Part VI
ECHONET Discrete Lower-layer Communication Interface Specification

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

Revision record

・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.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

	Revised entry	Revision/addition
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 n 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.

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.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

Contents

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

Chapter1 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 specification for the discrete lower-layer communication interface deals with interface services that are to be supported by the lower-layer communication software. It defines discrete lower-layer communication interface specification 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 specification levels 1 and 2 are based on the concepts of Basic API Levels 1 and 2.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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 communications middleware and the lower-layer communication software.

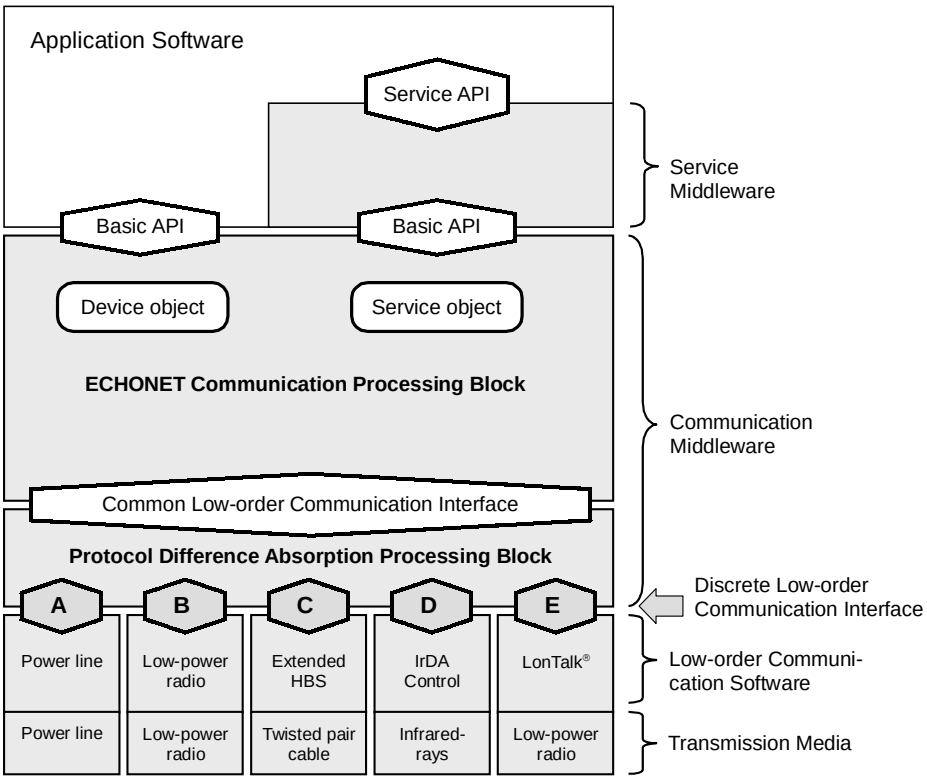

Fig. 1.1 Positioning of Discrete Lower-layer Communication Interface

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

Chapter2 ECHONET Discrete Lower-layer Communication Interface Function Specification

2.1 List of ECHONET Discrete 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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

2.2 ECHONET Discrete Lower-layer Communication Interface Detailed Specification

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 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 communication stop state to 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 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 normal operation state to suspension state.

(7) Request for operation restart

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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(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.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(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 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 communication stop state.

(21) Request for complete stop
Requests that lower-layer communication software switch to stop state.

(22) Stop notice
Lower-layer communication software notifies Protocol Difference Absorption Processing Block that lower-layer communication software has switched to stop state.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

Chapter3 Level 1 ECHONET Discrete Lower-layer Communication Interface Specification

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

For each service listed in Table 3.1, the level 1 ECHONET discrete lower-layer communication interface specification prescribes 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 specification. 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 Discrete 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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

Table 3.1 List of Level 1 ECHONET Discrete 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 communication stop state.	Optional
21	Request for complete stop	Requests that lower-layer communication software enter stop state.	Optional
22	Stop notice	Lower-layer communication software notifies Protocol Difference Absorption Processing Block that lower-layer communication software has switched to stop state.	Optional

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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

This section stipulates the data that are 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 the protocol difference absorption processing block. When these data transfer operations can be performed, the level 1 ECHONET discrete lower-layer communication interface specification is complied with. The data transfer method (use of a structure, delivery of data exchange buffer pointer information, etc.) is not stipulated here.

Further, 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 radio, etc.). Table 3.2 shows input/output specifications.

Table 3.2 List of Low-layer Communication Software Type Request API Input/Output Data

Direction	Data name	Contents and condition	Remarks
Input	–		
Output	device_id	- Indicates the type of lower-layer communication software. - The power line lower-layer communication software, specific low-power radio 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: Successful initialization, FALSE: Failed initialization	Optional

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(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 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	snfbuf	- Indicates buffer size, buffer location, and other relevant information for transmitted data between lower-layer communication software and communication medium.	Optional
Input	rnfbuf	- 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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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

Requests that lower-layer communication software start operation. Table 3.4 shows 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 Protocol Difference Absorption Processing Block. Table 3.5 shows 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 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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(6) Request for suspension

Requests that lower-layer communication software enter 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 suspension state and enter 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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(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 radio, 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 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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(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 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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(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 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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(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 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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(20) Request for communication stop

Requests that lower-layer communication software switch to 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 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 ECHONET communication processing block that lower-layer communication software has switched to stop state. Table 3.23 shows input and output specifications.

Table 3.23 Stop Notice Service Input/Output Data

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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

Chapter4 Level 2 ECHONET Discrete Lower-layer Communication Interface Specification

This Section provides the API detailed specification in consideration of the interchangeability of software to be developed using this interface as the level 2 ECHONET discrete lower-layer communication interface. The stipulations 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).

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.1 List of Level 2 ECHONET Discrete Lower-layer Communication Interfaces

The following 22 functions are stipulated as the level 2 ECHONET discrete 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 Discrete 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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

Table 1.17 List of Level 2 ECHONET Discrete Low-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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.2 Level 2 ECHONET Discrete Lower-layer Communication Interface Detail Specification

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.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

(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).

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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 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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

*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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(6) Structure

```
typedef strut {
    short           sfholdtime,   /*  Maximum holding time for transmission data */
    short           rfholdtime,   /*  Maximum holding time for received data */
    unsigned char   low_mode,     /*  Operation mode specification */
                                  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 cold start or warm start state, this function returns "Failed initialization".

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

(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 communication stop state, this function returns "Failure to start".

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.2.4 LowSetTrouble

(1) Name

Fault notice function

(2) Function

Notifies the lower-layer communication software of fault (error) status of high-order layer from 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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

htrouble_no : Trouble No.

-1 Trouble removed
1 Application software error
2 ECHONET communication 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.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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 into 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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

(5) Return value

0: Failed request

1: Successful request

(6) Structure

None

(7) Notes/restrictions

If the lower-layer communication software is already in 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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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 into 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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

(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 into 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 suspension state:

- Data reception

  No data is to be received.

- Data transmission request from ECHONET communication control processing block

  An error is returned.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.2.7 LowWakeup

(1) Name

Operation restart request function

(2) Function

Requests that lower-layer communication software exit suspension state. Upon receipt of this request, the lower-layer communication software switches into 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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

(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".

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

*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.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

**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 strut {
        unsigned char   kind;       /* Low-order medium types
                                           Power line: 0x31
                                           Low-power radio: 0x33
                                           Extended HBS: 0x34
                                           IrDA Control: 0x35
                                           LonTalk®: 0x36 */
        unsigned char   ver[3];     /* Lower-layer communication software version No. */
        unsinged 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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.2.9 LowGetStatus

(1) Name

Status data acquisition request function

(2) Function

Requests that lower-layer communication software provide status data for lower-layer communication software. Status data obtained by this function is dynamic information, such as error status and processing status.

(3) Syntax

```
BOOL   LowGetStatus (
           unsigned char   device_id      /* [IN] Lower-layer communication software ID */
           LOW_STATUS   *status           /* [OUT] Lower-layer communication software
                                             status */
)
```

(4) Explanation

device_id : Lower-layer communication software ID

Power line	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

*status : Pointer to status data structure is returned.

(5) Return value

0: Failed acquisition

1: Successful acquisition

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(6) Structure

```
typedef strut {
        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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

*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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(6) Structure

None

(7) Notes/restrictions

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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.2.11 LowGetSendResult

(1) Name

Transmission result acquisition request function

(2) Function

Requests result of latest ECHONET data transmission that lower-layer communication software performed in accordance with data transmission function (ClcSendData).

(3) Syntax

```
short    LowGetSendResult (
         unsigned char   device_id,     /* [IN] Lower-layer communication software ID */
         unsigned char *result          /* [OUT] Transmission result */
)
```

(4) Explanation

device_id : Lower-layer communication software identification information.

Power line	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

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_SENDEND(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 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).

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.2.12 LowSendCancel

(1) Name

Transmission stop request function

(2) Function

Requests that lower-layer communication software cancel an ECHONET data transmission being performed in accordance with 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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

(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 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.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

*buf : Specifies pointer (1st byte: EDC) to receiving buffer.
buf_sz : Specifies receiving buffer size.
rcv_sz : 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 normal operation state, this function returns "Internal error of lower-layer communication software".

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.2.14 LowGetAddress

(1) Name

Address information 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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.2.16 LowReqToMac

(1) Name

Physical address translation request function

(2) Function

Requests lower-layer communication software to furnish 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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.2.17 LowReqToID

(1) Name

Node ID translation request function

(2) Function

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

(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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

This function is not needed when the lower-layer communication software has broadcast capability.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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 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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

*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

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

(6) Structure

```
typedef strut {
    short           sfholdtime,   /* Information on maximum holding time for data
                                     transmitted by lower-layer communication software */
    short           rfholdtime,   /* 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.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.2.20 LowStop

(1) Name

Communication stop request function

(2) Function

Requests that lower-layer communication software stop communications. Upon receipt of this request, the lower-layer communication software enters 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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

(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 into 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 suspension state:

- Data reception

  No data is to be received.
- Data transmission request from ECHONET communication control processing block

  An error is returned.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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	0x11～0x1F
Specific low-power radio	0x31～0x3F
Extended HBS	0x41～0x4F
IrDA_Control	0x51～0x5F
LonTalk®	0x61～0x6F

(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 into 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 suspension state:

- Data reception

  No data is to be received.

- Data transmission request from ECHONET communication control processing block

  An error is returned.

© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.3 Initial Setting Information Specification

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 Remark below) for each of the following six types of lower-layer communication software:

(1) Power line lower-layer communication software
(2) Specific low-power radio 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

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) */
```


© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

4.3.1 Initialization parameter specifications for power line 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 */
} PLCA_INIT_DATA
```

4.3.2 Initialization parameter specifications for specific low-power radio 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
```


© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED

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
```


© 2000（2002） ECHONET CONSORTIUM ALL RIGHT RESERVED