ANSI/ASHRAE Addendum *c* to
ANSI/ASHRAE Standard 135-2001

ASHRAE® STANDARD

BACnet®—A Data Communication Protocol for Building Automation and Control Networks

Approved by the ASHRAE Standards Committee on October 5, 2003; by the ASHRAE Board of Directors on January 29, 2004; and by the American National Standards Institute on February 25, 2004.

This standard is under continuous maintenance by a Standing Standard Project Committee (SSPC) for which the Standards Committee has established a documented program for regular publication of addenda or revisions, including procedures for timely, documented, consensus action on requests for change to any part of the standard. The change submittal form, instructions, and deadlines may be obtained in electronic form from the ASHRAE web site, *http://www.ashrae.org*, or in paper form from the Manager of Standards. The latest edition of an ASHRAE Standard and printed copies of a public review draft may be purchased from ASHRAE Customer Service, 1791 Tullie Circle, NE, Atlanta, GA 30329-2305. E-mail: *orders@ashrae.org*. Fax: 404-321-5478. Telephone: 404-636-8400 (worldwide), or toll free 1-800-527-4723 (for orders in U.S. and Canada).

ISSN 1041-2336

Approved American National Standard
ANSI

AMERICAN SOCIETY OF HEATING, REFRIGERATING AND AIR-CONDITIONING ENGINEERS, INC.
1791 Tullie Circle, NE • Atlanta, GA 30329

SPECIAL NOTE

This American National Standard (ANS) is a national voluntary consensus standard developed under the auspices of the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE). Consensus is defined by the American National Standards Institute (ANSI), of which ASHRAE is a member and which has approved this standard as an ANS, as "substantial agreement reached by directly and materially affected interest categories. This signifies the concurrence of more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that an effort be made toward their resolution." Compliance with this standard is voluntary until and unless a legal jurisdiction makes compliance mandatory through legislation.

ASHRAE obtains consensus through participation of its national and international members, associated societies, and public review.

ASHRAE Standards are prepared by a Project Committee appointed specifically for the purpose of writing the Standard. The Project Committee Chair and Vice-Chair must be members of ASHRAE; while other members may or may not be members of ASHRAE, all must be technically qualified in the subject area of the standard. Every effort is made to balance the concerned interests on all Project Committees.

The Manager of Standards of ASHRAE should be contacted for:

a. interpretation of the contents of this Standard,
b. participation in the next review of the Standard,
c. offering constructive criticism for improving the Standard,
d. permission to reprint portions of the Standard.

(This foreword is not part of this standard. It is merely informative and does not contain requirements necessary for conformance to the standard.)

FOREWORD

The purpose of this addendum is to add a number of independent substantive changes to the BACnet standard. These modifications are the result of change proposals made pursuant to the continuous maintenance procedures contained in the *Manual for Processing ASHRAE Standards* and *PC Guidance* and of deliberations within Standing Standard Project Committee 135. The changes are summarized below.

135*c*-1. Allow Life Safety objects to advertise supported modes, p. 1.
135*c*-2. Add Unsilence Options to the LifeSafetyOperation Service, p.3.
135*c*-3. Specify the relationship between the Event_Type and Event_Parameter properties, p.4.
135*c*-4. Add a new Accumulator Object Type, p. 5.
135*c*-5. Add a new Pulse Converter Object Type, p. 19.
135*c*-6. Standardize event notification priorities, p.32.
135*c*-7. Define Abort reason when insufficient segments are available, p.38.
135*c*-8. Add new Error Codes and specify usage, p.39.

In the following document, language to be added to existing clauses of ANSI/ASHRAE 135-2001 and Addenda is indicated through the use of *italics* while deletions are indicated by ~~strikethrough~~. Where entirely new subclauses are to be added, plain type is used throughout.

135*c*-1. Allow Life Safety objects to advertise supported modes.

Addendum 135*c*-1

[Change **Table 12-17**, p. 184]

Table 12-17. Properties of the Life Safety Point Object Type

Property Identifier	Property Datatype	Conformance Code
…	…	…
Out_Of_Service	BOOLEAN	R
Mode	BACnetLifeSafetyMode	W
Accepted_Modes	*List of BACnetLifeSafetyMode*	*R*
Time_Delay	Unsigned	O[2]
…	…	…

[Add a new **Clause 12.14.13**, p. 186, and renumber the existing Clause 12.14.13 and subsequent clauses]

12.14.13 Accepted_Modes

This read-only property, of type List of BACnetLifeSafetyMode, shall specify all values the Mode property accepts when written to using BACnet services. Even though a mode is listed in this property, the write may be denied by the object due to the internal state of the object at that time. The value of the Accepted_Modes property does not depend on the internal state of the object and shall not change when the internal state changes. If a write is denied, a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned. Internal computation in the object may set the Mode property to a value other than one of those listed in the Accepted_Modes property.

[Change **Table 12-18**, p. 190]

Table 12-18. Properties of the Life Safety Zone Object Type

Property Identifier	Property Datatype	Conformance Code
…	…	…
Out_Of_Service	BOOLEAN	R
Mode	BACnetLifeSafetyMode	W
Accepted_Modes	*List of BACnetLifeSafetyMode*	*R*
Time_Delay	Unsigned	O[2]
…	…	…

[Add a new **Clause 12.15.13**, p. 192, and renumber the existing Clause 12.15.13 and subsequent clauses]

12.15.13 Accepted_Modes

This read-only property, of type List of BACnetLifeSafetyMode, shall specify all values the Mode property accepts when written to using BACnet services. Even though a mode is listed in this property, the write may be denied by the object due to the internal state of the object at that time. The value of the Accepted_Modes property does not depend on the internal state of the object and shall not change when the internal state changes. If a write is denied, a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned. Internal computation in the object may set the Mode property to a value other than one of those listed in the Accepted_Modes property.

[Change **Annex C**, pp. 433]

LIFE-SAFETY-POINT ::= SEQUENCE {

...		
out-of-service	[81]	BOOLEAN,
mode	[160]	BACnetLifeSafetyMode,
accepted_modes	*[175]*	*SEQUENCE OF BACnetLifeSafetyMode,*
time-delay	[113]	Unsigned OPTIONAL,
...		
}		

[Change **Annex C**, pp. 434]

LIFE-SAFETY-ZONE ::= SEQUENCE {

...		
out-of-service	[81]	BOOLEAN,
mode	[160]	BACnetLifeSafetyMode,
accepted_modes	*[175]*	*SEQUENCE OF BACnetLifeSafetyMode,*
time-delay	[113]	Unsigned OPTIONAL,
...		
}		

[Change **Annex D.14**, p.449]

D.14 Example of a Life Safety Point Object

...		
Property:	Mode =	ON
Property:	*Accepted_Modes =*	*{ENABLED, DISABLED, TEST}*
Property:	Time_Delay =	10
...		

[Change **Annex D.15**, pp. 449-450]

D.15 Example of a Life Safety Zone Object

...		
Property:	Mode =	ON
Property:	*Accepted_Modes =*	*{ENABLED, DISABLED, TEST}*
Property:	Time_Delay =	10
...		

135*c*-2. Add Unsilence Options to the LifeSafetyOperation Service.

Addendum 135*c*-2

[Change **Clause 13.13**, p. 265]

13.13 LifeSafetyOperation Service

The LifeSafetyOperation service is intended for use in fire, life safety and security systems to provide a mechanism for conveying specific instructions from a human operator to *accomplish any of the following objectives*:

(a) silence audible or visual notification appliances, or
(b) reset latched notification ~~appliances.~~ *appliances, or*
(c) unsilence previously silenced audible or visual notification appliances.

[Change **Clause 13.13.1.1.3**, p. 265]

13.13.1.1.3 Request

This parameter, of type BACnetLifeSafetyOperation, shall convey the requested operation:

{SILENCE, SILENCE_AUDIBLE, SILENCE_VISUAL, RESET, RESET_ALARM, ~~RESET_FAULT}~~
RESET_FAULT, UNSILENCE, UNSILENCE_AUDIBLE, UNSILENCE_VISUAL}

[Change BACnetLifeSafetyOperation production in **Clause 21**, p. 395]

```
BACnetLifeSafetyOperation ::= ENUMERATED {
    none                (0),
    silence             (1),
    silence-audible     (2),
    silence-visual      (3),
    reset               (4),
    reset-alarm         (5),
    reset-fault         (6),
    unsilence           (7),
    unsilence-audible   (8),
    unsilence-visual    (9),
    ...
    }
-- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
-- 64-65535 may be used by others subject to procedures and constraints described in
-- Clause 23.
```

135*c*-3. Specify the relationship between the Event_Type and Event_Parameter properties.

Addendum 135*c*-3

[Change **Clause 12.11.5**, p. 176]

12.11.5 Event_Type

This *read-only* property, of type BACnetEventType, indicates the type of event algorithm that is to be used to detect the occurrence of events and report to enrolled devices. This parameter is an enumerated type that may have any of the following values:

{CHANGE_OF_BITSTRING, CHANGE_OF_STATE, CHANGE_OF_VALUE, COMMAND_FAILURE, FLOATING_LIMIT, OUT_OF_RANGE, BUFFER_READY, CHANGE_OF_LIFE_SAFETY}.

There is a specific relationship between each event algorithm, the parameter list, and the event types that are valid for the event. The Event_Type ~~implies an~~ *reflects the* algorithm that is used to determine the state of an event. The algorithm for each Event_Type is specified in Clause 13. The Event_Parameters property provides the parameters needed by the algorithm.

…

[Change **Clause 12.11.7**, p.176]

12.11.7 Event_Parameters

~~Each Event_Type determines the method to be used to monitor the referenced object.~~ The Event_Parameters property, of type BACnetEventParameter, *determines the algorithm used to monitor the referenced object and* provides the parameter values needed for this algorithm. The meaning of each value in the Event_Parameters~~, therefore,~~ depends on *the algorithm as indicated by* the Event_Type ~~as shown in~~ *column* in Table 12-14. Each of the possible parameters is described below.

…

135*c*-4. Add a new Accumulator Object Type.

Addendum 135*c*-4

[Add a new **Clause 12.1**, p. 130, and renumber the existing Clause 12.1 and subsequent clauses, including tables and figures]

12.1 Accumulator Object Type

The Accumulator object type defines a standardized object whose properties represent the externally visible characteristics of a device that indicates measurements made by counting pulses.

This object maintains precise measurement of input count values, accumulated over time. The accumulation of pulses represents the measured quantity in unsigned integer units. This object is also concerned with the accurate representation of values presented on meter read-outs. This includes the ability to initially set the Present_Value property to the value currently displayed by the meter (as when the meter is installed), and to duplicate the means by which it is advanced, including simulating a modulo-N divider prescaling the actual meter display value, as shown in Figure 12-1.

Typical applications of such devices are in peak load management and in accounting and billing management systems. This object is not intended to meet all such applications. Its purpose is to provide information about the quantity being measured, such as electric power, water, or natural gas usage, according to criteria specific to the application.

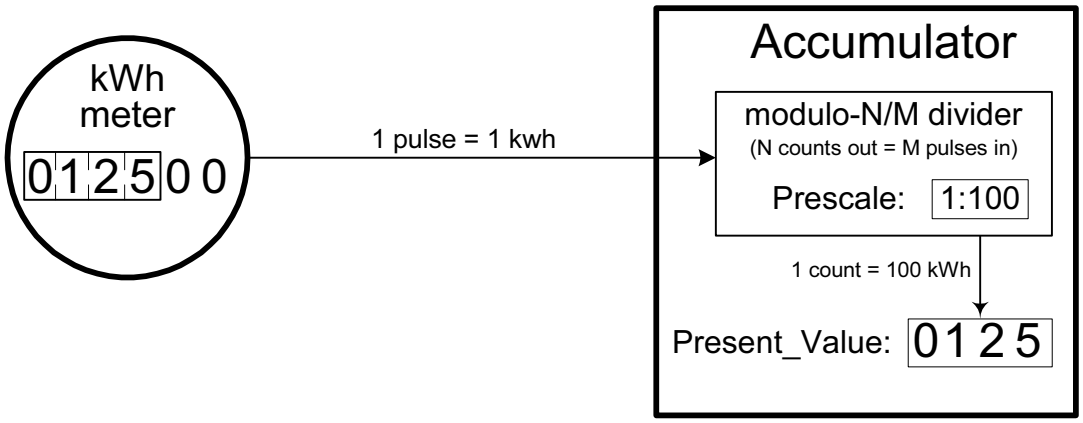

Figure 12-1. Example of an Accumulator object

The object and its properties are summarized in Table 12-1 and described in detail in this subclause.

Table 12-1. Properties of the Accumulator Object

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	Unsigned	R[1]
Description	CharacterString	O
Device_Type	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Scale	BACnetScale	R
Units	BACnetEngineeringUnits	R
Prescale	BACnetPrescale	O
Max_Pres_Value	Unsigned	R
Value_Change_Time	BACnetDateTime	O[2]
Value_Before_Change	Unsigned	O[2,3]
Value_Set	Unsigned	O[2,3]
Logging_Record	BACnetAccumulatorRecord	O
Logging_Object	BACnetObjectIdentifier	O
Pulse_Rate	Unsigned	O[1,4]
High_Limit	Unsigned	O[4]
Low_Limit	Unsigned	O[4]
Limit_Monitoring_Interval	Unsigned	O[4]
Notification_Class	Unsigned	O[4]
Time_Delay	Unsigned	O[4]
Limit_Enable	BACnetLimitEnable	O[4]
Event_Enable	BACnetEventTransitionBits	O[4]
Acked_Transitions	BACnetEventTransitionBits	O[4]
Notify_Type	BACnetNotifyType	O[4]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4]
Profile_Name	CharacterString	O

[1] This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if either Value_Before_Change or Value_Set is writable.
[3] Either Value_Before_Change or Value_Set may be writable, but not both.
[4] These properties are required if the object supports intrinsic reporting.

12.1.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet Device that maintains it.

12.1.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet Device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.1.3 Object_Type

This property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be ACCUMULATOR.

12.1.4 Present_Value

This property, of type Unsigned, indicates the count of the input pulses, prescaled if the Prescale property is present, acquired since the value was most recently set by writing to the Value_Set property.

The value of this property shall remain in the range from zero through Max_Pres_Value. All operations on the Present_Value property are performed modulo (Max_Pres_Value+1).

This property shall be writable when Out_Of_Service is TRUE.

12.1.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.1.6 Device_Type

This property, of type CharacterString, is a text description of the physical device represented by the Accumulator object. It will typically be used to describe the type of sensor represented by the Accumulator.

12.1.7 Status_Flags

This property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of an Accumulator object. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the Present_Value and Reliability properties are no longer tracking changes to the physical input. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

12.1.8 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 and if the Reliability property is not present, then the value of this property shall be NORMAL. If the Reliability property is present and does not have a

value of NO_FAULT_DETECTED, then the value of the Event_State property shall be FAULT. Changes in the Event_State property to the value FAULT are considered to be “fault” events.

12.1.9 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value property 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 Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, NO_SENSOR, OVER_RANGE, UNDER_RANGE, OPEN_LOOP, SHORTED_LOOP, UNRELIABLE_OTHER}

12.1.10 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 and Pulse_Rate properties are 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, Pulse_Rate 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, Pulse_Rate 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.

12.1.11 Scale

This property, of type BACnetScale, indicates the conversion factor to be multiplied with the value of the Present_Value property to provide a value in the units indicated by Units. The choice of options for this property determine how the scaling operation (which is performed by the client reading this object) is performed:

Option	Datatype	Indicated Value in Units
floatScale	REAL	Present_Value x Scale
integerScale	INTEGER	Present_Value x 10^{Scale}

12.1.12 Units

This property, of type BACnetEngineeringUnits, indicates the measurement units of the Present_Value when multiplied with the scaling factor indicated by Scale. See the BACnetEngineeringUnits ASN.1 production in Clause 21 for a list of engineering units defined by this standard.

12.1.13 Prescale

This property, of type BACnetPrescale, presents the coefficients that are used for converting the pulse signals generated by the measuring instrument into the value displayed by Present_Value. The conversions are performed using integer arithmetic in such a fashion that no measurement-generated pulse signals are lost in the conversion.

These coefficients might simply document a conversion performed prior to the reception of the input pulses by the Accumulator object, or they might actually be used by the Accumulator to convert input pulses into the value displayed by Present_Value. Whichever is done is a local matter.

The coefficients are as follows:

multiplier	The numerator of the conversion factor expressed as a ratio of integers.
moduloDivide	The denominator of the conversion factor expressed as a ratio of integers.

The conversion algorithm is performed as follows, utilizing a non-displayed variable called an accumulator:

For each input pulse:

Add the value of 'multiplier' to an accumulator and then,
while the accumulator is greater than or equal to the value of 'moduloDivide':
Increment the value of Present_Value by one, and
decrease the value of the accumulator by the value of 'moduloDivide'.

This procedure supports non-integral ratios of measurement pulses to Present_Value. For example, in an electrical metering application, the output of the voltage- and current-measuring systems might be 9000/1200 (scale / voltage*current) pulses per kWh, requiring the Accumulator object to accumulate 2/15 kWh/pulse. With this algorithm such pulses can be accurately accumulated and displayed when the units of Present_Value are KILOWATT_HOURS.

12.1.14 Max_Pres_Value

This property, of type Unsigned, indicates the maximum value of the Present_Value property.

12.1.15 Value_Change_Time

This read-only property, of type BACnetDateTime, shall be present if the Present_Value property is adjustable by writing to the Value_Before_Change or Value_Set properties. It represents the date and time of the most recent occurrence of such a write operation. If no such write has yet occurred, this property shall have wildcard values for all date and time fields.

12.1.16 Value_Before_Change

This property, of type Unsigned, indicates the value of the Present_Value property just prior to the most recent write to the Value_Set or Value_Before_Change properties. If no such write has yet occurred, this property shall have the value zero. If this property is writable, the Value_Set property shall be read-only.

If this property is writable, the following series of operations, for which the associated properties are present, shall be performed atomically by the object when this property is written:
(1) The value of Present_Value shall be copied to the Value_Set property.
(2) The value written to Value_Before_Change shall be stored in the Value_Before_Change property.
(3) The current date and time shall be stored in the Value_Change_Time property.

While this series of operations is being performed, it is critical that any other process not change the Present_Value, Value_Set and Value_Before_Change properties.

12.1.17 Value_Set

This property, of type Unsigned, indicates the value of the Present_Value property after the most recent write to the Value_Set or Value_Before_Change properties. If no such write has yet occurred, this property shall have the value zero. If this property is writable, the Value_Before_Change property shall be read-only.

If this property is writable, the following series of operations, for which the associated properties are present, shall be performed atomically by the object when this property is written:
(1) The value of Present_Value shall be copied to the Value_Before_Change property.
(2) The value written to Value_Set shall be stored in both the Value_Set and Present_Value properties.
(3) The current date and time shall be stored in the Value_Change_Time property.

While this series of operations are being performed, it is critical that any other process not change the Present_Value, Value_Set and Value_Before_Change properties.

12.1.18 Logging_Record

This read-only property, of type BACnetAccumulatorRecord, is a list of values that must be acquired and returned "atomically" in order to allow proper interpretation of the data.

If the Logging_Object property is present, then, when Logging_Record is acquired by the object identified by Logging_Object, this list of values shall be saved and returned when read by other objects or devices. If the Logging_Object property is present and Logging_Record has not yet been acquired by the object identified by Logging_Object, 'timestamp' shall contain all wildcards, 'present-value' and 'accumulated-value" shall contain the value zero, and 'accumulator-status' shall indicate STARTING.

The list of values ('timestamp', 'present-value', 'accumulated-value', and 'accumulator-status') shall be acquired from the underlying system when they reflect a stable state of the device (for example, they shall not be acquired when Present_Value has just been incremented but the corresponding increment of 'accumulated-value' has not yet occurred).

The items returned in the list of values are:

timestamp — The local date and time when the data was acquired.

present-value — The value of the Present_Value property.

accumulated-value — The short term accumulated value of the counter. The algorithm used to calculate accumulated-value is a function of the value of accumulator-status. If this is the initial read, the value returned shall be zero.

accumulator-status — An indication of the reliability of the data in this list of values.

The accumulator-status parameter may take on any of the following values:

{NORMAL, STARTING, RECOVERED, ABNORMAL, FAILED}

where the values are defined as follows:

NORMAL — No event affecting the reliability of the data has occurred during the period from the preceding to the current qualified reads of the Logging_Record property. In this case 'accumulated-value' shall be represented by the expression:

$$\text{(accumulated-value)} = (\text{Present\_Value}_{\text{current}}) - (\text{Present\_Value}_{\text{previous}})$$

STARTING — This value indicates that the data in Logging_Records is either the first data to be acquired since startup by the object identified by Logging_Object (if 'timestamp' has non-wildcard values) or that no data has been acquired since startup by the object identified by Logging_Object (in which case 'timestamp' has all wildcard values).

RECOVERED — One or more writes to Value_Before_Change or Value_Set have occurred since Logging_Record was acquired by the object identified by Logging_Object. For the case of a single write, 'accumulated-value' shall be represented by the expression:

$$\text{'accumulated-value'} = (\text{Present\_Value}_{\text{current}} - \text{Value\_Set}) + (\text{Value\_Before\_Change} - \text{Present\_Value}_{\text{previous}})$$

ABNORMAL — The accumulation has been carried out, but some unrecoverable event such as the clock's time being changed by a significant amount since Logging_Record was acquired by the object identified by Logging_Object. (How much time is considered significant shall be a local matter.)

FAILED — The 'accumulation-value' is not reliable due to some problem. The criteria for returning this value are a local matter.

Changes in the value of 'accumulator-status' shall occur only when the Logging_Record is acquired by the object identified by Logging_Object.

12.1.19 Logging_Object

This property, of type BACnetObjectIdentifier, indicates the object in the same device as the Accumulator object which, when it acquires Logging_Record data from the Accumulator object, shall cause the Accumulator object to acquire, present and store the data from the underlying system.

12.1.20 Pulse_Rate

This property, of type Unsigned, shall indicate the number of input pulses received during the most recent period specified by Limit_Monitoring_Interval. The mechanism that associates the input signal with the value indicated by this property is a local matter.

This property shall be writable when Out_Of_Service is TRUE.

12.1.21 High_Limit

This property, of type Unsigned, shall specify a limit that Pulse_Rate must exceed before an event is generated. This property is required if this object supports intrinsic reporting.

12.1.21.1 Conditions for Generating a TO-OFFNORMAL Event

A TO-OFFNORMAL event is generated under these conditions:

(a) Pulse_Rate must exceed High_Limit for a minimum period of time, specified in the Time_Delay property, and
(b) the HighLimitEnable flag must be set in the Limit_Enable property, and
(c) the TO-OFFNORMAL flag must be set in the Event_Enable property.

12.1.21.2 Conditions for Generating a TO-NORMAL Event

Once exceeded, Pulse_Rate must fall below High_Limit before a TO-NORMAL event is generated under these conditions:

(a) Pulse_Rate must remain below High_Limit for a minimum period of time, specified in the Time_Delay property, and
(b) the HighLimitEnable flag must be set in the Limit_Enable property, and
(c) the TO-NORMAL flag must be set in the Event_Enable property.

12.1.22 Low_Limit

This property, of type Unsigned, shall specify a limit that Pulse_Rate must fall below before an event is generated. This property is required if this object supports intrinsic reporting.

12.1.22.1 Conditions for Generating a TO-OFFNORMAL Event

A TO-OFFNORMAL event is generated under these conditions:

(a) Pulse_Rate must fall below Low_Limit for a minimum period of time, specified in the Time_Delay property, and
(b) the LowLimitEnable flag must be set in the Limit_Enable property, and
(c) the TO-OFFNORMAL flag must be set in the Event_Enable property.

12.1.22.2 Conditions for Generating a TO-NORMAL Event

Once Pulse_Rate has fallen below the Low_Limit, the Pulse_Rate must become greater than Low_Limit before a TO-NORMAL event is generated under these conditions:

(a) Pulse_Rate must become greater than Low_Limit for a minimum period of time, specified in the Time_Delay property, and
(b) the LowLimitEnable flag must be set in the Limit_Enable property, and
(c) the TO-NORMAL flag must be set in the Event_Enable property.

12.1.23 Limit_Monitoring_Interval

This property, of type Unsigned, specifies the monitoring period in seconds for determining the value of Pulse_Rate. The use of a fixed or sliding time window for detecting pulse rate is a local matter. This property is required if this object supports intrinsic reporting.

12.1.24 Notification_Class

This property, of type Unsigned, shall specify the notification class to be used when handling and generating event notifications for this object. The Notification_Class property implicitly refers to a Notification Class object that has a Notification_Class property with the same value. This property is required if this object supports intrinsic reporting.

12.1.25 Time_Delay

This property, of type Unsigned, shall specify the minimum period of time in seconds that Pulse_Rate must remain outside the range from Low_Limit through High_Limit, before a TO-OFFNORMAL event is generated, or within the same band before a TO-NORMAL event is generated. This property is required if this object supports intrinsic reporting.

12.1.26 Limit_Enable

This property, of type BACnetLimitEnable, shall convey two flags that separately enable and disable reporting of High_Limit and Low_Limit offnormal events and their return to normal. This property is required if this object supports intrinsic reporting.

12.1.27 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable reporting of TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events. In the context of Accumulator objects, transitions to the High_Limit or Low_Limit Event_States are considered to be "offnormal" events. This property is required if this object supports intrinsic reporting.

12.1.28 Acked_Transitions

This property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the receipt of acknowledgements for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events. In the context of Accumulator objects, transitions to High_Limit and Low_Limit Event_State are considered to be "offnormal" events. These flags shall be cleared upon the occurrence of the corresponding event and set under any of these conditions:

(a) upon receipt of the corresponding acknowledgement;
(b) upon the occurrence of the event if the corresponding flag is not set in the Event_Enable property (meaning event notifications will not be generated for this condition and thus no acknowledgement is expected);
(c) upon the occurrence of the event if the corresponding flag is set in the Event_Enable property and the corresponding flag in the Ack_Required property of the Notification Class object implicitly referenced by the Notification_Class property of this object is not set (meaning no acknowledgement is expected).

This property is required if this object supports intrinsic reporting.

12.1.29 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. This property is required if this object supports intrinsic reporting.

12.1.30 Event_Time_Stamps

This optional property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last event notifications for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively. Time stamps of type Time or Date shall have X'FF' in each octet and Sequence number time stamps shall have the value 0 if no event notification of that type has been generated since the object was created. This property is required if this object supports intrinsic reporting.

12.1. 31 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name must begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile document named by the remainder of the profile name. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. The definition of the profiles themselves is outside the scope of this standard.

[Change **Table 13-2** p. 237]

Table 13-2. Standard Objects ~~that~~ *That* May Support Intrinsic Reporting

Object Type	Criteria	Event Type
Accumulator	*If Pulse_Rate exceeds range from Low_Limit through High_Limit for longer than Time_Delay AND the new transition is enabled in Event_Enable and Limit_Enable,* *OR* *Pulse_Rate returns to range from Low_Limit through High_Limit for longer than Time_Delay AND the new transition is enabled in Event_Enable and Limit_Enable*	*UNSIGNED_RANGE*
Analog Input, Analog Output, Analog Value*,*	If Present_Value exceeds range between High_Limit and Low_Limit for longer than Time_Delay AND the new transition is enabled in Event_Enable and Limit_Enable, OR Present__Value returns within the High_Limit – Deadband to Low_Limit + Deadband range for longer than Time_Delay AND the new transition is enabled in Event_Enable and Limit_Enable	OUT_OF_RANGE
…		

[Change **Table 13-3** p. 238]

Table 13-3. Standard Object Property Values Returned in Notifications

Object	Event Type	Notification Parameters	Referenced Object's Properties
Accumulator	*UNSIGNED_RANGE*	*Exceeding_Value* *Status_Flags* *Exceeded_Limit*	*Pulse_Rate* *Status_Flags* *Low_Limit or High_Limit*
Analog Input, Analog Output, Analog Value	OUT_OF_RANGE	Exceeding_Value Status_Flags Deadband Exceeded_Limit	Present_Value Status_Flags Deadband Low_Limit or High_Limit
…			

[Change **Table 13-4**, p. 238]

Table 13-4. Notification Parameters for Standard Event Types

Event Type	Notification Parameters	Description
…		
CHANGE_OF_LIFE_SAFETY	New_State New_Mode Status_Flags Operation_Expected	The new value of the referenced property The new mode of the referenced object The Status_Flags of the referenced object The next operation requested by the referenced object
UNSIGNED_RANGE	*Exceeding_Value* *Status_Flags* *Exceeded_Limit*	*The value that exceeded a limit* *The Status_Flags of the referenced object* *The limit that was exceeded*

[Change **Clause 13.3**, p.239]

13.3 Algorithmic Change Reporting

…

The following event type algorithms are specified in this standard because of their widespread occurrence in building automation and control systems. They are:

(a) CHANGE_OF_BITSTRING
(b) CHANGE_OF_STATE
(c) CHANGE_OF_VALUE
(d) COMMAND_FAILURE
(e) FLOATING_LIMIT
(f) OUT_OF_RANGE
(g) BUFFER_READY
(h) CHANGE_OF_LIFE_SAFETY
(i) UNSIGNED_RANGE

…

[Add **Clause 13.3.9**, p. 245**]**

13.3.9 UNSIGNED_RANGE Algorithm

An UNSIGNED_RANGE occurs if the referenced property leaves the range of values from Low_Limit through High_Limit parameters and remains there for Time_Delay seconds. If the transition is to a value above High_Limit or below Low_Limit, the Event Enrollment object generates a TO-OFFNORMAL transition. The event notification shall show an 'Event Type' of UNSIGNED_RANGE.

An UNSIGNED_RANGE clears when the referenced property attains a value from Low_Limit through High_Limit and remains there for Time_Delay seconds. The Event Enrollment object generates a TO-NORMAL transition. The event notification shall show an 'Event Type' of UNSIGNED_RANGE. See Figure 13-10.

[Add new Figure **13-10**, p.245, and renumber existing Figure 13-10 and subsequent figures]

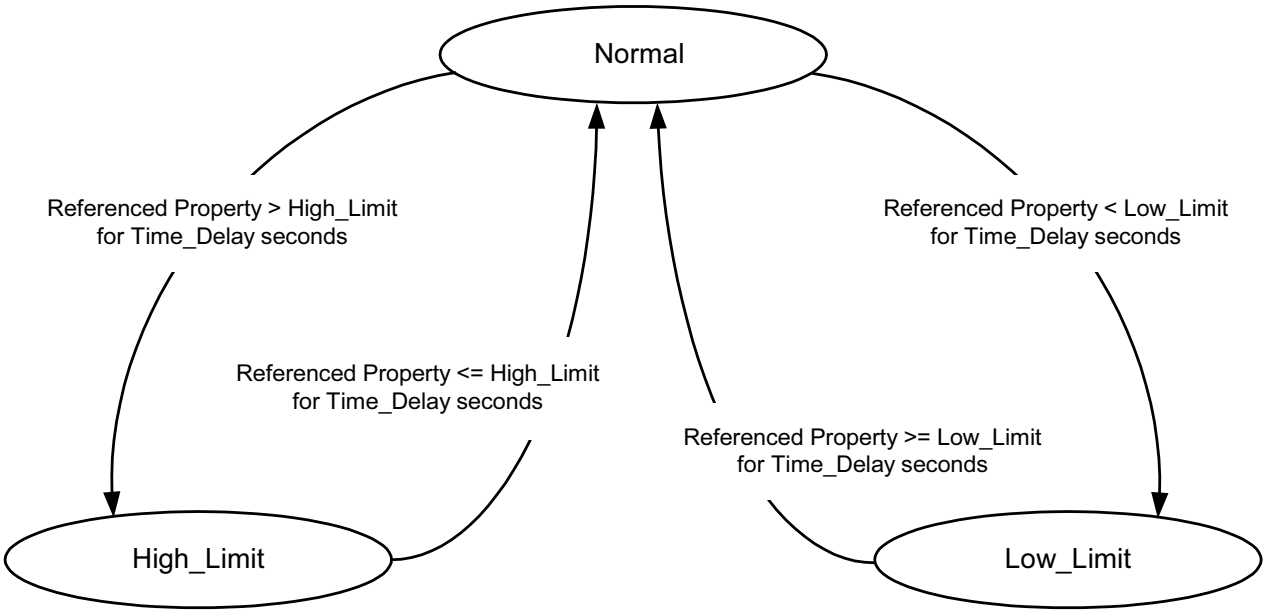

Figure 13-10. UNSIGNED_RANGE algorithm.

[Add new BACnetAccumulatorRecord production to **Clause 21**, p. 387]

```
BACnetAccumulatorRecord ::= SEQUENCE {
    timestamp           [0] BACnetDateTime,
    presentValue        [1] Unsigned,
    accumulatedValue    [2] Unsigned,
    accumulatortStatus  [3] ENUMERATED {
                            normal      (0),
                            starting    (1),
                            recovered   (2),
                            abnormal    (3),
                            failed      (4)
                            }
    }
```

[Change BACnetEventParameter production in **Clause 21**, p.393]

BACnetEventParameter ::= CHOICE {

-- These choices have a one-to-one correspondence with Event_Type enumeration
…
change-of-life-safety [8] SEQUENCE {
time-delay [0] Unsigned,
list-of-life-safety-alarm-values [1] SEQUENCE OF BACnetLifeSafetyState,
list-of-alarm-values [2] SEQUENCE OF BACnetLifeSafetyState,
mode-property-reference [3] BACnetDeviceObjectPropertyReference
~~,~~ *},*
-- *enumeration 9 is used in Addendum a to ANSI/ASHRAE 135-2001(135a-2)*
-- *enumeration 10 is used in Addendum b to ANSI/ASHRAE 135-2001(135b-1)*
unsigned-range [11] SEQUENCE {
time-delay [0] Unsigned,
low-limit [1] Unsigned,
high-limit [2] Unsigned
}
}

-- CHOICE [6] has been intentionally omitted. It parallels the complex-event-type CHOICE [6] of the
-- BACnetNotificationParameters production which was introduced to allow the addition of proprietary event
-- algorithms whose event parameters are not necessarily network-visible.

[Change BACnetEventType production in **Clause 21**, p.394]

BACnetEventType ::= ENUMERATED {
change-of-bitstring (0),
change-of-state (1),
change-of-value (2),
command-failure (3),
floating-limit (4),
out-of-range (5),
-- complex-event-type (6), -- see comment below
buffer-ready (7),
change-of-life-safety (8),
-- *enumeration 9 is used in Addendum a to ANSI/ASHRAE 135-2001(135a-2)*
-- *enumeration 10 is used in Addendum b to ANSI/ASHRAE 135-2001(135b-1)*
unsigned-range (11),
...
}
-- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
-- 64-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23. It is expected that these enumerated values will correspond to the use of the
-- complex-event-type CHOICE [6] of the BACnetNotificationParameters production.
-- The last enumeration used in this version is ~~8~~*. 11.*

[Change BACnetNotificationParameters production in **Clause 21**, p.396]

BACnetNotificationParameters ::= CHOICE {
-- These choices have a one-to-one correspondence with Event_Type enumeration
…
change-of-life-safety [8] SEQUENCE {
new-state [0] BACnetLifeSafetyState,
new-mode [1] BACnetLifeSafetyMode,

```
                    status-flags            [2] BACnetStatusFlags,
                    operation-expected      [3] BACnetLifeSafetyOperation
                    } },
    -- enumeration 9 is used in Addendum a to ANSI/ASHRAE 135-2001(135a-2)
    -- enumeration 10 is used in Addendum b to ANSI/ASHRAE 135-2001(135b-1)
    unsigned-range      [11] SEQUENCE {
                    exceeding-value     [0] Unsigned,
                    status-flags        [1] BACnetStatusFlags,
                    exceeded-limit      [2] Unsigned
                    }
    }
```

[Add new BACnetPrescale production to **Clause 21**, p. 399]

```
BACnetPrescale ::= SEQUENCE {
    multiplier          [0] Unsigned,
    moduloDivide        [1] Unsigned
    }
```

[Add new BACnetScale production to **Clause 21**, p. 405]

```
BACnetScale ::= CHOICE {
    floatScale          [0] REAL,
    integerScale        [1] INTEGER
    }
```

[Add to **Annex C**, p. 428]

ANNEX C - FORMAL DESCRIPTION OF OBJECT TYPE STRUCTURES (INFORMATIVE)

(This annex is not part of this standard but is included for informative purposes only.)

```
ACCUMULATOR ::= SEQUENCE {
    object-identifier           [75]    BACnetObjectIdentifier,
    object-name                 [77]    CharacterString,
    object-type                 [79]    BACnetObjectType,
    present-value               [85]    Unsigned,
    description                 [28]    CharacterString OPTIONAL,
    device-type                 [31]    CharacterString OPTIONAL,
    status-flags                [111]   BACnetStatusFlags,
    event-state                 [36]    BACnetEventState,
    reliability                 [103]   BACnetReliability OPTIONAL,
    out-of-service              [81]    BOOLEAN,
    scale                       [186]   BACnetScale,
    units                       [117]   BACnetEngineeringUnits,
    prescale                    [188]   BACnetPrescale OPTIONAL,
    max-pres-value              [65]    Unsigned,
    value-change-time           [192]   BACnetDateTime OPTIONAL,
    value-before-change         [190]   Unsigned OPTIONAL,
    value-set                   [191]   Unsigned OPTIONAL,
    logging-record              [184]   BACnetAccumulatorRecord OPTIONAL,
    logging-device              [183]   BACnetRecipient OPTIONAL,
    pulse-rate                  [186]   Unsigned OPTIONAL,
    high-limit                  [45]    Unsigned OPTIONAL,
    low-limit                   [59]    Unsigned OPTIONAL,
    limit-monitoring-interval   [182]   Unsigned OPTIONAL,
```

event-type *[37]* *BACnetEventType,*
notification-class *[17]* *Unsigned OPTIONAL,*
time-delay *[113]* *Unsigned OPTIONAL,*
limit-enable *[52]* *BACnetLimitEnable OPTIONAL,*
event-enable *[35]* *BACnetEventTransitionBits OPTIONAL,*
acked-transitions *[0]* *BACnetEventTransitionBits OPTIONAL,*
notify-type *[72]* *BACnetNotifyType OPTIONAL,*
event-time-stamps *[130]* *SEQUENCE OF BACnetTimeStamp OPTIONAL,*
profile-name *[167]* *CharacterString OPTIONAL*
}

ANALOG-INPUT ::= SEQUENCE {
…

[Add new **Clause D.1**, p. 439, and renumber existing Clause D.1 and subsequent clauses]

D.1 Example of an Accumulator object

Property:	Object_Identifier =	(Accumulator, Instance 1)
Property:	Object_Name =	"Tenant 1"
Property:	Object_Type =	ACCUMULATOR
Property	Present_Value =	323
Property:	Description =	""
Property:	Device_Type =	"Electric Pulse"
Property:	Status_Flags =	{FALSE, FALSE, FALSE, FALSE}
Property:	Event_State =	NORMAL
Property:	Out_Of_Service =	FALSE
Property:	Scale =	2
Property:	Units =	KILOWATT_HOURS
Property:	Prescale =	(1,10000)
Property:	Max_Pres_Value =	9999
Property:	Value_Change_Time =	(23-MAR-01,18:50:21.2)
Property:	Value_Before_Change =	0
Property:	Value_Set =	67
Property:	Logging_Record =	((13-JUL-01,13:00:00.2),120,1,NORMAL)
Property:	Logging_Object =	(Trend Log Multiple, Instance 100)
Property:	Pulse_Rate =	3
Property:	High_Limit =	15
Property:	Low_Limit =	0
Property:	Limit_Monitoring_Interval =	300
Property:	Event_Type =	UNSIGNED_RANGE
Property:	Notification_Class =	3
Property:	Limit_Enable =	{TRUE, FALSE}
Property:	Event_Enable =	{TRUE, FALSE, TRUE}
Property:	Acked_Transitions =	{TRUE, TRUE, TRUE}
Property:	Notify_Type =	ALARM
Property:	Event_Time_Stamps =	((12-JUL-01,18:50:21.2), (*-*-*,*:*:*.*), (12-JUL-01,19:01:34.0))

135*c*-5. Add a new Pulse Converter Object Type.

Addendum 135*c*-5

[Add a new **Clause 12.23**, p. 224, and renumber the existing Clause 12.22 and subsequent clauses including tables and figures]

12.23 Pulse Converter Object Type

The Pulse Converter object type defines a standardized object that represents a process whereby ongoing measurements made of some quantity, such as electric power or water or natural gas usage, and represented by pulses or counts, might be monitored over some time interval for applications such as peak load management, where it is necessary to make periodic measurements but where a precise accounting of every input pulse or count is not required.

The Pulse Converter object might represent a physical input. As an alternative, it might acquire the data from the Present_Value of an Accumulator object, representing an input in the same device as the Pulse Converter object. This linkage is illustrated by the dotted line in Figure 12-3. Every time the Present_Value property of the Accumulator object is incremented, the Count property of the Pulse Converter object is also incremented.

The Present_Value property of the Pulse Converter object can be adjusted at any time by writing to the Adjust_Value property, which causes the Count property to be adjusted, and the Present_Value recomputed from Count. In the illustration in Figure 12-3, the Count property of the Pulse Converter was adjusted down to 0 when the Total_Count of the Accumulator object had the value 0070.

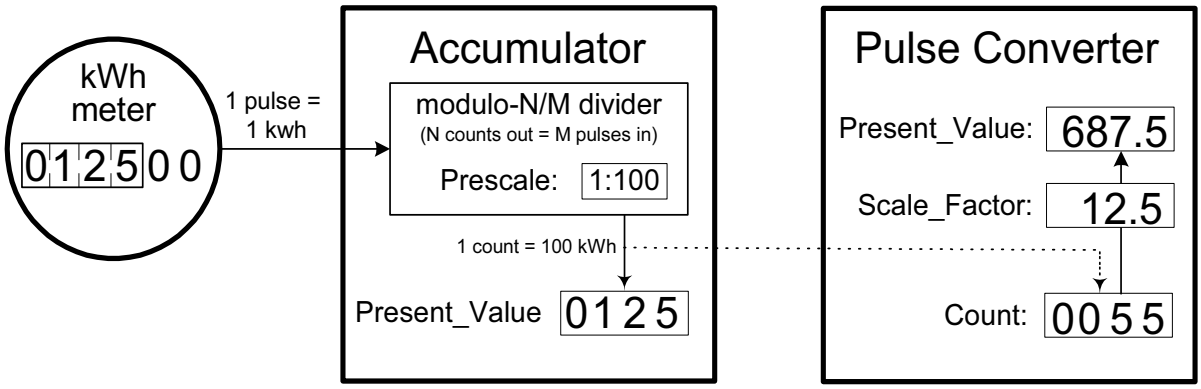

Figure 12-3. Relationship between the Pulse Converter and Accumulator objects.

The object and its properties are summarized in Table 12-27 and described in detail in this subclause.

Table 12-27. Properties of the Pulse Converter Object

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Present_Value	REAL	R[1]
Input_Reference	BACnetObjectPropertyReference	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Units	BACnetEngineeringUnits	R
Scale_Factor	REAL	R
Adjust_Value	REAL	W
Count	Unsigned	R
Update_Time	BACnetDateTime	R
Count_Change_Time	BACnetDateTime	R[2]
Count_Before_Change	Unsigned	R[2]
COV_Increment	REAL	O[3]
COV_Period	Unsigned	O[3]
Notification_Class	Unsigned	O[4]
Time_Delay	Unsigned	O[4]
High_Limit	REAL	O[4]
Low_Limit	REAL	O[4]
Deadband	REAL	O[4]
Limit_Enable	BACnetLimitEnable	O[4]
Event_Enable	BACnetEventTransitionBits	O[4]
Acked_Transitions	BACnetEventTransitionBits	O[4]
Notify_Type	BACnetNotifyType	O[4]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4]
Profile_Name	CharacterString	O

[1] This property is required to be writable when Out_Of_Service is TRUE.
[2] These properties are required if Count_Before_Change is writable.
[3] These properties are required if the object supports COV reporting.
[4] These properties are required if the object supports intrinsic reporting.

12.23.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet Device that maintains it.

12.23.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet Device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.23.3 Object_Type

This property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be PULSE_CONVERTER.

12.23.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.23.5 Present_Value

This property, of type REAL, indicates the accumulated value of the input being measured. It is computed by multiplying the current value of the Count property by the value of the Scale_Factor property. The value of the Present_Value property may be adjusted by writing to the Adjust_Value property. The Present_Value property shall be writable when Out_Of_Service is TRUE.

12.23.6 Input_Reference

This optional property, of type BACnetObjectPropertyReference, indicates the object and property (typically an Accumulator object's Present_Value property) representing the actual physical input that is to be measured and presented by the Pulse Converter object. The referenced property should have a datatype of INTEGER or Unsigned.

If this property is not present, the Pulse Converter object directly represents the physical input.

12.23.7 Status_Flags

This property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Pulse Converter. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical TRUE (1) if the point has been overridden by some mechanism local to the BACnet Device. In this context "overridden" is taken to mean that the Present_Value, Count and Reliability properties are no longer tracking changes to the input. Otherwise, the value is logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

12.23.8 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 and if the Reliability property is not present, then the value of this property shall be NORMAL. If the Reliability property is present and does not have a

value of NO_FAULT_DETECTED, then the value of the Event_State property shall be FAULT. Changes in the Event_State property to the value FAULT are considered to be “fault” events.

12.23.9 Reliability

The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value and/or Count properties 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 Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, NO_SENSOR, OVER_RANGE, UNDER_RANGE, OPEN_LOOP, SHORTED_LOOP, UNRELIABLE_OTHER, CONFIGURATION_ERROR}

If Input_Reference is configured to reference a property that is not of datatype Unsigned or INTEGER, or is otherwise not supported as an input source for this object, the Reliability property shall indicate CONFIGURATION_ERROR.

12.23.10 Out_Of_Service

The Out_Of_Service property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the input that the object directly represents, if any, is not in service. ("Directly represents" means that the Input_Reference property is not present in this object.) The Present_Value property is decoupled from the Count property and will not track changes to the 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 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 input.

If the Input_Reference property is present, the state of the Out_Of_Service property of the object referenced by Input_Reference shall not be indicated by the Out_Of_Service property of the Pulse Converter object.

12.23.11 Units

This property, of type BACnetEngineeringUnits, indicates the measurement units of the Present_Value property. See the BACnetEngineeringUnits ASN.1 production in Clause 21 for a list of engineering units defined by this standard.

12.23.12 Scale_Factor

This property, of type REAL, provides the conversion factor for computing Present_Value. It represents the change in Present_Value resulting from changing the value of Count by one.

12.23.13 Adjust_Value

This property, of type REAL, is written to adjust the Present_Value property (and thus the Count property also) by the amount written to Adjust_Value.

If this property is writable the following series of operations shall be performed atomically when this property is written:

(1) The value written to Adjust_Value shall be stored in the Adjust_Value property.
(2) The value of Count shall be copied to the Count_Before_Change property.
(3) The value of Count shall be decremented by the value calculated by performing the integer division (Adjust_Value/Scale_Factor) and discarding the remainder.
(4) The current date and time shall be stored in the Count_Change_Time property.

A write to this property results in a change in the value of Present_Value. Whether the new value is computed as part of the atomic series of operations or when Present_Value is read is a local matter.

If Adjust_Value has never been written, it shall have a value of zero.

12.23.14 Count

This read-only property, of type Unsigned, indicates the count of the input pulses as acquired from the physical input or the property referenced by the Input_Reference property.

If the property referenced by Input_Reference property is present, has datatype Unsigned or INTEGER, and is supported as an input source for this object, the value of the Count property is derived from the referenced property. An increment by one count in the referenced property is reflected by an increment of one count in the Count property. The means by which this is done shall be a local matter. Because the value of the Pulse Converter object Count property may be changed by a write to the Adjust_Value property, the value of the Count property can be different from the value of the referenced property.

12.23.15 Update_Time

This read-only property, of type BACnetDateTime, reflects the date and time of the most recent change to the Count property as a result of input pulse accumulation and is updated atomically with the Count property. If no such change has yet occurred, this property shall have wildcard values for all date and time fields.

12.23.16 Count_Change_Time

This read-only property, of type BACnetDateTime, represents the date and time of the most recent occurrence of a write to the Adjust_Value property. If no such write has yet occurred, this property shall have wildcard values for all date and time fields.

12.23.17 Count_Before_Change

This property, of type Unsigned, indicates the value of the Count property just prior to the most recent write to the Adjust_Value properties. If no such write has yet occurred, this property shall have the value zero.

12.23.18 COV_Increment

This property, of type REAL, shall specify the minimum change in Present_Value that will cause a COV notification to be issued to subscriber COV-clients. This property is required if COV reporting is supported by this object.

12.23.19 COV_Period

The COV_Period property, of type Unsigned, shall indicate the amount of time in seconds between the periodic COV notifications performed by this object. This property is required if COV reporting is supported by this object.

12.23.20 Notification_Class

This property, of type Unsigned, shall specify the notification class to be used when handling and generating event notifications for this object. The Notification_Class property implicitly refers to a Notification Class object that has a Notification_Class property with the same value. This property is required if intrinsic reporting is supported by this object.

12.23.21 Time_Delay

This property, of type Unsigned, shall specify the minimum period of time in seconds that the Present_Value must remain outside the band defined by the High_Limit and Low_Limit properties before a TO-OFFNORMAL event is generated or remain within the same band, including the Deadband property, before a TO-NORMAL event is generated. This property is required if intrinsic reporting is supported by this object.

12.23.22 High_Limit

This property, of type REAL, shall specify a limit that the Present_Value must exceed before an event is generated. This property is required if intrinsic reporting is supported by this object.

12.23.22.1 Conditions for Generating a TO-OFFNORMAL Event

A TO-OFFNORMAL event is generated under these conditions:

(a) the Present_Value must exceed the High_Limit for a minimum period of time, specified in the Time_Delay property, and
(b) the HighLimitEnable flag must be set in the Limit_Enable property, and
(c) the TO-OFFNORMAL flag must be set in the Event_Enable property.

12.23.22.2 Conditions for Generating a TO-NORMAL Event

Once exceeded, the Present_Value must fall below the High_Limit minus the Deadband before a TO-NORMAL event is generated under these conditions:

(a) the Present_Value must fall below the High_Limit minus the Deadband for a minimum period of time, specified in the Time_Delay property, and
(b) the HighLimitEnable flag must be set in the Limit_Enable property, and
(c) the TO-NORMAL flag must be set in the Event_Enable property.

12.23.23 Low_Limit

This property, of type REAL, shall specify a limit below which the Present_Value must fall before an event is generated. This property is required if intrinsic reporting is supported by this object.

12.23.23.1 Conditions for Generating a TO-OFFNORMAL Event

A TO-OFFNORMAL event is generated under these conditions:
(a) the Present_Value must fall below the Low_Limit for a minimum period of time, specified in the Time_Delay property, and
(b) the LowLimitEnable flag must be set in the Limit_Enable property, and
(c) the TO-OFFNORMAL flag must be set in the Event_Enable property.

12.23.23.2 Conditions for Generating a TO-NORMAL Event

Once the Present_Value has fallen below the Low_Limit, the Present_Value must exceed the Low_Limit plus the Deadband before a TO-NORMAL event is generated under these conditions:

(a) the Present_Value must exceed the Low_Limit plus the Deadband for a minimum period of time, specified in the Time_Delay property, and
(b) the LowLimitEnable flag must be set in the Limit_Enable property, and(c) the TO-NORMAL flag must be set in the Event_Enable property.

12.23.24 Deadband

This property, of type REAL, shall specify a range between the High_Limit and Low_Limit properties, which the Present_Value must remain within for a TO-NORMAL event to be generated under these conditions:

(a) the Present_Value must fall below the High_Limit minus Deadband, and
(b) the Present_Value must exceed the Low_Limit plus the Deadband, and
(c) the Present_Value must remain within this range for a minimum period of time, specified in the Time_Delay property, and
(d) either the HighLimitEnable or LowLimitEnable flag must be set in the Limit_Enable property, and
(e) the TO-NORMAL flag must be set in the Event_Enable property

This property is required if intrinsic reporting is supported by this object.

12.23.25 Limit_Enable

This property, of type BACnetLimitEnable, shall convey two flags that separately enable and disable reporting of HighLimit and LowLimit offnormal events and their return to normal. This property is required if intrinsic reporting is supported by this object.

12.23.26 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable reporting of TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events. In the context of Pulse Converter objects, transitions to the High_Limit or Low_Limit Event_States are considered to be "offnormal" events. This property is required if intrinsic reporting is supported by this object.

12.23.27 Acked_Transitions

This property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the receipt of acknowledgements for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events. In the context of Pulse Converter objects, transitions to High_Limit and Low_Limit Event_State are considered to be "offnormal" events. These flags shall be cleared upon the occurrence of the corresponding event and set under any of these conditions:

(a) upon receipt of the corresponding acknowledgement;
(b) upon the occurrence of the event if the corresponding flag is not set in the Event_Enable property (meaning event notifications will not be generated for this condition and thus no acknowledgement is expected);
(c) upon the occurrence of the event if the corresponding flag is set in the Event_Enable property and the corresponding flag in the Ack_Required property of the Notification Class object implicitly referenced by the Notification_Class property of this object is not set (meaning no acknowledgement is expected).

This property is required if intrinsic reporting is supported by this object.

12.23.28 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. This property is required if intrinsic reporting is supported by this object.

12.23.29 Event_Time_Stamps

This optional property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last event notifications for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively. Time stamps of type Time or Date shall have X'FF' in each octet and Sequence number time stamps shall have the value 0 if no event notification of that type has been generated since the object was created. This property is required if intrinsic reporting is supported by this object.

12.23.30 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name must begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile document named by the remainder of the profile name. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. The definition of the profiles themselves is outside the scope of this standard.

[Change **Clause 13.1**, p. 234]

13.1 Change of Value Reporting

Change of value (COV) reporting allows a COV-client to subscribe with a COV-server, on a permanent or temporary basis, to receive reports of some changes of value of some referenced property based on fixed criteria. If an object provides COV reporting, then changes of value of any subscribed-to properties of the object, in some cases based on programmable increments, trigger COV notifications to be sent to subscribing clients. Typically, COV notifications are sent to supervisory programs in COV-client devices or to operators or logging devices. Any object, proprietary or standard, may support COV reporting at the implementor's option.

COV subscriptions are established using the SubscribeCOV service or the SubscribeCOVProperty service. The subscription establishes a connection between the change of value detection and reporting mechanism within the COV-server device and a "process" within the COV-client device. Notifications of changes are issued by the COV-server when changes occur after the subscription has been established. The ConfirmedCOVNotification and UnconfirmedCOVNotification services are used by the COV-server to convey change notifications. The choice of confirmed or unconfirmed service is specified in the subscription.

When a BACnet standard object, of a type listed in Table 13-1, supports COV reporting, it shall support COV reporting for the property as listed in Table 13-1. At the implementor's discretion, COV reporting may also be supported for any other property of the object. For properties listed in Table 13-1 that have a REAL data type, the COV increment used to determine when to generate notifications will be the COV_Increment property of the object unless a COV_Increment parameter is supplied in the SubscribeCOVProperty service. For other properties that have a REAL data type, the COV increment to use when not supplied with the SubscribeCOVProperty service shall be a local matter. This is to allow multiple subscribers that do not require a specific increment to use a common increment to allow for the reduction of the processing burden on the COV-server. The criteria for COV reporting for properties other than those listed in Table 13-1 is based on the data type of the property subscribed to and is described in Table 13-1a.

If an object supports the COV_Period property and COV_Period is non-zero, it shall issue COV notifications to all subscribed recipients at the regular interval specified by COV_Period, in addition to the notifications initiated by the change of value of the monitored property. The value of the monitored property conveyed by the periodic COV notification shall be the basis for determining whether a subsequent COV notification is required by the change in value of the monitored property. If COV_Period is zero, the periodic notifications shall not be issued.

…

[Change **Table 13-1**, p. 235]

Table 13-1. Standardized Objects ~~Which~~ *That* May Support COV Reporting

Object Type	Criteria	Properties Reported
…	…	…
Loop	If Present_Value changes by COV_Increment or Status_Flags changes at all	Present_Value, Status_Flags, Setpoint, Controlled_Variable_Value
Pulse Converter	*If Present_Value changes by COV_Increment* *or* *Status_Flags changes at all* *or* *If COV_Period expires*	*Present_Value, Status_Flags, Update_Time*

[Change **Table 13-2**, p. 237]

Table 13-2. Standard Objects ~~that~~ *That* May Support Intrinsic Reporting

Object Type	Criteria	Event Type
…		
Analog Input, Analog Output, Analog ~~Value~~ *Value,* *Pulse Converter,*	If Present_Value exceeds range between High_Limit and Low_Limit for longer than Time_Delay AND the new transition is enabled in Event_Enable and Limit_Enable, OR Present__Value returns within the High_Limit – Deadband to Low_Limit + Deadband range for longer than Time_Delay AND the new transition is enabled in Event_Enable and Limit_Enable	OUT_OF_RANGE
…		

[Change **Table 13-3**, p. 238]

Table 13-3. Standard Object Property Values Returned in Notifications

Object	Event Type	Notification Parameters	Referenced Object's Properties
…			
Analog Input, Analog Output, Analog ~~Value~~ *Value,* *Pulse Converter*	OUT_OF_RANGE	Exceeding_Value Status_Flags Deadband Exceeded_Limit	Present_Value Status_Flags Deadband Low_Limit or High_Limit
…			

[Change BACnetObjectType production in **Clause 21**, pp. 397-398]

BACnetObjectType ::= ENUMERATED {
    accumulator *(23),*
    analog-input (0),
    …
    program (16),
    pulse-converter *(24),*
    schedule (17),
    -- see averaging (18),
    -- see multi-state-value (19),
    trend-log (20),
    -- see life-safety-point (21),

```
        -- see life-safety-zone     (22),
        -- see accumulator          (23),
        -- see pulse-converter      (24),
        ...
        }
    -- Enumerated values 0-127 are reserved for definition by ASHRAE. Enumerated values
    -- 128-1023 may be used by others subject to the procedures and constraints described
    -- in Clause 23.
```

[Change BACnetObjectTypesSupported production in **Clause 21**, pp. 398-399]

```
    BACnetObjectTypesSupported ::= BIT STRING {
        -- accumulator          (23),
        analog-input            (0),
        …
        program                 (16),
        -- pulse-converter      (24),
        schedule                (17),
        -- trend-log            (20),
    -- Objects added after 1995
        averaging               (18),
        multi-state-value       (19),
        trend-log               (20)
        life-safety-point       (21),
        life-safety-zone        (22),
    -- Objects added after 2001
        accumulator             (23),
        pulse-converter         (24)
        }
```

[Change BACnetPropertyIdentifier production in **Clause 21**, pp. 399-403]
[Note: schedule-default (174) appears in Addendum 135-2001*a* PR2]

```
    BACnetPropertyIdentifier ::= ENUMERATED {
        accepted-modes              (175),
        acked-transitions           (0),
        …
        active-cov-subscriptions    (152),
        adjust-value                (176),
        alarm-value                 (6),
        …
        controlled-variable-value   (21),
        count                       (177),
        count-before-change         (178),
        count-change-time           (179),
        cov-increment               (22),
        cov-period                  (180),
        cov-resubscription-interval (128),
        …
        in-process                  (47),
        input-reference             (181),
        instance-of                 (48),
            …
        limit-enable                (52),
        limit-monitoring-interval   (182),
        list-of-group-members       (53),
```

```
    …
    log-interval                    (134),
     logging-device                 (183),
     logging-record                 (184),
    low-limit                       (59),
    …
    polarity                        (84),
    prescale                        (185),
    present-value                   (85),
    …
    protocol-version                (98),
    pulse-rate                      (186),
    read-only                       (99),
    …
    resolution                      (106),
    scale                           (187),
    scale-factor                    (188),
    schedule-default                (174),
    …
    update-interval                 (118),
    update-time                     (189),
    utc-offset                      (119),
    …
    valid-samples                   (146),
    value-before-change             (190),
    value-set                       (191),
    value-change-time               (192),
    variance-value                  (151),
    …
    -- see schedule-default         (174),
    -- see accepted-modes           (175),
    -- see adjust-value             (176),
    -- see count                    (177),
    -- see count-before-change      (178),
    -- see count-change-time        (179),
    -- see cov-period               (180),
    -- see input-reference          (181),
    -- see limit-monitoring-interval(182),
    -- see logging-device           (183),
    -- see logging-record           (184),
    -- see prescale                 (185),
    -- see pulse-rate               (186),
    -- see scale                    (187),
    -- see scale-factor             (188),
    -- see update-time              (189),
    -- see value-before-change      (190),
    -- see value-set                (191),
    -- see value-change-time        (192),
    …
    }

-- The special property identifiers all, optional, and required are reserved for use in the ReadPropertyConditional and
-- ReadPropertyMultiple services or services not defined in this standard.
--
-- Enumerated values 0-511 are reserved for definition by ASHRAE. Enumerated values 512-4194303 may be used by
```

-- others subject to the procedures and constraints described in Clause 23. The highest enumeration used in this version is ~~174.~~ *192.*

[Add to **Clause 21**, BACnetReliability, pp. 404-405]
[Note: configuration-error (10) is defined in Addendum 135-2001*a* PR2]

BACnetReliability ::= ENUMERATED {
    ...
    multi-state fault (9),
    configuration-error (10),
    …
}

[Add to **Annex C**, p. 437]

PROGRAM :: = SEQUENCE {
    …
    }

PULSE-CONVERTER *::= SEQUENCE {*

object-identifier	*[75]*	*BACnetObjectIdentifier,*
object-name	*[77]*	*CharacterString,*
object-type	*[79]*	*BACnetObjectType,*
description	*[28]*	*CharacterString OPTIONAL,*
present-value	*[85]*	*REAL,*
input-reference	*[181]*	*BACnetObjectPropertyReference OPTIONAL,*
status-flags	*[111]*	*BACnetStatusFlags,*
event-state	*[36]*	*BACnetEventState,*
reliability	*[103]*	*BACnetReliability OPTIONAL,*
out-of-service	*[81]*	*BOOLEAN,*
units	*[117]*	*BACnetEngineeringUnits,*
scale-factor	*[188]*	*REAL,*
adjust-value	*[176]*	*REAL,*
count	*[177]*	*Unsigned,*
update-time	*[189]*	*BACnetDateTime,*
count-change-time	*[179]*	*BACnetDateTime,*
count-before-change	*[178]*	*Unsigned,*
cov-increment	*[22]*	*REAL OPTIONAL,*
cov-period	*[180]*	*Unsigned OPTIONAL,*
notification-class	*[17]*	*Unsigned OPTIONAL,*
time-delay	*[113]*	*Unsigned OPTIONAL,*
high-limit	*[45]*	*REAL OPTIONAL,*
low-limit	*[59]*	*REAL OPTIONAL,*
deadband	*[25]*	*REAL OPTIONAL,*
limit-enable	*[52]*	*BACnetLimitEnable OPTIONAL,*
event-enable	*[35]*	*BACnetEventTransitionBits OPTIONAL,*
acked-transitions	*[0]*	*BACnetEventTransitionBits OPTIONAL,*
notify-type	*[72]*	*BACnetNotifyType OPTIONAL,*
event-time-stamps	*[130]*	*SEQUENCE OF BACnetTimeStamp OPTIONAL,*
profile-name	*[167]*	*CharacterString OPTIONAL*

}

SCHEDULE ::= SEQUENCE {…

[Add new **Annex D.22**, p. 455, and renumber existing Annex D.22 and subsequent clauses]

D.22 Example of a Pulse Converter object

Property:	Object_Identifier =	(Pulse Converter, Instance 1)
Property:	Object_Name =	"Meter 5"
Property:	Object_Type =	PULSE_CONVERTER
Property:	Description =	""
Property :	Present_Value =	125.0
Property:	Input_Reference =	((Accumulator, Instance 1), Present_Value)
Property:	Status_Flags =	{FALSE, FALSE, FALSE, FALSE}
Property:	Event_State =	NORMAL
Property:	Out_Of_Service =	FALSE
Property:	Units =	LITERS_PER_HOUR
Property:	Scale_Factor =	0.5
Property:	Adjust_Value =	500.0
Property:	Count =	250
Property:	Update_Time =	(10-JUL-01,11:40:21.0),
Property:	Count_Change_Time =	(10-JUL-01,11:30:01.0),
Property:	Count_Before_Change =	523
Property:	COV_Increment =	10.0
Property:	COV_Period =	3600
Property:	Notification_Class =	5
Property:	Time_Delay =	0
Property:	High_Limit =	1000.0
Property:	Low_Limit =	0.0
Property:	Deadband =	0.0
Property:	Limit_Enable =	{FALSE, TRUE}
Property:	Event_Enable =	{TRUE, FALSE, TRUE}
Property:	Acked_Transitions =	{TRUE, TRUE, TRUE}
Property:	Notify_Type =	ALARM
Property:	Event_Time_Stamps =	((12-JUL-01,18:50:21.2), (*-*-*,*:*:*.*), (12-JUL-01,19:01:34.0))

135*c*-6. Standardize event notification priorities.

Addendum 135*c*-6

[Add new **Clause 13.4.1**, p. 247]

13.4.1 Alarm and Event Priority Classification

Alarms and events traversing the BACnet network need prioritization to assure that important information reaches its destination and is acted upon quickly. To assure alarm prioritization at the network level, the Network Priority as defined in 6.2.2 shall be set as a function of the alarm and event priority as defined in Table 13-5. Annex M provides additional clarity and examples of specific messages and priorities.

[Add new **Table 13-5**, p. 247, and renumber subsequent tables]

Table 13-5. Alarm and Event Priority - Network Priority Association

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

[Change **Clause 12.20**, p. 216]

12.21 Notification Class Object type

The Notification Class object type defines a standardized object that represents and contains information required for the distribution of event notifications within BACnet systems. Notification Classes are useful for event-initiating objects that have identical needs in terms of how their notifications should be handled, what the destination(s) for their notifications should be, and how they should be acknowledged.

A notification class defines how event notifications shall be prioritized in their handling according to TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events; whether these categories of events require acknowledgment (nearly always by a human operator); and what destination devices or processes should receive notifications.

The purpose of prioritization is to provide a means to ensure that alarms or event notifications with critical time considerations are not unnecessarily delayed. The possible range of priorities is 0 - 255. A lower number indicates a higher priority. *The priority and the Network Priority (Clause 6.2.2) are associated as defined in Table 13-5.* Priorities may be assigned to TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events individually within a notification class.
…

[Change **Clause 12.20.6**, Priority, p. 217]

12.21.6 Priority

This property, of type BACnetARRAY[3] of Unsigned, shall convey the priority to be used for event notifications for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively. Priorities shall range from 0 - 255 inclusive. A lower number indicates a higher priority. *The priority and the Network Priority (Clause 6.2.2) are associated as defined in Table 13-5.*

[Change **Clause 13.8.1.7**, Priority, p.254]

13.8.1.7 Priority

This parameter, of type Unsigned8, shall specify the priority of the event that has occurred. The priority is specified by the Priority property of the Notification Class or Event Enrollment objects associated with this event. The possible range of priorities is 0-255. A lower number indicates a higher priority. *The priority and the Network Priority (Clause 6.2.2) are associated as defined in Table 13-5.*

[Change **Clause 13.9.1.7**, Priority, p. 257]

13.9.1.7 Priority

This parameter, of type Unsigned8, shall specify the priority of the event that has occurred. The priority is specified by the Priority property of the Notification Class or Event Enrollment objects associated with the event. The possible range of priorities is 0-255. A lower number indicates a higher priority. *The priority and the Network Priority (Clause 6.2.2) are associated as defined in Table 13-5.*

[Add new **Annex M**, p. 555]

Annex M – Guide to Event Notification Priority Assignments (Informative)

[This annex is not part of this standard. It is merely informative and does not contain requirements for conformance to the standard.]

The Alarm and Event Priorities and Network Priorities defined in 13.4.1 broadly categorize the alarm and event notification priorities. This annex provides examples of various alarms and events that could be assigned into these categories.

Table M-1 extends Table 13-5 by adding semantic meaning to the priority classifications. The subsequent narrative details the classifications and provides examples of various alarm and event priorities in an interoperable system.

Table M-1. Message Groups Priorities

Message Group	Priority Range	Network Priority	Brief Description
Life Safety	00 – 31	Life Safety Message	Notifications related to an immediate threat to life, safety or health such as fire detection or armed robbery
Property Safety	32 - 63	Life Safety Message	Notifications related as an immediate threat to property such as forced entry
Supervisory	64 - 95	Critical Equipment Message	Notifications related to improper operation, monitoring failure (particularly of Life Safety or Property Safety monitoring), or monetary loss
Trouble	96 - 127	Critical Equipment Message	Notifications related to communication failure (particularly of Life Safety or Property Safety equipment)
Miscellaneous Higher Priority Alarm and Events	128 - 191	Urgent Message	Higher-level notifications related to occupant discomfort, normal operation, normal monitoring, or return to normal
Miscellaneous Lower Priority Alarm and Events	192 - 255	Normal Message	Lower-level notification related to occupant discomfort, normal operation, normal monitoring, or return to normal.

M.1 Life Safety Message Group (0 – 31)

This message group includes any event report related to an immediate threat to life, safety or health. Examples include fire detection, armed robbery and medical emergency.

M.1.1 Life Safety Message Group Examples

Criteria for membership in a particular life safety message group vary from jurisdiction to jurisdiction. The examples below are intended to clarify the intent of the grouping and are not meant to be prescriptive.

Event	Description/Examples
Reliable Fire Alarms	Fire alarm events produced by reliable fire alarm detection devices. Examples might include smoke detectors and heat detectors.
Life Safety Process Alarms	A process or equipment alarm that indicates an immediate threat to life, safety or health belongs at this priority. Examples might include carbon monoxide or explosive vapor detection and toxic chemical release.
Fire Alarms Requiring Verification	Fire alarm events requiring verification report. Examples might include pull stations and alarmed fire exit doors. This category is separated from reliable fire alarm because of the potential for false alarms caused by vandals or environmental contamination.
Medical Alarms	Immediate threats to life or health due to medical emergencies. Examples might include heart attack or stroke alarm and falls with injuries.
Hold Up And Duress Alarms	Potential threats to life, safety or health due to criminal activity belong at this priority. Examples might include armed robbery, kidnapping, and bomb threats.
Panic Alarms	Any condition requiring immediate outside intervention to prevent or reduce threats to life, safety, or health.
Life Safety PreAlarm Alerts	Conditions that are likely to become full-fledged Life Safety threats momentarily or tentative detection of Life Safety threats. Examples include fire prealarm or toxic gas nearing the alarm level.
Life Safety Return To Normal	Reporting or recording of returns to normal after a Life Safety Alarm or Alert. Examples include resetting a fire pull station or discontinuing a medical alarm.

M.2 Property Safety Message Group (32 – 63)

Any event report related as an immediate threat to property belongs in this group. Example events include forced entry, unlocked doors, and equipment above the allowed operating temperature.

M.2.1 Property Safety Message Group Examples

Event	Description/Examples
Burglar Alarms and Forced Door Alarms	Improper intrusion into a secure area where there is potential for property damage or theft. Examples include motion detected in an unoccupied space, locked door forced open, and broken exterior glass.
Security Alarms	Potential intrusion or unauthorized occupant alarms. Examples include interior door improperly opened or person without proper ID.

Watchman Tour Alarms	Alarms related to a predefined watch tour or other manual property supervision not being properly conducted. Examples include watchman late to station and watchman station out of order.
Property Process Alarms	Any process or equipment alarm that indicates a direct threat to property not covered elsewhere. Examples include freeze alarm and low duct pressure with danger of collapse.
Door Held Open Alarms	Alarms related to a door or other opened items that were previously opened properly and should now be closed and locked, but are not. An example would be a door propped open past normal business hours.
Property Safety Return To Normal	Reporting or recording of returns to normal after a Property Safety Alarm. Examples include locking door held open and resetting a burglar alarm.

M.3 Supervisory Message Group (64 – 95)

Any event report related to improper operation, monitoring failure (particularly of Life Safety or Property Safety monitoring), or monetary loss belongs in this group. Example events include fire sprinkler valve shut off, communication failure and excessive energy use.

M.3.1 Supervisory Message Group Examples

Event	Description/Examples
Fire Supervision (tamper)	Fire alarm and suppression components that are supervised against tampering. Examples include sprinkler valve shutoff and uninterruptable power supply disable.
Security Supervision (tamper)	Security and burglar alarm components that are supervised against tampering. Examples include box tamper switches and uninterruptable power supply disable.
Energy Alarms	Loss of energy management control likely to result in monetary loss. Examples include failure to control electrical demand within allowed limits and failure of incoming energy sources such as gas or steam.
Early Warning Alerts	Warnings used to eliminate future problems by initiating early corrective action. Examples include security video recording tape low and standby generator fuel tank not full.
Energy Warnings	Problems with energy management control that could result in monetary loss if left uncorrected. Examples include failure of loads to shed for automated electrical demand control, or reductions in available incoming energy sources such as low gas pressure.
Supervisory Return To Normal	Reporting or recording of return to normal after a Supervisory off-normal report. An example is a fire sprinkler valve returning to normal.

M.4 Trouble Message Group (96 – 127)

Any event report related to communication failure (particularly of Life Safety or Property Safety equipment) belongs in this group.

M.4.1 Trouble Message Group Examples

Event	Description/Examples
Fire Trouble (equipment failure)	Failure of fire alarm and suppression components. Examples include loss of communication with fire alarm components or failure of a smoke detector.
Security and Burglar Trouble (equipment failure)	Failure of security and burglar alarm components. Examples include loss of communication with security components or failure of an intrusion detector.
Communication Equipment Failure Trouble	Failure of equipment used for communication (but not directly related to fire or security applications). Examples include Local Area Network component failure, telephone system component failure, and Building Automation System communication failure.
Process Trouble	Problems with general processes or equipment not operating correctly. Examples include HVAC interlocks failing to operate, equipment not responding to commands, and control programs not operating.
Energy Warnings	Problems with energy management control that could result in monetary loss if left uncorrected. Examples include failure of loads to shed for automated electrical demand control, or reductions in available incoming energy sources such as low gas pressure.
Communication Equipment Warning Trouble	Warnings or troubles with equipment used for communication (but not directly related to fire or security applications). Examples include degraded throughput, excessive message retries, or low buffer warnings.
Trouble Return To Normal	Reporting or recording of return to normal after a Trouble off-normal report. An example is communication returning to normal.

M.5 Miscellaneous Higher Priority Message Group (128 – 191)

Any higher-level event report related to occupant discomfort, normal operation, normal monitoring, or return to normal belongs in this group. Example events include normal event logging, room temperature above setpoint and test result logging.

M.5.1 Miscellaneous Higher Priority Group Examples

Event	Description/Examples
Equipment And Industrial Supervision	Used for miscellaneous supervision of equipment or processes that are not likely to result in risks to people or property, or in loss of money.
Comfort Alarm	Reporting of temperature, humidity, noise levels or other conditions that will cause occupant discomfort with accompanying loss of productivity and discontent can be reported using this priority. Examples include high or low occupied space temperature, high or low humidity, and high carbon dioxide levels.
System Status Normal	Simple status changes to the normal or passive states that do not imply any problem or required action. Examples include preprogrammed or timed changes, and preprogrammed triggers operating properly.

Comfort Normal	Reporting of occupied space temperature, humidity, noise levels, or other conditions returning to their normal values after a comfort alarm or warning.

M.6 Miscellaneous Lower Priority Message Group (192 – 255)

Any lower-level event report related to occupant discomfort, normal operation, normal monitoring, or return to normal belongs in this group. Example events include normal event logging, room temperature above setpoint, return to normal events and test result logging.

M.6.1 Miscellaneous Lower Priority Group Examples

Event	Description/Examples
System Events	Simple system events that only require simple logging or noting for future reference can use this priority. Examples include access granted or denied and normal watchtour station reached.
System Status Active	Simple status changes to the active state that do not imply any problem or required action can use this priority. Examples include preprogrammed or timed changes and preprogrammed triggers operating properly.
Comfort Warning	Reporting of temperature, humidity, noise levels, or other conditions that are out of the usual range and could eventually lead to occupant discomfort with accompanying loss of productivity and discontent can be reported using this priority. Examples include high or low occupied space temperature, high or low humidity, and high carbon dioxide levels.
Test and Diagnostic Events	Reporting of normal test results or normal diagnostics such as fire alarm walk test events can use this priority.

135*c*-7. Define Abort reason when insufficient segments are available.

Addendum 135*c*-7

[Change **Clause 5.4.5.3**, pp.33-34]

5.4.5.3 AWAIT_RESPONSE

…

CannotSendSegmentedComplexACK

If a CONF_SERV.response(+) is received from the local application program, which is to be conveyed via a BACnet-ComplexACK-PDU, and the length of the APDU is greater than maximum-transmittable-length as determined according to 5.2.1, and either

(a) this device does not support the transmission of segmented messages or

(b) the client will not accept a segmented response (the 'segmented-response-accepted' parameter in BACnet-ConfirmedRequest-PDU is FALSE), or

(c) the client's max-segments-accepted parameter in the BACnet-ConfirmedRequest-PDU is fewer than required to transmit the total ~~APDU.~~ *APDU, or*

(d) the number of segments transmittable by this device is fewer than required to transmit the total APDU,

then~~, for case (a) and (b),~~ issue an N-UNITDATA.request with 'data_expecting_reply' = FALSE to transmit a BACnet-Abort-PDU with 'server' = TRUE and 'abort-reason' = SEGMENTATION_NOT_SUPPORTED *for case (a) and (b), or BUFFER_OVERFLOW for case (c) and (d),* and enter the IDLE state.

…

135*c*-8. Add new Error Codes and specify usage.

Addendum 135*c*-8.

[Add **Clause 15.3.2.1**, p. 283]

15.3.2.1 Error Class and Error Code Assignments

The ‘Error Class’ and ‘Error Code’ to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The device cannot allocate the space needed for the new object.	RESOURCES	NO_SPACE_FOR_OBJECT
The device does not support creation of this object for any reason other than space.	OBJECT	DYNAMIC_CREATION_NOT_SUPPORTED
The object being created already exists.	OBJECT	OBJECT_IDENTIFIER_ALREADY_EXISTS
A datatype of a property value specified in the List of Initial Values does not match the datatype of the property specified by the Property_Identifier.	PROPERTY	INVALID_DATATYPE
A value used in the List of Initial Values is outside the range of values defined for the property specified by the Property_Identifier.	PROPERTY	VALUE_OUT_OF_RANGE
A Property_Identifier has been specified in the List of Initial Values that is unknown for objects of the type being created.	PROPERTY	UNKNOWN_PROPERTY
A character string value was encountered in the List of Initial Values that is not a supported character set.	PROPERTY	CHARACTER_SET_NOT_SUPPORTED
A property specified by the Property_Identifier in the List of Initial Values does not support initialization during the CreateObject service.	PROPERTY	WRITE_ACCESS_DENIED

[Add **Clause 15.4.2.1**, p. 284]

15.4.2.1 Error Class and Error Code Assignments

The ‘Error Class’ and ‘Error Code’ to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The object to be deleted does not exist.	OBJECT	UNKNOWN_OBJECT
The object exists but cannot be deleted.	OBJECT	OBJECT_DELETION_NOT_PERMITTED

[Change **Clause 15.5.2**, p. 286]

15.5.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to access the specified property of the specified object. If the access is successful, a 'Result(+)' primitive, which returns the accessed value, shall be generated. If the access fails, a 'Result(-)' primitive shall be generated*, indicating the reason for the failure*.

[Add **Clause 15.5.2.1**, p. 286]

15.5.2.1 Error Class and Error Code Assignments

The ‘Error Class’ and ‘Error Code’ to be returned for specific situations are as follows:

Situation	Error Class	Error Code
Specified object does not exist.	OBJECT	UNKNOWN_OBJECT
Specified property does not exist.	PROPERTY	UNKNOWN_PROPERTY
An array index is provided but the property is not an array.	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
An array index is provided that is outside the range existing in the property.	PROPERTY	INVALID_ARRAY_INDEX

[Add **Clause 15.7.2.1**, pp.293]

15.5.2.1 Error Class and Error Code Assignments

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
Specified object does not exist.	OBJECT	UNKNOWN_OBJECT
Specified property does not exist.	PROPERTY	UNKNOWN_PROPERTY
An array index is provided but the property is not an array.	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
An array index is provided that is outside the range existing in the property.	PROPERTY	INVALID_ARRAY_INDEX

[Add **Clause 15.9.2.1**, p. 300]

15.9.2 Service Procedure

15.9.2.1 Error Class and Error Code Assignments

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
Specified object does not exist.	OBJECT	UNKNOWN_OBJECT
Specified property does not exist.	PROPERTY	UNKNOWN_PROPERTY
An array index is provided but the property is not an array.	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
An array index is provided that is outside the range existing in the property.	PROPERTY	INVALID_ARRAY_INDEX
The specified property is currently not writable by the requestor.	PROPERTY	WRITE_ACCESS_DENIED
The datatype of the value provided is incorrect for the specified property.	PROPERTY	INVALID_DATATYPE
The property is Object_Name and the name is already in use in the device.	PROPERTY	DUPLICATE_NAME
The property is Object Identifier and the identifier is already in use in the device.	PROPERTY	DUPLICATE_OBJECT_ID
The value provided is outside the range of values that the property can take on.	PROPERTY	VALUE_OUT_OF_RANGE
There is not enough space to store the new value.	RESOURCES	NO_SPACE_TO_WRITE_PROPERTY

[Add **Clause 15.10.2.1**, p. 302]

15.10.2.1 Error Class and Error Code Assignments

The 'Error Class' and 'Error Code' to be returned in a 'Result(-)' for specific situations are as follows:

Situation	Error Class	Error Code
Specified object does not exist.	OBJECT	UNKNOWN_OBJECT
Specified property does not exist.	PROPERTY	UNKNOWN_PROPERTY
An array index is provided but the property is not an array.	PROPERTY	PROPERTY_IS_NOT_AN_ARRAY
An array index is provided that is outside the range existing in the property.	PROPERTY	INVALID_ARRAY_INDEX
The specified property is currently read-only.	PROPERTY	WRITE_ACCESS_DENIED
The datatype of the value provided is incorrect for the specified property.	PROPERTY	INVALID_DATATYPE
The property is Object_Name and the name is already in use in the device.	PROPERTY	DUPLICATE_NAME
The property is Object Identifier and the identifier is already in use in the device.	PROPERTY	DUPLICATE_OBJECT_ID
The value provided is outside the range of values that the property can take on.	PROPERTY	VALUE_OUT_OF_RANGE
There is not enough space to store the new value.	RESOURCES	NO_SPACE_TO_WRITE_PROPERTY

[Add new **Clause 18.3.7**, p. 334, and renumber subsequent clauses]

18.3.7 PROPERTY_IS_NOT_AN_ARRAY - An attempt has been made to access a property as an array and that property does not have an array datatype.

[Add new **Clause 18.6.3 and 18.6.4**, p. 335, and renumber subsequent clauses]

18.6.3 DUPLICATE_NAME - An attempt has been made to write to an Object_Name property with a value that is already in use in a different Object_Name property within the device.

18.6.4 DUPLICATE_OBJECT_ID – An attempt has been made to write to an Object_Identifier property with a value that is already in use in a different Object_Identifier within the same device.

[Add new **Clause 18.6.14**, p. 335, and renumber subsequent clauses]

18.6.14 PROPERTY_IS_NOT_AN_ARRAY - An attempt has been made to access a property as an array and that property does not have an array datatype.

[Change **Clause 18.8.2** through **Clause 18.8.5**, p. 336]

18.8.2 INCONSISTENT_PARAMETERS - Generated in response to a confirmed request APDU that omits a conditional service argument that should be present or contains a conditional service argument that should not be present. *This condition could also elicit a Reject PDU with a Reject Reason of INVALID_TAG.*

18.8.3 INVALID_PARAMETER_DATA_TYPE - Generated in response to a confirmed request APDU in which the encoding of one or more of the service parameters does not follow the correct type specification. *This condition could also elicit a Reject PDU with a Reject Reason of INVALID_TAG.*

18.8.4 INVALID_TAG - While parsing a message, an invalid tag was encountered. *Since an invalid tag could confuse the parsing logic, any of the following Reject Reasons may also be generated in response to a confirmed request containing an invalid tag: INCONSISTENT_PARAMETERS, INVALID_PARAMETER_DATA_TYPE, MISSING_REQUIRED_PARAMETER, and TOO_MANY_ARGUMENTS.*

18.8.5 MISSING_REQUIRED_PARAMETER - Generated in response to a confirmed request APDU that is missing at least one mandatory service argument. *This condition could also elicit a Reject PDU with a Reject Reason of INVALID_TAG.*

[Change **18.8.7**, p. 336]

18.8.7 TOO_MANY_ARGUMENTS - Generated in response to a confirmed request APDU in which the total number of service arguments is greater than specified for the service. *This condition could also elicit a Reject PDU with a Reject Reason of INVALID_TAG.*

[Change Error production in **Clause 21**, pp. 385-386]

Error ::= SEQUENCE {

```
    …
    error-code      ENUMERATED {
                        other                                   (0),
                        authentication-failed                   (1),
                        character-set-not-supported             (41),
                        configuration-in-progress               (2),
                        device-busy                             (3),
                        duplicate-name                          (48),
                        duplicate-object-id                     (49),
                        dynamic-creation-not-supported          (4),
                        file-access-denied                      (5),
                        incompatible-security-levels            (6),
                        inconsistent-parameters                 (7),
                        inconsistent-selection-criterion        (8),
                        invalid-array-index                     (42),
                        invalid-configuration-data              (46),
                        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),
                        no-vt-sessions-available                (21),
                        object-deletion-not-permitted           (23),
                        object-identifier-already-exists        (24),
                        operational-problem                     (25),
                        optional-functionality-not-supported,   (45),
                        password-failure                        (26),
                        property-is-not-a-list                  (22),
                        property-is-not-an-array                (50),
                        read-access-denied                      (27),
                        security-not-supported                  (28),
                        service-request-denied                  (29),
                        timeout                                 (30),
                        unknown-object                          (31),
                        unknown-property                        (32),
                    -- this enumeration was removed             (33),
                        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),
        -- see  character-set-not-supported           (41),
        -- see  invalid-array-index                   (42),
                cov-subscription-failed                (43),
                not-cov-property                       (44),
        -- see  optional-functionality-not-supported, (45),
        -- see  invalid-configuration-data            (46),
        -- see  datatype-type-supported               (47),
        -- see  duplicate-name                        (48),
        -- see duplicate-object-id                    (49),
        -- see  property-is-not-an-array              (50),
                ...
    }
    -- Enumerated values 0-255 are reserved for definition by ASHRAE. Enumerated values
    -- 256-65535 may be used by others subject to the procedures and constraints described
    -- in Clause 23. The last enumeration used in this version is 46 50.
}
```

[Add a new entry to **History of Revisions**, p.557]

(This History of Revisions is not part of this standard. It is merely informative and does not contain requirements necessary for conformance to the standard.)

HISTORY OF REVISIONS

Protocol		***Summary of Changes to the Standard***
Version	***Revision***	
…	…	**…**
1	4	**Addendum *a* to ANSI/ASHRAE 135-2001** Approved by the ASHRAE Standards Committee January XX, 2004 and by the ASHRAE Board of Directors January XX, 2004; and by the American National Standards Institute XXX, 2004. 1. Add Partial Day Scheduling to the Schedule object. 2. Enable reporting of proprietary events by the Event Enrollment object. 3. Allow detailed error reporting when all ReadPropertyMultiple accesses fail. 4. Remove the Recipient property from the Event Enrollment object. 5. Add the capability to issue I-Am responses on behalf of MS/TP slave devices. 6. Add a new silenced mode to the DeviceCommunicationControl service. 7. Add 21 new engineering units. 8. Specify the behavior of a BACnetArray when its size is changed. 9. Clarify the behavior of a BACnet router when it receives an unknown network message type.
1	4	**Addendum *c* to ANSI/ASHRAE 135-2001** Approved by the ASHRAE Standards Committee January XX, 2004 and by the ASHRAE Board of Directors January XX, 2004; and by the American National Standards Institute XXX, 2004. 1. Allow Life Safety objects to advertise supported mode. 2. Add Unsilence Options to the LifeSafetyOperation Service. 3. Specify the relationship between the Event_Type and Event_Parameter properties.

		4. Add a new Accumulator Object Type. 5. Add a new Pulse Converter Object Type. 6. Standardize event notification priorities. 7. Define Abort reason when insufficient segments are available. 8. Add new Error Codes and specify usage.
1	4	**Addendum *d* to ANSI/ASHRAE 135-2001** Approved by the ASHRAE Standards Committee January XX, 2004 and by the ASHRAE Board of Directors January XX, 2004; and by the American National Standards Institute XXX, 2004. 1. Add clauses describing BACnet-EIB/KNX mapping.

POLICY STATEMENT DEFINING ASHRAE'S CONCERN FOR THE ENVIRONMENTAL IMPACT OF ITS ACTIVITIES

ASHRAE is concerned with the impact of its members' activities on both the indoor and outdoor environment. ASHRAE's members will strive to minimize any possible deleterious effect on the indoor and outdoor environment of the systems and components in their responsibility while maximizing the beneficial effects these systems provide, consistent with accepted standards and the practical state of the art.

ASHRAE's short-range goal is to ensure that the systems and components within its scope do not impact the indoor and outdoor environment to a greater extent than specified by the standards and guidelines as established by itself and other responsible bodies.

As an ongoing goal, ASHRAE will, through its Standards Committee and extensive technical committee structure, continue to generate up-to-date standards and guidelines where appropriate and adopt, recommend, and promote those new and revised standards developed by other responsible organizations.

Through its *Handbook*, appropriate chapters will contain up-to-date standards and design considerations as the material is systematically revised.

ASHRAE will take the lead with respect to dissemination of environmental information of its primary interest and will seek out and disseminate information from other responsible organizations that is pertinent, as guides to updating standards and guidelines.

The effects of the design and selection of equipment and systems will be considered within the scope of the system's intended use and expected misuse. The disposal of hazardous materials, if any, will also be considered.

ASHRAE's primary concern for environmental impact will be at the site where equipment within ASHRAE's scope operates. However, energy source selection and the possible environmental impact due to the energy source and energy transportation will be considered where possible. Recommendations concerning energy source selection should be made by its members.