

Part VI

ECHONET Discrete Lower-Layer Communication Interface Specifications

Revision History

Note) Version numbers except Ver.3.20 indicate Japanese editions.

- Version 1.0 March 18th 2000 Released / Open to consortium members
- July 2000 Open to the public
- Version 1.01 May 23rd 2001 Open to consortium members
- Version 1.0 addendum & corrigendum
- Version 2.00 August 7th 2001 Open to consortium members
- Since the power line A and power line B methods were integrated into a single power line method (based on the power line A method), the associated descriptions were corrected accordingly.

The following Table-of-Contents entries were revised:

	Revised entry	Revision/addition
1	4.2.1, 4.2.2, 4.2.8	Descriptions were changed because the power line A and power line B methods were integrated into a single method.
2	4.2.19, 4.2.20	Deleted because the power line A and power line B methods were integrated into a single method.
3	4.3	Descriptions were changed because the power line A and power line B methods were integrated into a single method.
4	4.3.2, 4.3.3, 4.3.4, 4.3.5	The sections indicated at left were renumbered because the power line A and power line B methods were integrated into a single method (the number of sections decreased by one from Version 1.01).

- Version 2.01 December 19th 2001 Open to consortium members
- Version 2.10 Preview December 28th 2001 Open to consortium members
- Version 2.10 Draft February 15th 2002 Open to consortium members
- Version 2.10 March 7th 2002 Open to consortium members

The following Table-of-Contents entries were revised:

	Revised entry	Revision/addition
1	2.1	- The following interfaces were added in accordance with revisions to the state transition stipulated in Part 2: "Request for complete initialization", "request for communication stop", and "request for complete stop"
2	2.1	- The request named "request for reset" was renamed "request for warm start" in accordance with revisions to the state transition stipulated in Part 2.
3	2.2	- The detailed interface descriptions were changed in accordance with revisions to the state transition stipulated in Part 2.
4	3.1	- The following interfaces were added in accordance with revisions to the state transition stipulated in Part 2: "Request for complete initialization", "request for communication stop", and "request for complete stop"
5	3.1	- The request named "request for reset" was renamed "request for warm start" in accordance with revisions to the state transition stipulated in Part 2.
6	3.2	- The detailed interface descriptions were changed in accordance with revisions to the state transition stipulated

		in Part 2.
7	4.1	- The following APIs were added in accordance with revisions to the state transition stipulated in Part 2: "LowInitAll", "LowStop", and "LowHalt"
8	4.1	- In accordance with revisions to the state transition stipulated in Part 2, "LowReset" was renamed "LowStart".
9	4.2	- The detailed API descriptions were changed in accordance with revisions to the state transition stipulated in Part 2.

•Version 2.11 April 26th 2002 Open to consortium members

The following Table-of-Contents entries were revised:

	Revised entry	Revision/addition
1	4.2.9	- The status of the lower-layer communication software block of the structure used was added.
2	4.2.17	- The type of syntax argument mac was corrected to pointer type, and mac_len was newly added.
3	4.2.18	- The type of syntax argument map was corrected to pointer type, and map_len was newly added.
4	4.2.20	- Function explanation was corrected.

•Version 3.00 Draft June 12th 2002 Open to consortium members

The parts to which changes have been made are as follows:

	Changed part (section number in the Table of Contents)	<Summary of additions/changes>
1	1.2	Addition of new transmission media in Fig. 1.1.
2	2.1	Addition of "Lower-layer communication software address table data request," "Master router notification" and "Hardware address data request."
3	2.2	Addition of explanation of lower-layer communication software address table data request, master router notification and hardware address data request.
4	3.1	Addition of "Lower-layer communication software address table data request," "Master router notification" and "Hardware address data request" in Table 3.1.
5	3.2	Addition of explanation of lower-layer communication software address table data request, master router notification and hardware address data request.
6	4.1	Addition of "LowGetAddressTableData," "LowSet MasterRouterFlag" and "LowGetHardwareAddress" in Table 4.1.
7	4.2.22	Addition of explanation of LowGetAddressTableData.
8	4.2.23	Addition of explanation of LowSetMasterRouterFlag.
9	4.2.24	Addition of explanation of LowGetHardwareAddress.

•Version 3.00 August 29th 2002 Open to consortium members

•Version 3.10Draft November 8th 2002 Open to consortium members

The parts to which changes have been made are as follows:

	Changed part (section number in the Table of Contents)	Summary of additions/changes
1	2.2	Addition of "(23) Lower-layer communication software address table data size acquisition."

2	4.1	* Correction of incorrect descriptions. * Correction of the table number of Table 4.1. * Correction of item numbers in Table 4.1. * Addition of "LowReceiveStop" in Table 4.1.
3	4.2.22	* Addition of the LowReceiveStop function.

- Version 3.10 December 18th 2002 Open to consortium members.
- Version 3.11 March 7th 2003 Open to consortium members
- Version 3.12 May 22nd 2003 Open to consortium members

The parts to which changes have been made are as follows:

	Changed part (section number in the Table of Contents)	Summary of additions/changes
1	3.1 3.2	* Addition of "Lower-layer communication software address table data size acquisition."

- Version 3.21Draft October 17th 2003 Open to consortium members
- Version 3.20 December 12th 2006 Open to the Public.

The parts to which changes have been made are as follows:

	Changed part (section number in the Table of Contents)	Summary of additions/changes
1	2.1	* Addition of "(27) ECHONET MAC address list acquisition," "(28) Master router data acquisition" and "(29) Hardware address conversion request."
2	2.2	* Addition of explanation of ECHONET MAC address list acquisition, master router data acquisition and hardware address conversion request.
3	3.1	* "No.5 warm start request" and "No.11 Transmission result acquisition" in Table 3.1 were changed from "Required" to "Optional." * Addition of "No.27 ECHONET MAC address list acquisition" and "No.28 Master router data acquisition" in Table 3.1. Addition of explanation of hardware address conversion request.
4	3.2	* (1) Correction of incorrect descriptions. * (2) Deletion of sdbuf and rdbuf. * (17) Correction of the table number of Table 3.18. * (23) Correction of the table number of Table 3.23. * (24) Correction of a number overlap. * Addition of (27), (28) and (29).
5	4.1	* Correction of incorrect descriptions. * Correction of an incorrect description in No.7 of Table 4.1. * "No. 6 Suspend request operation ," "No.7 Operation restart request," "No.11 Transmission result acquisition ," "No.12 Transmission cancellation request " and "No.15 Address data setting request" were changed from "Required" to "Optional" in Table 4.1. * Addition of "No.27 ECHONET MAC address list acquisition" and "No.28 Master router data acquisition" in Table 4.1. Addition of "No.29 Hardware address conversion request."
6	4.2.8	* Correction of incorrect descriptions.
7	4.2.18	* Correction of incorrect descriptions.
8	4.2.27	* Addition of the "LowGetEchonetMACList" function.
9	4.2.28	* Addition of the "LowGetMasterRouterInfo" function.

10	4.2.29	* Addition of the “LowReqToHardwareAddress” function.
----	--------	---

- Version 3.30 December 2nd 2004 Open to consortium members

The parts to which changes have been made are as follows:

	Changed part (section number in the Table of Contents)	Summary of additions/changes
1	1.2	• Explanations about IEEE802.11/11b were added in “Fig. 1.1 Positioning of Individual Lower-layer Communication Interfaces.”
2	4.2	• IEEE802.11/11b-related explanations were added to the explanations about device_id in 4.2.1 to 4.2.29.
3	4.3	• A new subsection entitled “4.3.8 Initialization parameter specifications for IEEE802.11/11b lower-layer communication software” was added.

- Version 3.40 Draft December 28th 2004 Open to consortium members

The parts to which changes have been made are as follows:

	Changed part (section number in the Table of Contents)	Summary of additions/changes
1	1.2	• Explanations about the Power Line Communication Protocol C and D Systems were added in “Fig. 1.1 Positioning of Individual Lower-layer Communication Interfaces.” • The layout was changed.
2	4.2	• “Power Line Communication Protocol System” in the explanations about device_id in 4.2.1 to 4.2.29 were changed to “Power Line Communication Protocol A and D Systems.” Explanations about the Power Line Communication Protocol C System were newly added.

- Version 3.40 February 3rd 2005 Open to consortium members
- Version 3.41 May 11th 2005 Open to consortium members
- Version 3.2 October 13th 2005 Open to the public
- Version 3.42 October 27th 2005 Open to consortium members

The parts to which changes have been made are as follows:

	Changed part (section number in the Table of Contents)	Summary of additions/changes
1	4.2.8	• Explanations about structures to be used were added.
2	4.2.18	• Explanations about arguments were changed.

- Version 3.50 Draft August 3rd 2006 Open to consortium members
- Version 3.5 September 20th 2006 Open to consortium members
- Version 3.51 Draft February 2nd 2007 Open to consortium members

- Version 3.60 March 5th 2007 Open to consortium members
 December 11th 2007 Open to the public

The specifications published by the ECHONET Consortium are established without regard to industrial property rights (e.g., patent and utility model rights). In no event will the ECHONET Consortium be responsible for industrial property rights to the contents of its specifications.

The publisher of this specification is not authorized to license and/or exempt any third party from responsibility for JAVA, IrDA, Bluetooth or HBS.

A party who intends to use JAVA, IrDA, Bluetooth or HBS should take action in being licensed for above-mentioned specifications.

In no event will the publisher of this specification be liable to you for any damages arising out of use of this specification.

The original language of The ECHONET Specification is Japanese. The English version of the Specification was translated the Japanese version. Queries in the English version should be referred to the Japanese version.

Contents

Chapter 1 Overview	1-1
1.1 Basic Concept.....	1-1
1.2 Positioning on Communication Layers.....	1-2
Chapter 2 ECHONET Individual Lower-Layer Communication Interface Function Specifications2-1	
2.1 List of ECHONET Individual Lower-Layer Communication Interface Functions .	2-1
2.2 ECHONET Individual Lower-Layer Communication Interface Detailed Specifications.....	2-3
Chapter 3 Level 1 ECHONET Individual Lower-Layer Communication Interface Specifications3-1	
3.1 List of Level 1 ECHONET Individual Lower-Layer Communication Interface Services	3-1
3.2 Detailed Specifications for Level 1 ECHONET Individual Lower-Layer Communication Interface Services	3-3
Chapter 4 Level 2 ECHONET Individual Lower-Layer Communication Interface Specifications4-1	
4.1 List of Level 2 ECHONET Individual Lower-Layer Communication Interfaces...	4-2
4.2 Level 2 ECHONET Individual Lower-Layer Communication Interface Detail Specifications.....	4-4
4.2.1 LowGetDevID	4-5
4.2.2 LowInit.....	4-6
4.2.3 LowRequestRun.....	4-8
4.2.4 LowSetTrouble	4-9
4.2.5 LowStart	4-10
4.2.6 LowSuspend	4-11
4.2.7 LowWakeup.....	4-13
4.2.8 LowGetProData.....	4-14
4.2.9 LowGetStatus	4-17
4.2.10LowSendData.....	4-19

4.2.11 LowGetSendResult	4-22
4.2.12 LowSendCancel	4-23
4.2.13 LowReceiveData	4-24
4.2.14 LowGetAddress	4-25
4.2.15 LowSetAddress	4-26
4.2.16 LowReqToMac	4-27
4.2.17 LowReqToID	4-28
4.2.18 LowReqBcastID	4-29
4.2.19 LowInitAll	4-31
4.2.20 LowStop	4-33
4.2.21 LowHalt	4-35
4.2.22 LowReceiveStop	4-37
4.2.23 LowGetAddressTableDataSize	4-38
4.2.24 LowGetAddressTableData	4-39
4.2.25 LowSetMasterRouterFlag	4-41
4.2.26 LowGetHardwareAddress	4-43
4.2.27 LowGetEchonetMACList	4-45
4.2.28 LowGetMasterRouterInfo	4-47
4.2.29 LowReqToHardwareAddress	4-49
4.3 Initial Setting Information Specifications	4-51
4.3.1 Initialization parameter specifications for lower-layer communication software for the Power Line Communication Protocol A, C and D Systems	4-52
4.3.2 Initialization parameter specifications for specific low-power RF lower-layer communication software	4-52
4.3.3 Initialization parameter specifications for extended HBS lower-layer communication software	4-52
4.3.4 Initialization parameter specifications for IrDA-dependent lower-layer communication software	4-52
4.3.5 Initialization parameter specifications for LonTalk®-dependent lower-layer communication software	4-53
4.3.6 Initialization parameter specifications for Bluetooth™ lower-layer communication software	4-53
4.3.7 Initialization parameter specifications for Ethernet lower-layer communication software	4-53
4.3.8 Initialization parameter specifications for IEEE802.11/11b lower-layer communication software	4-53

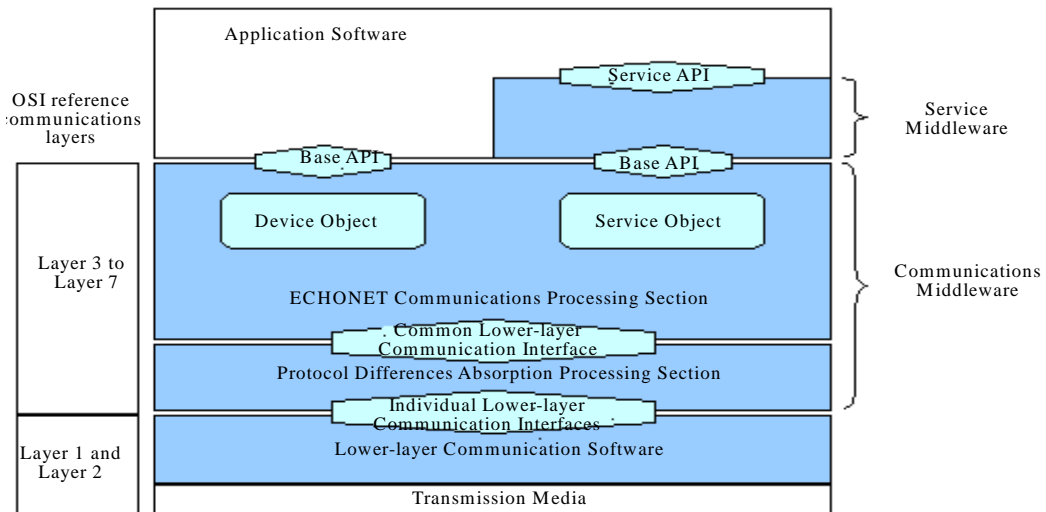
Chapter 1 Overview

1.1 Basic Concept

Part 6 provides specifications for the software interface that implements the processes and information exchanges performed between a Protocol Difference Absorption Processing Block and lower-layer communication software, which are illustrated in Fig. 1.1 on the next page. The Protocol Difference Absorption Processing Block can exchange information with the lower-layer communication software via a discrete lower-layer communication interface. The specifications for the discrete lower-layer communication interface deal with interface services that are to be supported by the lower-layer communication software. They define discrete lower-layer communication interface specifications Levels 1 and 2. Level 1 stipulates input/output data items; Level 2, the use of functions in situations in which a particular language is specified. Discrete lower-layer communication interface specifications Levels 1 and 2 are based on the concepts of Basic API Levels 1 and 2.

1.2 Positioning on Communication Layers

The shaded area in Fig. 1.1 illustrates the positioning of the discrete lower-layer communication interface. This interface is positioned between the Protocol Difference Absorption Processing Block and the lower-layer communication software and implements processing calls and information exchange, thereby connecting the communication middleware and the lower-layer communication software.



Lower-layer Communication Software Supported by the Current Version

Symbol	Name of Lower-layer Communication Software	Transmission Medium
A	Power Line Communication Protocol A System Power Line Communication Protocol D System	Power distribution lines
B	Low-power Radio	Low-power radio
C	Extended HBS	Twisted-pair cables
D	IrDA Control	Infrared
E	LonTalk®	Low-power radio
F	Bluetooth® (UDP/IP)	Low-power radio (BT)
G	Ethernet IEEE802.3 (UDP/IP)	Ethernet
H	IEEE802.11 IEEE802.11b (UDP/IP)	Low-power radio (WLAN)
I	Power Line Communication Protocol C System	Power distribution lines

LonTalk® is a registered trademark of the Echelon Corporation that is used in the United States and other countries.

Bluetooth® is a registered trademark of Bluetooth SIG, Inc.

Ethernet is a registered trademark of the Xerox Corporation.

All other trademarks are properties of their respective owners

Fig. 1.1 Positioning of Discrete Lower-Layer Communication Interface

Chapter 2 ECHONET Individual Lower-Layer Communication Interface Function Specifications

2.1 List of ECHONET Individual Lower-Layer Communication Interface Functions

The table below lists the ECHONET discrete lower-layer communication interface functions supported by the lower-layer communication software. The lower-layer communication software shall be provided with these interface functions. The individual interface functions are detailed in the next section.

- (1) Request for lower-layer communication software type information
- (2) Request for initialization
- (3) Request for operation start
- (4) Fault notice
- (5) Request for warm start
- (6) Request for suspension
- (7) Request for operation restart
- (8) Profile acquisition
- (9) Status acquisition
- (10) Request for data transmission
- (11) Transmission result acquisition
- (12) Request for transmission stop
- (13) Request for received data
- (14) Address information acquisition
- (15) Address information setting
- (16) Request for physical address translation
- (17) Request for node ID translation
- (18) Request for broadcast destination acquisition
- (19) Request for complete initialization
- (20) Request for communication stop
- (21) Request for complete stop
- (22) Stop notice
- (23) Lower-layer communication software address table data size request
- (24) Lower-layer communication software address table data request
- (25) Master router notification
- (26) Hardware address data request
- (27) ECHONET MAC address list acquisition
- (28) Master router data acquisition

(29) Hardware address conversion request

2.2 ECHONET Individual Lower-Layer Communication Interface Detailed Specifications

The ECHONET discrete lower-layer communication interface functions supported by the lower-layer communication software are described below. For the lower-layer communication software state transitions mentioned in this section, see the explanation of the associated lower-layer communication software in Part 3.

(1) Request for mounting information

Requests information about the lower-layer communication software (the number of mounted lower-layer communication software programs and their IDs).

(2) Request for initialization

Requests that lower-layer communication software effect initialization by performing a cold start and switch to the communication stop state. Here, the MAC address retained by the lower-layer communication software is discarded/updated.

(3) Request for operation start

Requests that lower-layer communication software switch from the communication stop state to the normal operation state.

(4) Fault notice

Reports a fault (abnormality) in a layer higher than the Protocol Difference Absorption Processing Block.

(5) Request for warm start

Requests that lower-layer communication software effect initialization by performing a warm start and then switch to the communication stop state. Here, the MAC address retained by the lower-layer communication software remains unchanged.

(6) Request for suspension

Requests that lower-layer communication software switch from the normal operation state to the suspension state.

(7) Request for operation restart

Requests that lower-layer communication software exit the suspension state and enter the normal operation state.

- (8) Profile acquisition
Requests that lower-layer communication software furnish profile data. The profile data requested by this function consists of static information about the lower-layer communication software, such as the software development manufacturer code and version number.
- (9) Status acquisition
Requests that lower-layer communication software furnish status data. The status data requested by this function consists of dynamic information about the lower-layer communication software, such as information about abnormalities and processing status.
- (10) Request for data transmission
Requests that lower-layer communication software transmit specified ECHONET data.
- (11) Transmission result acquisition
Requests that lower-layer communication software furnish information about the processing status of a data transmission requested immediately before this request.
- (12) Request for transmission stop
Requests that lower-layer communication software stop a data transmission process performed in compliance with the data transmission request issued immediately before this request.
- (13) Request for received data
Requests that the lower-layer communication software deliver received data.
- (14) Address information acquisition
Requests that the lower-layer communication software provide address information, such as recognized MAC addresses or house codes.
- (15) Address information setting
Sets the MAC address and house code information for the lower-layer communication software.
- (16) Request for physical address translation
Delivers node ID information to the lower-layer communication software and requests MAC address of corresponding communications software.

- (17) Request for node ID translation
Delivers a MAC address to the lower-layer communication software and requests the associated node ID.
- (18) Request for broadcast destination acquisition
Delivers to the lower-layer communication software a broadcast target selection code (DEA 2nd byte information for broadcast designation) for situations in which the broadcast type selection code (DEA 1st byte information for broadcast designation) indicates an intra-domain broadcast (0x00) or intra-subnet broadcast (0x01), and requests the broadcast target node ID (value extracted in accordance with a broadcast group selection for each lower-layer communication software program).
- (19) Request for complete initialization
Requests that lower-layer communication software effect initialization by performing a cold start and then switch to the communication stop state. Here, the house code information and MAC address are to be acquired again.
- (20) Request for communication stop
Requests that lower-layer communication software switch to the communication stop state.
- (21) Request for complete stop
Requests that lower-layer communication software switch to the stop state.
- (22) Stop notice
Lower-layer communication software notifies the Protocol Difference Absorption Processing Block that lower-layer communication software has switched to the stop state.
- (23) Lower-layer communication software address table data size acquisition
Acquires the number of lower-layer destination address table data pairs held (lower-layer communication software).
- (24) Lower-layer communication software address table data acquisition
Acquires lower-layer destination address table data held (lower-layer communication software).
- (25) Master router notification
Notifies the lower-layer communication software as to whether or not the home node is a master router.

(26) Hardware address data acquisition

Acquires the hardware address data held (lower-layer communication software).

(27) ECHONET MAC address list acquisition

Acquires the ECHONET MAC address list held (lower-layer communication software).

(28) Master router data acquisition

Acquires the master router data held (lower-layer communication software).

(29) Hardware address conversion request

Requests the hardware address that corresponds to the ECHONET MAC address data delivered to the lower-layer communication software.

Chapter 3 Level 1 ECHONET Individual Lower-Layer Communication Interface Specifications

3.1 List of Level 1 ECHONET Individual Lower-Layer Communication Interface Services

For each service listed in Table 3.1, the Level 1 ECHONET discrete lower-layer communication interface specifications prescribe the data to be exchanged between the Protocol Difference Absorption Processing Block and lower-layer communication software. The input/output data items stipulated in the next section shall be provided for mounting in compliance with the Level 1 ECHONET discrete lower-layer communication interface specifications. However, two or more services may be integrated into a single service, and a single service may be divided into two or more services. Further, two or more data items may be processed as a single data item, and a single data item may be processed as two or more data items.

Table 3.1 List of Level 1 ECHONET Individual Lower-Layer Communication Interfaces (1/2)

No.	API name	Function outline	Mounting specification
1	Request for lower-layer communication software type information	Requests type and ID of lower-layer communication software.	Required
2	Request for initialization	Requests that lower-layer communication software perform cold start for initialization.	Required
3	Request for operation start	Requests that lower-layer communication software start operation.	Required
4	Fault notice	Notifies lower-layer communication software of fault (error) status of high-order layer from Protocol Difference Absorption Processing Block.	Optional
5	Request for warm start	Requests that lower-layer communication software perform warm start for initialization.	Optional
6	Request for suspension	Requests that lower-layer communication software suspend operation.	Optional
7	Request for operation restart	Requests that lower-layer communication software restart operation.	Optional
8	Request for profile data acquisition	Acquires static information about lower-layer communication software.	Required
9	Request for status data acquisition	Acquires dynamic status information about lower-layer communication software (information about process abnormalities, duplicate addresses, etc.).	Required
10	Request for data transmission	Requests that lower-layer communication software transmit data.	Required
11	Transmission result acquisition	Requests that lower-layer communication software provide data transmission result.	Required
12	Request for transmission stop	Requests that lower-layer communication software stop data transmission.	Optional

Table 3.1 List of Level 1 ECHONET Individual Low-layer Communication Interfaces (2/2)

No.	API name	Function outline	Mounting specification
13	Request for received data	Requests that lower-layer communication software deliver received data.	Required
14	Address information acquisition	Requests that lower-layer communication software furnish retained MAC address and house code information.	Required
15	Address information setting	Sets MAC address and house code information for lower-layer communication software.	Optional
16	Request for physical address translation	Delivers node ID to lower-layer communication software and requests associated MAC address.	Optional
17	Request for node ID translation	Delivers MAC address to lower-layer communication software and requests associated node ID.	Optional
18	Request for broadcast destination acquisition	Requests that lower-layer communication software furnish node ID targeted for broadcast.	Optional
19	Request for complete initialization	Requests that lower-layer communication software perform cold start for initialization. Here, house code information is acquired again.	Optional
20	Request for communication stop	Requests that lower-layer communication software enter the communication stop state.	Optional
21	Request for complete stop	Requests that lower-layer communication software enter the stop state.	Optional
22	Stop notice	Lower-layer communication software notifies the Protocol Difference Absorption Processing Block that lower-layer communication software has switched to the stop state.	Optional
23	Lower-layer communication software address table data size acquisition	Acquires the number of lower-layer address table data pairs held (lower-layer communication software).	Optional
24	Lower-layer communication software address table data acquisition	Acquires lower-layer address table data held (lower-layer communication software).	Optional
25	Master router notification	Notifies the lower-layer communication software as to whether or not the home node is a master router.	Optional
26	Hardware address data acquisition	Acquires the hardware address data held (lower-layer communication software).	Optional
26	ECHONET MAC address list acquisition	Acquires the ECHONET MAC address list held (lower-layer communication software).	Optional
27	Master router data acquisition	Acquires the master router data held (lower-layer communication software).	Optional
28	Hardware address conversion request	Requests the hardware address that corresponds to the ECHONET MAC address data delivered (lower-layer communication software).	Optional

3.2 Detailed Specifications for Level 1 ECHONET Individual Lower-Layer Communication Interface Services

This section stipulates the data that is input or output by the various services indicated in Table 3.1 in the previous section. In the following tables, references to data input/output direction are made relative to the Protocol Difference Absorption Processing Block. More specifically, the term “input” denotes the transfer of data from the Protocol Difference Absorption Processing Block to the lower-layer communication software, and the term “output” indicates the transfer of data from that lower-layer communication software to the Protocol Difference Absorption Processing Block. When these data transfer operations can be performed, the Level 1 ECHONET discrete lower-layer communication interface specifications are complied with. The data transfer method (use of a structure, delivery of data exchange buffer pointer information, etc.) is not stipulated here.

Furthermore, the Level 1 ECHONET discrete lower-layer communication interface provides data input/output that remains compliant with these specifications even with different types of lower-layer communication software. Therefore, the argument for indicating the lower-layer communication software type is set as an input for interface services other than the request for lower-layer communication software type information. However, it is always handled as an optional argument because the lower-layer communication software type need not be specified for normal communication devices (in which no more than one lower-layer communication software program exists).

- (1) Request for lower-layer communication software (mandatory function for mounting)
Requests type of lower-layer communication software (power line, low-power RF, etc.).
Table 3.2 shows the input/output specifications.

Table 3.2 List of Lower-Layer Communication Software Type Request API Input/Output Data

Direction	Data name	Contents and condition	Remarks
Input	–		
Output	device_id	<ul style="list-style-type: none"> - Indicates the type of lower-layer communication software. - The power line lower-layer communication software, specific low-power RF lower-layer communication software, extended HBS lower-layer communication software, LonTalk®-dependent lower-layer communication software, IrDA-dependent lower-layer communication software, and other similar software shall be distinguishable. 	Required
Output	Return Value	TRUE: <Acquisition successfully completed FALSE: Acquisition failed>	Optional

(2) Request for initialization (mandatory function for mounting)

Requests that lower-layer communication software perform a cold start for initialization in accordance with specified information and then switch to the communication stop state. Within a series of processes performed in compliance with this request, the MAC address information is acquired again. When the lower-layer communication software has house code information, the house code information remains unchanged. Table 3.3 shows the input/output specifications.

Table 3.3 Input/Output Data List for Initialization Request Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Input	sfholdtime	- Information about maximum retention time for outgoing data. - Maximum time during which lower-layer communication software can retain outgoing data. Data will be discarded if not transmitted within this time.	Optional
Input	rfholdtime	- Information about maximum retention time for incoming data. - Maximum time during which lower-layer communication software can retain received data. Data will be discarded if not delivered to Protocol Difference Absorption Processing Block within this time.	Optional
Input	sfbuf	- Indicates buffer size, buffer location, and other information for transmitted data that lower-layer communication software receives from Protocol Difference Absorption Processing Block.	Optional
Input	rfbuf	- Indicates buffer size, buffer location, and other information for received data that lower-layer communication software delivers to Protocol Difference Absorption Processing Block.	Optional
Input	snbuf	- Indicates buffer size, buffer location, and other relevant information for transmitted data between lower-layer communication software and communication medium.	Optional
Input	rnbuf	- Indicates buffer size, buffer location, and other relevant information for received data between lower-layer communication software and communication medium.	Optional
Input	low_mode	- Indicates special mode selection of lower-layer communication software, such as test mode or networked data monitoring mode.	Optional
Input	mac_ad	- Indicates MAC address to be set for lower-layer communication software.	Optional
Input	mac_len	- Indicates information about size of MAC address to be set for lower-layer communication software.	Optional
Input	housecode	- Indicates house code information to be set for lower-layer communication software.	Optional
Input	housecode_len	- Indicates information about size of house code information to be set for lower-layer communication software.	Optional
Input	lowinit	- Indicates initialization parameter, which differs for each lower-layer communication software program	Optional
Output	Return Value	TRUE: Successful initialization, FALSE: Failed initialization	Optional

(3) Request for operation start (mandatory function for mounting)

Requests that lower-layer communication software start operation. Table 3.4 shows the input/output specifications.

Table 3.4 Input/Output Data List for Operation Start Request Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Output	Return Value	TRUE: Successful operation start, FALSE: Failed operation start	Optional

(4) Fault notice

Notifies the lower-layer communication software of fault (error) status of high-order layer from the Protocol Difference Absorption Processing Block. Table 3.5 shows the input/output specifications.

Table 3.5 Input/Output Data List for Fault Notice Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Input	htrouble_no	- Reports a number indicating type of trouble (abnormal state).	Required
Output	Return Value	TRUE: Successful notice, FALSE: Failed notice	Optional

(5) Request for warm start

Requests that lower-layer communication software perform a warm start for initialization and then switch to the communication stop state. Within a series of processes performed in compliance with this request, the house code information and MAC address information remain unchanged. Table 3.6 shows the input/output specifications.

Table 3.6 Input/Output Data List for Warm Start Request Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Output	Return Value	TRUE: Warm start request accepted, FALSE: Request denied	Optional

(6) Request for suspension

Requests that lower-layer communication software enter the suspension state. Table 3.7 shows the input/output specifications.

Table 3.7 Input/Output Data List for Suspension Request Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Output	Return Value	TRUE: Suspension acceptable, FALSE: Not acceptable	Optional

(7) Request for operation request

Requests that lower-layer communication software exit the suspension state and enter the normal operation state. Table 3.8 shows the input/output specifications.

Table 3.8 Input/Output Data List for Operation Restart Request Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Output	Return Value	TRUE: Successful restart, FALSE: Restart disabled (including failure)	Optional

(8) Request for profile data acquisition (mandatory function for mounting)

Requests profile data for lower-layer communication software. Profile data requested by this service consists of static information about the lower-layer communication software, such as the software development manufacturer code and version number. Table 3.9 shows the input/output specifications.

Table 3.9 Input/Output Data List for Profile Acquisition Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Output	kind	- Lower-layer communication software identification information. - Power line, low-power RF, extended HBS, IrDA control, and LonTalk [®] lower-layer communication software programs shall be distinguishable.	Required
Output	mac_ad	- Indicates retained MAC address.	Required
Output	housecode	- Indicates retained house code information.	Required
Output	version_No	- Indicates version information about lower-layer communication software.	Optional
Output	maker	- Indicates manufacturer code.	Optional
Output	srlen	- Indicates transmittable/receivable data length.	Optional
Output	broad	- Indicates whether or not broadcast function is enabled.	Optional
Output	baud	- Indicates baud rate.	Optional
Output	chmac_info	- Indicates information about MAC address-to-node ID translation (e.g., translation function address information).	Optional
Output	chnode_info	- Indicates information about node ID-to-MAC address translation (e.g., translation function address information).	Optional
Output	chbroad_info	- Indicates information about node ID-to-broadcast destination MAC address translation (e.g., translation function address information).	Optional
Output	Return Value	TRUE: Normal, FALSE: Error	Optional

(9) Request for status data acquisition (mandatory function for mounting)

Requests that lower-layer communication software furnish its status data. The status data requested by this service consists of dynamic information about the lower-layer communication software, such as information about abnormalities and processing status. Table 3.10 shows the input/output specifications.

Table 3.10 Input/Output Data List for Status Acquisition Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Output	state	- State transition information about lower-layer communication software. - States defined for each lower-layer communication software program in Part 3 shall be distinguishable.	Required
Output	upper_trouble	- Information recognized as a high-order layer fault	Optional
Output	low_trouble	- Indicates information about recognized trouble in lower-layer communication software.	Optional
Output	low_mode	- Indicates information about operation mode (monitoring mode, test mode, etc.).	Optional
Output	Return Value	TRUE: Normal, FALSE: Error	Optional

(10) Request for data transmission (mandatory function for mounting)

Requests that lower-layer communication software send specified ECHONET data. Table 3.11 shows the input/output specifications.

Table 3.11 Input/Output Data List for Data Transmission Request Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Input	send_data	- Information about requested outgoing data in ECHONET data format. - Uses format acceptable between protocol difference absorption processing blocks.	Required
Input	d_add	- Indicates MAC address of intra-subnet transmission destination.	Required
Input	mac_len	- Indicates size of MAC address of intra-subnet transmission destination.	Optional
Input	broad	- Indicates whether broadcast or individual transmission is selected.	Optional
Output	Return Value	TRUE: Normal, FALSE: Error	Optional

(11) Request for transmission result acquisition

Requests data received by lower-layer communication software. Table 3.12 shows the input/output specifications.

Table 3.12 Input/Output Data List for Transmission Result Acquisition Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Output	result	- Shows result to indicate whether transmission is in progress, ended normally, ended abnormally, or canceled.	Required
Output	Return Value	TRUE: Normal, FALSE: Error	Optional

(12) Request for transmission stop

Requests that lower-layer communication software stop data transmission process currently being executed. Table 3.13 shows the input/output specifications.

Table 3.13 Input/Output Data List for Transmission Stop Request Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Output	Return Value	TRUE: Successful stop, FALSE: Failure to stop (already transmitted)	Optional

(13) Request for data reception (mandatory function for mounting)

Requests data received by the lower-layer communication software. Table 3.14 shows the input/output specifications.

Table 3.14 Input/Output Data List for Received Data Request Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Output	receive_data	- Indicates received data in ECHONET data format. - Uses format acceptable between protocol difference absorption processing blocks.	Required
Output	s_add	- Indicates MAC address of intra-subnet transmission source.	Required
Output	mac_len	- Indicates size of MAC address of intra-subnet transmission source.	Optional
Output	Return Value	TRUE: Normal, FALSE: Error (error indication code such as "No received data")	Optional

- (14) Request for address information acquisition (mandatory function for mounting)
Requests address information retained by lower-layer communication software. Table 3.15 shows the input/output specifications.

Table 3.15 Input/Output Data List for Address Information Acquisition Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Output	mac_ad	- Indicates retained MAC address.	Required
Output	mac_len	- Indicates size of MAC address.	Optional
Output	houscode	- Indicates retained house code information.	Optional
Output	houscode_len	- Indicates size of house code information.	Optional
Output	Return Value	TRUE: Normal; FALSE: Error (error indication code such as "node ID not set" or "Specified device_id error")	Optional

- (15) Request for address information setup
Sets address information for lower-layer communication software. Table 3.16 shows the input/output specifications.

Table 3.16 Input/Output Data List for Address Setup Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Input	mac_ad	- Indicates MAC address to be set.	Required
Input	mac_len	- Indicates size of MAC address.	Optional
Input	houscode	- Indicates house code information to be set.	Optional
Input	houscode_len	- Indicates size of house code information.	Optional
Output	Return Value	TRUE: Normal, FALSE: Error (error indication code such as "Set disable")	Optional

(16) Request for physical address translation

Delivers a node ID to the lower-layer communication software and requests the corresponding MAC address for the associated lower-layer communication software.

Table 3.17 shows the input/output specifications.

Table 3.17 Input/Output Data List for Physical Address Translation Request Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Input	node_id	- Indicates node ID to be translated.	Required
Output	mac_ad	- Indicates MAC address corresponding to specified node ID.	Required
Output	mac_len	- Indicates size of MAC address.	Optional
Output	Return Value	TRUE: Normal, FALSE: Error (error indication code such as "Translate disable")	Optional

(17) Request for node ID translation

Delivers a MAC address to the lower-layer communication software and requests the associated node ID (value translated according to the translation rule specific to the lower-layer communication software). Table 3.18 shows the input/output specifications.

Table 3.18 Input/Output Data List for Node ID Translation Request Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Input	mac_ad	- Indicates MAC address to be translated.	Required
Output	mac_len	- MAC address size information.	Optional
Output	node_id	- Indicates node ID corresponding to specified MAC address.	Required
Output	Return Value	TRUE: Normal, FALSE: Error (error indication code such as "Translate disable")	Optional

(18) Request for broadcast destination acquisition

Delivers to the lower-layer communication software a broadcast target selection code (DEA 2nd byte information for broadcast designation) for situations in which the broadcast type selection code (DEA 1st byte information for broadcast designation) indicates an intra-domain broadcast (0x00) or intra-subnet broadcast (0x01), and requests the broadcast target node ID (value extracted in accordance with a broadcast group selection for each lower-layer communication software program). Table 3.19 shows the input/output specifications.

Table 3.19 Input/Output Data List for Broadcast Destination Acquisition Request Service

Direction	Data name	Contents and condition	Remarks
Input	device_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Optional
Input	broad_adinfo	- Indicates code that specifies broadcast target.	Required
Output	node_num	- Indicates number of node IDs targeted for broadcast.	Required
Output	node_idinfo	- Presents information about node IDs targeted for broadcast.	Required
Output	Return Value	TRUE: Normal, FALSE: Error (error indication code such as "Translate disable")	Optional

(19) Request for complete initialization

Requests that lower-layer communication software perform a cold start for initialization and then switch to the communication stop state. Within a series of processes performed in compliance with this request, the house code information and MAC address information are acquired again.

Table 3.20 Input/Output Data List for Complete Initialization Request Service

Direction	Data name	Contents and condition	Remarks
Input	software_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Required
Input	p_init	- Specifies initialization parameters. - Parameters include outgoing data maximum retention time and incoming data maximum retention time, but vary with that lower-layer communication software be initialized.	Required
Output	Return Value	TRUE: Successful initialization, FALSE: Failed initialization	Optional

(20) Request for communication stop

Requests that lower-layer communication software switch to the communication stop state. Table 3.21 shows the input/output specifications.

Table 3.21 Input/Output Data List for Communication Stop Request Service

Direction	Data name	Contents and condition	Remarks
Input	software_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Required
Output	Return Value	TRUE: Request accepted, FALSE: Request denied	Optional

(21) Request for complete stop

Requests that lower-layer communication software switch to the stop state. Table 3.22 shows the input/output specifications.

Table 3.22 Input/Output Data List for Complete Stop Request Service

Direction	Data name	Contents and condition	Remarks
Input	software_id	- Specifies lower-layer communication software ID obtained through lower-layer communication software type request service.	Required
Output	Return Value	TRUE: Request accepted, FALSE: Request denied	Optional

(22) Stop notice

Notifies the ECHONET Communications Processing Block that lower-layer communication software has switched to the stop state. Table 3.23 shows the input and output specifications.

Table 3.23 Stop Notice Service Input/Output Data

Direction	Data name	Contents and condition	Remarks
Output	software_id	• Indicates the lower-layer communication software that has switched to the stop state.	Required
Input	Return Value	TRUE: notice received, FALSE: notice cannot be received	Optional

(23) Lower-layer communication software address table data size acquisition (Optional)

Acquires the number of data pairs held by the lower-layer address table (lower-layer communication software). Table 3.24 shows the input and output specifications.

Table 3.24 Input and Output Data for the Lower-Layer Communication Software Address Table Data Size Acquisition Service

Direction	Data name	Content and conditions	Remark
Input	device_id	* Specifies the lower-layer communication software ID acquired by the lower-layer communication software type request service.	Optional

Output	data_number	* Indicates the number of address table pairs in the lower-layer address table data.	Required
Output	Return Value	TRUE: normal, FALSE: abnormal	Optional

(24) Lower-layer communication software address table data acquisition (Optional)

Acquires the lower-layer address table data held (lower-layer communication software). The output data consists of the number of data pairs and the data pairs. Each data pair contains hardware address data, MAC address data and a flag indicating whether or not the home node is a master router.

Table 3.25 Input and Output Data for the Lower-Layer Communication Software Address Table Data Acquisition Service

Direction	Data name	Content and conditions	Remark
Input	device_id	* Specifies the lower-layer communication software ID acquired by the lower-layer communication software type request service.	Optional
Output	data_number	* Indicates the number of address table pairs in the lower-layer address table data.	Required
Output	ListOfHardwareaddress	* Gives a list of the hardware addresses stored in the address table of the lower-layer communication software.	Required
Output	ListOfMac_ad	* Gives a list of the MAC addresses stored in the address table of the lower-layer communication software.	Required
Output	ListOfMasterRouter_Flag	* List of identifiers that indicate whether or not the nodes corresponding to the addresses stored in the address table of the lower-layer communication software are master routers. If a node is a master router, "1" is given. Otherwise, "0" is given.	Required
Output	Return Value	TRUE: normal, FALSE: abnormal	Optional

(25) Master router notification (Optional)

Notifies the lower-layer communication software as to whether or not the home node is a master router.

Table 3.26 Input and Output Data for the Master Router Notification Service

Direction	Data name	Content and conditions	Remarks
Input	device_id	* Specifies the lower-layer communication software ID acquired by the lower-layer communication software type request service.	Optional
Input	masterRouter_Flag	* If the node is a master router, "1" is specified. Otherwise, "0" is specified.	Required
Output	Return Value	TRUE: normal, FALSE: abnormal	Optional

(26) Hardware address data acquisition (Optional)

Acquires the hardware address data held (lower-layer communication software).

Table 3.27 Input and Output Data for the Hardware Address Data Acquisition Service

Direction	Data name	Content and conditions	Remarks
Input	device_id	* Specifies the lower-layer communication software ID acquired by the lower-layer communication software type request service.	Optional
Output	hardwareaddress	* Indicates the hardware address of the home node.	Required
Output	Return Value	TRUE: normal, FALSE: abnormal	Optional

(27) ECHONET MAC address list acquisition

Acquires the ECHONET MAC address list held (lower-layer communication software).

Table 3.28 Input and Output Data for the ECHONET MAC Address List Acquisition Service

Direction	Data name	Content and conditions	Remark
Input	device_id	* Specifies the lower-layer communication software ID acquired by the lower-layer communication software type request service.	Optional
Output	mac_addr_list	* Gives a list of the MAC addresses held (lower-layer communication software).	Required

(28) Master router data acquisition

Acquires the master router list held (lower-layer communication software).

Table 3.29 Input and Output Data for the Master Router Data Acquisition Service

Direction	Data name	Content and conditions	Remark
Input	device_id	* Specifies the lower-layer communication software ID acquired by the lower-layer communication software type request service.	Optional
Output	result,	* Indicates whether or not a master router is present	Required
Output	master_node_id	* Master router's NodeID	

(29) Hardware address conversion request

Requests the hardware address that corresponds to the NodeID delivered to the lower-layer communication software.

Table 3.30 Input and Output Data for the Hardware Address Conversion Request Service

Direction	Data name	Content and conditions	Remarks
Input	device_id	* Specifies the lower-layer communication software ID acquired by the lower-layer communication software type request service.	Optional
Input	mac_len	* Indicates the length of the MAC address to be converted.	Required

Input	mac	* Pointer to the MAC address to be converted.	
Output	hardwareaddress	* Indicates the pointer to the hardware address.	Required
Output	hardwareaddress_1 en	* Indicates the pointer to the hardware address size.	Required

Chapter 4 Level 2 ECHONET Individual Lower-Layer Communication Interface Specifications

This section provides the API detailed specifications in consideration of the interchangeability of software to be developed using this interface as the Level 2 ECHONET discrete lower-layer communication interface. The requirements set forth in this chapter presume that the API process is implemented in the lower-layer communication software (the Protocol Difference Absorption Processing Block calls a lower-layer communication software process).

ECHONET Standard Version 2.10 states the Level 2 ECHONET discrete lower-layer communication interface specifications for the ANSI standard C language (hereinafter referred to as the C language).

4.1 List of Level 2 ECHONET Individual Lower-Layer Communication Interfaces

The following 22 functions are stipulated as the Level 2 ECHONET discrete individual lower-layer communication interface functions for the C language. The term “Optional” for Level 2 indicates that the associated function need not be mounted at all times. However, if the capability of such a function is implemented for compliance with Level 2 specifications, the function defined in this section shall be implemented.

Table 4.1 List of Level 2 ECHONET Individual Low-layer Communication Interface Functions (1/2)

No.	API name	API function name	Function	Remarks
1	Request for lower-layer communication software type	LowGetDevID	Requests type and ID of lower-layer communication software.	Required
2	Request for initialization	LowInit	Requests initialization of lower-layer communication software.	Required
3	Request for operation start	LowRequestRun	Requests that lower-layer communication software start operation.	Required
4	Fault notice	LowSetTrouble	Notifies lower-layer communication software of fault (error) status of high-order layer from Protocol Difference Absorption Processing Block.	Optional
5	Request for warm start	LowStart	Requests that lower-layer communication software perform a warm start process.	Required
6	Request for suspension	LowSuspend	Requests that lower-layer communication software suspend operation.	Required
7	Request for operation restart	LowWakeup	Requests that lower-layer communication software restart operation.	Required
8	Request for profile data acquisition	LowGetProData	Gets profile data (static information) of lower-layer communication software.	Required
9	Request for status data acquisition	LowGetStatus	Gets dynamic status (processing fault, address redundancy, etc.) of lower-layer communication software.	Required
10	Request for data transmission	LowSendData	Requests that lower-layer communication software send data.	Required
11	Transmission result acquisition	LowGetSendResult	Requests data transmission result from lower-layer communication software.	Required
12	Request for transmission stop	LowSendCancel	Requests that lower-layer communication software stop data transmission.	Required
13	Request for received data	LowReceiveData	Requests that lower-layer communication software exchange received data.	Required
14	Address information acquisition	LowGetAddress	Gets address information, such as MAC addresses or house codes, recognized by lower-layer communication software.	Required
15	Request for address information setup	LowSetAddress	Sets such address information as MAC addresses and house codes for lower-layer communication software.	Required
16	Request for physical address translation	LowReqToMac	Requests translation of node ID into corresponding MAC address.	Optional

Table 4.1 List of Level 2 ECHONET Individual Lower-Layer Communication Interface Functions (2/2)

No.	API name	API function name	Function	Remarks
17	Request for node ID translation	LowReqToID	Requests translation of MAC address into corresponding node ID.	Optional
18	Request for broadcast destination address acquisition	LowReqBcastID	Requests target node ID for broadcast.	Optional
19	Request for complete initialization	LowInitAll	Requests that lower-layer communication software effect initialization and acquire house code information again.	Optional
20	Request for communication stop	LowStop	Requests that lower-layer communication software stop communications.	Optional
21	Request for complete stop	LowHalt	Requests that lower-layer communication software stop completely.	Optional
22	Reception of stop notification	LowReceiveStop	Requests the lower-layer communication software to provide a stop notification	Required
23	Lower-layer communication software address table data size acquisition	LowGetAddressTable DataSize	Acquires the number of lower-layer address table data pairs held (lower-layer communication software).	Optional
24	Lower-layer communication software address table data acquisition	LowGetAddressTable Data	Acquires lower-layer address table data held (lower-layer communication software).	Optional
25	Master router notification	LowSetMasterRouterFlag	Notifies the lower-layer communication software as to whether or not the home node is a master router.	Optional
26	Hardware address data acquisition	LowGetHardwareAddress	Acquires the hardware address data for the home node (lower-layer communication software).	Optional
27	ECHONET MAC address list acquisition	LowGetEchonetMAC List	Acquires the ECHONET MAC address list held (lower-layer communication software).	Optional
28	Master router data acquisition	LowGetMasterRouterInfo	Acquires the master router data held (lower-layer communication software).	Optional
29	Hardware address conversion request	LowReqToHardwareAddress	Requests the hardware address that corresponds to the ECHONET MAC address data delivered (lower-layer communication software).	Optional

4.2 Level 2 ECHONET Individual Lower-Layer Communication Interface Detail Specifications

This section provides detailed specifications for each function shown in Table 4.1 with regard to the following seven items:

- (1) Name
Indicates function name.
- (2) Function
Explains function.
- (3) Syntax
Indicates function syntax.
- (4) Explanation
Provides detailed specifications for arguments and variables.
- (5) Return value
Indicates return value.
- (6) Structure
Specifies structure, if any.
- (7) Notes/restrictions
Indicates notes or restrictions, if any.

4.2.1 LowGetDevID

(1) Name

Lower-layer communication software type request function

(2) Function

Requests lower-layer communication software ID indicating lower-layer communication software type.

(3) Syntax

```
BOOL LowGetDevID (
    unsigned char *device_id /* [OUT] Lower-layer communication software ID */
)
```

(4) Explanation

device_id	: Lower-layer communication software ID
	Power Line Communication Protocol A and D Systems
	0x11 – 0x1F
	Specific low-power RF
	0x31 – 0x3F
	Extended HBS
	0x41 – 0x4F
	IrDA_Control
	0x51 – 0x5F
	LonTalk [®]
	0x61 – 0x6F
	Bluetooth [®]
	0x71 – 0x7F
	Ethernet
	0x81 – 0x8F
	IEEE802.11/11b
	0x91 – 0x9F
	Power Line Communication Protocol C System
	0xA1

(5) Return value

0: Failed acquisition

1: Successful acquisition

(6) Structure

None

(7) Notes/restrictions

It is presumed that this function is called prior to the initialization request function (LowInit) and operation start request function (LowRequestRun).

4.2.2 LowInit

(1) Name

Initialization request function

(2) Function

Requests that lower-layer communication software effect initialization (by performing a cold start) and acquire MAC address again. Upon receipt of this request, the lower-layer communication software performs a cold start to switch to the communication stop state and then sets the initialization parameters for itself.

(3) Syntax

```

BOOL LowInit (
    unsigned char device_id,          /* [IN] Lower-layer communication software ID */
    LOW_INIT_DATA *init_data,       /* [IN] Pointer to initialization parameter (1) */
    void *low_init                   /* [IN] Pointer to initialization parameter (2) */
)

```

(4) Explanation

device_id	: Lower-layer communication software ID
	Power Line Communication Protocol A and D Systems
	0x11 – 0x1F
	Specific low-power RF
	0x31 – 0x3F
	Extended HBS
	0x41 – 0x4F
	IrDA_Control
	0x51 – 0x5F
	LonTalk [®]
	0x61 – 0x6F
	Bluetooth [®]
	0x71 – 0x7F
	Ethernet
	0x81 – 0x8F
	IEEE802.11/11b
	0x91 – 0x9F
	Power Line Communication Protocol C System
	0xA1

*init_data : Pointer to initialization parameter of the common specification item

*low_init : Pointer to initialization parameter, which differs for each lower-layer communication software. Parameter contents are specified for each discrete lower-layer communication software program. (See Section 4.3.)

(5) Return value

0: Failed initialization

1: Successful initialization

(6) Structure

```
typedef struct {
    short          sftohldtime, /* Maximum holding time for transmission data */
    short          rftohldtime, /* Maximum holding time for received data */
    unsigned char  low_mode,    /* Operation mode specifications */
                                     0x00 Normal operation mode
                                     0x01 Test/maintenance mode
                                     (Details are not stipulated.) */
    short          mac_len,     /* MAC address length */
    unsigned char  mac_ad[6],  /* MAC address */
} LOW_INIT_DATA
```

Except for mac_ad[6], when there is no initialization data, set to NULL.

When mac_len is set to NULL, mac_ad[6] is not significant. (When mac_len is NULL, the MAC address is not set.)

(7) Notes/restrictions

If the lower-layer communication software is already in the cold-start or warm-start state, this function returns "Failed initialization".

4.2.3 LowRequestRun

(1) Name

Operation start request function

(2) Function

Requests that lower-layer communication software start operation. Upon receiving this request, the lower-layer communication software starts operation.

(3) Syntax

```
BOOL LowRequestRun (
    unsigned char device_id /* [IN] Lower-layer communication software ID */
)
```

(4) Explanation

device_id	: Lower-layer communication software ID
	Power Line Communication Protocol A and D Systems
	0x11 – 0x1F
	Specific low-power RF
	0x31 – 0x3F
	Extended HBS
	0x41 – 0x4F
	IrDA_Control
	0x51 – 0x5F
	LonTalk [®]
	0x61 – 0x6F
	Bluetooth [®]
	0x71 – 0x7F
	Ethernet
	0x81 – 0x8F
	IEEE802.11/11b
	0x91 – 0x9F
	Power Line Communication Protocol C System
	0xA1

(5) Return value

0: Failure to start

1: Successful start

(6) Structure

None

(7) Notes/restrictions

If the lower-layer communication software is not in the communication stop state, this function returns “Failure to start”.

4.2.4 LowSetTrouble

(1) Name

Fault notice function

(2) Function

Notifies the lower-layer communication software of the fault (error) status of a high-order layer from the Protocol Difference Absorption Processing Block.

(3) Syntax

```

BOOL LowSetTrouble (
    unsigned char device_id, /* [IN] Lower-layer communication software ID */
    char htrouble_no /* [IN] Higher-layer trouble number */
)

```

(4) Explanation

device_id : Lower-layer communication software ID

Power Line Communication Protocol A and D Systems	0x11 – 0x1F
Specific low-power RF	0x31 – 0x3F
Extended HBS	0x41 – 0x4F
IrDA_Control	0x51 – 0x5F
LonTalk [®]	0x61 – 0x6F
Bluetooth [®]	0x71 – 0x7F
Ethernet	0x81 – 0x8F
IEEE802.11/11b	0x91 – 0x9F
Power Line Communication Protocol C System	0xA1

htrouble_no : Trouble No.

-1	Trouble removed
1	Application software error
2	ECHONET Communications Processing Block error
3	Protocol Difference Absorption Processing Block error

(5) Return value

0: Failed notice

1: Successful notice

(6) Structure

None

(7) Notes/restrictions

While an abnormality is reported, the lower-layer communication software performs the following operations:

- Data reception process

Refrains from performing data reception or discards received data.

- Data transmission request from Protocol Difference Absorption Processing Block

Causes an error to be returned.

4.2.5 LowStart

(1) Name

Warm start request function

(2) Function

Requests that lower-layer communication software effect initialization (by performing a warm start) while retaining the MAC address. Upon receipt of this request, the lower-layer communication software performs a warm start and switches to the communication stop state.

(3) Syntax

```

BOOL LowReset (
    unsigned char device_id    /* [IN] Lower-layer communication software ID */
)

```

(4) Explanation

device_id	: Lower-layer communication software ID
	Power Line Communication Protocol A and D Systems
	0x11 – 0x1F
	Specific low-power RF
	0x31 – 0x3F
	Extended HBS
	0x41 – 0x4F
	IrDA_Control
	0x51 – 0x5F
	LonTalk [®]
	0x61 – 0x6F
	Bluetooth [®]
	0x71 – 0x7F
	Ethernet
	0x81 – 0x8F
	IEEE802.11/11b
	0x91 – 0x9F
	Power Line Communication Protocol C System
	0xA1

(5) Return value

0: Failed request

1: Successful request

(6) Structure

None

(7) Notes/restrictions

If the lower-layer communication software is already in the cold-start or warm-start state, this function returns “Failed request”.

When this request is received, the following warm start process is performed:

- Clears transmitting and receiving buffers
- Resets higher-layer fault setup
- Resets various status/work areas
- Resets communication hardware block

4.2.6 LowSuspend

(1) Name

Suspension request function

(2) Function

Requests that lower-layer communication software suspend operation. Upon receipt of this request, the lower-layer communication software switches to the suspension state.

(3) Syntax

```
BOOL LowSuspend (
    unsigned char device_id /* [IN] Lower-layer communication software ID */
)
```

(4) Explanation

device_id	: Lower-layer communication software ID
	Power Line Communication Protocol A and D Systems
	0x11 – 0x1F
	Specific low-power RF
	0x31 – 0x3F
	Extended HBS
	0x41 – 0x4F
	IrDA_Control
	0x51 – 0x5F
	LonTalk [®]
	0x61 – 0x6F
	Bluetooth [®]
	0x71 – 0x7F
	Ethernet
	0x81 – 0x8F
	IEEE802.11/11b
	0x91 – 0x9F
	Power Line Communication Protocol C System
	0xA1

(5) Return value

0: Failed suspension

1: Successful suspension

(6) Structure

None

(7) Notes/restrictions

If the lower-layer communication software is in a state other than normal operation, this function returns “Failed suspension”.

If the lower-layer communication software is in the midst of data transmission when this request is received, it terminates a series of transmission processes and switches to the suspension state. If it is in the midst of data reception, on the other hand, it discards the received data and terminates the process.

The following operations are performed in the suspension state:

- Data reception

No data is to be received.

- Data transmission request from ECHONET communication control processing block

An error is returned.

4.2.7 LowWakeup

(1) Name

Operation restart request function

(2) Function

Requests that lower-layer communication software exit the suspension state. Upon receipt of this request, the lower-layer communication software switches to the normal operation state.

(3) Syntax

```

BOOL LowWakeup (
    unsigned char device_id          /* [IN] lower-layer communication software
                                     type ID */
)

```

(4) Explanation

device_id	: Lower-layer communication software ID
	Power Line Communication Protocol A and D Systems
	0x11 – 0x1F
	Specific low-power RF
	0x31 – 0x3F
	Extended HBS
	0x41 – 0x4F
	IrDA_Control
	0x51 – 0x5F
	LonTalk [®]
	0x61 – 0x6F
	Bluetooth [®]
	0x71 – 0x7F
	Ethernet
	0x81 – 0x8F
	IEEE802.11/11b
	0x91 – 0x9F
	Power Line Communication Protocol C System
	0xA1

(5) Return value

0: Failure to restart

1: Successful restart

(6) Structure

None

(7) Notes/restrictions

If the lower-layer communication software is in a state other than suspension, this function returns “Failure to restart”.

4.2.8 LowGetProData

(1) Name

Profile data acquisition request function

(2) Function

Acquires profile data for lower-layer communication software and a special process function address retained by the lower-layer communication software. Profile data requested by this function consists of property value information for the lower-layer communication software profile class, such as the software development manufacturer name and version number.

(3) Syntax

```

BOOL LowGetProData (
    unsigned char device_id,          /* [IN] Lower-layer communication software ID
                                     */
    LOW_PRO_DATA *pro_data,         /* [OUT] Profile data */
    short (**chmacfunc) (unsigned char node_id, unsigned char *mac),
                                     /* [OUT] Node ID   MAC address translation
                                     function address */
    unsigned char (**chnodefunc) (unsigned char *mac),
                                     /* [OUT] MAC address   Node ID translation
                                     function address */
    void(**broadfunc) (const char bcast, char map[32])
                                     /* [OUT] Broadcast destination acquisition
                                     function address */
)

```

(4) Explanation

device_id	: Lower-layer communication software ID
	Power Line Communication Protocol A and D Systems
	0x11 – 0x1F
	Specific low-power RF
	0x31 – 0x3F
	Extended HBS
	0x41 – 0x4F
	IrDA_Control
	0x51 – 0x5F
	LonTalk [®]
	0x61 – 0x6F
	Bluetooth [®]
	0x71 – 0x7F
	Ethernet
	0x81 – 0x8F
	IEEE802.11/11b
	0x91 – 0x9F
	Power Line Communication Protocol C System
	0xA1

*pro_data : Pointer to profile data structure of lower-layer communication software.

**chmacfunc : Pointer to address of function for translating a node ID to the MAC address specific to the lower-layer communication software is returned. If the lower-layer communication software has a node ID equal to the MAC address or effects simple linear translation, NULL is returned. Specifications for the function arguments to be delivered are as follows:

node_id:	[in] Node ID before translation
mac:	[out] MAC address after translation

This function returns MAC address size (in bytes).

**chnodefunc : Pointer to address of function for translating the MAC address specific to

that lower-layer communication software to a node ID is returned. If the lower-layer communication software has a node ID equal to the MAC address or effects simple linear translation, NULL is returned.

The specification for the function argument to be delivered is as follows:

mac: [out] MAC address before translation

As the return value, this function returns the node ID derived from translation.

****broadfunc** : Pointer to address of broadcast destination acquisition function is returned. If lower-layer communication software has broadcast capability, NULL is returned.

Specifications for function arguments to be delivered are as follows:

bcast : [in] Broadcast target designation code for intra-domain or intra-local-subnet broadcast designation.

map[32] : [out] Returns array for broadcast destination node ID bitmap. The relationship between broadcast destination node IDs and bits is shown below:

map[0]-bit0 : Node ID 0 (0x00)

map[0]-bit1 : Node ID 1 (0x01)

.....

map[1]-bit0 : Node ID 8 (0x08)

map[2]-bit1 : Node ID 9 (0x09)

.....

map[31]-bit7 : Node ID 255 (0xFF)

(5) Return value

0: Failed acquisition

1: Successful acquisition

(6) Structure

```
typedef struct {
    unsigned char    kind;        /* Low-order medium types
                                   Power line: 0x31
                                   Low-power RF: 0x33
                                   Extended HBS: 0x34
                                   IrDA Control: 0x35
                                   LonTalk®: 0x36 */
    unsigned char    ver[3];     /* Lower-layer communication software version No. */
    unsigned char    maker[3];   /* Manufacturer code */
    short            mac_len;    /* MAC address length */
    unsigned char    mac_ad[6];  /* MAC address */
    unsigned char    mac_mask[6]; /* MAC address mask value */
    short            house_len;  /* House code length */
    short            *housecode; /* Pointer to house code information */
    short            slen;       /* Transmittable data length */
    short            rlen;       /* Receivable data length */
    short            broad;      /* Existence/non-existence of broadcast function
                                   (0: Non-existence, 1: Existence) */
    short            baud;       /* Transmission rate */
} LOW_PRO_DATA
```

(7) Notes/restrictions

None

4.2.9 LowGetStatus

(1) Name

Status data acquisition request function

(2) Function

Requests the lower-layer communication software to provide the lower-layer communication software status data. The status data that can be obtained by this function is dynamic information such as that indicating a change to an abnormal state or the current processing state.

(3) Syntax

/*[IN] lower-layer communication software ID */

/*[OUT] status of the lower-layer communication software */

(4) Explanation

device_id	: Lower-layer communication software ID
	Power Line Communication Protocol A and D Systems
	0x11 – 0x1F
	Specific low-power RF
	0x31 – 0x3F
	Extended HBS
	0x41 – 0x4F
	IrDA_Control
	0x51 – 0x5F
	LonTalk [®]
	0x61 – 0x6F
	Bluetooth [®]
	0x71 – 0x7F
	Ethernet
	0x81 – 0x8F
	IEEE802.11/11b
	0x91 – 0x9F
	Power Line Communication Protocol C System
	0xA1

*status : Pointer to status data structure is returned.

(5) Return value

0: Failed acquisition

1: Successful acquisition

(6) Structure

```
typedef struct {
    char    upper_trouble; /* High-order layer fault code (0–127)
                          No fault or removal of trouble (0) */
    char    low_trouble;  /* Lower-layer communication software block fault
                          code (0–127)
                          No fault or removal of trouble (0) */
    char    low_mode;     /* Operation mode code
```

```

Normal operation (0)
Test mode, such as maintenance (1)
Monitoring mode (2) */
short    state;    /*
Lower-layer communication software block status
LOW_STS_STOP      : 0 Stop status
LOW_STS_INI       : 1 Initializing status
LOW_STS_RUN       : 2 Normal processing status
LOW_STS_ESTOP     : 3 Error stop status */
LOW_STS_RST       : 4 warm start state
LOW_STS_CSTOP     : 5 communication stop
state
LOW_STS_SPD       : 6 suspend status

} LOW_STATUS;

```

(7) Notes/restrictions

None

4.2.10 LowSendData

(1) Name

Data transmission request function

(2) Function

Requests that lower-layer communication software transmit ECHONET data.

(3) Syntax

```
short LowSendData (
    unsigned char device_id, /* [IN] Lower-layer communication software
                             type ID */
    const unsigned char *buf, /* [IN] Pointer to transmission data */
    short snd_sz, /* [IN] Transmission data size */
    const unsigned char *da, /* [IN] Physical address of transmission
                             destination */
    unsigned char broad, /* [IN] Broadcast specification */
)
```

(4) Explanation

device_id : Lower-layer communication software identification information.
Power Line Communication Protocol A and D Systems

	0x11 – 0x1F
Specific low-power RF	0x31 – 0x3F
Extended HBS	0x41 – 0x4F
IrDA_Control	0x51 – 0x5F
LonTalk [®]	0x61 – 0x6F
Bluetooth [®]	0x71 – 0x7F
Ethernet	0x81 – 0x8F
IEEE802.11/11b	0x91 – 0x9F
Power Line Communication Protocol C System	0xA1

*buf : Specifies pointer to ECHONET data to be transmitted. The ECHONET data to be delivered here is one of the data exchanged between protocol difference absorption processing blocks as stipulated in Part 2, Section 4.2.

snd_sz : Specifies transmission data size.

*da : Specifies pointer to MAC address of transmission destination within local subnet. If “broad” specifies a simultaneous broadcast within the domain or a broadcast within the local subnet, this parameter is not used and the lower-layer communication software performs a simultaneous broadcast.

broad : Specifies broadcast.

0x00 : Specifies no broadcast or a simultaneous broadcast within a specified subnet.

0xFF : Specifies a broadcast within the domain or within the local subnet.

(5) Return value

LOW_BUFFER_FULL(0) : Buffer full error

LOW_NO_ERROR(1) : Transmission accepted

LOW_BUFFER_SIZE_ERROR(2) : Buffer size error

LOW_STATE_ERROR(3) : Internal error in lower-layer communication

software

(6) Structure

None

(7) Notes/restrictions

If the specified lower-layer communication software is not in the normal operation state, this function returns “Internal error of lower-layer communication software”.

4.2.11 LowGetSendResult

(1) Name

Transmission result acquisition request function

(2) Function

Requests the results of the last transmission of an ECHONET message that was made by the lower-layer communication software in response to a request by the message transmission function (ClcSendData).

(3) Syntax

/*[IN] lower-layer communication software ID */

/*[OUT] transmission result */

(4) Explanation

device_id : Lower-layer communication software identification information.
Power Line Communication Protocol A and D Systems

0x11 – 0x1F

Specific low-power RF 0x31 – 0x3F

Extended HBS 0x41 – 0x4F

IrDA_Control 0x51 – 0x5F

LonTalk[®] 0x61 – 0x6F

Bluetooth[®] 0x71 – 0x7F

Ethernet 0x81 – 0x8F

IEEE802.11/11b 0x91 – 0x9F

Power Line Communication Protocol C System

0xA1

result : Transmission result. 0x00: Successful transmission, 0x01: Failed transmission, 0xFF: No response

(5) Return value

LOW_CANCEL(0) : Transmission stop

LOW_NO_ERROR(1) : Normal

LOW_NO_SENDDEND(2) : Transmitting status (transmission not completed)

LOW_INTERNAL_ERROR(3) : Internal error in lower-layer communication software

(6) Structure

None

(7) Notes/restrictions

If the lower-layer communication software is not in the normal operation state, this function returns “Internal error of lower-layer communication software”.

Note that “result” is meaningful only when the return value is normal (NO_ERROR).

4.2.12 LowSendCancel

(1) Name

Transmission stop request function

(2) Function

Requests that the lower-layer communication software cancel an ECHONET data transmission being performed in accordance with the data transmission function (ClcSendData).

(3) Syntax

```
unsigned char LowSendCancel (
    unsigned char device_id /*[IN] Lower-layer communication software ID */
)
```

(4) Explanation

device_id	: Lower-layer communication software identification information.
	Power Line Communication Protocol A and D Systems
	0x11 – 0x1F
	Specific low-power RF
	0x31 – 0x3F
	Extended HBS
	0x41 – 0x4F
	IrDA_Control
	0x51 – 0x5F
	LonTalk [®]
	0x61 – 0x6F
	Bluetooth [®]
	0x71 – 0x7F
	Ethernet
	0x81 – 0x8F
	IEEE802.11/11b
	0x91 – 0x9F
	Power Line Communication Protocol C System
	0xA1

(5) Return value

LOW_CANCEL(0)	: No execution of stop processing because transmission has been completed
LOW_NO_ERROR(1)	: Normal
LOW_INTERNAL_ERROR(3)	: Internal error in lower-layer communication software

(6) Structure

None

(7) Notes/restrictions

If the lower-layer communication software is not in the normal operation state, this function returns “Internal error of lower-layer communication software”.

Upon receipt of this request, the lower-layer communication software discards all data retained in the transmitting buffer.

4.2.13 LowReceiveData

(1) Name

Received-data request function

(2) Function

Requests received ECHONET data retained by lower-layer communication software.

(3) Syntax

```
short LowReceiveData (
    unsigned char device_id, /* [IN] Lower-layer communication software ID */
    unsigned char *buf, /* [IN] Pointer to receiving buffer */
    short buf_sz /* [IN] Receiving buffer size */
    short *rcv_cz /* [OUT] Received data size */
    unsigned char *sa /* [OUT] Transmission source MAC address */
)
```

(4) Explanation

device_id : Lower-layer communication software identification information.
Power Line Communication Protocol A and D Systems

	0x11 – 0x1F
Specific low-power RF	0x31 – 0x3F
Extended HBS	0x41 – 0x4F
IrDA_Control	0x51 – 0x5F
LonTalk [®]	0x61 – 0x6F
Bluetooth [®]	0x71 – 0x7F
Ethernet	0x81 – 0x8F
IEEE802.11/11b	0x91 – 0x9F
Power Line Communication Protocol C System	0xA1

*buf : Specifies pointer (1st byte: EDC) to receiving buffer.

buf_sz : Specifies receiving buffer size.

rcv_cz : Returns actual received data size.

sa : Returns transmission source MAC address.

(5) Return value

LOW_NO_RECEIVE(0) : No received data
 LOW_NO_ERROR(1) : Normal (with received data)
 LOW_BUFFER_SIZE_ERROR(2) : Buffer size error
 LOW_INTERNAL_ERROR(3) : Internal error in lower-layer communication software

(6) Structure

None

(7) Notes/restrictions

If the specified lower-layer communication software is not in the normal operation state, this function returns “Internal error of lower-layer communication software”.

4.2.14 LowGetAddress

(1) Name

Address data acquisition request function

(2) Function

Requests address information retained by lower-layer communication software.

(3) Syntax

```

BOOL LowGetAddress (
    unsigned char device_id,    /* [IN] Lower-layer communication software ID */
    short mac_len,             /* [OUT] MAC address length */
    unsigned char mac_ad[7],   /* [OUT] MAC address */
    unsigned char mac_mask[7], /* [OUT] MAC address mask value */
    short *housecode_len,     /* [OUT] Pointer to house code information size */
    unsigned char *housecode;  /* [OUT] Pointer to house code information */
)

```

(4) Explanation

device_id : Lower-layer communication software identification information.
Power Line Communication Protocol A and D Systems

	0x11 – 0x1F
Specific low-power RF	0x31 – 0x3F
Extended HBS	0x41 – 0x4F
IrDA_Control	0x51 – 0x5F
LonTalk [®]	0x61 – 0x6F
Bluetooth [®]	0x71 – 0x7F
Ethernet	0x81 – 0x8F
IEEE802.11/11b	0x91 – 0x9F
Power Line Communication Protocol C System	0xA1

mac_ad : Returns MAC address size.

mac_len : Returns MAC address.

housecode_len : Pointer to house code information size is returned. The value “0x00” indicates that no house code information is needed.

housecode : Pointer to house code information is returned.

(5) Return value

0: Failed address acquisition

1: Successful address acquisition

(6) Structure

None

(7) Notes/restrictions

None

4.2.15 LowSetAddress

(1) Name

Address information setup request information

(2) Function

Sets the address information for lower-layer communication software.

(3) Syntax

```
short LowSetAddress (
    unsigned char device_id, /* [IN] Lower-layer communication software ID */
    short mac_len, /* [IN] MAC address length */
    unsigned char mac_ad[7], /* [IN] MAC address */
    unsigned char mac_mask[7], /* [IN] MAC address mask value */
    short housecode_len, /* [IN] House code information size */
    unsigned char *housecode, /* [IN] Pointer to house code information */
)
```

(4) Explanation

device_id : Lower-layer communication software identification information.
Power Line Communication Protocol A and D Systems

	0x11 – 0x1F
Specific low-power RF	0x31 – 0x3F
Extended HBS	0x41 – 0x4F
IrDA_Control	0x51 – 0x5F
LonTalk [®]	0x61 – 0x6F
Bluetooth [®]	0x71 – 0x7F
Ethernet	0x81 – 0x8F
IEEE802.11/11b	0x91 – 0x9F
Power Line Communication Protocol C System	0xA1

mac_ad : Specifies MAC address size. The value “0x00” indicates that MAC address setup is not requested.

mac_len : Sets MAC address.

housecode_len : Specifies house code information size. The value “0x00” indicates that house code setup is not requested.

*housecode : Specifies pointer to the house code information.

(5) Return value

LOW_NO_CHANGE(0) : Unchangeable with software
 LOW_NO_ERROR(1) : Normal
 LOW_INTERNAL_ERROR(3) : Internal error in lower-layer communication software

(6) Structure

None

(7) Notes/restrictions

None

4.2.16 LowReqToMac

(1) Name

Physical address translation request function

(2) Function

Requests lower-layer communication software to furnish the MAC address corresponding to a delivered node ID.

(3) Syntax

```

BOOL LowReqToMac (
    unsigned char device_id, /* [IN] Lower-layer communication software ID */
    unsigned char node_id,  /* [IN] Node ID to be translated */
    unsigned char *mac,     /* [OUT] Pointer to MAC address derived from
                           translation */
    short *mac_len         /* [OUT] Pointer to MAC address size derived from
                           translation */
)

```

(4) Explanation

device_id	: Lower-layer communication software identification information.
	Power Line Communication Protocol A and D Systems
	0x11 – 0x1F
	Specific low-power RF
	0x31 – 0x3F
	Extended HBS
	0x41 – 0x4F
	IrDA_Control
	0x51 – 0x5F
	LonTalk [®]
	0x61 – 0x6F
	Bluetooth [®]
	0x71 – 0x7F
	Ethernet
	0x81 – 0x8F
	IEEE802.11/11b
	0x91 – 0x9F
	Power Line Communication Protocol C System
	0xA1

node_id : Sets node ID to be translated.

*mac : Pointer to MAC address derived from translation is returned.

*mac_len : Pointer to MAC address size derived from translation is returned.

(5) Return value

0: Failed translation

1: Successful translation

(6) Structure

None

(7) Notes/restrictions

None

4.2.17 LowReqToID

(1) Name

NodeID conversion request function

(2) Function

Requests the NodeID that corresponds to the MAC address delivered (lower-layer communication software).

(3) Syntax

```

BOOL LowReqToID (
  unsigned char  device_id,           /*[IN]  Lower-layer communication software ID */
                short  mac_len       /*[IN]  MAC address length to be translated */
  unsigned char *mac,               /*[IN]  MAC address to be translated */
  unsigned char *node_id,          /*[OUT] Node ID derived from translation */
)

```

(4) Explanation

device_id : Lower-layer communication software identification information.

	Power Line Communication Protocol A and D Systems	0x11 – 0x1F
	Specific low-power RF	0x31 – 0x3F
	Extended HBS	0x41 – 0x4F
	IrDA_Control	0x51 – 0x5F
	LonTalk [®]	0x61 – 0x6F
	Bluetooth [®]	0x71 – 0x7F
	Ethernet	0x81 – 0x8F
	IEEE802.11/11b	0x91 – 0x9F
	Power Line Communication Protocol C System	0xA1

mac_len : MAC address length to be translated

mac : Specifies MAC address to be translated.

*node_id : Pointer to node ID derived from translation is returned.

(5) Return value

0: Failed translation

1: Successful translation

(6) Structure

None

(7) Notes/restrictions

None

4.2.18 LowReqBcastID

(1) Name

Broadcast destination acquisition request function

(2) Function

Extracts target node ID from the DEA intra-domain or intra-local-subnet broadcast target designation code delivered to lower-layer communication software.

(3) Syntax

```

BOOL LowReqBcastID (
    unsigned char device_id, /* [IN] Lower-layer communication software ID */
    unsigned char bcast,    /* [IN] Broadcast target designation code */
    short *map_len,        /* [OUT] Address length for transmitting
                           destination node */
    unsigned char map      /* [OUT] Address information for transmitting
                           destination node */
)

```

(4) Explanation

device_id : Lower-layer communication software identification information.
Power Line Communication Protocol A and D Systems
0x11 – 0x1F
Specific low-power RF
0x31 – 0x3F
Extended HBS
0x41 – 0x4F
IrDA_Control
0x51 – 0x5F
LonTalk®
0x61 – 0x6F
Bluetooth®
0x71 – 0x7F
Ethernet
0x81 – 0x8F
IEEE802.11/11b
0x91 – 0x9F
Power Line Communication Protocol C System
0xA1

bcast : Broadcast target designation code to be targeted (broadcast target designation code in DEA 2nd byte position for intra-domain or intra-local-subnet broadcast designation).

map_len : Address length to bit map indicating translated NodeID

map : Returns address for bitmap indicating node ID derived from translation. The relationship between broadcast destination node IDs and bits is shown below:

```

map[0]-bit0 : Node ID 0 (0x00)
map[0]-bit1 : Node ID 1 (0x01)
.....
map[1]-bit0 : Node ID 8 (0x08)
map[2]-bit1 : Node ID 9 (0x09)
.....
map[31]-bit7 : Node ID 255 (0xFF)

```

(5) Return value

0: Failed translation

- 1: Successful translation

- (6) Structure
 - None

- (7) Notes/restrictions
 - This function is not needed when the lower-layer communication software has broadcast capability.

4.2.19 LowInitAll

(1) Name

Complete initialization request function

(2) Function

Requests that lower-layer communication software effect initialization (by performing a cold start) and acquire house code information and MAC address again. Upon receipt of this request, the lower-layer communication software performs a cold start to switch to the communication stop state and then sets the initialization parameters for itself.

(3) Syntax

```

BOOL LowInitAll (
    unsigned char device_id,          /* [IN] Lower-layer communication software
                                     ID */
    LOW_INIT_DATA *lowinit_data     /* [IN] Pointer to initialization parameter (1)
                                     */
    void *low_init                   /* [IN] Pointer to initialization parameter (2)
                                     */
)

```

(4) Explanation

device_id : Lower-layer communication software identification information.
Power Line Communication Protocol A and D Systems

	0x11 – 0x1F
Specific low-power RF	0x31 – 0x3F
Extended HBS	0x41 – 0x4F
IrDA_Control	0x51 – 0x5F
LonTalk [®]	0x61 – 0x6F
Bluetooth [®]	0x71 – 0x7F
Ethernet	0x81 – 0x8F
IEEE802.11/11b	0x91 – 0x9F
Power Line Communication Protocol C System	0xA1

*lowinit_data : Pointer to initialization parameter for lower-layer communication software common specification items.

*low_init : Pointer to initialization parameter, which varies with lower-layer communication software. The parameter is variously stipulated for all lower-layer communication software programs.

(5) Return value

0: Failed initialization

1: Successful initialization

(6) Structure

```
typedef struct {
    short          sftoldtime, /* Information on maximum holding time for data
                               transmitted by lower-layer communication software */
    short          rftoldtime, /* Information on maximum holding time for data
                               received by lower-layer communication software */
    unsigned char  low_mode,   /* Operation mode selection */
    short          mac_len,    /* MAC address length */
    unsigned char  mac_ad[7]  /* MAC address */
} LOW_INIT_DATA
```

* Except for mac_ad[7], NULL is to be set particularly when there is no initialization data.

* When mac_len is set to NULL, mac_ad[7] has no significance.
(When mac_len is NULL, there will be no MAC address setting.)

(7) Notes/restrictions

If the lower-layer communication software is in cold start, warm start, or communication stop state, this function returns “Failed initialization”.

For lower-layer communication software that does not use house code information, the same process will be performed as in the case of an initialization request.

4.2.20 LowStop

(1) Name

Communication stop request function

(2) Function

Requests the lower-layer communication software to stop the communication processing. Upon reception of this request, the lower-layer communication software will shift to the communication stop state.

(3) Syntax

```
BOOL LowStop (
    unsigned char device_id, /*[IN] Lower-layer communication software ID */
)
```

(4) Explanation

device_id : Lower-layer communication software identification information.

Power Line Communication Protocol A and D Systems	0x11 – 0x1F
Specific low-power RF	0x31 – 0x3F
Extended HBS	0x41 – 0x4F
IrDA_Control	0x51 – 0x5F
LonTalk [®]	0x61 – 0x6F
Bluetooth [®]	0x71 – 0x7F
Ethernet	0x81 – 0x8F
IEEE802.11/11b	0x91 – 0x9F
Power Line Communication Protocol C System	0xA1

(5) Return value

0: Failure to stop communications
1: Successful communication stop

(6) Structure

None

(7) Notes/restrictions

If the lower-layer communication software is in a state other than normal operation, this function returns “Failed suspension”.

If the lower-layer communication software is in the midst of data transmission when this request is received, it terminates a series of transmission processes and switches to the suspension state. If it is in the midst of data reception, on the other hand, it discards

the received data and terminates the process.

The following operations are performed in the suspension state:

- Data reception

No data is to be received.

- Data transmission request from ECHONET communication control processing block

An error is returned.

4.2.21 LowHalt

(1) Name

Complete stop request function

(2) Function

Requests lower-layer communication software to stop completely. Upon receipt of this request, the lower-layer communication software enters the stop state.

(3) Syntax

```

BOOL ClcLowHalt (
    unsigned char device_id,    /*[IN] Lower-layer communication software ID */
)

```

(4) Explanation

device_id : Lower-layer communication software identification information.

Power Line Communication Protocol A and D Systems	0x11 – 0x1F
Specific low-power RF	0x31 – 0x3F
Extended HBS	0x41 – 0x4F
IrDA_Control	0x51 – 0x5F
LonTalk [®]	0x61 – 0x6F
Bluetooth [®]	0x71 – 0x7F
Ethernet	0x81 – 0x8F
IEEE802.11/11b	0x91 – 0x9F
Power Line Communication Protocol C System	0xA1

(5) Return value

0: Failure to stop completely

1: Successful complete stop

(6) Structure

None

(7) Notes/restrictions

If the lower-layer communication software is in a state other than normal operation, this function returns “Failed suspension”.

If the lower-layer communication software is in the midst of data transmission when this request is received, it terminates a series of transmission processes and switches to the suspension state. If it is in the midst of data reception, on the other hand, it discards the received data and terminates the process.

The following operations are performed in the suspension state:

- Data reception

- No data is to be received.

- Data transmission request from ECHONET communication control processing block

- An error is returned.

4.2.22 LowReceiveStop

(1) Name

Stop notification request function

(2) Function

Receives stop notifications sent by the lower-layer communication software.

(3) Syntax

/* [IN] lower-layer communication software ID */

(4) Explanation

device_id : Lower-layer communication software identification information.

Power Line Communication Protocol A and D Systems

0x11 to 0x1F

Low-power RF

0x31 to 0x3F

Extended HBS

0x41 to 0x4F

IrDA_Control

0x51 to 0x5F

LonTalk[®]

0x61 to 0x6F

Bluetooth[™]

0x71 to 0x7F

Ethernet

0x81 to 0x8F

IEEE802.11/11b

0x91 to 0x9F

Power Line Communication Protocol C System

0xA1

(5) Return value

LOW_NO_RECEIVE (0)

: stop notification not provided

LOW_NO_ERROR (1)

: stop notification provided

LOW_INTERNAL_ERROR (3)

: internal error of the lower-layer communication software

(6) Structure used

None

(7) Notes and restrictions

This function returns LOW_NO_ERROR (1) upon reception of a stop notification from the specified lower-layer communication software program.

4.2.23 LowGetAddressTableDataSize

(1) Name

Lower-layer communication software address table data size acquisition function

(2) Function

Acquires the number of lower-layer address table data pairs held (lower-layer communication software).

(3) Syntax

/*[IN] lower-layer communication software ID */

/*[OUT] number of data pairs */

(4) Explanation

This function is designed to be used in combination with LowGetAddressTableData(). The output data is the number of data pairs stored in the address table held by the lower-layer communication software and is stored in the data_number field.

device_id : Lower-layer communication software identification information.

Power Line Communication Protocol A and D Systems

0x11 to 0x1F

Low-power RF

0x31 to 0x3F

Extended HBS

0x41 to 0x4F

IrDA_Control

0x51 to 0x5F

LonTalk[®]

0x61 to 0x6F

Bluetooth[™]

0x71 to 0x7F

Ethernet

0x81 to 0x8F

IEEE802.11/11b

0x91 to 0x9F

Power Line Communication Protocol C System

0xA1

data_number : Pointer to the number of address table pairs in the lower-layer address table data.

(5) Return value

0: Complete stop failed

1: Complete stop successfully completed

(6) Notes

None

4.2.24 LowGetAddressTableData

(1) Name

Lower-layer communication software address table data acquisition function

(2) Function

Acquires lower-layer address table data held (lower-layer communication software).

(3) Syntax

/*[IN] lower-layer communication software ID */

/*[IN/OUT] number of data pairs */

/*[OUT] address table structure */

(4) Explanation

The input data (data_number) is the pointer to the number of address table pairs acquired by MidGetAddressTableDataSize.

The output data comprises array data of a structure that consists of a flag to indicate whether or not the node is a master router, the MAC address, the hardware address of each data pair and the number of address table pairs stored.

device_id : Lower-layer communication software identification information.

Power Line Communication Protocol A and D Systems

0x11 to 0x1F

Low-power RF

0x31 to 0x3F

Extended HBS

0x41 to 0x4F

IrDA_Control

0x51 to 0x5F

LonTalk[®]

0x61 to 0x6F

Bluetooth[™]

0x71 to 0x7F

Ethernet

0x81 to 0x8F

IEEE802.11/11b

0x91 to 0x9F

Power Line Communication Protocol C System

0xA1

data_number : Indicates the pointer to the number of address table data pairs in the lower-layer address table.

Adresstable : Head pointer to the array of address table structure that contains a flag to indicate whether or not the node is a master router, the MAC

address and the hardware address in the lower-layer address table data.

(5) Return value

0: Failed

1: Successfully completed

(6) Structure used

/*[OUT] hardware address data size */

/*[OUT] hardware address (data is stored beginning with the lowest byte towards the highest byte)*/

/* [OUT] MAC address */

/* [OUT] Identifier to indicate whether or not the node is a master router. (The value is "1" if it is a master router and "0" otherwise.) */

(7) Notes and restrictions

Note that data overwrites will occur because the second argument `data_number` is used for both input and output.

4.2.25 LowSetMasterRouterFlag

(1) Name

Master router notification function

(2) Function

Notifies the lower-layer communication software as to whether or not the home node is a master router.

(3) Syntax

/*[IN] lower-layer communication software ID */

/*[IN] master router identification flag */

(4) Explanation

device_id : Lower-layer communication software identification information.

Power Line Communication Protocol A and D Systems

0x11 to 0x1F

Low-power RF

0x31 to 0x3F

Extended HBS

0x41 to 0x4F

IrDA_Control

0x51 to 0x5F

LonTalk[®]

0x61 to 0x6F

Bluetooth[™]

0x71 to 0x7F

Ethernet

0x81 to 0x8F

IEEE802.11/11b

0x91 to 0x9F

Power Line Communication Protocol C System

0xA1

masterRouter_Flag: “1” is specified if the node is a master router and “0” is specified if the node is not a master router. If it has not been determined whether or not the node is a master router, “0” is specified.

(5) Return value

0: Failed

1: Successfully completed

(6) Structure used

None

(7) Notes and restrictions

None

4.2.26 LowGetHardwareAddress

(1) Name

Hardware address data acquisition function

(2) Function

Acquires the hardware address data held (lower-layer communication software).

(3) Syntax

/*[IN] lower-layer communication software ID */

/*[OUT] hardware address size */

/*[OUT] hardware address */

(4) Explanation

The output data comprises the hardware type, hardware address and MAC address.

`device_id` : Lower-layer communication software identification information.

Power Line Communication Protocol A and D Systems

0x11 to 0x1F

Low-power RF

0x31 to 0x3F

Extended HBS

0x41 to 0x4F

IrDA_Control

0x51 to 0x5F

LonTalk[®]

0x61 to 0x6F

Bluetooth[™]

0x71 to 0x7F

Ethernet

0x81 to 0x8F

IEEE802.11/11b

0x91 to 0x9F

Power Line Communication Protocol C System

0xA1

`Hardwareaddresssize` : Indicates the pointer to the size of the hardware address held (lower-layer communication software).

`Hardwareaddress` : Indicates the pointer to the hardware address that corresponds to the address in the lower-layer address table data.

(5) Return value

0: Failed

1: Successfully completed

(6) Structure used

None

(7) Notes and restrictions

None

4.2.27 LowGetEchonetMACList

(1) Name

ECHONET MAC address list acquisition function

(2) Function

Acquires the ECHONET MAC address list held (lower-layer communication software).

(3) Syntax

/*[IN] lower-layer communication software ID */

/*[OUT] ECHONET MAC address list */

(4) Explanation

`device_id` : Lower-layer communication software identification information.

Power Line Communication Protocol A and D Systems

0x11 to 0x1F

Low-power RF

0x31 to 0x3F

Extended HBS

0x41 to 0x4F

IrDA_Control

0x51 to 0x5F

LonTalk[®]

0x61 to 0x6F

Bluetooth[™]

0x71 to 0x7F

Ethernet

0x81 to 0x8F

IEEE802.11/11b

0x91 to 0x9F

Power Line Communication Protocol C System

0xA1

* `mac_addr_list` : Pointer to the (32-element) array of the ECHONET MAC address list held by the lower-layer communication software. The form of data is as follows:

The ECHONET MAC address list sets “1” for the bits that correspond to the ECHONET MAC addresses present (hexadecimal notation), beginning with the first byte (see the 32-byte table below).

	Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
1 st byte	F8	F9	FA	FB	FC	FD	FE	FF
2 nd byte	F0	F1	F2	F3	F4	F5	F6	F7
3 rd byte	E8	E9	EA	EB	EC	ED	EE	EF
4 th byte	E0	E1	E2	E3	E4	E5	E6	E7
5 th byte	D8	D9	DA	DB	DC	DD	DE	DF
6 th byte	D0	D1	D2	D3	D4	D5	D6	D7
7 th byte	C8	C9	CA	CB	CC	CD	CE	CF
8 th byte	C0	C1	C2	C3	C4	C5	C6	C7
9 th byte	B8	B9	BA	BB	BC	BD	BE	BF
10 th byte	B0	B1	B2	B3	B4	B5	B6	B7
11 th byte	A8	A9	AA	AB	AC	AD	AE	AF
12 th byte	A0	A1	A2	A3	A4	A5	A6	A7
13 th byte	98	99	9A	9B	9C	9D	9E	9F
14 th byte	90	91	92	93	94	95	96	97
15 th byte	88	89	8A	8B	8C	8D	8E	8F
16 th byte	80	81	82	83	84	85	86	87
17 th byte	78	79	7A	7B	7C	7D	7E	7F
18 th byte	70	71	72	73	74	75	76	77
19 th byte	68	69	6A	6B	6C	6D	6E	6F
20 th byte	60	61	62	63	64	65	66	67
21 st byte	58	59	5A	5B	5C	5D	5E	5F
22 nd byte	50	51	52	53	54	55	56	57
23 rd byte	48	49	4A	4B	4C	4D	4E	4F
24 th byte	40	41	42	43	44	45	46	47
25 th byte	38	39	3A	3B	3C	3D	3E	3F
26 th byte	30	31	32	33	34	35	36	37
27 th byte	28	29	2A	2B	2C	2D	2E	2F
28 th byte	20	21	22	23	24	25	26	27
29 th byte	18	19	1A	1B	1C	1D	1E	1F
30 th byte	10	11	12	13	14	15	16	17
31 st byte	08	09	0A	0B	0C	0D	0E	0F
32 nd byte	00	01	02	03	04	05	06	07

(5) Return value

0: Abnormal

1: Normal

(6) Structure used

None

(7) Notes and restrictions

When the transmission medium is a power line whose ECHONET MAC address is a 2-byte value, the lower byte is listed.

4.2.28 LowGetMasterRouterInfo

(1) Name

Master router data acquisition function

(2) Function

Acquires the master router data held (lower-layer communication software).

(3) Syntax

/*[IN] lower-layer communication software ID */

/*[OUT] whether or not a master router is present */

/*[OUT] master router's ECHONET MAC address */

(4) Explanation

device_id : Lower-layer communication software identification information.

Power Line Communication Protocol A and D Systems

0x11 to 0x1F

Low-power RF

0x31 to 0x3F

Extended HBS

0x41 to 0x4F

IrDA_Control

0x51 to 0x5F

LonTalk[®]

0x61 to 0x6F

Bluetooth[™]

0x71 to 0x7F

Ethernet

0x81 to 0x8F

IEEE802.11/11b

0x91 to 0x9F

Power Line Communication Protocol C System

0xA1

* result : Indicates the pointer to whether or not a master router is present.

0x00: there is no master router

0x01: there is a master router

* master_mac_addr : Indicates the pointer to the ECHONET MAC address value of the master router.

(5) Return value

0: Abnormal

1: Normal

(6) Structure used

None

(7) Notes and restrictions

When the transmission medium is a power line whose ECHONET MAC address is a 2-byte value, the lower byte is listed.

4.2.29 LowReqToHardwareAddress

(1) Name

Hardware address conversion request function

(2) Function

Requests the hardware address that corresponds to the ECHONET MAC address data delivered (lower-layer communication software).

(3) Syntax

/*[IN] lower-layer communication software ID */

/*[IN] length of the MAC address to be converted */

/*[IN] pointer to the MAC address to be converted */

/*[OUT] pointer to the hardware address */

/*[OUT] pointer to the hardware address size */

(4) Explanation

device_id : Lower-layer communication software identification information.

Power Line Communication Protocol A and D Systems

0x11 to 0x1F

Low-power RF

0x31 to 0x3F

Extended HBS

0x41 to 0x4F

IrDA_Control

0x51 to 0x5F

LonTalk[®]

0x61 to 0x6F

Bluetooth[™]

0x71 to 0x7F

Ethernet

0x81 to 0x8F

IEEE802.11/11b

0x91 to 0x9F

Power Line Communication Protocol C System

0xA1

mac_len : Length of the MAC address before conversion.

* mac: Specifies the pointer to the MAC address before conversion

hardwareaddress: Pointer to the hardware address after conversion is returned.

* hardwareaddress_len: Pointer to the size of the hardware address after conversion is returned.

(5) Return value

0: Abnormal

1: Normal

(6) Structure used

None

(7) Notes and restrictions

None

4.3 Initial Setting Information Specifications

This section describes the initialization parameter specifications provided in the area indicated by the argument initialization parameter pointer “*low_init” of the “Request for initialization: LowInit” (see Remarks below) for each of the following six types of lower-layer communication software:

- (1) Power line lower-layer communication software
- (2) Specific low-power RF lower-layer communication software
- (3) Extended HBS lower-layer communication software
- (4) IrDA-dependent lower-layer communication software
- (5) LonTalk[®]-dependent lower-layer communication software
- (6) Bluetooth[®] lower-layer communication software
- (7) Ethernet lower-layer communication software
- (8) IEEE802.11/11b lower-layer communication software

Remark: Syntax of the LowInit function

```

BOOL LowInit (
    short device_id,           /* [IN] Lower-layer communication
                               software type ID */
    LOW_INIT_DATA *init_data, /* [IN] Pointer to initialization parameter (1)
                               */
    void *low_init            /* [IN] Pointer to initialization parameter (2)
                               */
)

```

4.3.1 Initialization parameter specifications for lower-layer communication software for the Power Line Communication Protocol A, C and D Systems

```
typedef struct {
    short    sbuf_len;        /* Transmitting buffer size */
    short    *sbuf;          /* Pointer to transmitting buffer */
    short    rbuf_len;       /* Receiving buffer size */
    short    *rbuf           /* Pointer to receiving buffer */
} PLCA_INIT_DATA
```

4.3.2 Initialization parameter specifications for specific low-power RF lower-layer communication software

```
typedef struct {
} RF_INIT_DATA
```

4.3.3 Initialization parameter specifications for extended HBS lower-layer communication software

```
typedef struct {
    short    sbuf_len;        /* Transmitting buffer size */
    short    *sbuf;          /* Pointer to transmitting buffer */
    short    rbuf_len;       /* Receiving buffer size */
    short    *rbuf           /* Pointer to receiving buffer */
} HBS_INIT_DATA
```

4.3.4 Initialization parameter specifications for IrDA-dependent lower-layer communication software

```
typedef struct {
    short    sbuf_len;        /* Transmitting buffer size */
    short    *sbuf;          /* Pointer to transmitting buffer */
    short    rbuf_len;       /* Receiving buffer size */
    short    *rbuf           /* Pointer to receiving buffer */
    short    mac_table_len;  /* MAC address translation table size */
    short    *mac_table      /* Pointer to MAC address translation table */
} IRDA_INIT_DATA
```

4.3.5 Initialization parameter specifications for LonTalk®-dependent lower-layer communication software

```
typedef struct {  
    short    sbuf_len;        /* Transmitting buffer size */  
    short    *sbuf;          /* Pointer to transmitting buffer */  
    short    rbuf_len;       /* Receiving buffer size */  
    short    *rbuf           /* Pointer to receiving buffer */  
} LON_INIT_DATA
```

4.3.6 Initialization parameter specifications for Bluetooth™ lower-layer communication software

```
typedef struct {  
    short    sbuf_len;        /* transmission buffer size */  
    short    *sbuf;          /* pointer to the transmission buffer */  
    short    rbuf_len;       /* receiving buffer size */  
    short    *rbuf           /* pointer to the receiving buffer */  
} BT_INIT_DATA
```

4.3.7 Initialization parameter specifications for Ethernet lower-layer communication software

```
typedef struct {  
    short    sbuf_len;        /* transmission buffer size */  
    short    *sbuf;          /* pointer to the transmission buffer */  
    short    rbuf_len;       /* receiving buffer size */  
    short    *rbuf           /* pointer to the receiving buffer */  
} ETHERNET_INIT_DATA
```

4.3.8 Initialization parameter specifications for IEEE802.11/11b lower-layer communication software

```
typedef struct {  
    short    sbuf_len;        /* transmission buffer size */  
    short    *sbuf;          /* pointer to the transmission buffer */  
    short    rbuf_len;       /* receiving buffer size */  
    short    *rbuf           /* pointer to the receiving buffer */  
    short    mode;           /* communication mode */  
    short    auto_ip;        /* IP address setup method */  
}
```

```

short  ssid_size;      /* SSID array size */
short  *SSID;         /* SSID array storage pointer */
short  auth_type;     /* authentication type */
short  wep_key_size;  /* authentication key size */
short  *wep_key;     /* authentication key array pointer */
short  channel;      /* frequency channel */
} IEEE802_11_INIT_DATA

```

1) Information on the transmission and receiving buffers

The first 4 parameters of the structure provide buffer information for referencing by the common lower-layer communication interface.

2) Communication mode

Specifies the communication mode to use.

mode = 0: Ad-hoc mode

mode = 1: Infrastructure mode

mode = -1: Indefinite

Whether or not to allow the lower-layer software to store the attribute value data and whether or not to allow overwriting shall be implementation-dependent.

In the case where the lower-layer software does not need this parameter, the value “-1” shall be used. In the case where overwriting is prohibited, a fixed value shall be returned.

3) IP address setup method

Specifies the IP address setup method.

auto_ip = 0: Automatic mode

auto_ip = 1: Manual mode

auto_ip = -1: Indefinite

The automatic mode is a mode in which the system can assign and deliver unique IP addresses using DHCP and the lower-layer communication software acquires IP addresses using a DHCP client function.

The manual mode is a mode that is used to determine IP addresses when the system does not have a DHCP server function or to let the lower-layer communication software determine IP addresses without using a DHCP client function.

The manual mode must not be used unless the lower-layer communication software is equipped with a means of determining IP addresses. In the case where it is necessary to notify the IP address to the lower-layer communication software, the IP address shall be given to the third argument of LowInit.

4) SSID array size

Contains the SSID array size value, which is managed by the lower-layer communication software.

5) SSID storage pointer

Gives the pointer to the beginning of the SSID array.

6) Authentication type

Specifies whether to use open system authentication or shared key authentication.

auth_type = 0: Use open system authentication.

auth_type = 1: Use shared key authentication.

auth_type = -1: Indefinite

Whether or not to allow the lower-layer software to store the attribute value data and whether or not to allow overwriting shall be implementation-dependent.

In the case where the lower-layer software does not need this parameter, the value “-1” shall be used. In the case where overwriting is prohibited, a fixed value shall be returned.

7) Authentication key size

Specifies or references the authentication key array size.

8) Authentication key array pointer

Gives the pointer to the beginning of the authentication key array.

9) Frequency channel

The frequency channel used by this medium.