTRACT-SYNTAX.&Type
0x42   Application Tag 4 (Real), length 4
Data: value = 46.4
0x39
0x99
0x9A
0x4F   Context Tag 4, closing tag
0x1F   Context Tag 1, closing tag
0xE5   Data CRC
0xD4

PTP REQUESTER acknowledges

0x55   Preamble
Note that the response in this example was conveyed in a Data 1 frame. Examples in this document should not be assumed to be consecutive. Note that transmit and receive sequence numbers (Data 0 versus Data 1) are independent. They must be processed as described in clause 10. Both sequence numbers must be reset to zero whenever a connection is made.

6.2.2 ReadPropertyMultiple - Response on ARCNET

ARCNET BCU sends ComplexACK

0x01 ARCNET Source ID
0x15 ARCNET Destination ID
0x19 ARCNET Frame Length = 25.
0xCD ARCNET System Code for BACnet
0x82 8802-2 SSAP (value assigned by IEEE to BACnet)
0x82 8802-2 DSAP
0x83 8802-2 function: UI
NPDU follows
0x01 Network version 1
0x00 Network control octet
    Bit7 = 0 "BACnet APDU"
    Bit6 = 0 not used
    Bit5 = 0 DNET not present
    Bit4 = 0 not used
    Bit3 = 0 SNET not present
    Bit2 = 0 no reply expected
    Bit1,0 = 00 normal priority
APDU follows
0x30 APDU type 3: Complex ACK
0x02 Invoke ID 2
0x0E Service Choice 14: Read Property Multiple ACK
Tagged Service parameters follow
0xC0 Context Tag 0, length 4: Object Identifier
0x00 Data: Object type 0 (Analog Input), Object Instance 1
0x00
0x00
0x1E Context Tag 1, opening tag: SEQUENCE OF SEQUENCE
0x29 Context Tag 2, length 1: BACnetPropertyIdentifier
0x55 Data: Property 85: present-value
0x4E Context Tag 4, opening tag: ABSTRACT-SYNTAX.&Type
0x44 Application Tag 4 (Real), length 4
0x42 Data: value = 46.4
0x39
0x99
0x9A
0x4F Context Tag 4, closing tag
0x1F Context Tag 1, closing tag
7.0 ReadPropertyMultiple - Multiple Properties of Analog Input

Example of a ReadPropertyMultiple for the present-value, status-flags, and engineering units properties of Analog Input Object instance number 1.

In this case, the request contains a single ReadAccessSpecification, which specifies a SEQUENCE OF (list of) three BACnetPropertyReferences, one for each of the three properties to be read.

The encoding is as described in the previous example, except that the SEQUENCE OF BACnetPropertyReference contains three elements between its opening and closing tags.

7.1 ReadPropertyMultiple - on PTP

PTP REQUESTER sends

0x55  Preamble
0xFF
0x02  Frame Type 2: Data 0
0x00  Length = 28
0x1C
0x28  Header CRC
NPDU follows
0x01  Network version 1
0x2C  Network control octet
    Bit7  = 0  "BACnet APDU"
    Bit6  = 0  not used
    Bit5  = 1  DNET present
    Bit4  = 0  not used
    Bit3  = 1  SNET present
    Bit2  = 1  reply expected (confirmed service request)
    Bit1,0 = 00  normal priority
0x00  DNET 0x0001
0x01
0x01  Destination address length 1 octet
0x01  DADR 0x01
0x00  SNET 0x0002
0x02
0x01  Source address length 1 octet
0x15  SADR 0x15
0xFF  Hop Count 255
APDU follows
0x02  APDU type 0: Confirmed Request; Bit1: Segmented Response Accepted
0x03  Max response 3: up to 480 octets
0x02  Invoke ID 2
0x0E  Service Choice 14: Read Property Multiple
Tagged Service parameters follow
0x0C  Context Tag 0, length 4: Object Identifier
0x00  Object type 0 (Analog Input), Object Instance 1
0x00
0x00
0x01
0x1E  Context Tag 1, opening tag: SEQUENCE OF BACnetPropertyReference
0x09  Context Tag 0, length 1: BACnetPropertyIdentifier
PTP BCU acknowledges

```
0x55  Preamble
0xFF
0x06  Frame Type 6: Data Ack 0 XON
0x00  Length = 0
0x00
0x23  Header CRC
```

In this case, the response contains a single ReadAccessResult, which specifies a SEQUENCE OF (list of) three SEQUENCE, one for each of the three properties to be returned.

PTP BCU responds with ComplexACK

```
0x55  Preamble
0xFF
0x03  Frame Type 3: Data 1
0x00  Length = 43
0x2B
0x5C  Header CRC
NPDU follows
0x01  Network version 1
0x28  Network control octet
    Bit7  =  0  "BACnet APDU"
    Bit6  =  0  not used
    Bit5  =  1  DNET present
    Bit4  =  0  not used
    Bit3  =  1  SNET present
    Bit2  =  0  no reply expected
    Bit1,0 = 00  normal priority
0x00  DNET 0x0002
0x02
0x01  Destination address length 1 octet
0x15  DADR 0x15
0x00  SNET 0x0001
0x01
0x01  Source address length 1 octet
0x01  SADR 0x01
0xFF  Hop Count 255
APDU follows
0x30  APDU type 3: Complex ACK
0x02  Invoke ID 2
0x0E  Service Choice 14:  Read Property Multiple ACK
Tagged Service parameters follow
0x0C  Context Tag 0, length 4:  Object Identifier
0x00  Data: Object type 0 (Analog Input), Object Instance 1
0x00
7.2 ReadPropertyMultiple - on ARCNET

ARCNET REQUESTER sends

0x15  ARCNET Source ID
0x01  ARCNET Destination ID
0x17  ARCNET Frame Length = 23.
0xCD  ARCNET System Code for BACnet
0x82  8802-2 SSAP (value assigned by IEEE to BACnet)
0x82  8802-2 DSAP
0x83  8802-2 function: UI
NPDU follows
0x01  Network version 1
0x04  Network control octet
  Bit7 = 0  "BACnet APDU"
  Bit6 = 0  not used
Bit5  = 0   DNET not present
Bit4  = 0   not used
Bit3  = 0   SNET not present
Bit2  = 1   reply expected (confirmed service request)
Bit1,0 = 00  normal priority

APDU follows
0x02  APDU type 0: Confirmed Request; Bit1: Segmented Response Accepted
0x03  Max response 3: up to 480 octets
0x02  Invoke ID 2
0x0E  Service Choice 14: Read Property Multiple
    Tagged Service parameters follow
0x0C  Context Tag 0, length 4: Object Identifier
0x00  Object type 0 (Analog Input), Object Instance 1
0x00
0x00
0x01
0x1E  Context Tag 1, opening tag: SEQUENCE OF BACnetPropertyReference
0x09  Context Tag 0, length 1: BACnetPropertyIdentifier
0x55  Property 85: present-value
0x09  Context Tag 0, length 1: BACnetPropertyIdentifier
0x6F  Property 111: status-flags
0x09  Context Tag 0, length 1: BACnetPropertyIdentifier
0x75  Property 117: units
0x1F  Context Tag 1, closing tag

In this case, the response contains a single ReadAccessResult, which specifies a SEQUENCE OF (list of) three SEQUENCE, one for each of the three properties to be returned.

**ARCNET BCU responds with ComplexACK**

0x01  ARCNET Source ID
0x15  ARCNET Destination ID
0x26  ARCNET Frame Length = 38.
0xCD  ARCNET System Code for BACnet
0x82  8802-2 SSAP (value assigned by IEEE to BACnet)
0x82  8802-2 DSAP
0x83  8802-2 function: UI
    NPDU follows
0x01  Network version 1
0x00  Network control octet
    Bit7  = 0   "BACnet APDU"
    Bit6  = 0   not used
    Bit5  = 0   DNET not present
    Bit4  = 0   not used
    Bit3  = 0   SNET not present
    Bit2  = 0   no reply expected
    Bit1,0 = 00  normal priority

APDU follows
0x30  APDU type 3: Complex ACK
0x02  Invoke ID 2
0x0E  Service Choice 14: Read Property Multiple ACK
    Tagged Service parameters follow
0x0C  Context Tag 0, length 4: Object Identifier
0x00  Data: Object type 0 (Analog Input), Object Instance 1
0x00
0x00
8.0 ReadPropertyMultiple - Single Property of Multiple Objects

Example of a ReadPropertyMultiple for the present-value of Binary Input Object number 1 and of Analog Input Object number 1 in a single request.

In this case, the request contains two ReadAccessSpecifications, one for each object. Each ReadAccessSpecification specifies a SEQUENCE OF (list of) one BACnetPropertyReference.

8.1 ReadPropertyMultiple - on PTP

PTP REQUESTER sends

| 0x55 | Preamble |
| 0xFF |          |
| 0x02 | Frame Type 2: Data 0 |
| 0x00 | Length = 33 |
| 0x21 |          |
| 0xC2 | Header CRC |
|      | NPDU follows |
| 0x01 | Network version 1 |
| 0x2C | Network control octet |
|      | Bit7 = 0 "BACnet APDU" |
|      | Bit6 = 0 not used |
|      | Bit5 = 1 DNET present |
|      | Bit4 = 0 not used |
|      | Bit3 = 1 SNET present |
Bit2 $= 1$  
  reply expected (confirmed service request)  
Bit1,0 $= 00$  
  normal priority  

0x00  
  DNET 0x0001  
0x01  
  Destination address length 1 octet  
0x01  
  DADR 0x01  
0x00  
  SNET 0x0002  
0x02  
  Source address length 1 octet  
0x15  
  SADR 0x15  
0xFF  
  Hop Count 255  
APDU follows  
0x02  
  APDU type 0: Confirmed Request; Bit1: Segmented Response Accepted  
0x03  
  Max response 3: up to 480 octets  
0x06  
  Invoke ID 6  
0x0E  
  Service Choice 14: Read Property Multiple  
  Tagged Service parameters follow  
0x0C  
  Context Tag 0, length 4: Object Identifier  
0x00  
  Data: Object type 3 (Binary Input), Object Instance 1  
0xC0  
0x01  
  Context Tag 1, opening tag: SEQUENCE OF BACnetPropertyReference  
0x09  
  Context Tag 0, length 1: BACnetPropertyIdentifier  
0x55  
  Data: Property 85: present-value  
0x1F  
  Context Tag 1, closing tag  
0x0C  
  Context Tag 0, length 4: Object Identifier  
0x00  
  Data: Object type 0 (Analog Input), Object Instance 1  
0x00  
0x00  
0x01  
0x1E  
  Context Tag 1, opening tag: SEQUENCE OF BACnetPropertyReference  
0x09  
  Context Tag 0, length 1: BACnetPropertyIdentifier  
0x55  
  Data: Property 85: present-value  
0x1F  
  Context Tag 1, closing tag  
0x98  
  Data CRC  
0x16  
  
**PTP BCU acknowledges**  
0x55  
  Preamble  
0xFF  
0x06  
  Frame Type 6: Data Ack 0 XON  
0x00  
  Length = 0  
0x00  
0xBB  
  Header CRC  

The response contains two ReadAccessResults, each of which specifies a SEQUENCE OF (list of) one property value.  

**PTP BCU responds with ComplexACK**  
0x55  
  Preamble  
0xFF  
0x03  
  Frame Type 3: Data 1  
0x00  
  Length = 43
BACnet Samples

0x2B  Header CRC
0x5C  NPDU follows
0x01  Network version 1
0x28  Network control octet
    Bit7  = 0  "BACnet APDU"
    Bit6  = 0  not used
    Bit5  = 1  DNET present
    Bit4  = 0  not used
    Bit3  = 0  SNET not present
    Bit2  = 1  reply expected (confirmed service request)
    Bit1,0  = 00  normal priority
0x00  DNET 0x0002
0x02
0x01  Destination address length 1 octet
0x15  DADR 0x15
0x00  SNET 0x0001
0x01
0x01  Source address length 1 octet
0x01  SADR 0x01
0xFF  Hop Count 255
APDU follows
0x30  APDU type 3: Complex ACK
0x06  Invoke ID 6
0x0E  Service Choice 14: Read Property Multiple ACK
Tagged Service parameters follow
0x0C  Context Tag 0, length 4: Object Identifier
0x00  Data: Object type 3 (Binary Input), Object Instance 1
0xc0
0x00
0x00
0x01
0x1E  Context Tag 1, opening tag: SEQUENCE OF SEQUENCE
0x29  Context Tag 2, length 1: BACnetPropertyIdentifier
0x55  Data: Property 85: present-value
0x4E  Context Tag 4, opening tag: ABSTRACT-SYNTAX.&Type
0x91  Application tag 9 (enumerated), length 1
0x01  Data: enumerated value 1: active
0x4F  Context Tag 4, closing tag
0x1F  Context Tag 1, closing tag
0x0C  Context Tag 0, length 4: Object Identifier
0x00  Data: Object type 0 (Analog Input), Object Instance 1
0x00
0x00
0x01
0x1E  Context Tag 1, opening tag: SEQUENCE OF SEQUENCE
0x29  Context Tag 2, length 1: BACnetPropertyIdentifier
0x55  Data: Property 85: present-value
0x4E  Context Tag 4, opening tag: ABSTRACT-SYNTAX.&Type
0x44  Application Tag 4 (Real), length 4
0x42  Data: value = 46.4
0x39
0x99
0x9A
0x4F  Context Tag 4, closing tag
0x1F  Context Tag 1, closing tag
0xCA  Data CRC
0x06

**PTP REQUESTER acknowledges**

0x55  Preamble
0xFF  Preamble
0x07  Frame Type 7: Data Ack 1 XON
0x00  Length = 0
0x00
0x23  Header CRC

8.2 **ReadPropertyMultiple - on ARCNET**

**ARCNET REQUESTER sends**

0x15  ARCNET Source ID
0x01  ARCNET Destination ID
0x1C  ARCNET Frame Length = 28.
0xCD  ARCNET System Code for BACnet
0x82  8802-2 SSAP (value assigned by IEEE to BACnet)
0x82  8802-2 DSAP
0x83  8802-2 function: UI
     NPDU follows
0x01  Network version 1
0x04  Network control octet
      Bit7 = 0  "BACnet APDU"
      Bit6 = 0  not used
      Bit5 = 0  DNET not present
      Bit4 = 0  not used
      Bit3 = 0  SN1ET not present
      Bit2 = 1  reply expected (confirmed service request)
      Bit1,0 = 00  normal priority
     APDU follows
0x02  APDU type 0: Confirmed Request; Bit1: Segmented Response Accepted
0x03  Max response 3: up to 480 octets
0x06  Invoke ID 6
0x0E  Service Choice 14: Read Property Multiple
     Tagged Service parameters follow
0xC0  Context Tag 0, length 4: Object Identifier
0x00  Data: Object type 3 (Binary Input), Object Instance 1
0xC0
0x00
0x00
0x1E  Context Tag 1, opening tag: SEQUENCE OF BACnetPropertyReference
0x09  Context Tag 0, length 1: BACnetPropertyIdentifier
0x55  Data: Property 85: present-value
0x1F  Context Tag 1, closing tag
0xC0  Context Tag 0, length 4: Object Identifier
0x00  Data: Object type 0 (Analog Input), Object Instance 1
0x00
0x00
0x01
0x1E  Context Tag 1, opening tag: SEQUENCE OF BACnetPropertyReference
0x09  Context Tag 0, length 1: BACnetPropertyIdentifier
The response contains two ReadAccessResults, each of which specifies a SEQUENCE OF (list of) one property.

**ARCNET BCU responds with ComplexACK**

```
0x01  ARCNET Source ID
0x15  ARCNET Destination ID
0x26  ARCNET Frame Length = 38.
0xCD  ARCNET System Code for BACnet
0x82  8802-2 SSAP (value assigned by IEEE to BACnet)
0x82  8802-2 DSAP
0x83  8802-2 function: UI
       NPDU follows
0x01  Network version 1
0x00  Network control octet
       Bit7  = 0  "BACnet APDU"
       Bit6  = 0  not used
       Bit5  = 0  DNET not present
       Bit4  = 0  not used
       Bit3  = 0  SNET not present
       Bit2  = 0  no reply expected
       Bit1,0 = 00  normal priority
       APDU follows
0x30  APDU type 3: Complex ACK
0x06  Invoke ID 6
0x0E  Service Choice 14:  Read Property Multiple ACK
       Tagged Service parameters follow
0x0C  Context Tag 0, length 4:  Object Identifier
0x00  Data: Object type 3 (Binary Input), Object Instance 1
0xC0
0x00
0x00
0x01
0x1E  Context Tag 1, opening tag:  SEQUENCE OF SEQUENCE
0x29  Context Tag 2, length 1:  BACnetPropertyIdentifier
0x55  Data: Property 85:  present-value
0x4E  Context Tag 4, opening tag: ABSTRACT-SYNTAX.&Type
0x91  Application tag 9 (enumerated), length 1
       Data: enumerated value 1: active
0x4F  Context Tag 4, closing tag
0x1F  Context Tag 1, closing tag
0x0C
0x0C  Context Tag 0, length 4:  Object Identifier
0x00  Data: Object type 0 (Analog Input), Object Instance 1
0x00
0x00
0x01
0x1E  Context Tag 1, opening tag:  SEQUENCE OF SEQUENCE
0x29  Context Tag 2, length 1:  BACnetPropertyIdentifier
0x55  Data: Property 85:  present-value
0x4E  Context Tag 4, opening tag: ABSTRACT-SYNTAX.&Type
0x44  Application Tag 4 (Real), length 4
       Data: value = 46.4
0x39
0x99
```
9.0 WriteProperty - Single Property of Binary Output

Example of a WriteProperty for the present-value property of Binary Output Object instance number 1.

9.1 WriteProperty - Request

The Binary Output Object and its properties are described in clause 12.5. The WriteProperty service is described in clause 15.8. The ASN.1 definition of WriteProperty is in clause 21. This is a Confirmed service. On page 348, a BACnet-Confirmed-Request-PDU is defined to be a SEQUENCE of items. The comment states that tags are not used in the header encoding. Header encoding for a BACnet-Confirmed-Request-PDU is shown in clause 20.1.2.

The service-choice is defined to be of type BACnetConfirmedServiceChoice, which is defined on page 350 as an enumeration. The value for writeProperty is 15 decimal.

The service-request is defined to be of type BACnet-Confirmed-Service-Request, which is defined on page 351. The CHOICE for writeProperty is the production WriteProperty-Request, which is defined on page 356 as a SEQUENCE consisting of "objectIdentifier", tagged with context tag 0 and of type BACnetObjectIdentifier; "propertyIdentifier", tagged with context tag 1 and of type BACnetPropertyIdentifier; "propertyArrayIndex", an OPTIONAL unsigned integer tagged with context tag 2; "propertyvalue", tagged with context tag 3 and of type ABSTRACT-SYNTAX.&Type; and "priority", an OPTIONAL unsigned integer in the range 1 to 16 tagged with context tag 4. Since the present-value of a Binary Output Object is not an array, propertyArrayIndex will not be present in this example. Since the present-value of a Binary Output Object is a commandable property, priority will be present in this example.

The BACnetPropertyIdentifier is an enumeration defined on page 375. present-value has an enumerated value of 85 decimal.

In this example, the value to written will be "inactive" (0), and the priority will be 7 (shown in Table 19-1 as available for general use).

9.1.1 WriteProperty - Request on PTP

PTP REQUESTER sends

0x55 Preamble
0xFF Preamble
0x03 Frame Type 3: Data 1
0x00 Length = 28
0x1C  
0xB0 Header CRC
     DPDU follows
0x01 Network version 1
0x2C Network control octet
     Bit7 = 0 "BACnet APDU"
Bit6 = 0  not used
Bit5 = 1  DNET present
Bit4 = 0  not used
Bit3 = 1  SNET present
Bit2 = 1  reply expected (confirmed service request)
Bit1,0 = 00  normal priority

0x00  DNET 0x0001
0x01
0x01  Destination address length 1 octet
0x01  DADR 0x01
0x00  SNET 0x0002
0x02
0x01  Source address length 1 octet
0x15  SADR 0x15
0xFF  Hop Count 255
APDU follows

0x02  APDU type 0: Confirmed Request; Bit1: Segmented Response Accepted
0x03  Max response 3: up to 480 octets
0x05  Invoke ID 5
0x0F  Service Choice 15: Write Property
Tagged Service parameters follow

0x0C  Context Tag 0, length 4: Object Identifier
0x01  Data: Object type 4 (Binary Output), Object Instance 1
0x00
0x00
0x01
0x19  Context Tag 1, length 1: BACnetPropertyIdentifier
0x55  Data: Property 85: present-value
0x3E  Context Tag 3, opening tag: ABSTRACT-SYNTAX.&Type (value)
0x91  Application tag 9 (enumerated), length 1
0x00  Data: enumerated value 0: inactive
0x3F  Context Tag 3, closing tag
0x49  Context Tag 4, length 1: unsigned (priority)
0x07  Data: value 7
0xEB  Data CRC
0xBE

PTP BCU acknowledges

0x55  Preamble
0xFF
0x07  Frame Type 7: Data Ack 1 XON
0x00  Length = 0
0x00
0x23  Header CRC

9.1.2  WriteProperty - Request on ARCNET

ARCNET REQUESTER sends

0x15  ARCNET Source ID
0x01  ARCNET Destination ID
0x17  ARCNET Frame Length = 23.
0xCD  ARCNET System Code for BACnet
0x82  8802-2 SSAP (value assigned by IEEE to BACnet)
9.2 WriteProperty - Response

When the BCU receives a WriteProperty-Request, it services the request as described in clause 15.8.2
The normal response is a Result(+), which is defined in Table 15-13 as containing no parameters. The definition in ASN.1 is in clause 21.

Since the response does not include data, it must be returned as a SimpleACK-PDU. On page 348, BACnet-SimpleACK-PDU is defined to be a SEQUENCE of an invokeID and a service-ACK-choice. The comment states that tags are not used in the header encoding. Header encoding for BACnet-SimpleACK-PDU is shown in clause 20.1.4.

The service-ACK-choice is defined to be of type BACnetConfirmedServiceChoice, which is defined on page 350 as an enumeration. The value for writeProperty is 15 decimal.

9.2.1 WriteProperty - Response on PTP

PTP BCU responds

0x55 Preamble
0xFF
0x02  Frame Type 2: Data 0
0x00  Length = 14
0x0E
0x26  Header CRC
      NPDU follows
0x01  Network version 1
0x28  Network control octet
      Bit7  = 0  "BACnet APDU"
      Bit6  = 0  not used
      Bit5  = 1  DNET present
      Bit4  = 0  not used
      Bit3  = 1  SNET present
      Bit2  = 0  no reply expected
      Bit1,0 = 00  normal priority
0x00  DNET 0x0002
0x02
0x01  Destination address length 1 octet
0x15  DADR 0x15
0x00  SNET 0x0001
0x01
0x01  Source address length 1 octet
0x01  SADR 0x01
0xFF  Hop Count 255
      APDU follows
0x20  APDU type 2: Simple ACK
0x05  Invoke ID 5
0x0F  ServiceACK Choice 15: Write Property
      No Tagged Service parameters follow
0xDD  Data CRC
0x95

**PTP REQUESTER acknowledges**

0x55  Preamble
0xFF
0x06  Frame Type 6: Data Ack 0 XON
0x00  Length = 0
0x00
0xBB  Header CRC

### 9.2.2 WriteProperty - Response on ARCNET

**ARCNET BCU responds**

0x01  ARCNET Source ID
0x15  ARCNET Destination ID
0x09  ARCNET Frame Length = 9.
0xCD  ARCNET System Code for BACnet
0x82  8802-2 SSAP (value assigned by IEEE to BACnet)
0x82  8802-2 DSAP
0x83  8802-2 function: UI
      NPDU follows
0x01  Network version 1
0x00  Network control octet
      Bit7  = 0  "BACnet APDU"
10.0 WritePropertyMultiple - Single Property of Binary Output

The previous example used the WriteProperty service to write a property of an object. This example uses the WritePropertyMultiple service to write the same property. Either service may be used. However, WritePropertyMultiple has additional capabilities, parallel to those of ReadPropertyMultiple.

Example of a WritePropertyMultiple for the present-value property of Binary Output Object instance number 1.

10.1 WritePropertyMultiple - on PTP

The WritePropertyMultiple service is described in clause 15.9. The ASN.1 definition of WritePropertyMultiple is in clause 21. This is a Confirmed service. On page 348, a BACnet-Confirmed-Request-PDU is defined to be a SEQUENCE of items. The comment states that tags are not used in the header encoding. Header encoding for a BACnet-Confirmed-Request-PDU is shown in clause 20.1.2.

The service-choice is defined to be of type BACnetConfirmedServiceChoice, which is defined on page 350 as an enumeration. The value for writePropertyMultiple is 16 decimal.

The service-request is defined to be of type BACnet-Confirmed-Service-Request, which is defined on page 351. The CHOICE for writePropertyMultiple is the production WritePropertyMultiple-Request, which is defined on page 357 as a SEQUENCE OF WriteAccessSpecifications. The definition as a SEQUENCE OF means that a WritePropertyMultiple request can write to one or more properties of one or more objects.

The WriteAccessSpecification is defined on page 382 as a SEQUENCE consisting of "objectIdentifier", tagged with context tag 0 and of type BACnetObjectIdentifier; and "listOfProperties", tagged with context tag 1 and defined as a SEQUENCE OF (list of) BACnetPropertyValue. The definition as a SEQUENCE OF means that a WriteAccessSpecification can include one or more properties.

The BACnetPropertyValue is defined on page 378 as a SEQUENCE consisting of "propertyIdentifier", tagged with context tag 0 and of type BACnetPropertyIdentifier; "propertyArrayIndex", an OPTIONAL unsigned integer tagged with context tag 1; "value", tagged with context tag 2 and of type ABSTRACT-SYNTAX.&Type; and "priority", an OPTIONAL unsigned integer in the range 1 to 16 tagged with context tag 3. Since the present-value of a Binary Output Object is not an array, propertyArrayIndex will not be present in this example. Since the present-value of a Binary Output Object is a commandable property, priority will be present in this example.
The BACnetPropertyIdentifier is an enumeration defined on page 375. The present-value has an enumerated value of 85 decimal.

In this example, the value to written will be "inactive", and the priority will be 7 (shown in Table 19-1 as available for general use).

**PTP REQUESTER sends**

0x55  Preamble
0xFF  Preamble
0x03  Frame Type 3: Data 1
0x00  Length = 30
0x1E  Header CRC
NPDU follows
0x01  Network version 1
0x2C  Network control octet
    Bit7 = 0  "BACnet APDU"
    Bit6 = 0  not used
    Bit5 = 1  DNET present
    Bit4 = 0  not used
    Bit3 = 1  SNET present
    Bit2 = 1  reply expected (confirmed service request)
    Bit1,0 = 00  normal priority
0x00  DNET 0x0001
0x01  Destination address length 1 octet
0x01  DADR 0x01
0x00  SNET 0x0002
0x02  Source address length 1 octet
0x15  SADR 0x15
0xFF  Hop Count 255
APDU follows
0x02  APDU type 0: Confirmed Request; Bit1: Segmented Response Accepted
0x03  Max response 3: up to 480 octets
0x07  Invoke ID 7
0x10  Service Choice 16 (escaped): Write Property Multiple
0x90  Tagged Service follows
0x0C  Context Tag 0, length 4: Object Identifier
   Data: Object type 4 (Binary Output), Object Instance 1
0x00  
0x00  
0x01  
0x1E  Context Tag 1, opening tag: SEQUENCE OF BACnetPropertyValue
0x09  Context Tag 0, length 1: BACnetPropertyIdentifier
0x55  Data: enumerated value 85: present-value
0x2E  Context Tag 2, opening tag: ABSTRACT-SYNTAX.&Type (value)
0x91  Application Tag 9 (enumerated), length 1
0x00  Data: 0 = inactive
0x2F  Context Tag 2, closing tag
0x39  Context Tag 3, length: unsigned (priority)
0x07  Data: Value 7
0x1F  Context Tag 1, closing tag
0xD0  Data CRC
0x81

**PTP BCU acknowledges**

0x55  Preamble
0xFF
0x07  Frame Type 7: Data Ack 1 XON
0x00  Length = 0
0x00
0x23  Header CRC

**PTP BCU responds**

0x55  Preamble
0xFF
0x02  Frame Type 2: Data 0
0x00  Length = 14
0x0E
0x26  Header CRC
NPDU follows
0x01  Network version 1
0x28  Network control octet
    Bit7 = 0   “BACnet APDU”
    Bit6 = 0   not used
    Bit5 = 1   DNET present
    Bit4 = 0   not used
    Bit3 = 1   SNET present
    Bit2 = 0   no reply expected
    Bit1,0 = 00 normal priority
0x00  DNET 0x0002
0x02
0x01  Destination address length 1 octet
0x15  DADR 0x15
0x00  SNET 0x0001
0x01
0x01  Source address length 1 octet
0x01  SADR 0x01
0xFF  Hop Count 255
APDU follows
0x20  APDU type 2: Simple ACK
0x07  Original Invoke ID 7
0x10  ServiceACK Choice 16 (escaped): Write Property Multiple
0x90
0x1B  Data CRC
0x4E

**PTP REQUESTER acknowledges**

0x55  Preamble
0xFF
0x06  Frame Type 6: Data Ack 0 XON
0x00  Length = 0
0x00
0xBB  Header CRC
### 10.2 WritePropertyMultiple - on ARCNET

**ARCNET REQUESTER sends**

<table>
<thead>
<tr>
<th>Hex</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x15</td>
<td>ARCNET Source ID</td>
</tr>
<tr>
<td>0x01</td>
<td>ARCNET Destination ID</td>
</tr>
<tr>
<td>0x19</td>
<td>ARCNET Frame Length = 25.</td>
</tr>
<tr>
<td>0xCD</td>
<td>ARCNET System Code for BACnet</td>
</tr>
<tr>
<td>0x82</td>
<td>8802-2 SSAP (value assigned by IEEE to BACnet)</td>
</tr>
<tr>
<td>0x82</td>
<td>8802-2 DSAP</td>
</tr>
<tr>
<td>0x83</td>
<td>8802-2 function: UI</td>
</tr>
<tr>
<td></td>
<td>NPDU follows</td>
</tr>
<tr>
<td>0x01</td>
<td>Network version 1</td>
</tr>
<tr>
<td>0x04</td>
<td>Network control octet</td>
</tr>
<tr>
<td>Bit7</td>
<td>= 0 &quot;BACnet APDU&quot;</td>
</tr>
<tr>
<td>Bit6</td>
<td>= 0 not used</td>
</tr>
<tr>
<td>Bit5</td>
<td>= 0 DNET not present</td>
</tr>
<tr>
<td>Bit4</td>
<td>= 0 not used</td>
</tr>
<tr>
<td>Bit3</td>
<td>= 0 SNET not present</td>
</tr>
<tr>
<td>Bit2</td>
<td>= 1 reply expected (confirmed service request)</td>
</tr>
<tr>
<td>Bit1,0</td>
<td>= 00 normal priority</td>
</tr>
<tr>
<td></td>
<td>APDU follows</td>
</tr>
<tr>
<td>0x02</td>
<td>APDU type 0: Confirmed Request; Bit1: Segmented Response Accepted</td>
</tr>
<tr>
<td>0x03</td>
<td>Max response 3: up to 480 octets</td>
</tr>
<tr>
<td>0x07</td>
<td>Invoke ID 7</td>
</tr>
<tr>
<td>0x10</td>
<td>Service Choice: Write Property Multiple</td>
</tr>
<tr>
<td></td>
<td>Tagged Service follows</td>
</tr>
<tr>
<td>0x0C</td>
<td>Context Tag 0, length 4: Object Identifier</td>
</tr>
<tr>
<td>0x01</td>
<td>Data: Object type 4 (Binary Output), Object Instance 1</td>
</tr>
<tr>
<td>0x00</td>
<td></td>
</tr>
<tr>
<td>0x00</td>
<td></td>
</tr>
<tr>
<td>0x01</td>
<td></td>
</tr>
<tr>
<td>0x1E</td>
<td>Context Tag 1, opening tag: SEQUENCE OF BACnetPropertyValue</td>
</tr>
<tr>
<td>0x09</td>
<td>Context Tag 0, length 1: BACnetPropertyIdentifier</td>
</tr>
<tr>
<td>0x55</td>
<td>Data: enumerated value 85: present-value</td>
</tr>
<tr>
<td>0x2E</td>
<td>Context Tag 2, opening tag: ABSTRACT-SYNTAX.&amp;Type (value)</td>
</tr>
<tr>
<td>0x91</td>
<td>Application Tag 9 (enumerated), length 1</td>
</tr>
<tr>
<td>0x00</td>
<td>Data: 0 = inactive</td>
</tr>
<tr>
<td>0x2F</td>
<td>Context Tag 2, closing tag</td>
</tr>
<tr>
<td>0x39</td>
<td>Context Tag 3, length: unsigned (priority)</td>
</tr>
<tr>
<td>0x07</td>
<td>Data: Value 7</td>
</tr>
<tr>
<td>0x1F</td>
<td>Context Tag 1, closing tag</td>
</tr>
</tbody>
</table>

**ARCNET BCU responds**

<table>
<thead>
<tr>
<th>Hex</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x01</td>
<td>ARCNET Source ID</td>
</tr>
<tr>
<td>0x15</td>
<td>ARCNET Destination ID</td>
</tr>
<tr>
<td>0x09</td>
<td>ARCNET Frame Length = 9.</td>
</tr>
<tr>
<td>0xCD</td>
<td>ARCNET System Code for BACnet</td>
</tr>
<tr>
<td>0x82</td>
<td>8802-2 SSAP (value assigned by IEEE to BACnet)</td>
</tr>
<tr>
<td>0x82</td>
<td>8802-2 DSAP</td>
</tr>
<tr>
<td>0x83</td>
<td>8802-2 function: UI</td>
</tr>
<tr>
<td></td>
<td>NPDU follows</td>
</tr>
<tr>
<td>0x01</td>
<td>Network version 1</td>
</tr>
<tr>
<td>0x00</td>
<td>Network control octet</td>
</tr>
</tbody>
</table>
Bit7 = 0  "BACnet APDU"
Bit6 = 0  not used
Bit5 = 0  DNET not present
Bit4 = 0  not used
Bit3 = 0  SNET not present
Bit2 = 0  no reply expected
Bit1,0 = 00  normal priority

APDU follows
0x20  APDU type 2: Simple ACK
0x07  Original Invoke ID 7
0x10  ServiceACK Choice 16: Write Property Multiple

11.0  Breaking a PTP Connection

This section does not apply to ARCNET interfaces.

To break a connection with the BCU, a PTP REQUESTER sends

0x55  Preamble
0xFF
0x0E  Frame Type 14: Disconnect Request
0x00  Length = 1
0x01
0x89  Header CRC
0x03  Disconnect reason 3: other
0xE3  Data CRC
0xC2

PTP BCU responds

0x55  Preamble
0xFF
0x0F  Frame Type 15: Disconnect Response
0x00  Length = 0
0x00
0xEF  Header CRC

12.0  Error APDUs

If a REQUESTER sends a correctly encoded Confirmed-Service-Request, and the BCU encounters some condition while servicing the request which necessitates the return of Result(-), the BCU will return a BACnet-Error-PDU. For example, when the BCU receives a WritePropertyMultiple-Request, it services the request as described in clause 15.9.2. If a specified object does not exist, or a specified property is not writable, the response is a Result(-), which is defined in Table 15-14 as containing two parameters, Error Type, and First Failed Write Attempt. The definition in ASN.1 is in clause 21.

Since the response is a Result(-), it must be returned as a BACnet-Error-PDU. On page 349, BACnet-Error-PDU is defined to be a SEQUENCE consisting of an invokeID, an error-choice, and an error. The comment states that tags are not used in the header encoding. Header encoding for BACnet-Error-PDU is shown in clause 20.1.7.
The error-choice is defined to be of type BACnetConfirmedServiceChoice, which is defined on page 350 as an enumeration. The value for writePropertyMultiple is 16 decimal.

The "error" is defined to be of type BACnet-Error, which is defined on page 361 as a CHOICE. The value of the CHOICE for writePropertyMultiple, defined to be of type WritePropertyMultiple-Error. The comment on BACnet-Error states that the context tags on the CHOICE are not to be used in the encoding.

WritePropertyMultiple-Error is defined on page 364 as a SEQUENCE consisting of errorType, tagged with context tag 0 and of type Error; and firstFailedWriteAttempt, tagged with context tag 1 and of type BACnetObjectPropertyReference.

Error is defined on page 363 as a sequence consisting of two enumerations: error-class and error-code. In the case of a non-existent object, the error-class will be "object" and the error-code will generally be "unknown-object".

In the case of an unknown object, the "firstFailedWriteAttempt" is the first property in the propertyList. The PTP framing and Network information of a BACnet-Error-PDU are as shown in the previous examples. The APDU portion is defined in clause 20.1.7, and might appear as follows:

```
0x50 APDU type:  Error
0x06 Original Invoke ID 6
0x10 Error Choice 16 (escaped): Write Property Multiple
0x90 Tagged Service parameters follow
  0x0E Context Tag 0, opening tag: Error
  0x91 Application Tag 9 (enumerated), length 1
  0x01 Data: value 1: "object"
  0x91 Application Tag 9 (enumerated), length 1
  0x1F Data: value 31: "unknown-object"
  0x0F Context Tag 0, closing tag
  0x1E Context Tag 1, opening tag: firstFailedWriteAttempt
  0x0C Context Tag 0, length 4: objectIdentifier
  0x00 Data: Object typ e 4 (Binary Output), Object Instance 1
  0x01 0x00 0x00 0x01
  0x19 Context Tag 1, length 1: propertyIdentifier
  0x55 Data: 55 (present value)
  0x1F Context Tag 1, closing tag
```

### 13.0 Errors Within the Response to ReadPropertyMultiple

If a REQUESTER sends a correctly encoded Confirmed-Service-Request containing a ReadPropertyMultiple request, and the BCU encounters some condition while servicing the request which necessitates the return an error to one or more of the requested, the BCU will issue a BACnet-ComplexACK-PDU. For example, if the request of section 6 were sent to a BCU which did not contain the specified object, the response would contain a tagged error (CHOICE [5]) rather than a tagged value (CHOICE [4]).
The PTP framing and Network information of a BACnet-Error-PDU are as shown in the previous examples. The APDU portion might appear as follows:

0x30  APDU type 3: Complex ACK
0x05  Invoke ID 5 (matching the Invoke ID of the request)
0x0E  Service Choice 14: Read Property Multiple ACK
       Tagged Service parameters follow
       0x0C  Context Tag 0, length 4: Object Identifier
       0x00  Data: Object type 0 (Analog Input), Object Instance 1
       0x00
       0x01
0x1E  Context Tag 1, opening tag: SEQUENCE OF SEQUENCE
0x29  Context Tag 2, length 1: BACnetPropertyIdentifier
0x55  Data: Property 85: present-value
0x5E  Context Tag 5, opening tag: Error
0x91  Application Tag 9 (enumerated), length 1: Error Class
0x01  Data: value = 01 (object)
0x91  Application Tag 9 (enumerated), length 1: Error Code
0x1F  Data: value = 31 (unknown-object)
0x1F  Context Tag 5, closing tag
0x1F  Context Tag 1, closing tag

14.0 Reject of Invalid APDUs

If a REQUESTER sends a Confirmed-Service-Request with a valid APDU header, but with errors in the tagged portion (for example, missing items, incorrect tag numbers, or incorrect encoding of data values), the BCU will return a BACnet-Reject-PDU.

The PTP framing and Network information of a BACnet-Reject-PDU are as shown in the previous examples. The APDU portion is defined in clause 20.1.8, and might appear as follows:

0x60  APDU type 6: Reject
0x06  Original Invoke ID 6
0x04  Reject Reason 4 (invalid-tag)

The BACnetRejectReason enumeration is on page 362.

15.0 Abort APDUs

If a BCU cannot acquire sufficient internal resources to service a request, or if the request contains a non-initial segment, the BCU will return a BACnet-Abort-PDU. Since the BCU is acting as a server for the request, it will set the “server” bit in the Abort PCU.

The PTP framing and Network information of a BACnet-Abort-PDU are as shown in the previous examples. The APDU portion is defined in clause 20.1.9, and might appear as follows:

0x71  APDU type 7: Abort, sent by server
0x06  Original Invoke ID 6
0x02  Abort Reason 2 (invalid-APDU-in-this-state)

The BACnetAbortReason enumeration is on page 361.