Data Communication Protocol for Building Automation and Control Networks

Approved by ASHRAE on December 30, 2014; and by the American National Standards Institute on December 31, 2014.

This addendum was approved 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 website (www.ashrae.org) or in paper form from the Senior Manager of Standards.

The latest edition of an ASHRAE Standard may be purchased on the ASHRAE website (www.ashrae.org) or from ASHRAE Customer Service, 1791 Tullie Circle, NE, Atlanta, GA 30329-2305. E-mail: orders@ashrae.org. Fax: 678-539-2129. Telephone: 404-636-8400 (worldwide), or toll free 1-800-527-4723 (for orders in US and Canada). For reprint permission, go to www.ashrae.org/permissions.

© 2014 ASHRAE  ISSN 1041-2336
ASHRAE Standards Committee 2014–2015

Richard L. Hall, Chair  
James W. Earley, Jr.  
Mark P. Modera

Douglass T. Reindl, Vice-Chair  
Steven J. Emmerich  
Cyrus H. Nasseri

Joseph R. Anderson  
Patricia T. Graef  
Heather L. Platt

James Dale Aswegan  
Rita M. Harrold  
Peter Simmonds

Charles S. Barnaby  
Adam W. Hinge  
Wayne H. Stoppelmoor, Jr.

Donald M. Brundage  
Srinivas Katipamula  
Jack H. Zarour

John A. Clark  
Debra H. Kennoy  
Julia A. Keen, BOD ExO

Waller S. Clements  
Malcolm D. Knight  
Bjarne Wilkens Olesen, CO

David R. Conover  
Rick A. Larson

John F. Dunlap  
Arsen K. Melkov

Stephanie C. Reiniche, Senior Manager of Standards

SPECIAL NOTE

This American National Standard (ANS) is a national voluntary consensus Standard developed under the auspices of 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 committee members may or may not be ASHRAE members, 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 Senior 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, or
d. permission to reprint portions of the Standard.

DISCLAIMER

ASHRAE uses its best efforts to promulgate Standards and Guidelines for the benefit of the public in light of available information and accepted industry practices. However, ASHRAE does not guarantee, certify, or assure the safety or performance of any products, components, or systems tested, installed, or operated in accordance with ASHRAE’s Standards or Guidelines or that any tests conducted under its Standards or Guidelines will be nonhazardous or free from risk.

ASHRAE INDUSTRIAL ADVERTISING POLICY ON STANDARDS

ASHRAE Standards and Guidelines are established to assist industry and the public by offering a uniform method of testing for rating purposes, by suggesting safe practices in designing and installing equipment, by providing proper definitions of this equipment, and by providing other information that may serve to guide the industry. The creation of ASHRAE Standards and Guidelines is determined by the need for them, and conformance to them is completely voluntary.

In referring to this Standard or Guideline and in marking of equipment and in advertising, no claim shall be made, either stated or implied, that the product has been approved by ASHRAE.
[This foreword and the “rationales” on the following pages are not part of this standard. They are merely informative and do not contain requirements necessary for conformance to the standard.]

FOREWORD

The purpose of this addendum is to present a proposed change for public review. These modifications are the result of change proposals made pursuant to the ASHRAE continuous maintenance procedures and of deliberations within Standing Standard Project Committee 135. The proposed changes are summarized below.

135-2012ay-1 Add a Timer Object Type, p. 3
135-2012ay-2 Correct Expiry_Time property name to Expiration_Time in the Access Credential Object, p. 23

In the following document, language to be added to existing clauses of ANSI/ASHRAE 135-2012 and Addenda is indicated through the use of italics, while deletions are indicated by strikethrough. Where entirely new subclauses are proposed to be added, plain type is used throughout. Only this new and deleted text is open to comment at this time. All other material in this addendum is provided for context only and is not open for public review comment except as it relates to the proposed changes.
135-2012ay-1 Add a Timer Object Type

Rationale

Many building automation applications have timer functionality, which should be made network visible for interoperable control, monitoring, and visualization.

A new object type Timer is added to allow timer functionality to be network visible. A new event algorithm CHANGE_OF_TIMER is added to accommodate the event detection characteristics and notification needs of the Timer object.

[Add new Clause 12.X, pp. 459]

12.X Timer Object Type

The Timer object type defines a standardized object whose properties represent the externally visible characteristics of a countdown timer.

The Timer object provides a network-visible view of selected parameters of a countdown timer. The operating state of the timer may be viewed and controlled through these properties.

The Timer object type supports a pre-configured timeout as well as providing a specific timeout on activation of the timer. A boolean indication of when the timer is counting down supports applications that monitor a boolean flag for temporary behavior. The commanding mechanism, similar to the Schedule object, supports applications that require commands to be initiated on change of the timer's state. The intrinsic event reporting supports applications that need to be notified on change of the timer's state.

The countdown timer represented by this object type maintains a state, represented by its Timer_State property, whose state-event diagram is depicted by the following figure.

![Figure 12-X. State Transitions for the Timer object.](image)

The timer operation may be controlled by writing to the Present_Value property, the Timer_Running property, or the Timer_State property. For the description of allowed values or value ranges, see the state machine description below, and property descriptions in respective sub-clauses.

The timer's state machine shall operate as follows:
State IDLE

In the IDLE state, the timer is inactive, waiting to be activated. The timer may be activated using a default timeout or a specific timeout. The Present_Value property shall have a value of zero. The Timer_Running property shall have a value of FALSE. The Expiration_Time property shall indicate the unspecified datetime value.

Start Request with Default Timeout

If a value of TRUE is written to the Timer_Running property,

then set Initial_Timeout to the value of Default_Timeout; set Timer_Running to TRUE; set Timer_State to RUNNING; set Last_State_Change to IDLE_TO_RUNNING; set Present_Value to the value of Initial_Timeout; set Update_Time to the current date and time; initiate the write requests for the IDLE_TO_RUNNING transition if present; and enter the RUNNING state.

Start Request with Specific Timeout

If a value within the supported range is written to the Present_Value property,

then set Timer_Running to TRUE; set Timer_State to RUNNING; set Last_State_Change to IDLE_TO_RUNNING; set Initial_Timeout to the value written to Present_Value; set Update_Time to the current date and time; initiate the write requests for the IDLE_TO_RUNNING transition if present; and enter the RUNNING state.

Clear Request

If a value of IDLE is written to the Timer_State property,

then no properties shall be changed; no write requests shall be initiated; and no state transition shall occur.

Expire Request

If a value of zero is written to the Present_Value property, or a value of FALSE is written to the Timer_Running property,

then no properties shall be changed; no write requests shall be initiated; and no state transition shall occur.

Out of Range Request

If a value outside the supported range and not zero is written to the Present_Value property, or a timer state other than IDLE is written to the Timer_State property,

then no properties shall be changed; no write requests shall be initiated, no state transition shall occur, and a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned.

State RUNNING

In the RUNNING state, the timer is active and is counting down the remaining time. The Present_Value property shall indicate the remaining time until expiration. The Timer_Running property shall have a value of TRUE. The Expiration_Time property shall indicate the date and time when the timer will expire. The value of Expiration_Time shall be calculated at the time the property is read.

Timer Expired

If the remaining time indicated by Present_Value reaches zero,

then set Timer_Running to FALSE; set Timer_State to EXPIRED; set Last_State_Change to RUNNING_TO_EXPIRED; set Expiration_Time to the current data and time; set Update_Time to the current date and time; initiate the write requests for the RUNNING_TO_EXPIRED transition if present; and enter the EXPIRED state.

Expire Request

If a value of zero is written to the Present_Value property, or a value of FALSE is written to the Timer_Running property,
then set Timer_Running to FALSE; set Timer_State to EXPIRED; set Last_State_Change to FORCED_TO_EXPIRED; set Expiration_Time to the current date and time; set Update_Time to the current date and time; initiate the write requests for the FORCED_TO_EXPIRED transition if present; and enter the EXPIRED state.

Start Request with Default Timeout
If a value of TRUE is written to the Timer_Running property,

then set Last_State_Change to RUNNING_TO_RUNNING; set Initial_Timeout to the value of Default_Timeout; set Present_Value to the value of Initial_Timeout; set Update_Time to the current date and time; initiate the write requests for the RUNNING_TO_RUNNING transition if present; and enter the RUNNING state.

Start Request with Specific Timeout
If a value within the supported range is written to the Present_Value property,

then set Last_State_Change to RUNNING_TO_RUNNING; set Initial_Timeout to the value written to Present_Value; set Update_Time to the current date and time; initiate the write requests for the RUNNING_TO_RUNNING transition if present; and enter the RUNNING state.

Clear Request
If a value of IDLE is written to the Timer_State property,

then set Timer_Running to FALSE; set Timer_State to IDLE; set Last_State_Change to RUNNING_TO_IDLE; set Present_Value to zero; set Expiration_Time to the unspecified datetime value; set Update_Time to the current date and time; initiate the write requests for the RUNNING_TO_IDLE transition if present; and enter the IDLE state.

Out of Range Request
If a value outside the supported range and not zero is written to the Present_Value property, or a timer state other than IDLE is written to the Timer_State property,

then no properties shall be changed; no write requests shall be initiated; no state transition shall occur; and a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned.

State EXPIRED
In the EXPIRED state, the timer has expired or was forced to be expired. The Present_Value property shall have a value of zero. The Timer_Running property shall be FALSE. The Expiration_Time property shall indicate the date and time when this state was entered.

Start Request with Default Timeout
If a value of TRUE is written to the Timer_Running property,

then set Timer_Running to TRUE; set Timer_State to RUNNING; set Last_State_Change to EXPIRED_TO_RUNNING; set Initial_Timeout to the value of Default_Timeout; set Present_Value to the value of Initial_Timeout; set Update_Time to the current date and time; initiate the write requests for the EXPIRED_TO_RUNNING transition if present; and enter the RUNNING state.

Start Request with Specific Timeout
If a value within the supported range is written to the Present_Value property,

then set Timer_Running to TRUE; set Timer_State to RUNNING; set Last_State_Change to EXPIRED_TO_RUNNING; set Initial_Timeout to the value written to Present_Value; set Update_Time to the current date and time; initiate the write requests for the EXPIRED_TO_RUNNING transition if present; and enter the RUNNING state.

Clear Request
If a value of IDLE is written to the Timer_State property,
then set Timer_State to IDLE; set Last_State_Change to EXPIRED_TO_IDLE; set Expiration_Time to the unspecified datetime value; set Update_Time to current date and time; initiate the write requests for the EXPIRED_TO_IDLE transition if present; and enter the IDLE state.

**Expire Request**

If a value of zero is written to the Present_Value property, or a value of FALSE is written to the Timer_Running property,

then no properties shall be changed; no write requests shall be initiated; and no state transition shall occur.

**Out of Range Request**

If a value outside the supported range and not zero is written to the Present_Value property, or a timer state other than IDLE is written to the Timer_State property,

then no properties shall be changed; no write requests shall be initiated, no state transition shall occur; and a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned.

Since the timer is counting down a duration that is set at the start of the timer, changes to the Local_Date or Local_Time properties have no effect on the timer operation. The Expiration_Time property, in timer state RUNNING, indicates the date and time of expiration. While in this state, Expiration_Time shall be calculated at the time the property is accessed. In other states, Expiration_Time shall be set upon entry to that state.

If the properties State_Change_Values, List_Of_Object_Property_References and Priority_For_Writing are present and writable, then the Timer object shall be capable of writing values of type NULL, BOOLEAN, Unsigned, INTEGER, REAL, and ENUMERATED to properties in the local device referenced by List_Of_Object_Property_References. Support for writing to properties in other devices is optional.

Devices that support Timer objects that contain the Update_Time or the Expiration_Time properties shall support the Local_Date and Local_Time properties in their Device object.

Timer objects that support intrinsic event reporting shall apply the CHANGE_OF_TIMER event algorithm.

The Timer object type and its standardized properties are summarized in Table 12-X and described in detail in this subclause.
Table 12-X. Properties of the Timer Object Type

<table>
<thead>
<tr>
<th>Property Identifier</th>
<th>Property Datatype</th>
<th>Conformance Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object_Identifier</td>
<td>BACnetObjectIdentifier</td>
<td>R</td>
</tr>
<tr>
<td>Object_Name</td>
<td>CharacterString</td>
<td>R</td>
</tr>
<tr>
<td>Object_Type</td>
<td>BACnetObjectType</td>
<td>R</td>
</tr>
<tr>
<td>Description</td>
<td>CharacterString</td>
<td>O</td>
</tr>
<tr>
<td>Present_Value</td>
<td>Unsigned</td>
<td>O</td>
</tr>
<tr>
<td>Status_Flags</td>
<td>BACnetStatusFlags</td>
<td>R</td>
</tr>
<tr>
<td>Event_State</td>
<td>BACnetEventState</td>
<td>O</td>
</tr>
<tr>
<td>Reliability</td>
<td>BACnetReliability</td>
<td>O</td>
</tr>
<tr>
<td>Out_Of_Service</td>
<td>BOOLEAN</td>
<td>O</td>
</tr>
<tr>
<td>Timer_State</td>
<td>BACnetTimerState</td>
<td>R</td>
</tr>
<tr>
<td>Timer_Running</td>
<td>BOOLEAN</td>
<td>R</td>
</tr>
<tr>
<td>Update_Time</td>
<td>BACnetDateTime</td>
<td>O</td>
</tr>
<tr>
<td>Last_State_Change</td>
<td>BACnetTimerTransition</td>
<td>O</td>
</tr>
<tr>
<td>Expiration_Time</td>
<td>BACnetDateTime</td>
<td>O</td>
</tr>
<tr>
<td>Initial_Timeout</td>
<td>Unsigned</td>
<td>O</td>
</tr>
<tr>
<td>Default_Timeout</td>
<td>Unsigned</td>
<td>O</td>
</tr>
<tr>
<td>Min_Pres_Value</td>
<td>Unsigned</td>
<td>O</td>
</tr>
<tr>
<td>Max_Pres_Value</td>
<td>Unsigned</td>
<td>O</td>
</tr>
<tr>
<td>Resolution</td>
<td>Unsigned</td>
<td>O</td>
</tr>
<tr>
<td>List_Of_Object_Property_References</td>
<td>BACnetLIST of BACnetDeviceObjectPropertyReference</td>
<td>O</td>
</tr>
<tr>
<td>Priority_For_Writing</td>
<td>Unsigned</td>
<td>O</td>
</tr>
<tr>
<td>Event_Detection_Enable</td>
<td>BOOLEAN</td>
<td>O</td>
</tr>
<tr>
<td>Notification_Class</td>
<td>Unsigned</td>
<td>O</td>
</tr>
<tr>
<td>Time_Delay</td>
<td>Unsigned</td>
<td>O</td>
</tr>
<tr>
<td>Time_Delay_Normal</td>
<td>Unsigned</td>
<td>O</td>
</tr>
<tr>
<td>Alarm_Values</td>
<td>BACnetLIST of BACnetTimerState</td>
<td>O</td>
</tr>
<tr>
<td>Event_Enable</td>
<td>BACnetEventTransitionBits</td>
<td>O</td>
</tr>
<tr>
<td>Acked_Transitions</td>
<td>BACnetEventTransitionBits</td>
<td>O</td>
</tr>
<tr>
<td>Notify_Type</td>
<td>BACnetNotifyType</td>
<td>O</td>
</tr>
<tr>
<td>Event_Algorithm_Inhibit_Ref</td>
<td>BACnetObjectPropertyReference</td>
<td>O</td>
</tr>
<tr>
<td>Event_Algorithm_Inhibit</td>
<td>BOOLEAN</td>
<td>O</td>
</tr>
<tr>
<td>Reliability_Evaluation_Inhibit</td>
<td>BOOLEAN</td>
<td>O</td>
</tr>
<tr>
<td>Property_List</td>
<td>BACnetARRAY[N] of BACnetPropertyIdentifier</td>
<td>R</td>
</tr>
<tr>
<td>Profile_Name</td>
<td>CharacterString</td>
<td>O</td>
</tr>
</tbody>
</table>

1 These properties are required to be writable when Out_Of_Service is TRUE.
2 If Present_Value is writable when Out_Of_Service is FALSE, then these properties shall be writable.
3 These properties are required if the object supports intrinsic reporting.
4 If Present_Value is writable when Out_Of_Service is FALSE, then these properties are required to be present.
5 If one of these properties is present, then all of these properties shall be present.
6 These properties shall be present only if the object supports intrinsic reporting.
7 Event_Algorithm_Inhibit shall be present if Event_Algorithm_Inhibit_Ref is present.
8 If this property is present, then the Reliability property shall be present.

12.X.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.X.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.X.3 Object_Type
This property, of type BACnetObjectType, indicates membership in a particular object-type class. The value of this property shall be TIMER.

12.X.4 Description
This property is a string of printable characters that may be used to describe the timer or other locally desired descriptive information.

12.X.5 Present_Value
This property, of type Unsigned, indicates, in the RUNNING state, the remaining time, in milliseconds, until the timer expires and the EXPIRED state is entered. This property shall indicate a value of zero if the timer is in the IDLE or EXPIRED state.

Writing a value to this property that is within the supported range, defined by Min_Pres_Value and Max_Pres_Value, shall force the timer to transition to the RUNNING state. The value written shall be used as the initial timeout and set into the Initial_Timeout property.

Writing a value of zero to this property while the timer is in the RUNNING state shall be considered an expire request and force the timer state to transition to state EXPIRED. If a value of zero is written to the property while the timer is in the EXPIRED or IDLE state, then no transition of the timer state shall occur.

Writing a value to this property that is outside the supported range and not zero shall cause a Result(-) to be returned specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

12.X.6 Status_Flags
This property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the 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 does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).

**OVERRIDDEN** The value of this flag shall be logical FALSE (0).

**OUT_OF_SERVICE** Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.X.7 Event_State
The Event_State property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.
12.X.8 Reliability
The Reliability property, of type BACnetReliability, provides an indication of whether the Present_Value or the operation of the timer is "reliable" as far as the BACnet Device or operator can determine and, if not, why. This property shall be writable when Out_Of_Service is TRUE.

12.X.9 Out_Of_Service
The Out_Of_Service property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the timer this object represents is in service and will count down. If Out_Of_Service is FALSE, the timer is functioning as specified. If Out_Of_Service is TRUE, the object shall behave as specified, except that Present_Value shall not automatically count down in the RUNNING state.

While Out_Of_Service is TRUE, the Present_Value and Reliability properties may be changed as a means of simulating states and transitions, or for testing purposes. Writing values to Present_Value shall cause the timer to perform respective timer state transitions as specified in the state machine description. If an event algorithm and/or reliability evaluation is in place, it shall perform its evaluations as specified, regardless of the value of this property.

12.X.10 Timer_State
This property, of type BACnetTimerState, indicates the current state of the timer this object represents. To clear the timer, i.e., to request the timer to enter the IDLE state, a value of IDLE is written to this property.

The values that may be taken on by this property are:

- IDLE: The timer is not running and not expired. Writing this value to this property while in the RUNNING or EXPIRED state will force the timer to enter the IDLE state. If already in the IDLE state, no state transition occurs if this value is written.
- RUNNING: The timer is counting down, and will expire when Present_Value reaches zero or the timer is forced to expire.
- EXPIRED: The timer has expired.

If the object supports event reporting, then this property shall be the pMonitoredValue parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

Writing a value other than IDLE to this property shall cause a Result(-) to be returned specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE.

12.X.11 Timer_Running
This property, of type BOOLEAN, shall have a value of TRUE if the current state of the timer is RUNNING, otherwise FALSE. This property may be used by other objects that require a simple BOOLEAN flag for determining if the timer is in RUNNING state.

Writing a value of TRUE to this property, in any timer state, shall be considered a start request. Present_Value shall be set to the value specified in the Default_Timeout property.

Writing a value of FALSE to this property while the timer is in the RUNNING state shall be considered an expire request and shall force the timer to transition to state EXPIRED. When writing a value of FALSE to this property while the timer is in the EXPIRED or IDLE state, no transition of the timer state shall occur.

12.X.12 Update_Time
This read-only property, of type BACnetDateTime, indicates the date and time of the last transition of the timer state.

If a transition of the timer state has never occurred, then this property shall take on the unspecified datetime value.
If the object supports event reporting, then this property shall be the pUpdateTime parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

### 12.X.13 Last_State_Change
This read-only property, of type BACnetTimerTransition, shall indicate the last transition the timer state machine performed. The values that may be taken on by this property are:

- **NONE**: No timer state transition has occurred since initialization of the object.
- **IDLE_TO_RUNNING**: The last state transition performed by the timer was from IDLE to RUNNING, using either the default or a specific initial timeout.
- **RUNNING_TO_IDLE**: The last state transition performed by the timer was from RUNNING to IDLE. A write of value IDLE to Timer_State occurred.
- **RUNNING_TO_RUNNING**: The last state transition performed by the timer was from RUNNING to RUNNING, using either the default or a specific initial timeout.
- **RUNNING_TO_EXPIRED**: The last state transition performed by the timer was from RUNNING to EXPIRED. The Present_Value reached zero while counting down.
- **FORCED_TO_EXPIRED**: The last state transition performed by the timer was from RUNNING to EXPIRED. A value of zero was written to Present_Value, or a value of FALSE was written to Timer_Running while in the RUNNING state, before Present_Value reached zero.
- **EXPIRED_TO_IDLE**: The last state transition performed by the timer was from EXPIRED to IDLE. The timer expired and a clear request was performed.
- **EXPIRED_TO_RUNNING**: The last state transition performed by the timer was from EXPIRED to RUNNING, using either the default or a specific initial timeout.

If the object supports event reporting, then this property shall be the pLastStateChange parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

### 12.X.14 Expiration_Time
This read-only property, of type BACnetDateTime, indicates the date and time when the timer will expire or has expired.

The value indicated by this property depends on the current timer state, as follows:

- **In timer state IDLE**: This property shall contain the unspecified datetime value.
- **In timer state RUNNING**: The value indicated by this property is the date and time when the timer will expire and will perform a transition to EXPIRED timer state. The value shall be calculated at the time this property is accessed.
- **In timer state EXPIRED**: This property indicates the time at which the object transitioned to the EXPIRED state.

If the object supports event reporting, then this property shall be the pExpirationTime parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.
12.X.15  Initial_Timeout
This read-only property, of type Unsigned, indicates the initial timeout, in milliseconds, that was taken as initial duration
to count down when the timer last transitioned to state RUNNING.

If the object supports event reporting, then this property shall be the pInitialTimeout parameter for the object's event
algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.X.16  Default_Timeout
This property, of type Unsigned, specifies the default initial timeout, in milliseconds. This default timeout is used as the
initial timeout when a value of TRUE is written to the Timer_Running property.

The value of this property shall be within the supported range specified by the values of the Min_Pres_Value and
Max_Pres_Value properties. If this property is writable, writing a value to this property that is outside the supported range
shall cause a Result(-) to be returned specifying an 'Error Class' of PROPERTY and an 'Error Code' of
VALUE_OUT_OF_RANGE.

12.X.17  Min_Pres_Value
This property, of type Unsigned, indicates the minimum initial timeout the timer supports, in milliseconds.

12.X.18  Max_Pres_Value
This property, of type Unsigned, indicates the maximum initial timeout the timer supports, in milliseconds.

12.X.19  Resolution
This property, of type Unsigned, indicates the resolution of the timer, in milliseconds. The time of expiration of the timer
may vary plus or minus this resolution from the remaining time indicated by the Present_Value property.

12.X.20  State_Change_Values
This property, of type BACnetARRAY[7] of BACnetTimerStateChangeValue, represents the values that are to be written
to the referenced properties when a change of the timer state occurs. Each of the elements 1-7 of this array may contain a
value to be written for the respective change of timer state. The array index of the element is equal to the numerical value
of the BACnetTimerTransition enumeration for the respective timer state change. The timer state change NONE has no
corresponding array element.

Each element of this array is of type BACnetTimerStateChangeValue, which is a choice of the following options.

- **no-value**
  This option, of type NULL, is used to indicate that no value shall be written
  when the respective change of timer state occurs.

- **null, boolean, unsigned,
  signed, real, double, octet-
  string, character-string,
  bit-string, enumerated,
  date, time, object-id**
  These options specify a value of the respective primitive datatype. The
  option 'null' may be used in combination with other datatypes for
  specifying a relinquish command for a commandable property.

- **constructed-value**
  This option specifies a value of complex datatype. This option shall not be
  selected if the respective data type is supported by a defined option.

- **date-time**
  This option specifies a value of datatype BACnetDateTime.

- **lighting-command**
  This option specifies a value of datatype BACnetLightingCommand.

All non-NULL values used in this property shall be of the same datatype, and all properties referenced by the
List_Of_Object_Property_References shall be writable with that datatype. If these conditions are not met, then the
Reliability property shall have the value CONFIGURATION_ERROR.

If this property is written with a state change value containing a datatype not supported by this instance of the Timer object
(e.g., the List_Of_Object_Property_References property cannot be configured to reference a property of the unsupported
datatype), the device shall return a Result(-) response, specifying an 'Error Class' of PROPERTY and an 'Error Code' of DATATYPE_NOT_SUPPORTED.

12.X.21 List_Of_Object_Property_References
This property, of type BACnetLIST of BACnetDeviceObjectPropertyReference, specifies the Device Identifiers, Object Identifiers, and Property Identifiers of the properties to be written with specific values at changes of the timer state.

If this property is writable, it may be restricted to only support references to objects inside of the device containing the Timer object. If the property is restricted to referencing objects within the containing device, an attempt to write a reference to an object outside the containing device into this property shall cause a Result(-) to be returned with 'Error Class' of PROPERTY and 'Error Code' of OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED.

If this property is set to reference an object outside the device containing the Timer object, the method used for writing to the referenced property for the purpose of controlling the property is a local matter. The only restriction on the method of writing to the referenced property is that the device be capable of using WriteProperty for this purpose so as to be interoperable with all BACnet devices.

12.X.22 Priority_For_Writing
This property, of type Unsigned, defines the priority at which the referenced properties are commanded. It corresponds to the 'Priority' parameter of the WriteProperty service. It is an unsigned integer in the range 1-16, with 1 being considered the highest priority and 16 the lowest. See Clause 19.

12.X.23 Event_Detection_Enable
This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.X.24 Notification_Class
This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.X.25 Time_Delay
This property, of type Unsigned, is the pTimeDelay parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.X.26 Time_Delay_Normal
This property, of type Unsigned, is the pTimeDelayNormal parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.X.27 Alarm_Values
This property, of type BACnetLIST of BACnetTimerState, is the pAlarmValues parameter of the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.X.28 Event_Enable
This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO-OFFNORMAL, TO-FAULT, and TO-NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.X.29 Acked_Transitions
This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.X.30 Notify_Type
This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the ‘Notify Type’ service parameter in event notifications generated by the object.

12.X.31 Event_Time_Stamps
This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events (see 13.2.2.1). 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 ever occurred for the object.

12.X.32 Event_Message_Texts
This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.X.33 Event_Message_Texts_Config
This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the ‘Message Text’ parameter for the event notifications of TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.X.34 Event_Algorithm_Inhibit_Ref
This property, of type BACnetObjectPropertyReference, indicates the property which controls the value of property Event_Algorithm_Inhibit. When this property is present and initialized (contains an instance other than 4194303), the referenced property shall be of type BACnetBinaryPV or BOOLEAN.

12.X.35 Event_Algorithm_Inhibit
This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the event algorithm has been disabled for the object (see Clause 13.2.2.1). This property is a runtime override that allows temporary disabling of the event algorithm.

If the Event_Algorithm_Inhibit_Ref property is present and initialized (contains an instance other than 4194303), then the Event_Algorithm_Inhibit property shall be read only and shall reflect the value of the property referenced by Event_Algorithm_Inhibit_Ref. A BACnetBinaryPV value of INACTIVE shall map to a value of FALSE, and a value of ACTIVE shall map to a value of TRUE. If the referenced property does not exist, it shall be assumed to have a value of FALSE.

If the Event_Algorithm_Inhibit_Ref property is absent or is uninitialized and Event_Detection_Enable is TRUE, then the Event_Algorithm_Inhibit property shall be writable.

12.X.36 Reliability_Evaluation_Inhibit
This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.X.37 Property_List
This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.X.38 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 12-15, p. 210]

Table 12-15. Event Algorithm, Event Parameters and Event Algorithm Parameters

<table>
<thead>
<tr>
<th>Event Algorithm</th>
<th>Event Parameters</th>
<th>Event Algorithm Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>none</td>
<td>none</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>CHANGE_OF_TIMER</td>
<td>Time_Delay</td>
<td>pTimeDelay</td>
</tr>
<tr>
<td></td>
<td>Alarm_Values</td>
<td>pAlarmValues</td>
</tr>
<tr>
<td></td>
<td>Update_Time_Reference</td>
<td>Referent's value is pUpdateTime</td>
</tr>
</tbody>
</table>

[Change Table 12-15.1, p. 211]

Table 12-15.1. Additional Monitored Object Properties by Event Algorithm

<table>
<thead>
<tr>
<th>Event Algorithm</th>
<th>Additional Monitored Object Properties</th>
<th>Event Algorithm Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>none</td>
<td>none</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>CHANGE_OF_TIMER</td>
<td>Status_Flags</td>
<td>pStatusFlags</td>
</tr>
<tr>
<td></td>
<td>Initial_Timeout</td>
<td>pInitialTimeout</td>
</tr>
<tr>
<td></td>
<td>Expiration_Time</td>
<td>pExpirationTime</td>
</tr>
<tr>
<td></td>
<td>Last_State_Change</td>
<td>pLastStateChange</td>
</tr>
</tbody>
</table>

[Change Table 13-5, p. 473]

Table 13-5. Properties Reported in CHANGE_OF_RELIABILITY Notifications

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Door</td>
<td>Door_Alarm_State</td>
</tr>
<tr>
<td></td>
<td>Present_Value</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Timer</td>
<td>Present_Value</td>
</tr>
<tr>
<td></td>
<td>Timer_State</td>
</tr>
<tr>
<td></td>
<td>Update_Time(^1)</td>
</tr>
<tr>
<td></td>
<td>Last_State_Change(^3)</td>
</tr>
<tr>
<td></td>
<td>Initial_Timeout(^2)</td>
</tr>
<tr>
<td></td>
<td>Expiration_Time(^2)</td>
</tr>
</tbody>
</table>

\(^1\) This value may be excluded from the property-value parameter due to security requirements.
\(^2\) This property is, or may be, from a referenced object. If the value is not known by the event-initiating object, then it shall not be included in the property-value parameter.
\(^3\) These properties are optional and are included only if present in the object.

[Change Table 13-7, p. 475]

Table 13-7. Standardized Event Algorithms

<table>
<thead>
<tr>
<th>Event Algorithm</th>
<th>Clause</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>13.3.17</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>CHANGE_OF_TIMER</td>
<td>13.3.Y</td>
</tr>
</tbody>
</table>
13.3.Y CHANGE_OF_TIMER Event Algorithm

The CHANGE_OF_TIMER event algorithm detects whether the monitored value equals a value that is listed as an alarm value. The monitored value shall be of type BACnetTimerState.

The parameters of this event algorithm are:

- **pCurrentState**
  - This parameter, of type BACnetEventState, represents the current value of the Event_State property of the object that applies the event algorithm.

- **pMonitoredValue**
  - This parameter, of type BACnetTimerState, represents the current value of the monitored property.

- **pStatusFlags**
  - This parameter, of type BACnetStatusFlags, represents the current value of the Status_Flags property of the object containing the property that provides the value of the pMonitoredValue parameter. If no value is available for this parameter, then it takes on the value {FALSE, FALSE, FALSE, FALSE}.

- **pUpdateTime**
  - This parameter, of type BACnetDateTime, represents the date and time of update of the monitored timer state value.

- **pLastStateChange**
  - This parameter, of type BACnetTimerTransition, represents the last transition of the monitored timer state value.

- **pInitialTimeout**
  - This parameter, of type Unsigned, represents the initial timeout value in milliseconds at the last transition to the RUNNING timer state.

- **pExpirationTime**
  - This parameter, of type BACnetDateTime, represents the expiration time that was calculated at the time of application of the algorithm.

- **pAlarmValues**
  - This parameter, of type List Of BACnetTimerState, represents a list of timer state values that are considered offnormal values.

- **pTimeDelay**
  - This parameter, of type Unsigned, represents the time, in seconds, that the offnormal conditions must exist before an offnormal event state is indicated.

- **pTimeDelayNormal**
  - This parameter, of type Unsigned, represents the time, in seconds, that the Normal conditions must exist before a NORMAL event state is indicated. If no value is available for this parameter, then it takes on the value of the pTimeDelay parameter.

The conditions evaluated by this event algorithm are:

(a) If pCurrentState is NORMAL, and pMonitoredValue is equal to any of the values contained in pAlarmValues for pTimeDelay, then indicate a transition to the OFFNORMAL event state.

(b) If pCurrentState is OFFNORMAL, and pMonitoredValue is not equal to any of the values contained in pAlarmValues for pTimeDelayNormal, then indicate a transition to the NORMAL event state.

(c) If pCurrentState is OFFNORMAL, and pMonitoredValue is equal to one of the values contained in pAlarmValues, and pUpdateTime contains a value that is different from the value present at the last transition to OFFNORMAL, then indicate a transition to the OFFNORMAL event state.

Figure 13-Y depicts those transitions of Figure 13-X3 that this event algorithm may indicate:
Figure 13-Y. Transitions indicated by CHANGE_OF_TIMER algorithm

The notification parameters of this algorithm are:

- **New_State**: This notification parameter, of type BACnetTimerState, conveys the value of pMonitoredValue.
- **Status_Flags**: This notification parameter, of type BACnetStatusFlags, conveys the value of pStatusFlags.
- **Update_Time**: This notification parameter, of type BACnetDateTime, conveys the value of pUpdateTime.
- **Last_State_Change**: This optional notification parameter, of type BACnetTimerTransition, conveys the value of pLastStateChange, if one is available.
- **Initial_Timeout**: This optional notification parameter, of type Unsigned, conveys the value of pInitialTimeout, if one is available.
- **Expiration_Time**: This optional notification parameter, of type BACnetDateTime, conveys the value of pExpirationTime, if one is available.

[Add new ASN.1 productions to Clause 21, p. 639]

\[
\textbf{BACnetTimerStateChangeValue} ::= \text{CHOICE} \{
\begin{array}{ll}
\text{null} & \text{NULL}, \\
\text{boolean} & \text{BOOLEAN}, \\
\text{unsigned} & \text{Unsigned}, \\
\text{signed} & \text{INTEGER}, \\
\text{real} & \text{REAL}, \\
\text{double} & \text{Double}, \\
\text{octet-string} & \text{OCTET STRING}, \\
\text{character-string} & \text{CharacterString}, \\
\text{bit-string} & \text{BIT STRING}, \\
\text{enumerated} & \text{ENUMERATED}, \\
\text{date} & \text{Date}, \\
\text{time} & \text{Time}, \\
\text{object-id} & \text{BACnetObjectIdentifier}, \\
\text{no-value} & [0] \text{NULL}, \\
\text{constructed-value} & [1] \text{ABSTRACT-SYNTAX.&Type}, \\
\text{date-time} & [2] \text{BACnetDateTime}, \\
\text{lighting-command} & [3] \text{BACnetLightingCommand} \\
\end{array}
\]

--- This choice may be extended in the future with additional context
-- tagged options for supporting other standardized complex data types

\textbf{BACnetTimerState} ::= \texttt{ENUMERATED} {
  idle \hspace{1em} (0),
  running \hspace{1em} (1),
  expired \hspace{1em} (2)
}

\textbf{BACnetTimerTransition} ::= \texttt{ENUMERATED} {
  none \hspace{1em} (0),
  idle-to-running \hspace{1em} (1),
  running-to-idle \hspace{1em} (2),
  running-to-running \hspace{1em} (3),
  running-to-expired \hspace{1em} (4),
  forced-to-expired \hspace{1em} (5),
  expired-to-idle \hspace{1em} (6),
  expired-to-running \hspace{1em} (7)
}

[Change ASN.1 productions in \textbf{Clause 21}, p. 639]

\textbf{BACnetEventParameter} ::= \texttt{CHOICE} {

-- These choices have a one-to-one correspondence with the Event_Type enumeration with the exception of the
-- complex-event-type, which is used for proprietary event types.

...,

\textit{change-of-status-flags} \hspace{1em} [18] \texttt{SEQUENCE} {
  \textit{time-delay} \hspace{1em} [0] \texttt{Unsigned},
  \textit{selected-flags} \hspace{1em} [1] \texttt{BACnetStatusFlags}
}\texttt{,}

\textit{change-of-timer} \hspace{1em} [22] \texttt{SEQUENCE} {
  \textit{time-delay} \hspace{1em} [0] \texttt{Unsigned},
  \textit{alarm-values} \hspace{1em} [1] \texttt{SEQUENCE OF BACnetTimerState},
  \textit{update-time-reference} \hspace{1em} [2] \texttt{BACnetDeviceObjectPropertyReference}
}\texttt{,}

}\texttt{,}

\textbf{BACnetEventType} ::= \texttt{ENUMERATED} {

...,

\textit{none} \hspace{1em} (20),
\textit{change-of-timer} \hspace{1em} (22),

...\texttt{,}

}\texttt{,}

\textbf{BACnetNotificationParameters} ::= \texttt{CHOICE} {

-- These choices have a one-to-one correspondence with the Event_Type enumeration with the exception of the
-- complex-event-type, which is used for proprietary event types.

...,

\textit{change-of-reliability} \hspace{1em} [19] \texttt{SEQUENCE} {
  \textit{reliability} \hspace{1em} [0] \texttt{BACnetReliability},
  \textit{status-flags} \hspace{1em} [1] \texttt{BACnetStatusFlags},
  \textit{property-values} \hspace{1em} [2] \texttt{SEQUENCE OF BACnetPropertyValue}
}\texttt{,}

-- context tag [20] is not used, see note below

© ASHRAE (www.ashrae.org). For personal use only. Additional reproduction, distribution, or transmission in either print or digital form is not permitted without ASHRAE's prior written permission.
change-of-timer [22] SEQUENCE {
  new-state [0] BACnetTimerState,
  status-flags [1] BACnetStatusFlags,
  update-time [2] BACnetDateTime,
  last-state-change [3] BACnetTimerTransition OPTIONAL,
  initial-timeout [4] Unsigned OPTIONAL,
  expiration-time [5] BACnetDateTime OPTIONAL
}

BACnetObjectType ::= ENUMERATED { -- see below for numerical order
  ... time-value (50),
  timer (31),
  trend-log (20),
  trend-log-multiple (27),
  -- numerical order reference
  ...
  -- see access-door (30),
  value 31 is unassigned
  -- see timer (31),
  -- see access-credential (32),
  ...
  -- 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.

BACnetObjectTypesSupported ::= BIT STRING {
  ...
  access-door (30),
  value 31 is unassigned
  timer (31),
  access-credential (32),
  ...
}

BACnetPropertyIdentifier ::= ENUMERATED { -- see below for numerical order
  ...
  default-step-increment (376),
  default-timeout (393),
  derivative-constant (26),
  ...
  inactive-text (46),
  initial-timeout (394),
  in-process (47),
  ...
  last-restore-time (157),
  last-state-change (395),
  last-use-time (281),
  ...
  start-time (142),
  state-change-values (396),
  state-description (222),
  ...
  time-synchronization-recipients (116),
  timer-running (397),
  timer-state (398),
  total-record-count (145),
  ...

© ASHRAE (www.ashrae.org). For personal use only. Additional reproduction, distribution, or transmission in either print or digital form is not permitted without ASHRAE’s prior written permission.
...  

-- numerical order reference  

...  

-- see egges-active (386),  
-- see default-timeout (393),  
-- see initial-timeout (394),  
-- see last-state-change (395),  
-- see state-change-values (396),  
-- see timer-running (397),  
-- see timer-state (398),  
...  

}  

BACnetPropertyStates ::= CHOICE {  

  ...  

  lighting-transition [40] BACnetLightingTransition,  
  timer-state [43] BACnetTimerState,  
  timer-transition [44] BACnetTimerTransition,  
...  

}  

[Change Clause 22.2.1.3, p. 716]  

22.2.1.3 Scheduling  

"Scheduling" is the exchange of data between BACnet devices related to the establishment and maintenance of dates and times at which specified output actions are to be taken. Interoperability in this area permits the use of date and time schedules or timers for starting and stopping equipment and changing control setpoints as well as other analog or binary parameters.  

[Add new BIBBs to Annex K.3, pp. 893]  

K.3.X1 BIBB - Scheduling-Timer-Internal-B (SCHED-TMR-I-B)  

The B device provides timed modification of the values of specific properties of specific objects within the device. In addition to supporting the DS-RP-B and DS-WP-B BIBBs, each device claiming conformance to SCHED-TMR-I-B shall also be capable of possessing at least one Timer object.  

The Present_Value, the Timer_State, and the Timer_Running properties in the Timer object shall be writable.  

The Default_Timeout property in the Timer object shall be present and writable. The Timer object shall support a writable State_Change_Values property. The Priority_For_Writing property in the Timer object shall be writable.  

K.3.X2 BIBB - Scheduling-Timer-External-B (SCHED-TMR-E-B)  

The B device provides timed modification of the values of specific properties of specific objects in other devices. Devices claiming conformance to SCHED-TMR-E-B shall also support SCHED-TMR-I-B and DS-WP-A. The List_Of_Object_Property_References property of the Timer object shall support references to objects in external devices and be writable.  

[Change Clause K.3.5, p. 890]  

K.3.5 BIBB - Scheduling-Advanced View and Modify-A (SCHED-AVM-A)  

The A device manipulates schedules and calendars, schedules, calendars, and timers on the B device. The A device shall support the DM-ODC-A, DS-RP-A and DS-WP-A BIBBs.
The A device uses ReadProperty to retrieve for presentation and WriteProperty to modify each of the Schedule and Calendar, Schedule, Calendar, and Timer properties listed below. The A device shall be capable of using the CreateObject and DeleteObject services to create and delete Schedule and Calendar, Schedule, Calendar, and Timer objects on the B device. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

<table>
<thead>
<tr>
<th>BACnet Service</th>
<th>Initiate</th>
<th>Execute</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateObject</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>DeleteObject</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>ReadProperty</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>WriteProperty</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

The A device shall support the creation, presentation and modification of all forms of the Weekly_Schedule, Exception_Schedule and Date_List properties with the following limitations. At a minimum, the A device shall be capable of handling Exception_Schedule properties with up to 255 entries, and 12 BACnetTimeValue tuples per entry, Weekly_Schedule properties with up to 6 entries per day, and Date_List properties with up to 32 entries.

Devices claiming support for this BIBB shall be capable of creating, deleting, presenting, and modifying Schedule objects that schedule any of the following types:

**REAL, ENUMERATED, Unsigned32**

and which may contain NULL values and shall be capable of changing the datatype that a Schedule object schedules. Schedule objects contain a number of properties that need to be consistent in the datatype of the values they contain. Devices claiming support for this BIBB shall be prepared to interact, allow display and modification of, Schedule objects that are self-inconsistent. A self-inconsistent Schedule object is one in which the scheduled values in the Weekly_Schedule, Exception_Schedule, and Schedule_Default properties are not all of the same datatype or in which the controlled objects are not all of the same datatype or are of a datatype different than the scheduled values.

Devices claiming support for this BIBB shall be capable of creating, deleting, presenting, and modifying Timer objects that write any of the following types:

**REAL, ENUMERATED, Unsigned32, NULL**

and shall be capable of changing the datatype that a Timer object writes, and shall be capable of setting the 'No Value' option. Timer objects contain a number of properties that need to be consistent in the datatype of the values they contain. Devices claiming support for this BIBB shall be prepared to interact, allow display and modification of, Timer objects that are self-inconsistent. A self-inconsistent Timer object is one in which the values to write in the State_Change_Values property are not all of the same datatype or in which the controlled objects are not all of the same datatype or are of a datatype different than the values to be written.
The A device shall be capable of creating, deleting, presenting and modifying schedules and timers in any B device regardless of the B device’s claimed Protocol Revision.

The A device shall be capable of creating, deleting, presenting and modifying schedule objects that do not contain an Exception_Schedule.

Devices claiming support for this BIBB shall be capable of providing times and values in the time/value pairs within the full range as defined in Tables K-4 and K-6.

Actions taken by Device A when retrieval of a value for display fails are a local matter.

A device claiming support for SCHED-AVM-A is interoperable with devices that support any of the B side schedule and timer BIBBs.

Support for timer objects is not required for A side devices that claim a protocol revision less than X.

[Change Clause K.3.6, p. 891]

K.3.6 BIBB - Scheduling-View and Modify-A (SCHED-VM-A)

The A device manipulates schedules and calendars schedules, calendars, and timers on the B device. The A device shall support the DS-RP-A and DS-WP-A BIBBs.

<table>
<thead>
<tr>
<th>BACnet Service</th>
<th>Initiate</th>
<th>Execute</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadProperty</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>WriteProperty</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

Device A shall be capable of using ReadProperty to retrieve for presentation and WriteProperty to modify each of the Schedule, Calendar, and Timer properties listed below. Device A may use alternate services where support for execution of the alternate service is supported by Device B.

**Table K-13. Properties That SCHED-VM-A Devices Shall Be Capable of Presenting and Modifying**

<table>
<thead>
<tr>
<th>Schedule</th>
<th>Calendar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Effective_Period</td>
<td>Date_List</td>
</tr>
<tr>
<td>Weekly_Schedule</td>
<td></td>
</tr>
<tr>
<td>Exception_Schedule</td>
<td></td>
</tr>
</tbody>
</table>

**Timer**

| Present_Value             |                   |
| Timer_State               |                   |
| Timer_Running             |                   |
| Default_Timeout           |                   |
| State_Change_Values       |                   |

The A device shall support the presentation and modification of all forms of the Weekly_Schedule, Exception_Schedule and Date_List properties with the following limitations. At a minimum, the A device shall be capable of handling Exception_Schedule properties with up to 255 entries, and 12 BACnetTimeValue tuples per entry, Weekly_Schedule properties with up to 6 entries per day, and Date_List properties with up to 32 entries.

Devices claiming support for this BIBB shall be capable of presenting, and modifying Schedule objects that schedule any of the following types:

REAL, ENUMERATED, Unsigned32

and which may contain NULL values.

Devices claiming support for this BIBB shall be capable of presenting and modifying Timer objects that write any of the following types:
REAL, ENUMERATED, Unsigned32, NULL

and which may contain the 'No-Value' option.

The A device shall be capable of presenting and modifying schedules and timers in any B devices regardless of the B device's claimed Protocol_Revision.

The A device shall be capable of presenting and modifying schedule objects that do not contain an Exception_Schedule.

Devices claiming support for this BIBB shall be capable of providing times and values in the time/value pairs within the full range as defined in Table K-6.

A device claiming support for SCHED-VM-A is interoperable with devices that support any of the B-side schedule and timer BIBBs.

Support for timer objects is not required for A side devices that claim a protocol revision less than X.
135-2012ay-2 Correct Expiry_Time property name to Expiration_Time in the Access Credential Object

**Rationale**

The name Expiry_Time for the Access Credential expiration time property is corrected to Expiration_Time.

[Change Table 12-40, p. 360]

**Table 12-40. Properties of the Access Credential Object Type**

<table>
<thead>
<tr>
<th>Property Identifier</th>
<th>Property Datatype</th>
<th>Conformance Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Expiry_Time</td>
<td>Expiration_Time</td>
<td>BACnetDateTime</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

[Change Clause 12.35.9, p. 362]

**12.35.9 Reason_For_Disable**

...  
DISABLED_EXPIRED The credential is no longer valid. The current time is after the Expiry_Time Expiration_Time.

...  

[Change Clause 12.35.12, p. 363]

**12.35.12 Expiry_Time Expiration_Time**

This property, ...

[Change ASN.1 productions in Clause 21, p. 639]

```
BACnetPropertyIdentifier ::= ENUMERATED { -- see below for numerical order
  ...
  expiry time        (270),
  expiration-time     (270),
  ...
  -- numerical order reference
  ...
  -- see expiry-time     (270),
  -- see expiration-time (270),
  ...
}
```
[Add a new entry to History of Revisions, p. 1039]

(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**

<table>
<thead>
<tr>
<th>Protocol Version</th>
<th>Revision</th>
<th>Summary of Changes to the Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>
| 1                | 17       | Addendum *ay* to ANSI/ASHRAE 135-2012  
Approved by the ASHRAE Standards Committee *Xxxx xx*, 2015; by the ASHRAE  
Board of Directors *Xxxx xx*, 2015; and by the American National Standards Institute  
*Xxxx xx*, 2015.  
1. Add a Timer Object Type  
2. Correct Expiry_Time property name to Expiration_Time in the Access Credential Object |

© ASHRAE (www.ashrae.org). For personal use only. Additional reproduction, distribution, or transmission in either print or digital form is not permitted without ASHRAE's prior written permission.
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.
About ASHRAE

ASHRAE, founded in 1894, is a global society advancing human well-being through sustainable technology for the built environment. The Society and its members focus on building systems, energy efficiency, indoor air quality, refrigeration, and sustainability. Through research, Standards writing, publishing, certification and continuing education, ASHRAE shapes tomorrow’s built environment today.

For more information or to become a member of ASHRAE, visit www.ashrae.org.

To stay current with this and other ASHRAE Standards and Guidelines, visit www.ashrae.org/standards.

Visit the ASHRAE Bookstore

ASHRAE offers its Standards and Guidelines in print, as immediately downloadable PDFs, on CD-ROM, and via ASHRAE Digital Collections, which provides online access with automatic updates as well as historical versions of publications. Selected Standards and Guidelines are also offered in redline versions that indicate the changes made between the active Standard or Guideline and its previous version. For more information, visit the Standards and Guidelines section of the ASHRAE Bookstore at www.ashrae.org/bookstore.

IMPORTANT NOTICES ABOUT THIS STANDARD

To ensure that you have all of the approved addenda, errata, and interpretations for this Standard, visit www.ashrae.org/standards to download them free of charge.

Addenda, errata, and interpretations for ASHRAE Standards and Guidelines are no longer distributed with copies of the Standards and Guidelines. ASHRAE provides these addenda, errata, and interpretations only in electronic form to promote more sustainable use of resources.