



# ADDENDA

**ANSI/ASHRAE Addendum ca to  
ANSI/ASHRAE Standard 135-2020**



# A Data Communication Protocol for Building Automation and Control Networks

Approved by ASHRAE and the American National Standards Institute on January 21, 2022.

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. Instructions for how to submit a change can be found on the ASHRAE® website (<https://www.ashrae.org/continuous-maintenance>).

The latest edition of an ASHRAE Standard may be purchased on the ASHRAE website ([www.ashrae.org](http://www.ashrae.org)) or from ASHRAE Customer Service, 180 Technology Parkway NW, Peachtree Corners, GA 30092. E-mail: [orders@ashrae.org](mailto: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](http://www.ashrae.org/permissions).

© 2022 ASHRAE

ISSN 1041-2336



**ASHRAE Standing Standard Project Committee 135**

**Cognizant TC: 1.4, Control Theory and Application**

**SPLS Liaison: Charles S Barnaby**

Coleman L. Brumley, Jr.*, <i>Chair</i>	Bernhard Isler	Frank Schubert
Scott Ziegenfus, <i>Vice-Chair</i>	Jake Kopocis*	Marcelo Richter da Silva
Nate Benes*	Thomas Kurowski*	Ted Sunderland
Steven T Bushby*	Shahid Naeem	Lori Tribble
Salvatore Cataldi*	Frank V. Neher*	James F. Butler
Tyler Cove	Carl Neilson*	David Fisher
Brandon M. DuPrey*	Michael Osborne	Grant N. Wichenko*
Siddharth Goyal	Scott Reed	
Alexander Gurciullo*	David Robin*	

\* Denotes members of voting status when the document was approved for publication

---

**ASHRAE STANDARDS COMMITTEE 2021–2022**

Rick M. Heiden, <i>Chair</i>	Essam E. Khalil	Justin M. Prosser
Susanna S. Hanson, <i>Vice-Chair</i>	Malcolm D. Knight	David Robin
Charles S. Barnaby	Jay A. Kohler	Lawrence J. Schoen
Robert B. Burkhead	Cesar L. Lim	Steven C. Sill
Thomas E. Cappellin	Paul A. Lindahl, Jr.	Christian R. Taber
Douglas D. Fick	James D. Lutz	Russell C. Tharp
Michael W. Gallagher	Julie Majurin	William F. Walter
Patricia Graef	Lawrence C. Markel	Craig P. Wray
Srinivas Katipamula	Margret M. Mathison	Jaap Hogeling, BOD ExO
Gerald J. Kettler	Gwelen Paliaga	Tim J. McGinn, CO

Connor Barbaree, *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

- interpretation of the contents of this Standard,
- participation in the next review of the Standard,
- offering constructive criticism for improving the Standard, or
- 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.*

***135-2020ca-1. Introduce the Concept of Color for BACnet, p. 3***

***135-2020ca-2. Add new Color object type, p. 4***

***135-2020ca-3. Add new Color Temperature object type, p. 11***

***135-2020ca-4. Add color-reference properties to LO and BLO object types, p. 19***

***135-2020ca-5. Add high/low trim to LO object type, p.23***

***135-2020ca-6. Aggregated changes to Clause 21 and 25, p. 26***

***135-2020ca-7. BIBB Changes to Support Additional Object Types, p. 29***

In the following document, language to be added to existing clauses of ANSI/ASHRAE 135-2020 and Addenda is indicated through the use of *italics*, while deletions are indicated by ~~strike through~~. 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 document is provided for context only and is not open for public review comment except as it relates to the proposed changes.

The use of placeholders like X, Y, Z, X1, X2, N, NN, x, n, ?, etc., should not be interpreted as literal values of the final published version. These placeholders will be assigned actual numbers/letters only after final publication approval of the addendum.

## 135-2020ca-1. Introduce the Concept of Color for BACnet

[The following introduction is informative and provided as background information and rationale for this addendum. It will not be part of the standard.]

### INTRODUCTION

The BACnet Lighting Output (LO) and Binary Lighting Output (BLO) objects provide support for lighting applications and a standardized object model for white light endpoints that perform lighting control. In representing lighting applications with these object types, the central idea is the control of luminance on a normalized scale from off to some maximum lighting level. These objects do not address the concept of color in their object models. There are architectural and even commercial lighting applications, where color is a useful and important concept. There are other applications where it would be advantageous to model color, and changing color, as a characteristic not even necessarily coupled to lighting.

It is increasingly common to incorporate color control into lighting applications. However, most lighting applications separate the concept of luminance from the concept of color itself. There are several different color models in widespread use. These proposals focus on the use of the **CIE 15: 2004 Chromaticity Diagram** that models color as an x-y space and specific colors are (x,y) coordinates within this space. Color endpoints such as lighting controller outputs or luminaires are thus represented in BACnet as a pair of objects such as an LO and a Color object. The LO controls the luminance while the Color controls the x-y color.

The proposed Color object incorporates ideas from the lighting output objects in terms of fading, ramping and stepping. There is a significant difference between the concept in the lighting objects where Present\_Value is *commandable*, and the Color object where Present\_Value is not commandable. This difference is intentional and represents the different way that color lighting is commonly controlled vs. simple luminance in white light applications. In white light applications it is common to have more than one source of control, for example scheduled lighting level changes and changes initiated by a local controller device (dimmer or switch). The prioritization scheme that BACnet uses is directly applicable to this kind of application. However, in color lighting applications there is typically only one source for the desired color. Because of this, there is no use case where arbitration is required between multiple color control clients and a color endpoint.

There is a notable exception. In some color applications such as vivaria, it is desirable to control color over a long period of time, for example fading between specific colors over many hours. The scenario when a second “color” might be needed would be for example an emergency or maintenance situation when the normal long-term color fade must be temporarily interrupted perhaps with pure white light until the emergency is dealt with. At the end of the emergency the normal color control can take over and restore the color to where it would have been if it was not interrupted. The point is that there are only ever two sources for color information: normal operation and emergency operation. As a result, color does not require the complexity of commandability when a much simpler mechanism can provide a solution to the only use cases we have identified.

There is also the concept of color temperature. In some cases the “color” that is to be controlled is actually the white light color temperature, which is a specific region of the x-y color space called the white light curve. Color temperature applications are distinctly different from full color applications, sufficiently so that a separate Color Temperature object is also proposed.

### 135-2020ca-2. Add new Color object type

#### Rationale

The 135-2020 Lighting Output and Binary Lighting Output objects do not address the concept of color in their object models. There are architectural and even commercial lighting applications, where color is a useful and important concept. There are other applications where it would be advantageous to model color, and changing color, as a characteristic not even necessarily coupled to lighting.

These changes extend the BACnet object model to accommodate color.

[Change 3.2]

**fading:** the gradual increase or decrease of the ~~actual output~~ *luminance* from one setting to another over a fixed period of time, *or the change of color or color temperature from one color to another over a fixed period of time, or a change of both luminance level and color.*

**xyColor:** *The numeric representation of a color in terms of its (x,y) coordinates in the CIE chromaticity diagram, independent of its luminance.*

[Change Table 12-63]

**Table 12-63 – Datatype Coercion Rules**

Datatype in Present_Value write	Datatype of referenced property															
	unknown	BOOLEAN	Unsigned	INTEGER	REAL	Double	OCTET STRING	CharacterString	BIT STRING	ENUMERATED	Date	Time	BACnetObjectIdentifier	BACnetLightingCommand	BACnetColorCommand	BACnetxyColor
NULL	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	ID	ID	ID
BOOLEAN	NC	NC	2	2	2	2	ID	ID	ID	2	ID	ID	ID	ID	ID	ID
Unsigned	NC	1	NC	3	3	3	ID	ID	ID	NC	ID	ID	NC	ID	ID	ID
INTEGER	NC	1	4	NC	4	4	ID	ID	ID	4	ID	ID	ID	ID	ID	ID
REAL	NC	1	5	5	NC	5	ID	ID	ID	5	ID	ID	ID	ID	ID	ID
Double	NC	1	6	6	6	NC	ID	ID	ID	6	ID	ID	ID	ID	ID	ID
OCTET STRING	NC	ID	ID	ID	ID	ID	NC	ID	ID	ID	ID	ID	ID	ID	ID	ID
CharacterString	NC	ID	ID	ID	ID	ID	NC	ID	ID	ID	ID	ID	ID	ID	ID	ID
BIT STRING	NC	ID	ID	ID	ID	ID	ID	NC	ID	ID	ID	ID	ID	ID	ID	ID
ENUMERATED	NC	1	NC	3	3	3	ID	ID	ID	NC	ID	ID	ID	ID	ID	ID
Date	NC	ID	ID	ID	ID	ID	ID	ID	ID	ID	NC	ID	ID	ID	ID	ID
Time	NC	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	NC	ID	ID	ID	ID
BACnetObjectIdentifier	NC	ID	NC	ID	ID	ID	ID	ID	ID	ID	ID	ID	NC	ID	ID	ID
BACnetLightingCommand	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	NC	ID	ID
BACnetColorCommand	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	NC	ID
BACnetxyColor	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	NC

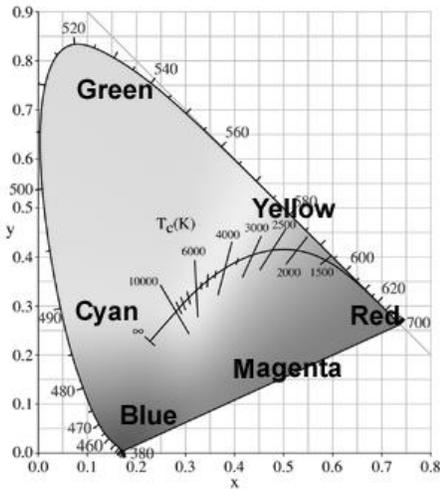
NC=No Coercion ID=Invalid Datatype

[Add 12.X, Color Object Type]

## 12.X Color Object Type

The Color object type defines a standardized object whose properties represent the externally visible characteristics of a color. This color may be used to affect the visible color produced by any device, typically, but not limited to, lighting outputs. The Color object includes dedicated functionality specific to color control that would otherwise require explicit programming. The color itself is analog in nature and shall be referred to as “color output.” The color may be used by the device in various applications, for example to represent the color of light produced by a lighting device. There is an implicit connection in such cases between the object that controls luminance, e.g. a Lighting Output or Binary Lighting Output, and a Color object that controls its color. This connection is BACnet-visible in the Color\_Reference and Override\_Color\_Reference properties of those objects. See 12.54.X, 12.54.Z, 12.55.X and 12.55.Z.

The color can be changed directly to an absolute color by writing to the Present\_Value of the Color object. The color may also be changed by writing to the Color\_Command property. The color command provides additional color control functionality with specific functions such as fading.



The Color object supports a single color-model called xyColor. In the xyColor model, the parameters are the x and y coordinates of the color on the CIE chromaticity diagram (CIE 15: 2004), expressed as a pair of numbers each in the range 0 to 1. Writes of BACnetxyColor values that are outside the range 0 to 1 shall be considered “values out of range.” Values within the range 0 to 1 but not within the curved space of the CIE chromaticity diagram, or xy values not supported by the device, may be considered “values out-of-range” or choose the closest color supported, the choice being a local matter. The algorithm used to determine the closest color shall be a local matter. When fading from one color to another the algorithm used to determine the intermediate colors shall be a local matter.

Figure 12-X: CIE Chromaticity Diagram

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

**Table 12-X.** Properties of the Color Object

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	BACnetxyColor	W
Tracking_Value	BACnetxyColor	R
Color_Command	BACnetColorCommand	W
In_Progress	BACnetColorOperationInProgress	R
Default_Color	BACnetxyColor	R
Description	CharacterString	O
Default_Fade_Time	Unsigned	R
Transition	BACnetColorTransition	O
Value_Source	BACnetValueSource	O <sup>1,2,3</sup>
Audit_Level	BACnetAuditLevel	O <sup>4</sup>
Auditable_Operations	BACnetAuditOperationFlags	O <sup>4</sup>
Tags	BACnetARRAY[N] of BACnetNameValue	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

<sup>1</sup> This property is required if the object supports the value source mechanism.

<sup>2</sup> This property shall be present only if the object supports the value source mechanism.

<sup>3</sup> This property shall be writable as described in Clause 19.5.

<sup>4</sup> This property shall be present only if the device supports audit reporting.

### 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 COLOR.

### 12.X.4 Present\_Value

This property, of type BACnetxyColor, shall indicate the target color value for the color output. The range of values for x and y shall be 0.0 to 1.0.

Present\_Value may also be affected by writes to the Color\_Command property that initiate color control commands. These commands may asynchronously affect the color output by establishing a new target for Present\_Value and carrying out the requested operation. Transitioning from one color to another is supported by writing a FADE\_TO\_COLOR command to the property Color\_Command. The current color is always indicated in the Tracking\_Value property. If a color command is currently in progress and the Present\_Value is written, the color command shall be halted (see Clause 12.X.6.1 Halting a Color Command in Progress).

The color output shall be updated whenever the Present\_Value is written or changed as a result of executing a color command. However, when the device starts up or is reset, it is a local matter as to whether the color output is updated with the current value of Present\_Value or whether the value of the color output before startup or reset is retained (see 12.X.8). When the color output is not updated at startup or reset, the property In\_Progress shall be set to NOT\_CONTROLLED until the color output is updated with the current value of Present\_Value. Writes to Present\_Value outside of the valid range of values shall cause a Result(-) to be returned with an Error Class of PROPERTY and an Error Code of VALUE\_OUT\_OF\_RANGE.

### 12.X.5 Tracking\_Value

This property, of type BACnetxyColor, indicates the components of the object's actual color output. If the Color\_Command is written with a color operation that affects color, causing Present\_Value to be changed over time, or Present\_Value is written to directly, then Tracking\_Value shall indicate the calculated color for the color output from moment to moment while the fade is in progress.

When the value of In\_Progress is IDLE, Tracking\_Value shall be equal to Present\_Value.

When the value of In\_Progress is FADE\_ACTIVE, Tracking\_Value shall indicate the current calculated value of the fade algorithm. The manner by which the Tracking\_Value is calculated in this situation shall be a local matter.

When the value of In\_Progress is NOT\_CONTROLLED or OTHER, the value of Tracking\_Value shall be a local matter.

### 12.X.6 Color\_Command

This property, of type BACnetColorCommand, is used to request specific behaviors. Color\_Command is written with compound values that specify particular color operations. Devices containing Color objects shall support all BACnetColorOperations shown in Table 12-X2. If Color\_Command 'operation' is written with an enumeration value not shown in Table 12-X2 a Result(-) shall be returned with an Error Class of PROPERTY and an Error Code of VALUE\_OUT\_OF\_RANGE. BACnetColorOperations may include three optional fields. The 'target-color' field, of type BACnetxyColor, represents the target color for FADE\_TO\_COLOR operations in Color objects. The 'fade-time' field is an optional amount of time during which a fade-to-color operation occurs.

The color commands are described in Table 12-X2. The notation to specify the syntax of the color commands is as follows:

<field in angle brackets>	required field of the BACnetColorCommand
<field in angle brackets = value>	required field of the BACnetColorCommand with a specified value
[fields in square brackets]	optional fields of the BACnetColorCommand.

**Table 12-X2. Color Commands Applicable to Color Objects**

Operation	Description
NONE	This operation is used to indicate that no color command has been written to the Color_Command property. This operation shall not be written to the Color_Command property. Attempts to write this operation to the Color_Command property shall cause a Result(-) to be returned with an Error Class of PROPERTY and an Error Code of VALUE_OUT_OF_RANGE.
FADE_TO_CCT RAMP_TO_CCT STEP_UP_CCT STEP_DOWN_CCT	These operations <u>shall not be written</u> to the Color_Command property. Attempts to write these operations to the Color_Command property shall cause a Result(-) to be returned with an Error Class of PROPERTY and an Error Code of VALUE_OUT_OF_RANGE.
FADE_TO_COLOR	Commands Present_Value to fade from the current Tracking_Value to the target-color specified in the command. The fade operation changes the color output from its current value to target-color, over a period of time defined by fade-time. While the fade operation is executing, In_Progress shall be set to FADE_ACTIVE, and Tracking_Value shall be updated to reflect the current progress of the fade.  Syntax: <operation = FADE_TO_COLOR> <target-color> [fade-time]
STOP	Stops any FADE_TO_COLOR command in progress and sets In_Progress to IDLE. If there is no fade command currently executing then this operation is ignored.  syntax: <operation = STOP>

### 12.X.6.1 Halting a Color Command in Progress

Some color commands (e.g. FADE\_TO\_COLOR) are executed over a period of time. While a color command is in progress, it shall be halted under the following conditions:

- (k) A STOP or FADE\_TO\_COLOR, is written to the Color\_Command property, or
- (l) The Present\_Value is written.

When a FADE\_TO\_COLOR command that is currently in progress is halted, the internal fade algorithm is halted.

### 12.X.7 In\_Progress

This property, of type BACnetColorOperationInProgress, shall indicate that there may be processes in the color object that may cause the Tracking\_Value and Present\_Value to differ temporarily. The processes indicated in the property are summarized in Table 12-X3.

**Table 12-X3. BACnetColorOperationInProgress Values**

Value	Description
IDLE	The default value that indicates that no processes are executing which would cause the Present_Value and Tracking_Value to differ.
FADE_ACTIVE	Indicates that a fade color command is currently being executed.
NOT_CONTROLLED	Indicates that on startup or reset the color output has not been updated with the current value of Present_Value.
OTHER	Indicates that the Tracking_Value and Present_Value may differ but none of the other conditions describe the nature of the process.

### 12.X.8 Default\_Color

This property, of type BACnetxyColor, shall indicate the color to be used for the color output when the device is restarted until such time as Present\_Value or Color\_Command are written. As a special case, the BACnetxyColor value (0,0) for Default\_Color shall be interpreted to mean “restore to previous color that was in effect prior to restart.” The Color object is not required to retain the moment-to-moment color value in non-volatile memory. However, if the implementation supports this feature then Default\_Color of (0,0) shall use this retained color instead of a specific default color.

### **12.X.9 Description**

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

### **12.X.10 Default\_Fade\_Time**

This property, of type `Unsigned`, indicates the amount of time in milliseconds over which changes to the color output reflected in the `Tracking_Value` property of the color object shall occur when the `Color_Command` property is written with a fade request that does not include a fade-time value. The range of allowable fade-time values is 100 ms to 86400000 ms (1 day) inclusive.

Values written outside of the allowable range shall cause a `Result(-)` to be returned with an Error Class of `PROPERTY` and an Error Code of `VALUE_OUT_OF_RANGE`.

### **12.X.11 Transition**

This property, of type `BACnetColorTransition`, specifies how a change in the `Present_Value` transitions from the current color to the target color. A transition comes into effect when the `Present_Value` is directly commanded. Writing the color command `FADE_TO_COLOR` shall ignore the `Transition` property.

The transition may be `NONE` or `FADE`. The transition `NONE` causes the `Present_Value` to immediately be set to the target color when `Present_Value` is written. The `FADE` transition allows a smooth transition of the color when the `Present_Value` changes. A `FADE` transition executes a fade operation from the `Tracking_Value` to the target color using the fade time specified in `Default_Fade_Time`.

When a transition results in an operation that may cause the `Tracking_Value` to differ from the `Present_Value`, then the `In_Progress` property shall be set to the value that reflects the operation in progress.

If the `Transition` property is not present, then `Present_Value` shall behave as if `Transition` is `NONE`.

### **12.X.12 Value\_Source**

This property, of type `BACnetValueSource`, indicates the source of the value of the `Present_Value`. The `Value_Source` property and its use in the value source mechanism are described in Clause 19.5.

### **12.X.13 Audit\_Level**

[Note to reviewer: see Addendum bi]

### **12.X.14 Auditable\_Operations**

[Note to reviewer: see Addendum bi]

### **12.X.15 Tags**

This property, of type `BACnetARRAY` of `BACnetNameValue`, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tag and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a `BACnetNameValue` construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the `BACnetNameValue` shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

### **12.X.16 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.17 Profile\_Location**

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile\_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile\_Location value is not provided for a particular object, then the client shall use the Profile\_Location of the Device object, if provided, to find the definition of the Profile\_Name.

#### **12.X.18 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.

### 135-2020ca-3. Add new Color Temperature object type

#### Rationale

The 135-2020 Lighting Output and Binary Lighting Output objects do not address the concept of color temperature in their object models. There are architectural and even commercial lighting applications, where color temperature is a useful and important concept.

These changes extend the BACnet object model to accommodate color temperature.

[Change 3.2]

**correlated color temperature (CCT):** A concept that is defined in *ANSI/ANSI C78.377-2015* to specify the color temperature along a white light curve through the CIE chromaticity space.

...

**clamped:** The act of limiting a value based on another parameter to a restricted range.

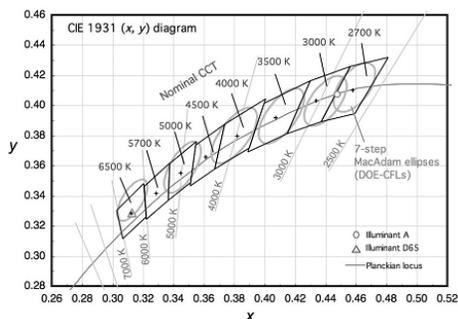
...

[Add 12.Y, Color Temperature Object Type]

## 12.Y Color Temperature Object Type

The Color Temperature object type defines a standardized object whose properties represent the externally visible characteristics of white light color temperature control of a color temperature output and includes dedicated functionality specific to lighting control of color temperature that would otherwise require explicit programming. The color temperature output is analog in nature.

The color temperature of the lighting output can be changed directly to an absolute color temperature by writing to the Present\_Value. The color temperature may also be changed by writing to the Color\_Command property. The color command provides additional color control functionality such as fading.



The concept of closest correlated color temperature (CCT) locus is defined in *ANSI/ANSI C78.377-2015* (see Figure 12-Y). This idea is based on a white light curve through the CIE chromaticity space. The standard defines eight rhomboid regions centered on the white light curve. However, the curve extends to infinity and to zero within the CIE chromaticity space.

The Color Temperature object supports a single color-temperature model based on temperatures as Kelvin.

**Figure 12-Y: White Light Curve**

Writes of Color Temperatures that are outside the range 1000 to 30000 Kelvin shall be considered “values out of range.” When fading/ramping from one color temperature to another the algorithm used to determine the intermediate colors shall be a local matter.

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

**Table 12-Y.** Properties of the Color Temperature Object

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	Unsigned	W
Tracking_Value	Unsigned	R
Color_Command	BACnetColorCommand	W
In_Progress	BACnetColorOperationInProgress	R
Default_Color_Temperature	Unsigned	R
Description	CharacterString	O
Default_Fade_Time	Unsigned	R
Default_Ramp_Rate	Unsigned	R
Default_Step_Increment	Unsigned	R
Min_Pres_Value	Unsigned	O <sup>1</sup>
Max_Pres_Value	Unsigned	O <sup>1</sup>
Transition	BACnetColorTransition	O
Value_Source	BACnetValueSource	O <sup>2,3,4</sup>
Audit_Level	BACnetAuditLevel	O <sup>5</sup>
Auditable_Operations	BACnetAuditOperationFlags	O <sup>5</sup>
Tags	BACnetARRAY[N] of BACnetNameValue	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

<sup>1</sup> If either of these properties are present then both shall be present

<sup>2</sup> This property is required if the object supports the value source mechanism.

<sup>3</sup> This property shall be present only if the object supports the value source mechanism

<sup>4</sup> This property shall be writable as described in Clause 19.5.

<sup>5</sup> This property shall be present only if the device supports audit reporting.

### 12.Y.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.Y.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.Y.3 Object\_Type

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

### 12.Y.4 Present\_Value

This property, of type Unsigned, shall indicate the target Color Temperature in Kelvin for the color temperature output. The range of values shall be 1000 to 30000 Kelvin. When writing to Present\_Value, the algorithm used to determine the closest supported color temperature shall be a local matter.

Present\_Value may also be affected by writes to the Color\_Command property that initiate color commands. These commands may asynchronously affect the color temperature output by establishing a new target for Present\_Value and carrying out the requested operation. Transitioning from one color temperature to another is supported by writing a FADE\_TO\_CCT, RAMP\_TO\_CCT, STEP\_UP\_CCT or STEP\_DOWN\_CCT command to the property Color\_Command. The current color temperature is always indicated in the Tracking\_Value property. If a color command is currently in progress and the Present\_Value is written, the color command shall be halted (see Clause 12.Y.6.1 Halting a Color Command in Progress).

The color temperature output shall be updated whenever the Present\_Value is written or changed as a result of executing a color command.

On a device restart, if Default\_Color\_Temperature property is non-zero, the color temperature output shall be updated to the value of the Default\_Color\_Temperature. If the Default\_Color\_Temperature property is zero, the color temperature output shall be updated to the last value of the Present\_Value property if this value is preserved. If the Default\_Color\_Temperature property is zero and the value of the Present\_Value property not preserved over a restart, the property In\_Progress shall be set to NOT\_CONTROLLED until the color temperature output is updated with the current value of Present\_Value.

If the Min\_Pres\_Value and Max\_Pres\_Value properties are present then writes to Present\_Value shall be further restricted to the range Min\_Pres\_Value through Max\_Pres\_Value. Values greater than 1000 and less than Min\_Pres\_Value shall be clamped to Min\_Pres\_Value. Values greater than Max\_Pres\_Value and less than 30000 shall be clamped to Max\_Pres\_Value.

If the Min\_Pres\_Value and Max\_Pres\_Value properties are not present then writes to Present\_Value outside of the valid range of values shall cause a Result(-) to be returned with an Error Class of PROPERTY and an Error Code of VALUE\_OUT\_OF\_RANGE.

#### **12.Y.5 Tracking\_Value**

This property, of type Unsigned, indicates the device's actual light output color temperature in Kelvin. If the Color\_Command is written with a color operation that affects color temperature, causing Present\_Value to be changed over time, or Present\_Value is written-to directly, then Tracking\_Value shall indicate the calculated color temperature for the color temperature output from moment to moment while the fade is in progress.

When the value of In\_Progress is IDLE, Tracking\_Value shall be equal to Present\_Value.

When the value of In\_Progress is FADE\_ACTIVE or RAMP\_ACTIVE, Tracking\_Value shall indicate the current calculated value of the fade/ramp algorithm. The manner by which the Tracking\_Value is calculated in this situation shall be a local matter.

When the value of In\_Progress is NOT\_CONTROLLED or OTHER, the value of Tracking\_Value shall be a local matter.

#### **12.Y.6 Color\_Command**

This property, of type BACnetColorCommand, is used to request color commands with specific behaviors. Color\_Command is written with compound values that specify particular color operations. Devices containing Color Temperature objects shall support all BACnetColorOperations shown in Table 12-Y2. If Color\_Command is written with an enumeration value not shown in Table 12-Y2 a Result(-) shall be returned with an Error Class of PROPERTY and an Error Code of VALUE\_OUT\_OF\_RANGE. If Color\_Command is written with the value FADE\_TO\_COLOR a Result(-) shall be returned with an Error Class of PROPERTY and an Error Code of VALUE\_OUT\_OF\_RANGE. The 'target-color-temperature' field, of type Unsigned, represents the target color temperature in Kelvin for FADE\_TO\_CCT and RAMP\_TO\_CCT operations in Color Temperature objects. The 'fade-time' field is an optional amount of time during which the fade operation occurs for FADE\_TO\_CCT operations. The 'ramp-rate' field is an optional rate of change per second during ramp operation for RAMP\_TO\_CCT operations. The 'step-increment' field is an optional amount that Present\_Value is incremented or decremented for each STEP\_UP\_CCT or STEP\_DOWN\_CCT operation. If a BACnetColorCommand is sent that includes an optional field that is not explicitly described for that operation in Table 12-Y2, then the field value shall be ignored. Color commands written with a required or optional field, explicitly specified for this command, which are outside of the allowable range of values, shall cause a Result(-) to be returned with an Error Class of PROPERTY and an Error Code of VALUE\_OUT\_OF\_RANGE.

The color commands are described in Table 12-Y2. The notation to specify the syntax of the color commands is as follows:

- <field in angle brackets> required field of the BACnetColorCommand
- <field in angle brackets = value> required field of the BACnetColorCommand with a specified value
- [fields in square brackets] optional fields of the BACnetColorCommand.

**Table 12-Y2. Color Commands Applicable to Color Temperature Objects**

Operation	Description
NONE	This operation is used to indicate that no color command has been written to the Color_Command property. This operation shall not be written to the Color_Command property. Attempts to write this operation to the Color_Command property shall cause a Result(-) to be returned with an Error Class of PROPERTY and an Error Code of VALUE_OUT_OF_RANGE.
FADE_TO_COLOR	This operation <u>shall not be written</u> to the Color_Command property. Attempts to write this operation to the Color_Command property shall cause a Result(-) to be returned with an Error Class of PROPERTY and an Error Code of VALUE_OUT_OF_RANGE.
FADE_TO_CCT	Commands Present_Value to fade from the current Tracking_Value to the target-color-temperature specified in the command. The fade operation changes the output color temperature from its current value to target-color-temperature, over a period of time defined by fade-time. While the fade operation is executing, In_Progress shall be set to FADE_ACTIVE, and Tracking_Value shall be updated to reflect the current progress of the fade. <target-color-temperature> shall be clamped to Min_Pres_Value and Max_Pres_Value as described in 12.Y.4. syntax: <operation = FADE_TO_CCT> <target-color-temperature> [fade-time]
RAMP_TO_CCT	Commands Present_Value to ramp from the current Tracking_Value to the target-color-temperature specified in the command. The ramp operation changes the output color temperature from its current value to target-color-temperature, at a particular Kelvin per second defined by ramp-rate. While the ramp operation is executing, In_Progress shall be set to RAMP_ACTIVE, and Tracking_Value shall be updated to reflect the current progress of the fade. <target-color-temperature> shall be clamped to Min_Pres_Value and Max_Pres_Value as described in 12.Y.4. syntax: <operation = RAMP_TO_CCT> <target-color-temperature> [ramp-rate]
STEP_UP_CCT	Commands Present_Value to a value equal to the Tracking_Value plus the step-increment. The resulting sum shall be clamped to Min_Pres_Value and Max_Pres_Value as described in 12.Y.4. syntax: <operation = STEP_UP_CCT> [step-increment]
STEP_DOWN_CCT	Commands Present_Value to a value equal to the Tracking_Value minus the step-increment. The resulting difference shall be clamped to Min_Pres_Value and Max_Pres_Value as described in 12.Y.4. syntax: <operation = STEP_DOWN_CCT> [step-increment]
STOP	Stops any FADE_TO_CCT or RAMP_TO_CCT command in progress and sets In_Progress to IDLE. If there is no fade command currently executing then this operation is ignored. syntax: <operation = STOP>

#### 12.Y.6.1 Halting a Color Command in Progress

Some color commands (e.g. FADE\_TO\_CCT) are executed over a period of time. While a color command is in progress, it shall be halted under the following conditions:

- (a) A STOP or FADE\_TO\_CCT or RAMP\_TO\_CCT or STEP\_UP\_CCT or STEP\_DOWN\_CCT, is written to the Color\_Command property, or

(b) The Present\_Value is written.

When a FADE\_TO\_CCT or RAMP\_TO\_CCT or STEP\_UP\_CCT or STEP\_DOWN\_CCT command that is currently in progress is halted, the internal fade/ramp algorithm is halted.

### 12.Y.7 In\_Progress

This property, of type BACnetColorOperationInProgress, shall indicate that there may be processes in the Color Temperature object that may cause the Tracking\_Value and Present\_Value to differ temporarily. The processes indicated in the property are summarized in Table 12-Y3.

**Table 12-Y3.** BACnetColorOperationInProgress Values

Value	Description
IDLE	The default value that indicates that no processes are executing which would cause the Present_Value and Tracking_Value to differ.
FADE_ACTIVE	Indicates that a fade-to-cct command is currently being executed.
RAMP_ACTIVE	Indicates that a ramp-to-cct command is currently being executed.
NOT_CONTROLLED	Indicates that on startup or reset the color temperature output has not been updated with the current value of Present_Value.
OTHER	Indicates that the Tracking_Value and Present_Value may differ but none of the other conditions describe the nature of the process.

### 12.Y.8 Default\_Color\_Temperature

This property, of type Unsigned, indicates the color temperature in Kelvin to be used for the color temperature output on a device restart.

Writes to Default\_Color\_Temperature shall be clamped to Min\_Pres\_Value and Max\_Pres\_Value as described in 12.Y.4.

### 12.Y.9 Description

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

### 12.Y.10 Default\_Fade\_Time

This property, of type Unsigned, indicates the amount of time in milliseconds over which changes to the Color Temperature reflected in the Tracking\_Value property shall occur when the Color\_Command property is written with a fade request that does not include a fade-time value. The range of allowable fade-time values is 100 ms to 86400000 ms (1 day) inclusive.

Values written outside of the allowable range shall cause a Result(-) to be returned with an Error Class of PROPERTY and an Error Code of VALUE\_OUT\_OF\_RANGE.

### 12.Y.11 Default\_Ramp\_Rate

This property, of type Unsigned, indicates the rate of change, in Kelvin per second, to the Color Temperature reflected in the Tracking\_Value property that shall occur when the Color\_Command property is written with a ramp request that does not include a ramp-rate value. The range of allowable ramp-rate values is 1 to 30000 inclusive.

Values written outside of the allowable range shall cause a Result(-) to be returned with an Error Class of PROPERTY and an Error Code of VALUE\_OUT\_OF\_RANGE.

### 12.Y.12 Default\_Step\_Increment

This property, of type Unsigned, indicates the amount of change in Kelvin to the Color Temperature reflected in the Tracking\_Value property that shall occur when the Color\_Command property is written with a step-up or step-down request that does not include a step-increment value. The range of allowable step-increment values is 1 to 30000 inclusive.

Values written outside of the allowable range shall cause a Result(-) to be returned with an Error Class of PROPERTY and an Error Code of VALUE\_OUT\_OF\_RANGE.

### **12.Y.13 Min\_Pres\_Value**

This property, of type Unsigned, specifies the minimum value allowed for Present\_Value. Writes to Present\_Value shall be clamped to the range Min\_Pres\_Value to Max\_Pres\_Value. See 12.Y.4. Min\_Pres\_Value shall be greater than or equal to 1000.

### **12.Y.14 Max\_Pres\_Value**

This property, of type Unsigned, specifies the maximum value allowed for Present\_Value. Writes to Present\_Value shall be clamped to the range Min\_Pres\_Value to Max\_Pres\_Value. See 12.Y.4. Max\_Pres\_Value shall be less than or equal to 30000.

### **12.Y.15 Transition**

This property, of type BACnetColorTransition, specifies how a change in the Present\_Value transitions from the current level to the target level. A transition comes into effect when the Present\_Value is directly commanded. Writing the Color command FADE\_TO\_CCT, RAMP\_TO\_CCT, STEP\_UP\_CCT or STEP\_DOWN\_CCT shall ignore the Transition property.

The transition may be NONE or FADE or RAMP. The transition NONE causes the Present\_Value to immediately be set to the target level when Present\_Value is written. The FADE transition allows a smooth transition of the color temperature when the Present\_Value changes. A FADE transition executes a fade operation from the Tracking\_Value to the target level using the fade time specified in Default\_Fade\_Time. The RAMP transition also allows a smooth transition of the color temperature when the Present\_Value changes. A RAMP transition executes a ramp operation from the Tracking\_Value to the target level using the ramp-rate specified in Default\_Ramp\_Rate.

When a transition results in an operation that may cause the Tracking\_Value to differ from the Present\_Value, then the In\_Progress property shall be set to the value that reflects the operation in progress.

If the Transition property is not present, then Present\_Value shall behave as if Transition is NONE.

### **12.Y.16 Value\_Source**

This property, of type BACnetValueSource, indicates the source of the value of the Present\_Value. The Value\_Source property and its use in the value source mechanism are described in Clause 19.5.

### **12.Y.17 Audit\_Level**

[Note to reviewer: see Addendum bi]

### **12.Y.18 Auditable\_Operations**

[Note to reviewer: see Addendum bi]

### **12.Y.19 Tags**

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tag and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

### **12.Y.20 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.Y.21 Profile\_Location**

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile\_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile\_Location value is not provided for a particular object, then the client shall use the Profile\_Location of the Device object, if provided, to find the definition of the Profile\_Name.

#### **12.Y.22 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.

**135-2020ca-4. Add color-reference properties to LO and BLO object types**

**Rationale**

The 135-2020 Lighting Output (LO) and Binary Lighting Output (BLO) objects only affect luminance of lighting output. The newly proposed Color (135-2020ca-2) and Color Temperature (135-2020ca-3) objects provide control of color and color temperature. When a lighting output device needs to control both luminance and color or color temperature, there is no obvious connection between the LO or BLO object and its color-describing companion object. Further, there are use cases for color and color temperature control that require explicit override of color temporarily.

These changes introduce new properties to LO and BLO objects that expose those relationships.

[Change 12.54, Lighting Output Object Type]

...

**Figure 12-14.** Daily Schedule with Blink-Warn Example

*In some cases, lighting outputs may also need to control color or color temperature of the physical device that the output represents. In those instances, the color characteristics of the physical output are controlled by companion Color or Color Temperature objects. The Color\_Reference property defines the companion Color or Color Temperature object. Multiple Lighting Output objects may reference the same Color or Color Temperature object when required.*

*When there are companion color objects, they operate asynchronously with respect to the Lighting Output object. That can create circumstances that need special handling. For example, a physical output is producing a dim light of a specific color. During an emergency or maintenance situation that output might need to be changed to a bright white light instead and then after the situation is resolved, the output should return to its previous state. However, when there is a companion color object that is asynchronously performing a transition such as a FADE operation, upon restoration the color should return to the then-current color, not the color that existed at the start of the emergency. The Override\_Color\_Reference property defines a Color or Color Temperature object to be used as the color during an override situation.*

The object and its properties are summarized in Table 12-64 and described in detail in this clause.

...

[Add to Table 12-64]

...

Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O <sup>7</sup>
Color_Reference	BACnetObjectIdentifier	O <sup>12</sup>
Color_Override	Boolean	O <sup>13</sup>
Override_Color_Reference	BACnetObjectIdentifier	O <sup>13</sup>
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

...

<sup>8</sup> This property shall be writable as described in Clause **Error! Reference source not found.**

<sup>12</sup> This property is required if the object supports color or color override

<sup>13</sup> This property is required if, and shall be present only if, the object supports color override

[Add 12.54.X, 12.54.Y and 12.54.Z]

#### **12.54.X Color\_Reference**

This property, of type BACnetObjectIdentifier, when present, shall specify the object identifier of a Color or Color Temperature object within the same device that controls the color aspects of this Lighting Output. If the object instance portion of the object identifier has the value 4194303, then there is no color companion object associated with this output. In that case the applicable color or color temperature shall be a local matter.

#### **12.54.Y Color\_Override**

This property, of type Boolean, when present, shall specify whether the currently active color or color temperature comes from the Color\_Reference (FALSE) or Override\_Color\_Reference (TRUE). If Color\_Override is present it shall be writable.

#### **12.54.Z Override\_Color\_Reference**

This property, of type BACnetObjectIdentifier, when present, shall specify the object identifier of a Color or Color Temperature object within the same device that controls the color override aspects of this Lighting Output. Color override occurs when the Color\_Override property of the Lighting Output is written with TRUE. In this case the Override\_Color\_Reference points to an object whose color shall be used to control the actual color of the lighting output. While color-overridden, any fade that may be in progress for the lighting output, as well as any fade that may be in progress for the Color\_Reference object, shall continue without interruption, except that the actual color output shall use the override color instead. See Clause 12.54 for a description of color override. Color override shall cease when Color\_Override is written with FALSE.

If the object instance portion of the object identifier has the value 4194303, then there is no color companion override object associated with this output. In that case the applicable color or color temperature shall be a local matter.

[Change 12.55, Binary Lighting Output Object Type]

...

**Figure 12-16.** Daily Schedule with Blink-Warn Example

*In some cases, binary lighting outputs may also need to control color or color temperature of the physical device that the output represents. In those instances, the color characteristics of the physical output are controlled by companion Color or Color Temperature objects. The Color\_Reference property defines the companion Color or Color Temperature object. Multiple Binary Lighting Output objects may reference the same Color or Color Temperature object when required.*

*When there are companion color objects, they operate asynchronously with respect to the Binary Lighting Output object. That can create circumstances that need special handling. For example, a physical output is producing a light of a specific color. During an emergency or maintenance situation that output might need to be changed to a bright white light instead and then after the situation is resolved, the output should return to its previous state. However, when there is a companion color object that is asynchronously performing a transition such as a FADE operation, upon restoration the color should return to the then-current color, not the color that existed at the start of the emergency. The Override\_Color\_Reference property defines a Color or Color Temperature object to be used as the color during an override situation.*

The object and its properties are summarized in Table 12-69 and described in detail in this clause.

...

[Add to Table 12-69]

...

Command_Time_Array	BACnetARRAY[16] of BACnetTimeStamp	O <sup>7</sup>
Color_Reference	BACnetObjectIdentifier	O <sup>12</sup>
Color_Override	Boolean	O <sup>13</sup>
Override_Color_Reference	BACnetObjectIdentifier	O <sup>13</sup>
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

...

<sup>8</sup> This property shall be writable as described in Clause **Error! Reference source not found.**

<sup>12</sup> This property is required if the object supports color or color override

<sup>13</sup> This property is required if, and shall be present only if, the object supports color override

[Add 12.55.X, 12.55.Y and 12.55.Z]

### **12.55.X Color\_Reference**

This property, of type BACnetObjectIdentifier, when present, shall specify the object identifier of a Color or Color Temperature object within the same device that controls the color aspects of this Binary Lighting Output. If the object instance portion of the object identifier has the value 4194303, then there is no color companion object associated with this output. In that case the applicable color or color temperature shall be a local matter.

### **12.55.Y Color\_Override**

This property, of type Boolean, when present, shall specify whether the currently active color or color temperature comes from the Color\_Reference (FALSE) or Override\_Color\_Reference (TRUE). If Color\_Override is present it shall be writable.

### **12.55.Z Override\_Color\_Reference**

This property, of type BACnetObjectIdentifier, when present, shall specify the object identifier of a Color or Color Temperature object within the same device that controls the color override aspects of this Binary Lighting Output. Color override occurs when the Color\_Override property of the Binary Lighting Output is written with TRUE. In this case the Override\_Color\_Reference points to an object whose color shall be used to control the actual color of the lighting output. While color-overridden, any fade that may be in progress for the Color\_Reference object, shall continue without interruption, except that the actual color output shall use the override color instead. See Clause 12.55 for a description of color override. Color override shall cease when Color\_Override is written with FALSE.

If the object instance portion of the object identifier has the value 4194303, then there is no color companion override object associated with this output. In that case the applicable color or color temperature shall be a local matter.

**135-2020ca-5. Add high/low trim to LO object type**

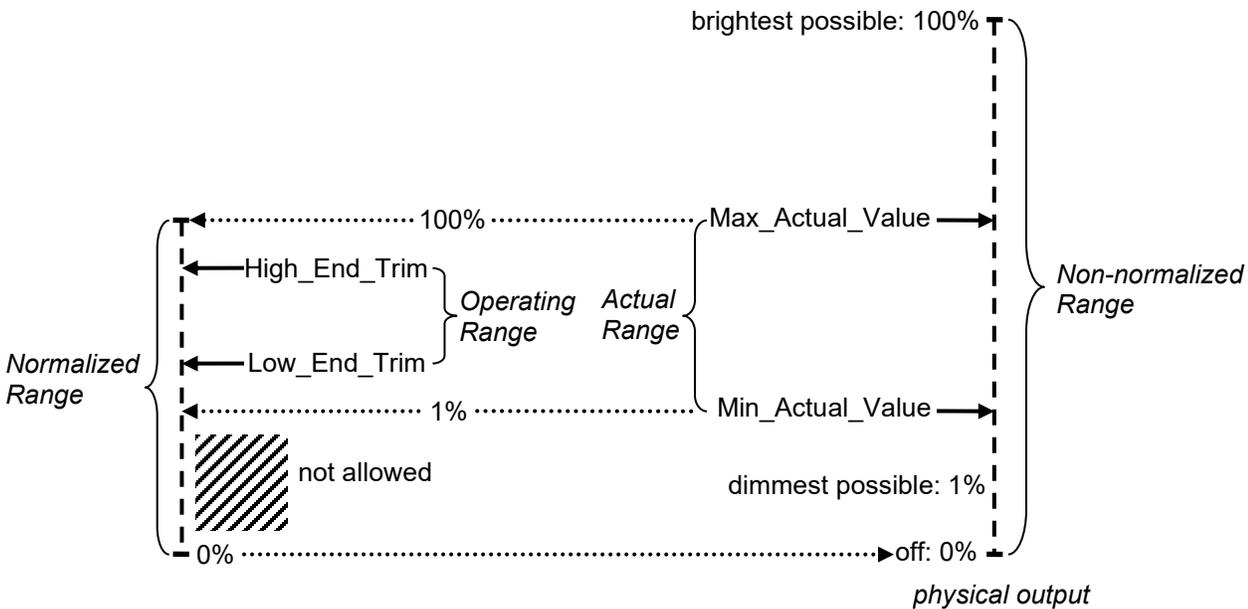
[change Clause 12.54, p. 495]

**12.54 Lighting Output Object Type**

...

The physical output level, or non-normalized range, is specified as the linearized percentage (0..100%) of the possible light output range with 0.0% being off, 1.0% being dimmest, and 100.0% being brightest. The actual range represents the subset of physical output levels defined by `Min_Actual_Value` and `Max_Actual_Value` (or 1.0 to 100.0% if these properties are not present). The normalized range is always 0.0 to 100.0% where 1.0% = bottom of the actual range and 100.0% = top of the actual range. All 0.0% to 100.0% properties of the Lighting Output object shall use the normalized range except for `Min_Actual_Value` and `Max_Actual_Value`. If `Min_Actual_Value` and `Max_Actual_Value` are not present, then the normalized and non-normalized ranges shall be the same.

*The Operating Range is a subset of the Normalized Range, that represents the range of acceptable values for control of the object. The Operating Range is defined by the `High_End_Trim` and `Low_End_Trim` property values. When values are written outside of the Operating Range, the `Tracking_Value` will reflect the actual, clamped normalized light output while the `Present_Value` will reflect the original target value.*



**Figure 12-13.** Normalized Range and Operating Range of the Lighting Output

...

The object and its properties are summarized in Table 12-64 and described in detail in this clause.

**Table 12-64.** Properties of the Lighting Output Object Type

Property Identifier	Property Datatype	Conformance Code
...	...	
<code>Profile_Name</code>	CharacterString	O
<code>High_End_Trim</code>	REAL	O
<code>Low_End_Trim</code>	REAL	O
<code>Trim_Fade_Time</code>	Unsigned	O <sup>14</sup>

...

<sup>14</sup> This property is required if and only if either the `High_End_Trim` or `Low_End_Trim` properties are present.

[change Clause 12.54.5, p. 498]

### 12.54.5 Tracking\_Value

This property, of type REAL, indicates the value at which the physical lighting output is being controlled within the normalized range at all times. *If the High\_End\_Trim or Low\_End\_Trim properties are present, the Tracking\_Value shall be clamped inclusively within the Operating Range.*

When the value of In\_Progress is IDLE, Tracking\_Value shall be equal to Present\_Value.

When the value of In\_Progress is RAMP\_ACTIVE or FADE\_ACTIVE, Tracking\_Value shall indicate the current calculated value of the ramp or fade algorithm. The manner by which the Tracking\_Value is calculated in this situation shall be a local matter.

*When the value of In\_Progress is TRIM\_ACTIVE, Tracking\_Value shall indicate the clamped value is equal to either the High\_End\_Trim or Low\_End\_Trim properties.*

When the value of In\_Progress is NOT\_CONTROLLED or OTHER, the value of Tracking\_Value shall be a local matter.

[change Table 12-68, p. 503]

### 12.54.7 In\_Progress

...

**Table 12-68. BACnetLightingInProgress Values**

Value	Description
IDLE	The default value that indicates that no <del>processes are executing</del> condition is in effect, which would cause the Present_Value and Tracking_Value to differ.
RAMP_ACTIVE	Indicates that a ramp lighting command is currently being executed.
FADE_ACTIVE	Indicates that a fade lighting command is currently being executed.
TRIM_ACTIVE	Indicates that the Present_Value is currently outside of the Operating Range (lower than Low_End_Trim or higher than High_End_Trim) regardless of ramp or fade status, which would cause the Present_Value and Tracking_Value to differ.
NOT_CONTROLLED	Indicates that on startup or reset the physical output has not been updated with the current value of Present_Value.
OTHER	Indicates that the Tracking_Value and Present_Value may differ but none of the other conditions describe the nature of the process.

[add new Clauses 12.54.Yn, p. 507]

### 12.54.Y1 High\_End\_Trim

This property, of type REAL, specifies a physical lighting output value within the normalized range that acts as an upper-end limit for the Tracking\_Value. If the Present\_Value is written with a value above the High\_End\_Trim, the Tracking\_Value shall be clamped to High\_End\_Trim and the property In\_progress shall be set to Trim\_Active. If the High\_End\_Trim value is less than the Low\_End\_Trim value, the Reliability property shall indicate CONFIGURATION\_ERROR.

High\_End\_Trim shall always be a positive number in the normalized range 1.0% to 100.0%. This property does not change the normalized range.

When High\_End\_Trim is being used and the value of the In\_Progress property is TRIM\_ACTIVE, the method for adjusting the calculation of fade-time shall be a local matter.

If Present\_Value is commanded at priority 1 or 2, High\_End\_Trim shall not be applied and the Tracking\_Value shall not be clamped.

### 12.54.Y2 Low\_End\_Trim

This property, of type REAL, specifies a physical lighting output value within the normalized range that acts as a lower-end limit for the Tracking\_Value. If the Present\_Value is written with a value above 0.0% but below the Low\_End\_Trim, the

Tracking\_Value shall be clamped to Low\_End\_Trim and the property In\_progress shall be set to Trim\_Active. If the High\_End\_Trim value is less than the Low\_End\_Trim value, the Reliability property shall indicate CONFIGURATION\_ERROR.

Low\_End\_Trim shall always be a positive number in the normalized range 1.0% to 100.0%. This property does not change the normalized range.

When Low\_End\_Trim is being used and the value of the In\_Progress property is TRIM\_ACTIVE, the method for adjusting the calculation of fade-time shall be a local matter.

If Present\_Value is commanded at priority 1 or 2, Low\_End\_Trim shall not be applied and the Tracking\_Value shall not be clamped.

#### **12.54.Y3 Trim\_Fade\_Time**

This property, of type Unsigned, indicates the amount of time in milliseconds over which changes to the normalized value reflected in the Tracking\_Value property of the lighting output shall occur when the High\_End\_Trim or Low\_End\_Trim properties are changed such that the current value of the Present\_Value property falls outside of the Operating Range. The range of allowable fade-time values is 0 ms to 86400000 ms (1 day) inclusive.

Values written outside of the allowable range shall cause a Result(-) to be returned with an Error Class of PROPERTY and an Error Code of VALUE\_OUT\_OF\_RANGE.

## 135-2020ca-6. Aggregated changes to Clause 21 and 25

### [Clause 21 changes]

```
BACnetChannelValue ::= CHOICE {  
    null                NULL,  
    real                REAL,  
    enumerated          ENUMERATED,  
    unsigned            Unsigned,  
    boolean             BOOLEAN,  
    integer             INTEGER,  
    double              Double,  
    time                Time,  
    characterstring     CharacterString,  
    octetstring         OCTET STRING,  
    bitstring           BIT STRING,  
    date                Date,  
    objectidentifier    BACnetObjectIdentifier,  
    lighting-command    [0] BACnetLightingCommand,  
    xycolor             [1] BACnetxyColor  
    color-command     [2] BACnetColorCommand  
}
```

```
BACnetColorCommand ::= SEQUENCE {  
    operation            [0] BACnetColorOperation,  
    target-color         [1] BACnetxyColor OPTIONAL,  
    target-color-temperature [2] Unsigned OPTIONAL,  
    fade-time           [3] Unsigned (100.. 86400000) OPTIONAL,  
    ramp-rate           [4] Unsigned (1..30000) OPTIONAL,  
    step-increment      [5] Unsigned (1..30000) OPTIONAL  
}
```

```
BACnetColorOperationInProgress ::= ENUMERATED {  
    idle                (0),  
    fade-active         (1),  
    ramp-active         (2),  
    not-controlled      (3),  
    other                (4)  
}
```

```
BACnetColorTransition ::= ENUMERATED {  
    none                (0),  
    fade                (1),  
    ramp                (2)  
}
```

```
BACnetxyColor::= SEQUENCE {  
    x-coordinate        REAL,  --(0.0 to 1.0)  
    y-coordinate        REAL  --(0.0 to 1.0)  
}
```

```
BACnetLightingInProgress ::= ENUMERATED {  
    idle (0),  
    fade-active (1),  
    ramp-active (2),  
    not-controlled (3),  
    other (4)  
    other(4),
```

*trim-active (5)*

}

```
BACnetPriorityValue ::= CHOICE {
    null           NULL,
    real           REAL,
    enumerated     ENUMERATED,
    unsigned       Unsigned,
    boolean        BOOLEAN,
    signed         INTEGER,
    double         Double,
    time           Time,
    characterString CharacterString,
    octetString    OCTET STRING,
    bitString      BIT STRING,
    date           Date,
    objectid       BACnetObjectIdentifier,
    constructedValue [0] ABSTRACT-SYNTAX.&Type,
    datetime       [1] BACnetDateTime,
    xycolor       [2] BACnetxyColor
}
```

```
BACnetColorOperation ::= ENUMERATED {
    none           (0),
    fade-to-color  (1),
    fade-to-cct    (2),
    ramp-to-cct    (3),
    step-up-cct    (4),
    step-down-cct  (5),
    stop           (6)
}
```

```
BACnetObjectType ::= ENUMERATED { -- see below for numerical order
```

```
    ...
    characterstring-value (40),
    color                  (63),
    color-temperature     (64),
    command               (7),
    ...
-- numerical order reference
    ...
see audit reporter (62)
-- see audit-reporter   (62),
-- color                 (63),
-- color-temperature    (64)
    ...
}
```

```
BACnetObjectTypesSupported ::= BIT STRING {
```

```
    ...
audit reporter (62)
    audit-reporter     (62),
    color               (63),
    color-temperature (64)
}
```

[Insert into production **BACnetPropertyIdentifier** in **Clause 21**, p. 845]

```
BACnetPropertyIdentifier ::= ENUMERATED { -- see below for numerical order
```

...	
client-cov-increment	(127),
color-override	(4194328),
color-reference	(4194329),
default-color	(4194330),
default-color-temperature	(4194331),
...	
output-units	(82),
override-color-reference	(4194332)
...	
-- numerical order reference	
...	
-- see represents	(491),
-- see color-override	(4194328),
-- see color-reference	(4194329),
-- see default-color	(4194330),
-- see default-color-temperature	(4194331),
-- see override-color-reference	(4194332)

**[Add to Clause 25]**

*CIE 13.3-1995, Commission Internationale de l'Eclairage, Method of Measuring and Specifying Colour Rendering of Light Sources.*

*CIE 15: 2004, Commission Internationale de l'Eclairage, Colorimetry, 3rd edition.)*

*ANSI\_ANSLG C78.377-2015, Specifications for the Chromaticity of Solid State Lighting Products.*

135-2020ca-7. BIBB Changes to Support Additional Object Types

[Change K.1.16]

K.1.16 BIBB - Data Sharing-Advanced View-A (DS-AV-A)

...

**Table K-3. Presentation Requirements by Datatype**

NULL	Present NULL values. The format is unrestricted as long as NULL is distinguishable from other values.
BOOLEAN	Present all valid values. The format is unrestricted as long as each valid value is distinguishable.
Unsigned, Unsigned8, Unsigned16, Unsigned32	Present the complete value range, unless specifically restricted by the standard for the property being displayed. The minimum displayable range for Unsigned by DS-AV-A devices is the same as Unsigned32 with the exception of array indexes, which shall have a minimum displayable range of Unsigned16. In addition, any Unsigned property whose value is also used as an array index, such as a Multi-state object's Present Value, shall have a minimum displayable range of Unsigned16.
INTEGER	Present the complete value range, unless specifically restricted by the standard for the property being displayed. The minimum displayable range for INTEGER shall be -2147483648...2147483647.
REAL, Double	Present the complete value range, including special values such as +-INF and NaN, unless specifically restricted by the standard for the property being displayed.
OCTET STRING	Present octet strings up to a length of 8 octets. The actual presentation of the values is unrestricted (text, numeric, iconic, etc.), as long as the individual values are distinguishable.
Character String	Present strings up to the length specified in Table K-4, encoded in any of the character sets supported by the A device.
BIT STRING	<i>Present the complete range of standard values defined for all standard bit string types for the Protocol_Revision claimed by the A device. (Note that in Protocol_Revision 24, BACnetObjectTypesSupported exceeded the historical minimum display requirement of 64 bits.) For properties where the length of the bit string is not directly defined by the standard (such as the Present_Value of the BitString Value object), present bit strings up to a length of 64 bits. Present bit strings up to a length of 64 bits. Present the complete range of standard values defined for all standard bit string types for the Protocol_Revision claimed by the A device.</i> —The actual presentation of the values is unrestricted (text, numeric, iconic, etc.) as long as the individual values are distinguishable.
Enumerated	Present the standard values defined for all standard enumerated types for the Protocol_Revision claimed by the A device. The actual presentation of the values is unrestricted (text, numeric, iconic, etc.) as long as the individual values are distinguishable.
Date	Present all valid dates, including values that contain unspecified octets (X'FF') or special date values (such as 'even days') which are defined for the Protocol_Revision claimed by the A device. Where the month, day and year fields all contain singular specified values, the content of the DayOfWeek field may be ignored. The format is unrestricted as long as each valid value is uniquely presented.
Time	Present all valid times, including values that contain unspecified octets (X'FF'). The format is unrestricted as long as each valid value is uniquely presented.
BACnetObjectIdentifier	Present all valid values. The format is unrestricted as long as each valid value is distinguishable. It is acceptable that BACnetObjectIdentifier values be replaced with unique object identification values such as the object's name, where available.

...

[Add a new entry to **History of Revisions**, p. 1411]

### HISTORY OF REVISIONS

...	...	...
1	24	<p>Addendum ca to ANSI/ASHRAE Standard 135-2020 Approved by ASHRAE on MONTH DAY, 20XX; and by the American National Standards Institute on MONTH DAY, 20XX.</p> <ol style="list-style-type: none"><li>1. Add new Color object type.</li><li>2. Add new Color Temperature object type.</li><li>3. Add color-reference properties to LO and BLO object types.</li><li>4. Add high/low trim to LO object type.</li><li>5. BIBB Changes to Support Additional Object Types.</li></ol>

## **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.

**ASHRAE · 180 Technology Parkway NW · Peachtree Corners, GA 30092 · [www.ashrae.org](http://www.ashrae.org)**

### **About ASHRAE**

Founded in 1894, ASHRAE is a global professional society committed to serve humanity by advancing the arts and sciences of heating, ventilation, air conditioning, refrigeration, and their allied fields.

As an industry leader in research, standards writing, publishing, certification, and continuing education, ASHRAE and its members are dedicated to promoting a healthy and sustainable built environment for all, through strategic partnerships with organizations in the HVAC&R community and across related industries.

To stay current with this and other ASHRAE Standards and Guidelines, visit [www.ashrae.org/standards](http://www.ashrae.org/standards), and connect on LinkedIn, Facebook, Twitter, and YouTube.

### **Visit the ASHRAE Bookstore**

ASHRAE offers its Standards and Guidelines in print, as immediately downloadable PDFs, 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](http://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](http://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.**