SLOVENSKI STANDARD
SIST EN ISO 16484-5:2018/oprA1:2020

01-februar-2020

Sistemi za avtomatizacijo stavb in regulacijo - 5. del: Protokol za izmenjavo podatkov - Dopolnilo A1 (ISO 16484-5:2017/FDAMD 1:2019)

Building automation and control systems (BACS) - Part 5: Data communication protocol - Amendment 1 (ISO 16484-5:2017/FDAMD 1:2019)

Systèmes d'automatisation et de gestion technique du bâtiment - Partie 5: Protocole de communication de données - Amendement 1 (ISO 16484-5:2017/FDAMD 1:2019)

Ta slovenski standard je istoveten z: **EN ISO 16484-5:2017/prA1**

ICS:

35.240.67	Uporabniške rešitve IT v gradbeništvu	IT applications in building and construction industry
97.120	Avtomatske krmilne naprave za dom	Automatic controls for household use

SIST EN ISO 16484-5:2018/oprA1:2020 **en,fr,de**

FINAL
DRAFT

AMENDMENT

ISO
16484-5:2017
FDAM 1

ISO/TC **205**

Secretariat: ANSI

Voting begins on:
2019-12-24

Voting terminates on:
2020-03-17

Building automation and control systems (BACS) —

Part 5:
Data communication protocol

AMENDMENT 1

Systèmes d'automatisation et de gestion technique du bâtiment —

Partie 5: Protocole de communication de données

AMENDEMENT 1

ISO/CEN PARALLEL PROCESSING

RECIPIENTS OF THIS DRAFT ARE INVITED TO SUBMIT, WITH THEIR COMMENTS, NOTIFICATION OF ANY RELEVANT PATENT RIGHTS OF WHICH THEY ARE AWARE AND TO PROVIDE SUPPORTING DOCUMENTATION.

IN ADDITION TO THEIR EVALUATION AS BEING ACCEPTABLE FOR INDUSTRIAL, TECHNOLOGICAL, COMMERCIAL AND USER PURPOSES, DRAFT INTERNATIONAL STANDARDS MAY ON OCCASION HAVE TO BE CONSIDERED IN THE LIGHT OF THEIR POTENTIAL TO BECOME STANDARDS TO WHICH REFERENCE MAY BE MADE IN NATIONAL REGULATIONS.

Reference number
ISO 16484-5:2017/FDAM 1:2020(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. International Standards are drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 205, *Building environmental design,* in collaboration with the European Committee for Standardization (CEN) Technical Committee CEN/TC 247, *Building Automation, Controls and Building Management*, in accordance with the Agreement on technical cooperation between ISO and CEN (Vienna Agreement).

A list of all parts in the ISO 16484 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

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

FOREWORD

The purpose of this addendum is to add several independent substantive changes to the BACnet standard. The changes are summarized below.

135-2016bd-1. Add Staging Object Type.
135-2016be-1. Add Lighting BIBBS and Device Profiles.
135-2016bi -1. Add Audit Reporting.
135-2016bi -2. Change DeviceCommunicaitonControlService for Audit Reporting.
135-2016bi -3. Modify Logging Objects to Allow for Extremely Large Logs.
135-2016bk -1. Expand the reserved range of BACnetPropertyIdentifier.
135-2016bl - 1. Clarify Result(-) response for failed WritePropertyMultiple requests.
135-2016bl - 2. Clarify ReadPropertyMultiple response on OPTIONAL when empty.
135-2016bl - 3. Clarify Out_Of_Service.
135-2016bm-1. Reduce allowed range for Usage Timeout.
135-2016bm-2. Specify design choices for MS/TP devices.
125-2016bm-3. Handle unwanted MS/TP frames in IDLE state.
135-2016bn-1. Make SCHED BIBBs consistent on supported datatypes, and add BOOLEAN.
135-2016bn-2. Clarify COV and COVP related BIBBs.
135-2016bn-3. Clock is required for support of AE-ACK-A.
135-2016bp-1. Make rules for POST consistent with rules for PUT.
135-2016bp-2. Make 'type' consistent at all levels and introduce 'effectiveType'.
135-2016bp-3. Fully specify the behavior of "includes".
135-2016bp-4. Remove the path syntax from the 'select' query parameter.
135-2016bp-5. Resolve conflicting statements about configuring external authorization servers.
135-2016bp-6. Remove incorrect table for callback formats.
135-2016bp-7. Allow plain text POSTs for primitive data.
135-2016bp-8. Allow extended error numbers.
135-2016bp-9. Add new error numbers.
135-2016bp-10. Add formal definition for JSON equivalent to XML's <CSML>.
135-2016bp-11. Specify 'name' safety check for setting data.
135-2016bp-12. Specify how to evaluate relative paths for collections of links.
135-2016bp-13. Allow proprietary categories for the 'metadata' query.
135-2016bq-1. Fix the Absentee_Limit property of the Access Credential object type.
135-2016bq-2. Ensure that the denied or granted access event is generated last.

In the following document, language to be added to existing clauses of EN ISO 16484-5 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.

The use of placeholders like X, Y, Z, X1, X2, etc., should not be interpreted as literal values of the final standard. These placeholders will be assigned actual numbers/letters only with incorporation of this addendum into the standard for republication.

Building automation and control systems (BACS) —

Part 5:
Data communication protocol

AMENDMENT 1

135-2016*bd*-1 Add a Staging Object Type

Rationale

The Staging object type provides a way for BACnet devices to map analog values onto multiple Binary Value, Binary Output, or Binary Lighting Output objects.

A common use case is in lighting applications, where a level, identified by a numeric value, sets the appropriate values of multiple binary outputs (on or off).

Support of this new object type is excluded from all data sharing BIBBs for life safety and access control.

[Insert new **Clause 12.X**]

12.X Staging Object Type

The Staging object type defines a standardized object whose properties represent the externally visible characteristics of a staged value. A "Staging" maps a numeric value onto multiple discrete ranges that define individual “stages” (**N**$_{\textbf{stages}}$). Each Staging object is associated with a collection of references to binary valued objects (**N**$_{\textbf{references}}$). Each Staging object may therefore control Binary Output, Binary Value, or Binary Lighting Output objects. Every stage specifies an arbitrary combination of ACTIVE/INACTIVE values to be written to these referenced objects. Stages are defined by a limit, a deadband, and the collection of values for the referenced objects.

Figure 12-X shows a typical Staging object application with four stages (**N**$_{\textbf{stages}}$ =4) and two referenced binary objects (**N**$_{\textbf{references}}$ = 2).

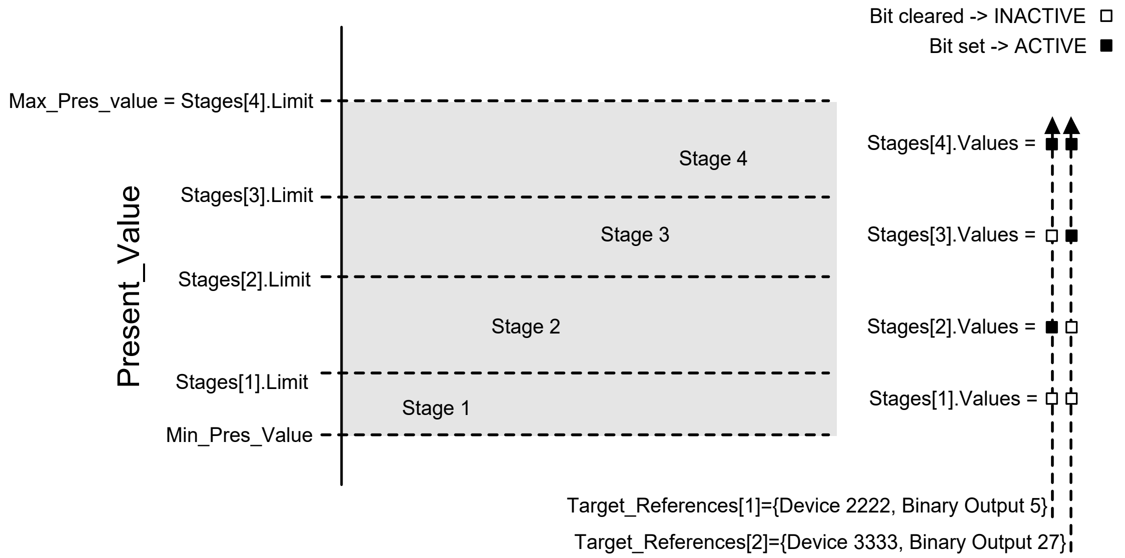


Figure 12-X. Typical Staging Application ($\mathbf{N}_{\text{stages}}$ =4, $\mathbf{N}_{\text{references}}$ = 2)

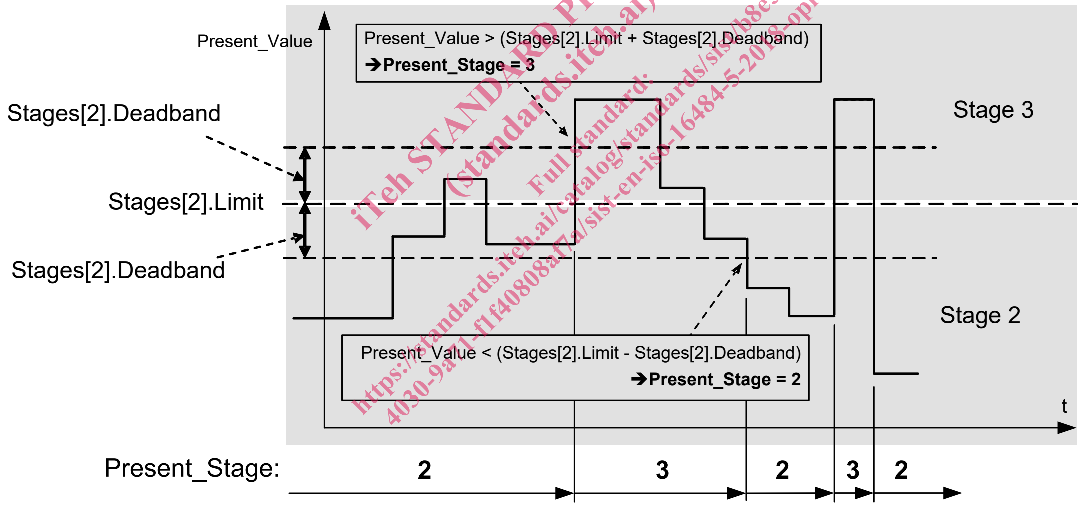
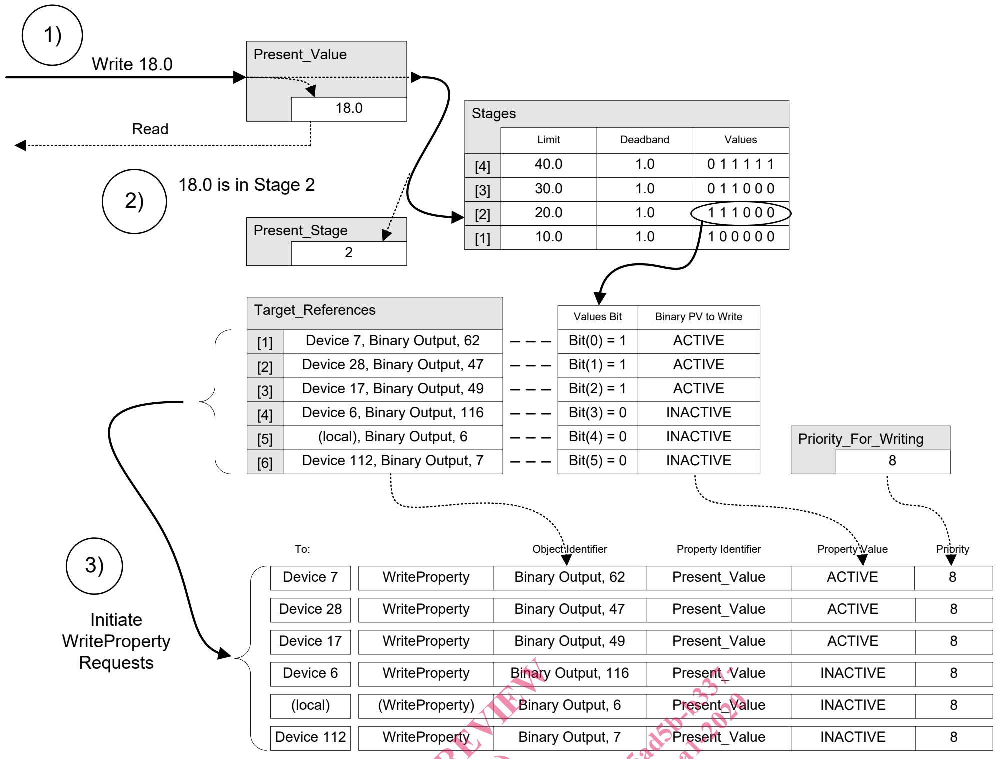


Figure 12-X2. Stage Limits incorporate hysteresis through the use of a Deadband around each Limit

Stages are defined by limits with a symmetrical deadband. A deadband greater than zero is used to prevent unwanted oscillation when the Present_Value is close to a limit. As the Present_Value increases, if it rises above the limit for a stage plus the deadband for that stage, the Present_Stage transitions to that stage+1. Similarly, as the Present_Value decreases, it must fall below the limit for a stage minus the deadband for that stage before Present_Stage transitions to that stage. The deadband is allowed to be zero (0.0).



Stages	Limit	Deadband	Values
[4]	40.0	1.0	0 1 1 1 1 1
[3]	30.0	1.0	0 1 1 0 0 0
[2]	20.0	1.0	1 1 1 0 0 0
[1]	10.0	1.0	1 0 0 0 0 0

Target_References		Values Bit	Binary PV to Write
[1]	Device 7, Binary Output, 62	Bit(0) = 1	ACTIVE
[2]	Device 28, Binary Output, 47	Bit(1) = 1	ACTIVE
[3]	Device 17, Binary Output, 49	Bit(2) = 1	ACTIVE
[4]	Device 6, Binary Output, 116	Bit(3) = 0	INACTIVE
[5]	(local), Binary Output, 6	Bit(4) = 0	INACTIVE
[6]	Device 112, Binary Output, 7	Bit(5) = 0	INACTIVE

To:		Object Identifier	Property Identifier	Property Value	Priority
Device 7	WriteProperty	Binary Output, 62	Present_Value	ACTIVE	8
Device 28	WriteProperty	Binary Output, 47	Present_Value	ACTIVE	8
Device 17	WriteProperty	Binary Output, 49	Present_Value	ACTIVE	8
Device 6	WriteProperty	Binary Output, 116	Present_Value	INACTIVE	8
(local)	(WriteProperty)	Binary Output, 6	Present_Value	INACTIVE	8
Device 112	WriteProperty	Binary Output, 7	Present_Value	INACTIVE	8

Figure 12-X3. Pipeline of operations when Present_Value is written

Staging objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Staging objects that support intrinsic reporting shall apply the NONE event algorithm.

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

Table 12-X. Properties of the Staging Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	REAL	W
Present_Stage	Unsigned	R
Stages	BACnetARRAY[N] of BACnetStageLimitValue	R[1]
Stage_Names	BACnetARRAY[N] of CharacterString	O[1]
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	R
Out_Of_Service	BOOLEAN	R
Description	CharacterString	O
Units	BACnetEngineeringUnits	R
Target_References	BACnetARRAY[N] of BACnetDeviceObjectReference	R[2]
Priority_For_Writing	Unsigned(1..16)	R
Default_Present_Value	REAL	O
Min_Pres_Value	REAL	R
Max_Pres_Value	REAL	R
COV_Increment	REAL	O[3]
Notification_Class	Unsigned	O[4,5]
Event_Enable	BACnetEventTransitionBits	O[4,5]
Acked_Transitions	BACnetEventTransitionBits	O[4,5]
Notify_Type	BACnetNotifyType	O[4,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[5,6]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Event_Detection_Enable	BOOLEAN	O[4,5]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Value_Source	BACnetValueSource	O[7,8,9]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] The array size of this property is $\mathbf{N}_{\mathbf{stages}}$.
[2] The array size of this property is $\mathbf{N}_{\mathbf{references}}$.
[3] This property is required if the object supports COV reporting.
[4] These properties are required if the object supports intrinsic reporting.
[5] These properties shall be present only if the object supports intrinsic reporting.
[6] This property, if present, is required to be read-only.
[7] This property is required if the object supports the value source mechanism.
[8] This property shall be present only if the object supports the value source mechanism.
[9] This property shall be writable as described in Clause 19.5.

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 STAGING.

12.X .4 Present_Value

This property, of type REAL, indicates the current value, in engineering units, of the Staging object. If Present_Value is written with a value less than Min_Pres_Value, then it shall be clamped to Min_Pres_Value. If Present_Value is written with a value greater than Max_Pres_Value, then it shall be clamped to Max_Pres_Value.

Whenever Present_Value is changed, the new value shall be compared with the 'Limit' values for the entire Stages array using the following algorithm:

```
ops = current Present_Stage
npv = new Present_Value

//check if value should remain in the current stage
If (ops != 0) then
   upperBound = Stages[ops].Limit + Stages[ops].Deadband
   if (ops > 1) then
      lowerBound = Stages[ops-1].Limit - Stages[ops-1].Deadband
   else
      lowerBound = Min_Pres_Value
   endif
   if (npv <= upperBound AND npv >= lowerBound) then
      Present_Value = npv
      exit algorithm //no change to current stage, stop algorithm
   endif
endif

// calculate the new stage
Present_Stage = Nstages
for (i=1 to Nstages-1, step +1)
   if npv <= (Stages[i].Limit) then
      Present_Stage = i
      exit for    //found the correct stage, stop iteration
   endif
next i

Present_Value = npv
```

Figure 12-X4. Pseudocode Algorithm for Evaluating Present_Value and Present_Stage

12.X.4.1 Writing to Referenced Objects

Changes to Present_Stage shall cause a write of ACTIVE or INACTIVE to the corresponding object's Present_Value for each Target_References array element. For each bit (Index = 0 to $\mathbf{N}_{\mathbf{references}}$ -1) in the Stages[Present_Stage].Values bitstring, if the bit is set (1), an ACTIVE value shall be written, or if clear (0), then an INACTIVE value shall be written to the Target_References[Index +1] object's Present_Value.

Writes to Present_Value that subsequently trigger writing to referenced objects due to reevaluation of Present_Stage, are not expected to wait until the reference writes occur before returning a Result(+) or Result(-) for the write to Present_Value. Subsequently if any write to a referenced object fails, Reliability shall be changed to COMMUNICATION_FAILURE. The COMMUNICATION_FAILURE shall remain in effect until all reference writes have been completed successfully. How a particular implementation handles other failures during writing to referenced objects shall be a local matter except that Reliability shall indicate a value other than NO_FAULT_DETECTED.

The order of evaluation of references and any referenced object write delay shall be a local matter.

12.X.5 Present_Stage

This property, of type Unsigned, shall indicate the array index (1 to $\mathbf{N}_{\mathbf{stages}}$) that corresponds to the current active stage or 0 meaning that the Present_Stage has not yet been initialized. Upon device restart, or when the Stages property is written to any of its elements, or the size of the Stages array changes, Present_Stage shall be set to 0 temporarily and then Present_Value shall be reevaluated as described in Clause 12.X.4.1.

Attempts to read Present_Stage when it is internally set to 0 shall return a Result(-) with an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_NOT_INITIALIZED.

12.X.6 Stages

This property, of type BACnetARRAY[N] of BACnetStageLimitValue, is an array representing the stages by limit, desired present values for the objects referenced by Target_References, and deadband. The size of the array is $\mathbf{N}_{\mathbf{stages}}$, where $\mathbf{N}_{\mathbf{stages}}$ shall be greater than 1. BACnetStageLimitValue is a tuple consisting of the following fields:

Limit	REAL
Values	BIT STRING
Deadband	REAL

The 'Limit' values for all elements shall be strictly ascending, such that:

```
for Index=1 to Nstages-1
{
    lowerbound = (Stages[Index].Limit + Stages[Index].Deadband)
    upperbound = (Stages[Index+1].Limit - Stages[Index+1].Deadband)
    lowerbound <= upperbound
}
```

If any of the stages do not meet this criterion, then the Reliability property shall have a value of CONFIGURATION_ERROR.

The bits in 'Values' correspond to references in the Target_References array, such that bit (Index = 0 to $\mathbf{N}_{\mathbf{references}}$ - 1) corresponds to Target_References[Index + 1], etc. The length of the 'Values' bitstring shall be $\mathbf{N}_{\mathbf{references}}$ bits.

The 'Deadband' shall be zero or positive. A negative value for 'Deadband' shall cause the value of Reliability to be CONFIGURATION_ERROR.

If the size of the Stages array is increased, then the new array elements, for which no initial value is provided, shall be initialized to contain 'Limit' = 0.0, 'Deadband' = 0.0, and 'Values' = {0…0}, and the value of Reliability shall be set to CONFIGURATION_ERROR.

If the size of the Stages array is less than 2, then the Reliability property shall have a value of CONFIGURATION_ERROR.

If Reliability has the value CONFIGURATION_ERROR, then Present_Value shall be set to Min_Pres_Value and Present_Stage to 1.

If Stages[$\mathbf{N}_{\mathbf{stages}}$].Limit becomes smaller than Present_Value, then Present_Value shall be set to Stages[$\mathbf{N}_{\mathbf{stages}}$].Limit.

If the Stages property is written, the value of Present_Stage shall be reevaluated and corresponding writes of new values to Target_References’ objects shall be triggered.

If the size of this array is changed, the size of the Stage_Names array shall also be changed to the same size.

12.X.7 Stage_Names

This property, of type BACnetARRAY[N] of CharacterString, is an array representing a name for each stage. The number of array elements in Stage_Names shall be the same as the number of array elements in the Stages property. Stage_Names[1] shall correspond to the name for Present_Stage=1. Stage_Names[2] shall correspond to the name for Present_Stage=2, etc.

If the size of this array is changed, the size of the Stages array shall also be changed to the same size. If the size of Stage_Names is increased, then it shall be a local matter what the uninitialized array elements contain.

12.X.8 Status_Flags

This property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of a Staging 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 Always 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.9 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.10 Reliability

The Reliability property, of type BACnetReliability, provides an indication that the properties of the Staging object are in a consistent state and that Target_References are being reliably written. See Clauses 12.X.4.1, 12.X.6, and 12.X.17.

Table 12-X2 summarizes scenarios when Reliability has a value other than NO_FAULT_DETECTED.

Table 12-X2. Reliability Scenarios

Scenario	Reliability Value	See Clause
Communication error when writing	COMMUNICATION_FAILURE	12.X.4.1
Non-communication-related failure when writing	Local Matter	12.X.4.1
'Limit' values out of order	CONFIGURATION_ERROR	12.X.6
'Deadband' is negative	CONFIGURATION_ERROR	12.X.6
Stages array size is increased	CONFIGURATION_ERROR	12.X.6
Min_Pres_Value is >= (Stages[1].Limit - Stages[1].Deadband)	CONFIGURATION_ERROR	12.X.17

12.X.11 Out_Of_Service

The Out_Of_Service property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value property is controllable by software local to the BACnet device.

When Out_Of_Service is TRUE:

a) changes to the Present_Value property are decoupled from the Target_References. This means that the objects referenced by Target_References shall not be updated;