INTERNATIONAL STANDARD

ISO 16484-6

First edition
2005-12-15

Building automation and control systems (BACS) —

Part 6:
Data communication conformance testing

Systèmes d'automatisation et de gestion technique du bâtiment —
Partie 6: Essais de conformité de la communication de données

Reference number
ISO 16484-6:2005(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web 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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this Technical Report may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO 16484-6 was prepared by the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) as ANSI/ASHRAE 135.1-2003, and was adopted (without modifications except those stated in Clause 2 of this International Standard) by Technical Committee ISO/TC 205, *Building environment design*.

ISO 16484 consists of the following parts, under the general title *Building automation and control systems (BACS)*:

Part 2: Hardware

Part 3: Functions

Part 5: Data communication protocol

Part 6: Data communication conformance testing

The task of updating this part of ISO 16484 has been delegated to an ISO Maintenance Agency. Details are given in the "Maintenance" clause overleaf.

Building automation and control systems (BACS) —

Part 6:
Data communication conformance testing

Scope

This part of ISO 16484 defines a standard method for verifying that an implementation of the BACnet protocol provides each capability claimed in its Protocol Implementation Conformance Statement (PICS) in conformance with the BACnet standard (ISO 16484-5).

The scope is further detailed in Clause 2 of the enclosed ANSI/ASHRAE publication.

Recommendations

The technical recommendations are those made in the following publication (reproduced on the following pages), which is adopted as an International Standard:

ANSI/ASHRAE 135.1-2003, Method for Test for Conformance to BACnet

International Standards cited in the text are the following:

ISO 16484-5, *Building automation and control systems — Part 5: Data communication protocol*

ISO 10646, *Information technology — Universal Multiple-Octet Coded Character Set (UCS)*

ISO/IEC 8802-2*, Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements — Part 2: Logical link control*

ISO/IEC 8802-3*, Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements — Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer*

ISO/IEC 8859-1, *Information technology — 8-bit single-byte coded graphic character sets — Part 1: Latin alphabet No. 1*

Maintenance

The ISO Maintenance Agency for ISO 16484-5 and ISO 16484-6, as designated by the ISO Council, is

American Society of Heating, Refrigerating and Air-Conditioning Engineers, Inc.,
1791 Tullie Circle NE,
Atlanta,
GA 30329,
USA.

E-mail: ISOMA@ashrae.org

The procedures for the maintenance of ISO 16484-5 and ISO 16484-6 are available at

http://www.iso.org/tc205

or from the Maintenance Agency Secretariat at the above address.

Contents

CLAUSE		PAGE
1.	PURPOSE	7
2.	SCOPE	7
3.	DEFINITIONS	7
4.	ELECTRONIC PICS FILE FORMAT	7
4.1	Character Encoding	7
4.2	Structure of EPICS Files	8
4.3	Character Strings	9
4.4	Notational Rules for Parameter Values	9
4.5	Sections of the EPICS File	10
5.	EPICS CONSISTENCY TESTS	27
6.	CONVENTIONS FOR SPECIFYING BACnet CONFORMANCE TESTS	28
6.1	TCSL Components	28
6.2	TCSL Statements	29
6.3	Time Dependencies	34
6.4	BACnet References	34
7.	OBJECT SUPPORT TESTS	35
7.1	Read Support for Properties in the Test Database	35
7.2	Write Support for Properties in the Test Database	35
7.3	Object Functionality Tests	36
8.	APPLICATION SERVICE INITIATION TESTS	106
8.1	AcknowledgeAlarm Service Initiation Tests	106
8.2	ConfirmedCOVNotification Service Initiation Tests	107
8.3	UnconfirmedCOVNotification Service Initiation Tests	116
8.4	ConfirmedEventNotification Service Initiation Tests	118
8.5	UnconfirmedEventNotification Service Initiation Tests	151
8.6	GetAlarmSummary Service Initiation Tests	158
8.7	GetEnrollmentSummary Service Initiation Tests	158
8.8	GetEventInformation Service Initiation Tests	160
8.9	LifeSafetyOperation Service Initiation Tests	160
8.10	SubscribeCOV Service Initiation Tests	161
8.11	SubscribeCOVProperty Service Initiation Tests	162
8.12	AtomicReadFile Service Initiation Tests	163
8.13	AtomicWriteFile Service Initiation Tests	163
8.14	AddListElement Service Initiation Tests	164
8.15	RemoveListElement Service Initiation Tests	165
8.16	CreateObject Service Initiation Tests	165
8.17	DeleteObject Service Initiation Tests	166
8.18	ReadProperty Service Initiation Tests	167
8.19	ReadPropertyConditional Service Initiation Tests	167
8.20	ReadPropertyMultiple Service Initiation Tests	168
8.21	ReadRange Service Initiation Tests	169
8.22	WriteProperty Service Initiation Tests	170
8.23	WritePropertyMultiple Service Initiation Tests	171
8.24	DeviceCommunicationControl Service Initiation Tests	173
8.25	ConfirmedPrivateTransfer Service Initiation Test	174
8.26	UnconfirmedPrivateTransfer Service Initiation Test	174

8.27 ReinitializeDevice Service Initiation Tests 175
8.28 ConfirmedTextMessage Service Initiation Tests 176
8.29 UnconfirmedTextMessage Service Initiation Tests 177
8.30 TimeSynchronization Service Initiation Tests 178
8.31 UTCTimeSynchronization Service Initiation Tests 178
8.32 Who-Has Service Initiation Tests 178
8.33 I-Have Service Initiation Tests 179
8.34 Who-Is Service Initiation Tests 179
8.35 I-Am Service Initiation Tests 180
8.36 VT-Open Service Initiation Tests 180
8.37 VT-Close Service Initiation Tests 181
8.38 VT-Data Service Initiation Tests 183
8.39 RequestKey Service Initiation Tests 184
8.40 Authenticate Service Initiation Tests 185
9. APPLICATION SERVICE EXECUTION TESTS 190
9.1 AcknowledgeAlarm Service Execution Tests 190
9.2 ConfirmedCOVNotification Service Execution Tests 203
9.3 UnconfirmedCOVNotification Service Execution Tests 207
9.4 ConfirmedEventNotification Service Execution Tests 207
9.5 UnconfirmedEventNotification Service Execution Tests 209
9.6 GetAlarmSummary Service Execution Tests 209
9.7 GetEnrollmentSummary Service Execution Tests 210
9.8 GetEventInformation Service Execution Tests 214
9.9 LifeSafetyOperation Service Execution Test 216
9.10 SubscribeCOV Service Execution Tests 218
9.11 SubscribeCOVProperty Service Execution Tests 223
9.12 AtomicReadFile Service Execution Tests 230
9.13 AtomicWriteFile Service Execution Tests 237
9.14 AddListElement Service Execution Tests 248
9.15 RemoveListElement Service Execution Tests 250
9.16 CreateObject Service Execution Tests 252
9.17 DeleteObject Service Execution Tests 255
9.18 ReadProperty Service Execution Tests 256
9.19 ReadPropertyConditional Service Execution Tests 258
9.20 ReadPropertyMultiple Service Execution Tests 259
9.21 ReadRange Service Execution Tests 264
9.22 WriteProperty Service Execution Tests 267
9.23 WritePropertyMultiple Service Execution Tests 270
9.24 DeviceCommunicationControl Service Execution Test 274
9.25 ConfirmedPrivateTransfer Service Execution Tests 278
9.26 UnconfirmedPrivateTransfer Service Execution Tests 278
9.27 ReinitializeDevice Service Execution Tests 278
9.28 ConfirmedTextMessage Service Execution Tests 281
9.29 UnconfirmedTextMessage Service Execution Tests 282
9.30 TimeSynchronization Service Execution Tests 282
9.31 UTCTimeSynchronization Service Execution Tests 284
9.32 Who-Has Service Execution Tests 284
9.33 Who-Is Service Execution Tests 290
9.34 VT-Open Service Execution Tests 293
9.35 VT-Close Service Execution Tests 295
9.36 VT-Data Service Execution Tests 296
9.37 RequestKey Service Execution Test 296
9.38 Authenticate Service Execution Tests 298

10. NETWORK LAYER PROTOCOL TESTS 304
10.1 Processing Application Layer Messages Originating from Remote Networks 304
10.2 Router Functionality Tests 304
10.3 Half-Router Functionality Tests 328
10.4 B/IP PAD Tests 336
10.5 Initiating Network Layer Messages 338
11. LOGICAL LINK LAYER PROTOCOL TESTS 340
11.1 UI Command and Response 340
11.2 XID Command and Response 340
11.3 TEST Command and Response 341
12. DATA LINK LAYER PROTOCOLS TESTS 342
12.1 MS/TP State Machine Tests 342
12.2 PTP State Machine Tests 399
13. SPECIAL FUNCTIONALITY TESTS 440
13.1 Segmentation 440
13.2 Time Master 449
13.3 Character Sets 450
13.4 Malformed PDUs 450
14. BACnet/IP Functionality Tests 453
14.1 Non-BBMD B/IP Device 453
14.2 Non-BBMD B/IP device Device with a Server Application 455
14.3 Broadcast Distribution Table Operations 456
14.4 Foreign Device Table Operations (Negative Tests) 459
14.5 BACnet Broadcast Management (No Foreign Device Table, No Applications) 460
14.6 Foreign Device Management 460
14.7 Broadcast Management (BBMD, Foreign Devices, Local Application) 463
15. Reporting Test Results 467
ANNEX A - Example EPICS (INFORMATIVE) 468

1. PURPOSE

To define a standard method for verifying that an implementation of the BACnet protocol provides each capability claimed in its Protocol Implementation Conformance Statement (PICS) in conformance with the BACnet standard.

2. SCOPE

This standard provides a comprehensive set of procedures for verifying the correct implementation of each capability claimed on a BACnet PICS including:

(a) support of each claimed BACnet service, either as an initiator, executor, or both,
(b) support of each claimed BACnet object-type, including both required properties and each claimed optional property,
(c) support of the BACnet network layer protocol,
(d) support of each claimed data link option, and
(e) support of all claimed special functionality.

3. DEFINITIONS

All definitions from ISO 16484-5 also apply to this standard.

3.1 local network: the network to which a BACnet device is directly connected.

3.2 remote network: a network that is accessible from a BACnet device only by passing through one or more routers.

3.3 test database: a database of BACnet functionality and objects created by reading the contents of an EPICS.

3.4 Abbreviations and Acronyms Used in the Standard

BNF Backus-Naur Form syntax
EPICS electronic protocol implementation conformance statement
IUT implementation under test
TCSL testing and conformance scripting language
TD testing device
TPI text protocol information

4. ELECTRONIC PICS FILE FORMAT

An electronic protocol implementation conformance statement (EPICS) file contains a BACnet protocol implementation conformance statement expressed in a standardized text form. EPICS files are machine and human readable representations of the implementation of BACnet objects and services within a given device. EPICS files shall use the extension ".TPI" (text protocol information) and contain normal editable text lines consisting of text character codes ending in carriage return/linefeed pairs (X'0D', X'0A').

EPICS files are used by software testing tools to conduct and interpret the results of tests defined in this standard. An EPICS file shall accompany any device tested according to the procedures of this standard.

4.1 Character Encoding

BACnet provides for a variety of possible character encodings. The character encodings in BACnet fall into three groups: octet streams, double octet streams and quad octet streams. Octet streams represent characters as single octet values. In some cases, such as Microsoft DBCS and JIS C 6226, certain octet values signal that the second octet which follows should be viewed along with the leading octet as a single value, thus extending the range to greater than 256 possible characters. In contrast, double octet streams view pairs of octets as representing single characters. The ISO 10646 UCS-2 encoding is an example. The first or leading octet of the pair is the most significant part of the value. Quad octet streams, such as ISO 10646 UCS-4, treat tuples of four octets at a time as single characters with the first or leading octet being the most significant.

To accommodate the various encodings that may be used with BACnet device descriptions, EPICS files begin with a header that serves both to identify the file as an EPICS file, and to identify the particular encoding used. The header begins with the string "PICS #" where # is replaced by a numeral representing the character set as shown in Table 4-1.

Table 4-1. Character Set Codes

code	*character set*
0	ANSI 3.4
1	Microsoft DBCS
2	JIS C 6226
3	ISO 10646 (UCS-4)
4	ISO 10646 (UCS-2)
5	ISO 8859-1

An octet stream format can be recognized by examining the first eight octets of the EPICS file. Using ANSI 3.4 encoding as an example these eight octets will contain: X'50' X'49' X'43' X'53' X'20' X'30' X'0D' X'0A'. This represents the text "PICS 0" followed by carriage return and linefeed.

A double octet stream format can be recognized by examining the first 16 octets of the EPICS file. Using ISO 10646 UCS-2 encoding as an example these 16 octets will contain:

X'00' X'50' X'00' X'49' X'00' X'43' X'00' X'53'
X'00' X'20' X'00' X'34' X'00' X'0D' X'00' X'0A'

This represents the text "PICS 4" followed by carriage return and linefeed.

A quad octet stream format can be recognized by examining the first 32 octets of the EPICS file. Using ISO 10646 UCS-4 as an example these 32 octets will contain:

X'00' X'00' X'00' X'50' X'00' X'00' X'00' X'49'
X'00' X'00' X'00' X'43' X'00' X'00' X'00' X'53'
X'00' X'00' X'00' X'20' X'00' X'00' X'00' X'33'
X'00' X'00' X'00' X'0D' X'00' X'00' X'00' X'0A'

This represents the text "PICS 3" followed by carriage return and linefeed.

4.2 Structure of EPICS Files

EPICS files consist of text lines ending in carriage return/linefeed pairs (X'0D', X'0A') encoded as octet, double octet or quad octet streams as defined in 4.1. In the rest of this standard, the term "character" will be used to mean one symbol encoded as one, two, or four octets based on the character encoding used in the EPICS file header. For example, the character space may be encoded as X'20' or X'0020' or X'00000020'. In this standard all characters will be shown in their single octet form.

The special symbol ↵ is used in this Clause to signify the presence of a carriage return/linefeed pair (X'0D0A'). Except within character strings, the character codes tab (X'09'), space (X'20'), carriage return (X'0D') and linefeed (X'0A') shall be considered to be white space. Any sequence of 1 or more white space characters shall be equivalent to a single white space character. Except within a character string, a sequence of two dashes (X'2D') shall signify the beginning of a comment which shall end with the next carriage return/linefeed pair, i.e., the end of the line upon which the -- appears. Comments shall be considered to be white space, and may thus be inserted freely.

EPICS files shall have, as their first line following the header, the literal text:

BACnet Protocol Implementation Conformance Statement ↵

This text serves as a signature identifying the EPICS file format.

Lines that define the sections of the EPICS (see 4.5) and the particular implementation data for a given device follow the signature line.

The EPICS file ends with a line containing the following literal text:

End of BACnet Protocol Implementation Conformance Statement ↵

4.3 Character Strings

The occurrence of a double quote (X'22'), single quote (X'27') or accent grave (X'60') shall signify character strings. For double quotes, the end of the string shall be signified by the next occurrence of a double quote, or the end of the line. For single quote or accent grave, the end of the string shall be signified by the next occurrence of a single quote (X'27'), or the end of the line. Thus strings which need to include a single quote or accent grave as a literal character in the string shall use the double quote quoting method, while strings which need to include double quote shall use the single quote or accent grave quoting method.

4.4 Notational Rules for Parameter Values

Within each section, parameters may need to be expressed in one of several forms. The following rules govern the format for parameters:

(a) key words are case insensitive so that X'41' through X'5A' are equivalent to X'61' through X'7A';
(b) null values are shown by the string "NULL";
(c) Boolean values are shown by the strings "T" or "TRUE" if the value is true, or "F" or "FALSE" if the value is false;
(d) integer values are shown as strings of digits, possibly with a leading minus (-): 12345 or -111;
(e) real values are shown with a decimal point, which may not be the first or last character: 1.23, 0.02, 1.0 but not .02;
(f) octet strings are shown as pairs of hex digits enclosed in either single quotes (X'2D') or accent graves (X'60'), and preceded by the letter "X": X'001122';
(g) character strings are represented as one or more characters enclosed in double, single or accent grave quotes as defined in 4.3: 'text' or 'text' or "text";
(h) bitstrings are shown as a list, enclosed by curly brackets ({ } or X'7B' and X'7D'), of true and false values: {T,T,F} or {TRUE, TRUE, FALSE}. When the actual value of a bit does not matter, a question mark is used: {T,T,?};
(i) enumerated values are represented as named, rather than numeric, values. Enumeration names are case insensitive so that X'41' through X'5A' are equivalent to X'61' through X'7A'. The underscore (X'5F') and dash (X'2D') are considered equivalent in enumeration names. Proprietary values are shown as a named text with no whitespace and ending in a non-negative decimal numeric. Each must start with the word "proprietary": Object_Type, proprietary-object-type-653;
(j) dates are represented enclosed in parenthesis: (Monday, 24-January-1998). Any "wild card" or unspecified field is shown by an asterisk (X'2A'): (Monday, *-January-1998). The omission of day of week implies that the day is unspecified: (24-January-1998);
(k) times are represented as hours, minutes, seconds, hundredths in the format hh:mm:ss.xx: 2:05:44.00, 16:54:59.99. Any "wild card" field is shown by an asterisk (X'2A'): 16:54:*.*;
(l) object identifiers are shown enclosed by parentheses, with commas separating the object type and the instance number: (analog-input, 56). Proprietary object types replace the object type enumeration with the word "proprietary" followed by the numeric value of the object type: (proprietary 700,1);
(m) constructed data items are represented enclosed by curly brackets ({ } or X'7B' and X'7D'), with elements separated by commas. If an element is itself a constructed value, then that element shall be enclosed in curly brackets.

4.4.1 Complex Parameter Values

Some parameter values, notably property values for constructed or CHOICE types of encoded values, need to use a more complex notation to represent their values. This notation is tied to the ASN.1 encoding for those property values and may appear obscure out of context. These additional rules govern the presentation of those types of parameter values:

(a) values which are a CHOICE of application-tagged values are represented by the value of the chosen item encoded as described in 4.4;
(b) values which are a CHOICE of context-tagged values are represented by the context tag number enclosed in square brackets, followed by the representation of the value of the chosen item;
(c) list values (ASN.1 "SEQUENCE OF") are represented enclosed in parenthsis, with the elements of the list separated by commas. If an element is itself a constructed value, then that element shall be enclosed in curly brackets;
(d) array values are represented enclosed in curly brackets, with the elements of the array separated by commas. If an element is itself a constructed value, then that element shall be enclosed in curly brackets.

4.4.2 Specifying Limits on Parameter Values

Some properties may have restrictions on the range or resolution of their values. In order to correctly interpret the results of tests in which the value of a property is changed using WriteProperty, WritePropertyMultiple, or AddListElement then read back using ReadProperty or ReadPropertyMultiple, it is necessary to know what these restrictions are. The test database may contain restriction statements that define these constraints. The permissible restrictions and the datatypes they apply to are:

(a) **minimum** - the minimum value for Unsigned, Integer, Real, or Double datatypes. The earliest date for the Date datatype;
(b) **maximum** - the maximum value for Unsigned, Integer, Real, or Double datatypes. The latest date for the Date datatype;
(c) **resolution** - the minimum guaranteed resolution for Real and Double datatypes. The minimum time resolution in seconds for the Time datatype;
(d) **maximum length string** - the maximum length of a CharacterString or OctetString;
(e) **maximum length list** - the maximum number of elements guaranteed to fit in a list;
(f) **maximum length array** - the maximum number of elements in an array;
(g) **allowed values** - a comma-delimited list of supported enumerations for an Enumerated datatype. A comma-delimited list of object types for properties that reference an external object identifier.

Restriction statements shall be listed within pointed brackets (< and >) following the default value. If there are multiple restrictions within a single set of angle brackets, then the restrictions shall be separated by a semicolon (;). A restriction statement consists of the restriction name followed by a colon (:) followed by the restriction value or, where appropriate, a comma-delimited list of possible values.

Here are some examples of property values with restriction statements as they could appear in the test database.

present-value: 13.4 <minimum: 0.0; maximum: 20.0; resolution: 0.1>
description: "this is a description" <maximum length string: 30>
units: milliamperes <allowed values: milliamperes, amperes>
object-property-reference: (analog input,12) <allowed values: analog input, analog value>

The Units property is a special case, because changing the units can change the value of the Present_Value property as well as any restrictions on its value. Therefore, minimum, maximum, and resolution restrictions are only valid for the default value of the Units property.

It is possible to specify default restrictions for most datatypes as described in 4.5.8. Restriction statements in the test database override the default restrictions for the individual property that contains the restriction statement.

4.5 Sections of the EPICS File

Each section of the EPICS file begins with a section name followed by a colon (: or X'3A'). After the colon is a set of one or more parameters delimited by a set of curly braces ({ } or X'7B' X'7D').

The following symbols are used as placeholders to indicate the presence of parameter information:
(a) the open box symbol inside quotation marks, "❑", is used to indicate that a character string parameter shall be present;
(b) the open box symbol with no quotation marks, ❑, is used to indicate that a parameter with a datatype other than a character string shall be present;
(c) a question mark, ?, is used in the test database to indicate that the property is present but the value is unknown because it depends on hardware input or is being changed by an internal algorithm.

An example EPICS file may be found in Annex A.

4.5.1 General Information Sections

These sections provide general information about the BACnet device. The syntax for these sections is shown below.

Vendor Name: "❑"↲
Product Name: "❑"↲
Product Model Number: "❑"↲
Product Description: "❑"↲

4.5.2 Conformance Sections

These sections provide information about the BACnet functionality that the device claims to support.

4.5.2.1 BIBBs Supported

This section indicates which BIBBs are supported. The syntax is shown below. Each BIBB shall be listed, one per line between the curly braces. An empty list indicates that no BIBBs are supported.

BIBBs Supported: ↲
{↲
 ❑↲
}↲

The BIBBs may be any of:

DS-RP-A	DS-RP-B	
DS-RPM-A	DS-RPM-B	
DS-RPC-A	DS-RPC-B	
DS-WP-A	DS-WP-B	
DS-WPM-A	DS-WPM-B	
DS-COV-A	DS-COV-B	
DS-COVP-A	DS-COVP-B	
DS-COVU-A	DS-COVU-B	
AE-N-A	AE-N-I-B	AE-N-E-B
AE-ACK-A	AE-ACK-B	
AE-ASUM-A	AE-ASUM-B	
AE-ESUM-A	AE-ESUM-B	
AE-INFO-A	AE-INFO-B	
AE-LS-A	AE-LS-B	
SCHED-A	SCHED-I-B	SCHED-E-B
T-VMT-A	T-VMT-I-B	T-VMT-E-B
T-ATR-A	T-ATR-B	
DM-DDB-A	DM-DDB-B	
DM-DOB-A	DM-DOB-B	
DM-DCC-A	DM-DCC-B	
DM-PT-A	DM-PT-B	
DM-TM-A	DM-TM-B	
DM-TS-A	DM-TS-B	
DM-UTC-A	DM-UTC-B	
DM-RD-A	DM-RD-B	
DM-BR-A	DM-BR-B	
DM-R-A	DM-R-B	
DM-LM-A	DM-LM-B	
DM-OCD-A	DM-OCD-B	
DM-VT-A	DM-VT-B	
NM-CE-A	NM-CE-B	
NM-RC-A	NM-RC-B	