BACnet *Errata* ANSI/ASHRAE STANDARD 135.1-2019 A Data Communication Protocol for Building Automation and Control Networks

June 18, 2021

This document lists *errata additions* to ANSI/ASHRAE Standard 135.1-2019 as of the above date. Each entry is cited first by clause, then page number, except where an erratum covers more than one clause. The back page marking identifying the electronic publication of Standard 135.1-2019 is "Product code: D-86437 9/19".

Changes are indicated by using strikeout for text to be removed and *italics* for text to be added, unless noted otherwise. Grey highlighting is used for marking small changes.

1) Clause 12.1.3.9 – Frame Type explicitly stated

12.1.3.9 Verify Tusage_timeout w/ Serial Analyzer

•••

- Test Steps:
- 1. MAKE (Power on both devices.)
- 2. WAIT (several seconds)
- 3. VERIFY (Has Token passing been established between the devices?)
- 4. MAKE (Power off the other master device, but not the IUT.)
- 5. WAIT (10 seconds)
- 6. MAKE (Stop the data capture.)
- 7. CHECK (Did the IUT send a type 0 Token frame to the other master, and, when the other master did not use the Token (because it was powered off), did the IUT follow the type 0 Token frame with one type 0 Token frame (Token retry) followed by a series of type 1 Poll For Master frames?)
- 8. CHECK (Is the time difference between the last octet of the type 0 *Token* frame sent by the IUT and the first octet of the immediately following type 1 *Poll For Master* frame transmitted by the IUT greater than 20 millseconds T_{neg_err} and less than 100 millseconds + T_{pos_err} ?)
- 9. CHECK (Is the time gap (last character to first character) between any two type 1 *Poll For Master* frames (Poll For Master) sent by the IUT greater than 20 millseconds T_{neg_err}, but less than 100 millseconds + T_{pos_err}?)

2) Clause 12.1.3.3 – Should not require a Data Frame

12.1.3.3 Verify Tframe_gap

Purpose: Verify that the maximum idle time between data octets when transmitting a frame is 20 bit times or less.

Test Steps:

- 1. Elicit the transmission of any data frame from the IUT.
- 2. Measure the longest EIA-485 idle time that appears between octets within the data frame transmitted by the IUT. If there is no idle time between octets, pass the IUT.
- 3. Fail the IUT if the time measured in step 2 is greater than the time intervals shown below for each baud rate.

9600 baud:	fail if interval is greater than 2,083 microseconds
19200 baud:	fail if interval is greater than 1,042 microseconds
38400 baud:	fail if interval is greater than 521 microseconds
57600 baud:	fail if interval is greater than 347 microseconds
76800 baud:	fail if interval is greater than 261 microseconds
115200 baud:	fail if interval is greater than 173 microseconds
x baud:	fail if interval is greater than (20/x) seconds

3) Document to PDF Conversion failed for Some '≥' and '≤' Characters

7.3.1.11

3. IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN

10. IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN

17. IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN

•••

. .

7.3.1.16

•••

- 3. WRITE (the array property being tested) = (array of non-zero size N₂, where N₂ $\leq d$ N₁)
- 4. VERIFY (array is as written in step 3)
- 5. WRITE (the array property being tested) = (array of non-zero size N_3 , where $N_3 \ge e N_1$)
- 6. VERIFY (array is as written in step 5)
- 7. WRITE (the array property being tested) = (a non-zero unsigned value N₄, where N₄ $\leq d$ N₁), ARRAY INDEX = 0
- 8. VERIFY (array contains first N₄ elements of the array written in step 5)
- 9. WRITE (the array property being tested) = (N₅, where N₅ $\ge e$ N₄), ARRAY INDEX = 0
- 10. VERIFY (array contains first N₄ elements of the array written in step 5, plus N₅ N₄ additional elements, initialized to particular values if specified for the array property being tested)
- 11. TRANSMIT WriteProperty-Request,

'Object Identifier' =	(the object being tested),
'Property Identifier' =	(the array property being tested),
'Property Array Index' =	(N ₆ , where N ₆ $\geq $ N ₅),
'Property Value' =	(one array element)

•••

7.3.2.9.7

• • •

3. IF (Protocol_Revision is present and Protocol_Revision ≥ 10) THEN

8.4.1

. . .

6. IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN

13. IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN

•••

8.4.2

- • •
- 2. IF ((Protocol_Revision is present AND Protocol_Revision ≥e 13) OR ((Protocol_Revision is present AND Protocol_Revision d-12 < 13) AND (pAlarmValues contains at least one value))) THEN { IF (pMonitoredValue is writable) THEN WRITE pMonitoredValue = (a value from pAlarmValues) ELSE

MAKE (pMonitoredValue have a valu	e pAlarmValues)
WAIT (pTimeDelay)	
BEFORE Notification Fail Time	
RECEIVE ConfirmedEventNotificatio	n-Request,
'Process Identifier' =	(any valid process ID),
'Initiating Device Identifier' =	IUT,
'Event Object Identifier' =	(the intrinsic reporting object being tested or the Event Enrollment object being tested).
'Time Stamp' =	(T1, any valid time stamp).
'Notification Class' =	(the configured notification class)
'Priority' =	(the value configured to correspond to a TO_OFFNORMAL transition)
'Event Type' –	CHANGE OF STATE
'Message Text' –	(optional any valid message text)
'Notify Type' –	EVENT ALADM
'A alc Paquirad' -	TDIE FAI SE
AckRequired =	I KUE FALSE,
From State =	NORMAL,
To State $=$	OFFNORMAL,
'Event Values' =	pMonitored Value, pStatusFlags
TRANSMIT BACnet-SimpleACK-PDU	
IF (Protocol_Revision is present AND Prot	ocol_Revision ≥e 13) THEN
VERIFY pStatusFlags = (TRUE, FAL	SE, ?, ?)
VERIFY pCurrentState = OFFNORMAL	
IF (Protocol_Revision is present AND Prot	$ocol_Revision \ge 1$) THEN
VERIFY Event Time Stamps = $(T1, T)$	Γa, Tb)
IF (pMonitoredValue is writable) THEN	
WRITE pMonitoredValue = (a value the second	hat corresponds to a NORMAL state)
ELSE	1
MAKE (pMonitoredValue have a valu	e that corresponds to a NORMAL state)
WAIT (pTimeDelay)	· ····· ······························
BEFORE Notification Fail Time	
BECEIVE ConfirmedEventNotificatio	n-Request
'Process Identifier' -	(any valid process ID)
'Initiating Davida Identifiar' –	(any value process iD),
'Event Object Identifier' =	101, (the intrinsic reporting chiest being tested on the Event Enrollment.
Event Object Identifier =	(the intrinsic reporting object being tested or the Event Enrollment object being tested),
"Time Stamp' =	(12, any valid time stamp),
'Notification Class' =	(the configured notification class),
'Priority' =	(the value configured to correspond to a TO_NORMAL transition),
'Event Type' =	CHANGE_OF_STATE,
'Message Text' =	(optional, any valid message text),
'Notify Type' =	EVENT ALARM,
'AckRequired' =	TRUE FALSE,
'From State' =	OFFNORMAL.
'To State' =	NORMAL.
'Event Values' =	pMonitoredValue_nStatusFlags
TRANSMIT BACnet-SimpleACK-PDU	pittointored value, potatasi lago
IE (Protocol Revision is present AND Prot	acal Revision > 13) THEN
VEDIEV pStatusElage - (EALSE EAL	$COL_{COL_{A}} = 13 (11121)$
VENIF I POLIUSFIAGS = (FALSE, FAL VEDIEV poliumentState = NODMAL	(، , :)
$v \in Kir i p \in Uirentstate = NOKWAL$	
IF (Protocol_Revision is present AND Prot VERIFY Event_Time_Stamps = (T1, T	$\operatorname{Ocol}_{\operatorname{Kevision}} \geq 1$) THEN Γ_a, T_2)
F ((Protocol_Revision is present AND Protoco	l_Revision $\frac{d}{d} = 12 < 13$)

...

8.4.3.1

IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN

8.4.3.2

9. IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN ...

8.4.4

IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN
 IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN
 IF (Protocol_Revision is present AND Protocol_Revision >> 13) THEN

14. IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN ...

8.4.5

10. IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN

21. IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN

32. IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN

43. IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN ...

8.4.6

10. IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN

21. IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN

32. IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN

43. IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN ...

8.4.8.1

- 7. IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN VERIFY pStatusFlags = (TRUE, FALSE, ?, ?)
- 8. VERIFY pCurrentState = OFFNORMAL
- 9. IF (Protocol_Revision is present AND Protocol_Revision ≥ 1) THEN

•••

4.	IF (latching is supported) THEN {	
	CHECK (pCurrentState = OFFNOR	MAL)
	MAKE (the object reset)	
	BEFORE Notification Fail Time	
	RECEIVE ConfirmedEventNotifica	tion-Request.
	'Process Identifier' =	(any valid process ID)
	'Initiating Device Identifier' –	IIIT
	'Event Object Identifier' –	(the intrinsic reporting object being tested or the Event Enrollment
	Event Object Identifier =	chiest being tested)
	'Timo Stomp' -	(T1: any valid time stamp)
	'Notification Class' –	(11. any value time stamp), (the configured notification class)
	'Driority' -	(the value configured to correspond to a TO OPENODMAL transition)
	Fliolity –	(the value configured to correspond to a TO-OFTWORWAL transition),
	'Event Type' =	CHANGE_OF_LIFE_SAFETY,
	'Message Text' =	(S1: optional, any valid message text),
	'Notify Type' =	EVENT ALARM,
	'AckRequired' =	TRUE FALSE,
	'From State' =	OFFNORMAL,
	'To State' =	NORMAL,
	'Event Values' =	pMonitoredValue, pMode, pStatusFlags, pOperationExpected
	TRANSMIT BACnet-SimpleACK-	PDU,
	IF (Protocol_Revision is present AN	ID Protocol_Revision ≥e 13) THEN
	VERIFY pStatusFlags = (FALS	SE, FALSE, ?, ?),
	VERIFY pCurrentState = NORMA	L,
	IF (Protocol_Revision is present AN	ID Protocol_Revision ≥e 1) THEN
	VERIFY Event_Time_Stamps	$=(*, *, T1)^{-1}$
	IF (Event_Message_Texts property	exists) THEN
	VERIFY Event_Message_Text	s = (*, *, S1)
	}	
	ELSE {	
	BEFORE Notification Fail Time	
	RECEIVE ConfirmedEventNot	ification-Request,
	'Process Identifier' =	(any valid process ID),
	'Initiating Device Identifier	S' = IUT,
	'Event Object Identifier' =	(the intrinsic reporting object being tested or the Event Enrollment
	-	object being tested),
	'Time Stamp' =	(T2: any valid time stamp),
	'Notification Class' =	(the configured notification class),
	'Priority' =	(the value configured to correspond to a TO-OFFNORMAL transition),
	'Event Type' =	CHANGE_OF_LIFE_SAFETY,
	'Message Text' =	(S2: optional, any valid message text),
	'Notify Type' =	EVENT ALARM,
	'AckRequired' =	TRUE FALSE,
	'From State' =	OFFNORMAL.
	'To State' =	NORMAL.
	'Event Values' =	pMonitoredValue, pMode, pStatusFlags, pOperationExpected
	TRANSMIT BACnet-SimpleACK-	PDU
	IF (Protocol Revision is present AN	ID Protocol Revision ≥ 13) THEN
	VERIFY $pStatusFlags = (FALS)$	SE FALSE $(2, 2)$
	VERIEV pCurrentState = NORMA	
	IF (Protocol Revision is present AN	- JD Protocol Revision ≫ 1) THEN
	VERIFY Event Time Stamps	$= (* * T^2)$
	IF (Event Message Texts property	exists) THEN
	VERIFY Event Message Text	$s = (* * S^2)$
	, Liti i Litem_message_fext	~ (, , >=)

}

8.4.8.3

7.	IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN
	VERIFY pStatusFlags = (TRUE, FALSE, ?, ?)

- 8. VERIFY pCurrentState = LIFE_SAFETY_ALARM
- 9. IF (Protocol_Revision is present AND Protocol_Revision ≥ 1) THEN
- •••

4.	IF (latching is supported) THEN {	
	CHECK (pCurrentState = LIFE_SAFE'	TY_ALARM)
	MAKE (the object reset)	
	BEFORE Notification Fail Time	
	RECEIVE ConfirmedEventNotification	n-Request,
	'Process Identifier' =	(any valid process ID),
	'Initiating Device Identifier' =	IUT,
	'Event Object Identifier' =	(the intrinsic reporting object being tested or the Event Enrollment object
		being tested),
	'Time Stamp' =	(T1: any valid time stamp),
	'Notification Class' =	(the configured notification class),
	'Priority' =	(the value configured to correspond to a TO-OFFNORMAL transition),
	'Event Type' =	CHANGE_OF_LIFE_SAFETY,
	'Message Text' =	(S1: optional, any valid message text),
	'Notify Type' =	EVENT ALARM,
	'AckRequired' =	TRUE FALSE,
	'From State' =	LIFE_SAFETY_ALARM,
	'To State' =	NORMAL,
	'Event Values' =	pMonitoredValue, pMode, pStatusFlags, pOperationExpected
	TRANSMIT BACnet-SimpleACK-PDU	
	IF (Protocol_Revision is present AND)	Protocol_Revision ≥ 13) THEN
	VERIFY pStatusFlags = (FALSE,	FALSE, ?, ?)
	VERIFY pCurrentState = NORMAL	
	IF (Protocol_Revision is present AND)	Protocol_Revision ≥e 1) THEN
	VERIFY Event_Time_Stamps = (*	^c , *, T1)
	IF (Event_Message_Texts property exis	sts) THEN
	VERIFY Event_Message_Texts =	(*, *, S1)
	}	
	ELSE {	
	BEFORE Notification Fail Time	
	RECEIVE ConfirmedEventNotific	ation-Request,
	'Process Identifier' =	(any valid process ID),
	'Initiating Device Identifier' =	IUT,
	'Event Object Identifier' =	(the intrinsic reporting object being tested or the Event Enrollment object being tested),
	'Time Stamp' =	(T2: any valid time stamp).
	'Notification Class' =	(the configured notification class),
	'Priority' =	(the value configured to correspond to a TO-OFFNORMAL transition),
	'Event Type' =	CHANGE_OF_LIFE_SAFETY,
	'Message Text' =	(S2: optional, any valid message text),
	'Notify Type' =	EVENT ALARM,
	'AckRequired' =	TRUE FALSE,
	*	

'From State' =	LIFE_SAFETY_ALARM,	
'To State' =	NORMAL,	
'Event Values' =	pMonitoredValue, pMode, pStatusFlags, pOperationExpected	
TRANSMIT BACnet-SimpleACK-PDU		
IF (Protocol_Revision is present AND)	Protocol_Revision ≥e 13) THEN	
VERIFY pStatusFlags = (FALSE,	FALSE, ?, ?)	
VERIFY pCurrentState = NORMAL		
IF (Protocol Revision is present AND Protocol Revision ≥ 1) THEN		
VERIFY Event_Time_Stamps = (*	*, *T2)	
IF (Event Message Texts property exists) THEN		
VERIFY Event_Message_Texts =	(*, *, S2)	
_ 0 _		

}

•••		
4.	IF (latching is supported) THEN {	
	CHECK (pCurrentState = LIFE_SAFET	TY_ALARM)
	MAKE (the object reset)	
	BEFORE Notification Fail Time	
	RECEIVE ConfirmedEventNotification	-Request,
	'Process Identifier' =	(any valid process ID),
	'Initiating Device Identifier' =	IUT.
	'Event Object Identifier' =	(the intrinsic reporting object being tested or the Event Enrollment
	3	object being tested).
	'Time Stamp' =	(T1: any valid time stamp).
	'Notification Class' =	(the configured notification class).
	'Priority' =	(the value configured to correspond to a TO-OFFNORMAL transition).
	'Event Type' =	CHANGE OF LIFE SAFETY.
	'Message Text' =	(S1: optional, any valid message text).
	'Notify Type' =	EVENT ALARM.
	'AckRequired' =	TRUE FALSE.
	'From State' =	LIFE SAFETY ALARM.
	'To State' =	OFFNORMAL.
	'Event Values' =	pMonitoredValue, pMode, pStatusFlags, pOperationExpected
	TRANSMIT BACnet-SimpleACK-PDI]
	IF (Protocol Revision is present AND F	Protocol Revision >= 13) THEN
	VERIFY nStatusFlags = (TRUE, E)	ALSE 2 2)
	VERIFY pCurrentState = OFFNORMA	L.
	IF (Protocol Revision is present AND F	Protocol Revision >= 1) THEN
	VERIEV Event Time Stemps = (T	
	IF (Event Message Texts property exis	te) THEN
	VERIEV Event Message Texts - ((S1 * *)
	VERI I Event_Wessage_Texts = (51, ,)
	f SE /	
	BEFORE Notification Fail Time	
DEFORE INCLUE TO Firmed Event Notification Request		ation-Request
	'Process Identifier' =	(any valid process ID)
	'Initiating Device Identifier' =	IIIT
	'Event Object Identifier' –	(the intrinsic reporting object being tested or the Event Enrollment
	Event object identifier =	object being tested)
	'Time Stamp' =	(T2: any valid time stamp)
	'Notification Class' –	(the configured notification class)
	'Priority' –	(the value configured to correspond to a TO-OFFNORMAL transition)
	'Fvent Type' –	CHANGE OF LIFE SAFFTY
	'Message Text' -	(\$2: ontional any valid message text)
	wiessage reat -	(52. optional, any value message (ext),

}	'Notify Type' = 'AckRequired' = 'From State' = 'To State' = 'Event Values' = TRANSMIT BACnet-SimpleACK-PDU IF (Protocol_Revision is present AND I VERIFY pStatusFlags = (TRUE, F VERIFY pCurrentState = OFFNORMA IF (Protocol_Revision is present AND I VERIFY Event_Stamps = (T IF (Event_Message_Texts property exis VERIFY Event_Message_Texts = (EVENT ALARM, TRUE FALSE, LIFE_SAFETY_ALARM, OFFNORMAL, pMonitoredValue, pMode, pStatusFlags, pOperationExpected Protocol_Revision ≥ 13) THEN ALSE, ?, ?) L Protocol_Revision ≥ 1) THEN '2, *, *) its) THEN (S2, *, *)
8.4.8.6		
 4. IF	(latching is supported) THEN { CHECK (pCurrentState = OFFNORMA MAKE (the object reset) BEFORE Notification Fail Time RECEIVE ConfirmedEventNotifica 'Process Identifier' = 'Initiating Device Identifier' = 'Event Object Identifier' = 'Time Stamp' = 'Notification Class' = 'Priority' =	L) ation-Request, (any valid process ID), IUT, (the intrinsic reporting object being tested or the Event Enrollment object being tested), (T1: any valid time stamp), (the configured notification class), (the value configured to correspond to a TO-OFFNORMAL transition),
	'Event Type' = 'Message Text' = 'Notify Type' = 'AckRequired' = 'From State' = 'To State' = 'Event Values' = TRANSMIT BACnet-SimpleACK-PDU IF (Protocol_Revision is present AND I VERIFY pStatusFlags = (TRUE, F VERIFY pCurrentState = LIFE_SAFET IF (Protocol_Revision is present AND I VERIFY Event_Time_Stamps = (T IF (Event_Message_Texts property exis VERIFY Event_Message_Texts =	CHANGE_OF_LIFE_SAFETY, (S1: optional, any valid message text), EVENT ALARM, TRUE FALSE, OFFNORMAL, LIFE_SAFETY_ALARM, pMonitoredValue, pMode, pStatusFlags, pOperationExpected Protocol_Revision \geq 13) THEN ALSE, ?, ?) CY_ALARM Protocol_Revision \geq 1) THEN '1, *, *) tts) THEN (S1, *, *)
} EL	SE { BEFORE Notification Fail Time RECEIVE ConfirmedEventNotifica 'Process Identifier' = 'Initiating Device Identifier' = 'Event Object Identifier' = 'Time Stamp' = 'Notification Class' = 'Priority' =	ation-Request, (any valid process ID), IUT, (the intrinsic reporting object being tested or the Event Enrollment object being tested), (T2: any valid time stamp), (the configured notification class), (the value configured to correspond to a TO-OFFNORMAL transition).

'Event Type' =	CHANGE_OF_LIFE_SAFETY,	
'Message Text' =	(S2: optional, any valid message text),	
'Notify Type' =	EVENT ALARM,	
'AckRequired' =	TRUE FALSE,	
'From State' =	OFFNORMAL,	
'To State' =	LIFE_SAFETY_ALARM,	
'Event Values' =	pMonitoredValue, pMode, pStatusFlags, pOperationExpected	
TRANSMIT BACnet-SimpleACK-PDU		
IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN		
VERIFY pStatusFlags = (TRUE, FALSE, ?, ?)		
VERIFY pCurrentState = LIFE_SAFETY_ALARM		
IF (Protocol_Revision is present AND Protocol_Revision ≥ 1) THEN		
VERIFY Event_Time_Stamps = (T2, *, *)		
IF (Event_Message_Texts property exists) THEN		
VERIFY Event_Message_Texts =	(S2, *, *)	

. . .

}

8.	IF (Protocol_Revision is present AND	Protocol_Revision ≥ 13) THEN
	VERIFY pStatusFlags =	(FALSE, FALSE, ?, ?)

- 9. VERIFY pCurrentState = NORMAL
- 10. IF (Protocol_Revision is present AND Protocol_Revision ≥ 1) THEN
- 17. IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN VERIFY pStatusFlags = (TRUE, FALSE, ?, ?)
- 18. VERIFY pCurrentState = OFFNORMAL
- 19. IF (Protocol_Revision is present AND Protocol_Revision ≥ 1) THEN
- 26. IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN VERIFY pStatusFlags = (TRUE, FALSE, ?, ?)
- 27. VERIFY pCurrentState = OFFNORMAL
- 28. IF (Protocol_Revision is present AND Protocol_Revision \geq 1) THEN ...

8.4.8.8

. . .

- 6. IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN VERIFY pStatusFlags = (TRUE, FALSE, ?, ?)
- 7. VERIFY pCurrentState = OFFNORMAL
- 8. IF (Protocol_Revision is present AND Protocol_Revision ≥ 1) THEN

```
• • •
```

8.4.8.9

3. IF (latching is supported) THEN {
 CHECK (pCurrentState = OFFNORMAL)
 MAKE (the object reset)
 BEFORE Notification Fail Time
 RECEIVE ConfirmedEventNotification-Request,
 'Process Identifier' = (any valid process ID),
 'Initiating Device Identifier' = IUT,
 'Event Object Identifier' = (the intrinsic reporting object being tested or the Event Enrollment

'Time Stamp' = 'Notification Class' = 'Priority' =	object being tested), (T1: any valid time stamp), (the configured notification class), (the value configured to correspond to a TO-OFFNORMAL transition),
'Event Type' = 'Message Text' = 'Notify Type' = 'AckRequired' = 'From State' = 'To State' = 'Event Values' = TRANSMIT BACnet-SimpleACK-PDU IF (Protocol_Revision is present AND I VERIFY pStatusFlags = (FALSE, I VERIFY pCurrentState = NORMAL IF (Protocol_Revision is present AND I VERIFY Event_Time_Stamps = (* IF (Event_Message_Texts property exis VERIFY Event_Message_Texts = (*	CHANGE_OF_LIFE_SAFETY, (S1: optional, any valid message text), EVENT ALARM, TRUE FALSE, OFFNORMAL, pMonitoredValue, pMode, pStatusFlags, pOperationExpected J Protocol_Revision \geq 13) THEN FALSE, ?, ?) Protocol_Revision \geq 1) THEN $\frac{1}{2}$, *, T1) sts) THEN (*, *, S1)
SE { BEFORE Notification Fail Time RECEIVE ConfirmedEventNotifica 'Process Identifier' = 'Initiating Device Identifier' = 'Event Object Identifier' = 'Time Stamp' = 'Notification Class' = 'Priority' = 'Event Type' = 'Message Text' = 'Notify Type' = 'AckRequired' = 'From State' = 'TRANSMIT BACnet-SimpleACK-PDU IF (Protocol_Revision is present AND I VERIFY pStatusFlags = (FALSE, I VERIFY pCurrentState = NORMAL IF (Protocol_Revision is present AND I VERIFY Event_Time_Stamps = (* IF (Event_Message_Texts property existed)	ation-Request, (any valid process ID), IUT, (the intrinsic reporting object being tested or the Event Enrollment object being tested), (T2: any valid time stamp), (the configured notification class), (the value configured to correspond to a TO-OFFNORMAL transition), CHANGE_OF_LIFE_SAFETY, (S2: optional, any valid message text), EVENT ALARM, TRUE FALSE, OFFNORMAL, NORMAL, pMonitoredValue, pMode, pStatusFlags, pOperationExpected J Protocol_Revision ≥ 13) THEN FALSE, ?, ?) Protocol_Revision ≥ 1) THEN *, *, T2) sts) THEN
, Draf i Dront_Nessage_reats =	(,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,

}

} ELSE

```
•••
```

```
6. IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN
VERIFY pStatusFlags = (TRUE, FALSE, ?, ?)
```

```
7. VERIFY pCurrentState = LIFE_SAFETY_ALARM
```

```
8. IF (Protocol_Revision is present AND Protocol_Revision ≥ 1) THEN
```

```
•••
```

```
IF (latching is supported) THEN {
3.
        CHECK (pCurrentState = LIFE SAFETY ALARM)
        MAKE (the object reset)
        BEFORE Notification Fail Time
            RECEIVE ConfirmedEventNotification-Request,
                'Process Identifier' =
                                             (any valid process ID),
                'Initiating Device Identifier' =
                                             IUT.
                'Event Object Identifier' =
                                             (the intrinsic reporting object being tested or the Event Enrollment
                                             object being tested),
                'Time Stamp' =
                                             (T1: any valid time stamp),
                'Notification Class' =
                                             (the configured notification class),
                'Priority' =
                                             (the value configured to correspond to a TO-OFFNORMAL transition),
                'Event Type' =
                                             CHANGE_OF_LIFE_SAFETY,
                'Message Text' =
                                             (S1: optional, any valid message text),
                'Notify Type' =
                                             EVENT | ALARM,
                'AckRequired' =
                                             TRUE | FALSE,
                'From State' =
                                             LIFE SAFETY ALARM,
                'To State' =
                                             NORMAL,
                'Event Values' =
                                             pMonitoredValue, pMode, pStatusFlags, pOperationExpected
        TRANSMIT BACnet-SimpleACK-PDU
        IF (Protocol Revision is present AND Protocol Revision ≥ 13) THEN
            VERIFY pStatusFlags = (FALSE, FALSE, ?, ?)
        VERIFY pCurrentState = NORMAL
        IF (Protocol Revision is present AND Protocol Revision at 1) THEN
            VERIFY Event_Time_Stamps = (*, *, T1)
        IF (Event_Message_Texts property exists) THEN
            VERIFY Event_Message_Texts = (*, *, S1)
    ELSE {
        BEFORE Notification Fail Time
            RECEIVE ConfirmedEventNotification-Request,
                'Process Identifier' =
                                             (any valid process ID),
                'Initiating Device Identifier' =
                                             IUT.
                'Event Object Identifier' =
                                             (the intrinsic reporting object being tested or the Event Enrollment
                                             object being tested),
                'Time Stamp' =
                                             (T2: any valid time stamp),
                'Notification Class' =
                                             (the configured notification class),
                                             (the value configured to correspond to a TO-OFFNORMAL transition),
                'Priority' =
                                             CHANGE_OF_LIFE_SAFETY,
                'Event Type' =
                                             (S2: optional, any valid message text),
                'Message Text' =
                'Notify Type' =
                                             EVENT | ALARM,
                'AckRequired' =
                                             TRUE | FALSE,
                'From State' =
                                             LIFE_SAFETY_ALARM,
                'To State' =
                                             NORMAL.
                'Event Values' =
                                             pMonitoredValue, pMode, pStatusFlags, pOperationExpected
        TRANSMIT BACnet-SimpleACK-PDU
        IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN
            VERIFY pStatusFlags = (FALSE, FALSE, ?, ?)
        VERIFY pCurrentState = NORMAL
        IF (Protocol_Revision is present AND Protocol_Revision ≥ 1) THEN
            VERIFY Event Time Stamps = (*, *, T2)
        IF (Event_Message_Texts property exists) THEN
            VERIFY Event Message Texts = (*, *, S2)
```

}

 2	MAKE (pMode a different value that forces	the state to OFFNORMAL)		
2.	IF (latching is supported) THEN {			
	CHECK (nCurrentState - OEENORMAL)			
	MAKE (the object reset)			
	REFORE Notification Fail Time			
	RECEIVE ConfirmedEventNotification_Request			
	'Process Identifier' =	(any valid process ID)		
	'Initiating Device Identifier' =	IUT.		
	'Event Object Identifier' =	(the intrinsic reporting object being tested or the Event Enrollment		
		object being tested).		
	'Time Stamp' =	(T1: any valid time stamp).		
	'Notification Class' =	(the configured notification class).		
	'Priority' =	(the value configured to correspond to a TO-OFFNORMAL transition).		
	'Event Type' -	CHANGE OF LIFE SAFETY		
	'Message Text' -	(S1: optional any valid massage text)		
	Netify Type' –	EVENT ALADM		
	Noury Type –			
	'Erom State' –	LIEE SAFETY ALADM		
	To State' –	OFENODMAI		
	'Event Values' –	nMonitoredValue nMode nStatusElags nOnerationExpected		
	TRANSMIT BACnet-SimpleACK-PDI	I		
	IF (Protocol Revision is present AND)	- Protocol Revision >= 13) THEN		
	VERIFY pStatusFlags = (TRUE F	ALSE ? ?)		
	VERIEV nCurrentState – OFENORMAI			
	IF (Protocol Revision is present AND)	Protocol Revision >= 1) THEN		
	VERIFY Event Time Stamps = (1)	1 * *)		
	IF (Event Message Texts property exis	sts) THEN		
VERIFY Event Message Texts $P(S1 + *)$				
	}			
	ELSE {			
	BEFORE Notification Fail Time			
	RECEIVE ConfirmedEventNotific	ation-Request,		
'Process Identifier' = (any valid process ID).		(any valid process ID),		
	'Initiating Device Identifier' =	IUT,		
	'Event Object Identifier' =	(the intrinsic reporting object being tested or the Event Enrollment		
		object being tested),		
	'Time Stamp' =	(T2: any valid time stamp),		
	'Notification Class' =	(the configured notification class),		
	'Priority' =	(the value configured to correspond to a TO-OFFNORMAL transition),		
	'Event Type' =	CHANGE_OF_LIFE_SAFETY,		
	'Message Text' =	(S2: optional, any valid message text),		
	'Notify Type' =	EVENT ALARM,		
	'AckRequired' =	TRUE FALSE,		
	'From State' =	LIFE_SAFETY_ALARM,		
	'To State' =	OFFNORMAL,		
	'Event Values' =	pMonitoredValue, pMode, pStatusFlags, pOperationExpected		
	TRANSMIT BACnet-SimpleACK-PDU			
	IF (Protocol_Revision is present AND Protocol_Revision 2 13) THEN			
	VERIFY pStatusFlags = (TRUE, FALSE, ?, ?)			
	VERIFY pCurrentState = OFFNORMAL			
	IF (Protocol_Revision is present AND I	Protocol_Revision ≥ 1) THEN		

```
VERIFY Event_Time_Stamps = (T2, *, *)
IF (Event_Message_Texts property exists) THEN
VERIFY Event_Message_Texts = (S2, *, *)
```

```
}
```

```
3. IF (latching is supported) THEN
        CHECK (pCurrentState = OFFNORMAL)
        MAKE (the object reset)
        BEFORE Notification Fail Time
            RECEIVE ConfirmedEventNotification-Request,
                'Process Identifier' =
                                             (any valid process ID),
                'Initiating Device Identifier' =
                                             IUT,
                'Event Object Identifier' =
                                             (the intrinsic reporting object being tested or the Event Enrollment
                                             object being tested),
                'Time Stamp' =
                                             (T1: any valid time stamp),
                'Notification Class' =
                                             (the configured notification class),
                                             (the value configured to correspond to a TO-OFFNORMAL transition),
                'Priority' =
                'Event Type' =
                                             CHANGE_OF_LIFE_SAFETY,
                'Message Text' =
                                             (S1: optional, any valid message text),
                'Notify Type' =
                                             EVENT | ALARM,
                'AckRequired' =
                                             TRUE | FALSE,
                                             LIFE SAFETY ALARM,
                'From State' =
                'To State' =
                                             OFFNORMAL.
                'Event Values' =
                                             pMonitoredValue, pMode, pStatusFlags, pOperationExpected
        TRANSMIT BACnet-SimpleACK-PDU
        IF (Protocol Revision is present AND Protocol Revision ≥e 13) THEN
            VERIFY pStatusFlags = (TRUE, FALSE, ?, ?)
        VERIFY pCurrentState = OFFNORMAL
        IF (Protocol_Revision is present AND Protocol_Revision ≥ 1) THEN
            VERIFY Event Time Stamps = (T1, *, *)
        IF (Event_Message_Texts property exists) THEN
            VERIFY Event Message Texts = (S1, *, *)
    ELSE {
        BEFORE Notification Fail Time
            RECEIVE ConfirmedEventNotification-Request,
                'Process Identifier' =
                                             (any valid process ID),
                'Initiating Device Identifier' = IUT,
                'Event Object Identifier' =
                                             (the intrinsic reporting object being tested or the Event Enrollment
                                             object being tested),
                                             (T2: any valid time stamp),
                'Time Stamp' =
                'Notification Class' =
                                             (the configured notification class),
                'Priority' =
                                             (the value configured to correspond to a TO-OFFNORMAL transition),
                'Event Type' =
                                             CHANGE OF LIFE SAFETY.
                                             (S2: optional, any valid message text),
                'Message Text' =
                'Notify Type' =
                                             EVENT | ALARM,
                'AckRequired' =
                                             TRUE | FALSE,
                'From State' =
                                             LIFE SAFETY ALARM,
                'To State' =
                                             OFFNORMAL,
                'Event Values' =
                                             pMonitoredValue, pMode, pStatusFlags, pOperationExpected
        TRANSMIT BACnet-SimpleACK-PDU
        IF (Protocol Revision is present AND Protocol Revision ≥e 13) THEN
            VERIFY pStatusFlags = (TRUE, FALSE, ?, ?)
        VERIFY pCurrentState = OFFNORMAL
```

```
June 18, 2021
```

```
IF (Protocol_Revision is present AND Protocol_Revision ≥e 1) THEN
VERIFY Event_Time_Stamps = (T2, *, *)
IF (Event_Message_Texts property exists) THEN
VERIFY Event_Message_Texts = (S2, *, *)
}
```

8.4.9

```
1. IF (the object generates TO-OFFNORMAL transitions) THEN {
        READ CS1 = pCurrentState
        MAKE (an OFFNORMAL condition exist)
        WAIT D1
        BEFORE Notification Fail Time
            RECEIVE ConfirmedEventNotification-Request,
                 'Process Identifier' =
                                                   (any valid process ID),
                 'Initiating Device Identifier' =
                                                   IUT.
                 'Event Object Identifier' =
                                                   (the event generating object),
                 'Time Stamp' =
                                                   (TS1: the current local time),
                 'Notification Class' =
                                                   (the configured notification class),
                 'Priority' =
                                                   (the value configured for TO_OFFNORMAL),
                 'Event Type' =
                                                   EXTENDED,
                 'Message Text' =
                                                   (optional, any valid message text),
                 'Notify Type' =
                                                   EVENT | ALARM,
                 'AckRequired' =
                                                   TRUE | FALSE,
                 'From State' =
                                                   CS1.
                 'To State' =
                                                   (CS2: any offnormal valid event state),
                 'Event Values' =
                                                   (pVendorId: any valid vendor id),
                                                   (pEventType: any valid event-type),
                                                   (a list of 0 or more valid parameters as defined by the Vendor)
        TRANSMIT BACnet-SimpleACK-PDU
        IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN
            VERIFY pStatusFlags = (TRUE, FALSE,?,?)
        VERIFY pCurrentState = CS2
        IF (Protocol Revision is present AND Protocol Revision ≥ 1) THEN
            VERIFY Event_Time_Stamps = (TS1, *, *)
    IF (the object generates TO_NORMAL transitions) THEN {
        READ CS2 = pCurrentState
        MAKE (a NORMAL condition exist)
        WAIT D2
        BEFORE Notification Fail Time
            RECEIVE ConfirmedEventNotification-Request,
                 'Process Identifier' =
                                                   (any valid process ID),
                 'Initiating Device Identifier' =
                                                   IUT.
                 'Event Object Identifier' =
                                                   (the intrinsic reporting object being tested),
                 'Time Stamp' =
                                                   (TS2: the current local time),
                 'Notification Class' =
                                                   (the configured notification class),
                 'Priority' =
                                                   (the value configured for TO-NORMAL),
                 'Event Type' =
                                                   EXTENDED,
                 'Message Text' =
                                                   (optional, any valid message text),
                 'Notify Type' =
                                                   EVENT | ALARM,
                 'AckRequired' =
                                                   TRUE | FALSE,
                 'From State' =
                                                   CS2,
                 'To State' =
                                                   NORMAL,
                 'Event Values' =
                                                   (pVendorId: any valid vendor id),
```

(pEventType: any valid event-type), (a list of 0 or more valid parameters as defined by the Vendor)

TRANSMIT BACnet-SimpleACK-PDU IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN VERIFY pStatusFlags = (FALSE, FALSE, ?, ?) VERIFY pCurrentState = NORMAL IF (Protocol_Revision is present AND Protocol_Revision ≥ 1) THEN VERIFY Event_Time_Stamps = (*, *, TS2)

8.4.13

}

 2.	IF ((Protocol_Revision is present AND Protocol_Revision ≥ 13) OR ((Protocol_Revision is present AND Protocol_Revision d 12 < 13) AND (pAlarmValues contains at least one characterstring))) THEN {
 7.	IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN
 15.	IF (Protocol_Revision is present AND Protocol_Revision 2 13) THEN
 22.	IF (Protocol_Revision is present AND Protocol_Revision ≥ 13) THEN

8.4.15

• • •

. . .

...

...

6.	IF (Protocol	Revision is	present AND	Protocol	Revision ≥e	13) THEN
	(

12. IF (Protocol_Revision is present AND Protocol_Revision ≥e 13) THEN

8.23.2

1.	RECEIVE WritePropertyMultiple-Request,	
	'Object Identifier' = (any valid object identifier),	
	'Property Identifier' = (any valid non-array property of the specified object),	
	'Property Value' = (any value appropriate to the specified property),	
	'Property Identifier' = (any valid property of the specified object that was not previously used),	
	'Property Value' = (any value appropriate to the specified property)	
	(Any number of properties ≥ 2 is acceptable.)	

• • •

8.23.3

•••	
1.	RECEIVE WritePropertyMultiple-Request,

'Object Identifier' = (any valid object identifier),
'Property Identifier' = (any valid non-array property of the specified object),
'Property Value' = (any value appropriate to the specified property),
'Object Identifier' = (any valid object identifier not previously used),
'Property Identifier' = (any valid property of the specified object),
'Property Value' = (any value appropriate to the specified property)
 (Any number of (object, property, value) tuples ≥ 2 is acceptable.)

9.1.1.10

•••

Test Concept: An event is triggered and the IUT notifies the TD and one other device. The TD acknowledges the event and verifies that the acknowledgment is properly noted by the IUT. The IUT notifies all recipients that the event has been acknowledged. The TD then acknowledges the event again, and the IUT again notifies all recipients. This behavior was not defined before Protocol_Revision 7 and so this test shall only be performed if Protocol_Revision is present (i.e., Protocol_Revision ≥ 7).

•••

9.1.1.11

...

Test Concept: An event is triggered and the IUT notifies the TD and one other device. The TD acknowledges the event and verifies that the acknowledgment is properly noted by the IUT. The IUT notifies all recipients that the event has been acknowledged. The TD then acknowledges the event again, and the IUT again notifies all recipients. This behavior was not defined before Protocol_Revision 7 and so this test shall only be performed if Protocol_Revision is present (i.e., Protocol_Revision ≥ 7).

•••

... 1.

9.21.1.4

TRANSMIT ReadRange-Re	quest,
'Object Identifier' =	(the log object configured for this test),
'Property Identifier' =	Log_Buffer,
'Reference Time' =	(any value x: x is older (of earlier time) than the time of an entry in the buffer which has a sequence number of S, and x is newer
	than or equal to the time of any preceding entry),
'Count' =	(any value y: $0 < y \leq t$ Total_Record_Count - S + 1)

•••

9.21.1.4.1

1. TRANSMIT ReadRange-Request,

'Object Identifier' =	(the log object configured for this test),
'Property Identifier' =	Log_Buffer,
'Reference Time' =	(x, selected as described above),
'Count' =	(any value y: $0 < y \le d$ number of records in the buffer)

...

13.2.2

13.2.2 TimeSynchronization Recipients Test, Protocol_Revision ≥e7

•••

13.2.6

Dependencies: 13.2.2, "TimeSynchronization Recipients Test, Protocol_Revision $\geq e$ 7".

13.8.2.1

. . .

Backup Characteristics:

8. The TD is configured with a Protocol_Revision ≥ 10 . This is only used if the IUT claims Protocol_Revision ≥ 10 .

Restore Characteristics:

8. The TD is configured with a Protocol_Revision ≥ 10 . This is only used if the IUT claims Protocol_Revision ≥ 10 .

•••

. . .