



YVCAPI Functional Specifications

Version 1.03.00

October 17, 2019

Yamaha Corporation

Copyright© 2017-2019 Yamaha Corporation

Revision History

Date	Version	Description
Aug. 10, 2017	1.00.00	First edition
Dec. 05, 2017	1.01.00	Supported Mac Renamed the API function “YVC_DLLInfo” to “ YVC_APIInfo ” Renamed the data type definition “YVC_DLL_INFO” to “ YVC_API_INFO ” Renamed the constant definition “YVC_LEN_DLL_VERSION” to “ YVC_LEN_API_VERSION ”
July 24, 2018	1.02.00	Added YVC-200 to the target equipment
Oct. 17, 2019	1.03.00	Added YVC-330 to the target equipment

Contents

1. Overview	4
1.1 Scope	4
1.2 Target equipment	4
1.3 Terminology	4
1.4 Reference documentation	5
2. Operating Environment	6
2.1 Software environment	6
2.2 Supported versions	6
3. Structure of the API	7
3.1 File components	8
4. API Specifications	9
4.1 List of API functions	9
4.2 API functions in detail	10
4.3 Control commands	15
4.4 Control command list	16
4.5 Control command in detail	22
4.6 Event details	89
4.7 Constants and type definitions	91
4.8 Error definition	102
4.9 Limitations	103
4.10 Notes	103
5. API Usage Example	104
6. Appendices	108
6.1 Lists of modes and firmware versions supported by the commands	108
6.2 Microphone capsule positions	111
6.3 Audio block diagram	114

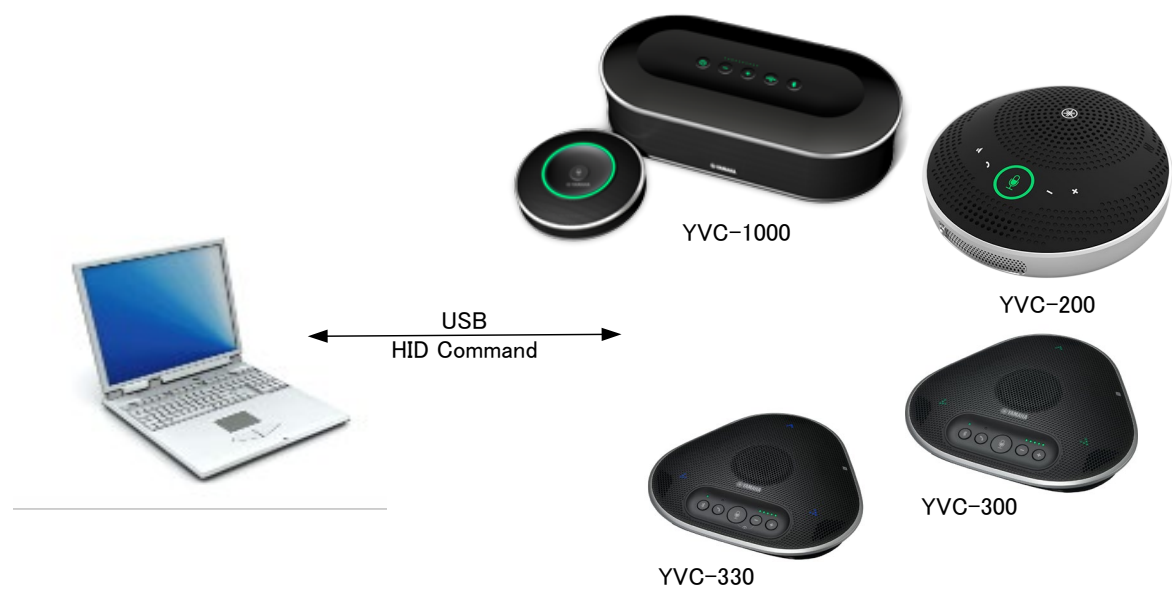
1. Overview

This specification document contains the API specifications for the API (Application Programming Interface) that controls YVC-Series equipment from a PC.

Using the API of the API enables application software, such as web conference applications, on PCs to control YVC-Series equipment, set parameters, and get the operating status.

1.1 Scope

The specification document covers the functional specifications of the API, which is used to control YVC-Series equipment with dedicated commands through HID communication to set and get parameters. The equipment is recognized by PCs as HID devices.



1.2 Target equipment

- Yamaha YVC-1000
- Yamaha YVC-300
- Yamaha YVC-330
- Yamaha YVC-200

1.3 Terminology

This section describes the terms used in this specification document and header files:

Term	Description
HID	An abbreviation for Human Interface Device
NORMAL	Represents the normal startup mode (with the audio function enabled).
FWUP	Represents the mode for updating the firmware (with the audio function disabled).
AGC	An abbreviation for Automatic Gain Controller
AAT	An abbreviation for automatic audio tuning
BT	An abbreviation for Bluetooth
SP	An abbreviation for speaker
MIC	An abbreviation for microphone

EQ	An abbreviation for equalizer
CH	An abbreviation for channel
FW	An abbreviation for firmware
INFO	An abbreviation for information
CMD	An abbreviation for command
ARG	An abbreviation for argument
PRM	An abbreviation for parameter

1.4 Reference documentation

For details on the functions of each device, refer to the following manuals:

- Unified Communications Microphone & Speaker System YVC-1000 User's Manual
- Unified Communications Speakerphone YVC-300 User's Manual
- Unified Communications Speakerphone YVC-330 User's Manual
- Unified Communications Speakerphone YVC-200 User's Manual

2. Operating Environment

This chapter describes the operating environment of the API.

2.1 Software environment

The API runs on the following OSs:

- **Windows**

- Microsoft Windows 7 (32bit/64bit)
- Microsoft Windows 8.1 (32bit/64bit)
- Microsoft Windows 10 (32bit/64bit)

- **Mac**

- macOS 10.12
- macOS 10.13
- macOS 10.14

* It is assumed that YVC-Series equipment is recognized as HID devices and operates properly on these OSs.

2.2 Supported versions

The following table lists the versions of the firmware on YVC-Series equipment that is supported by the following version of the API:

- **Windows**

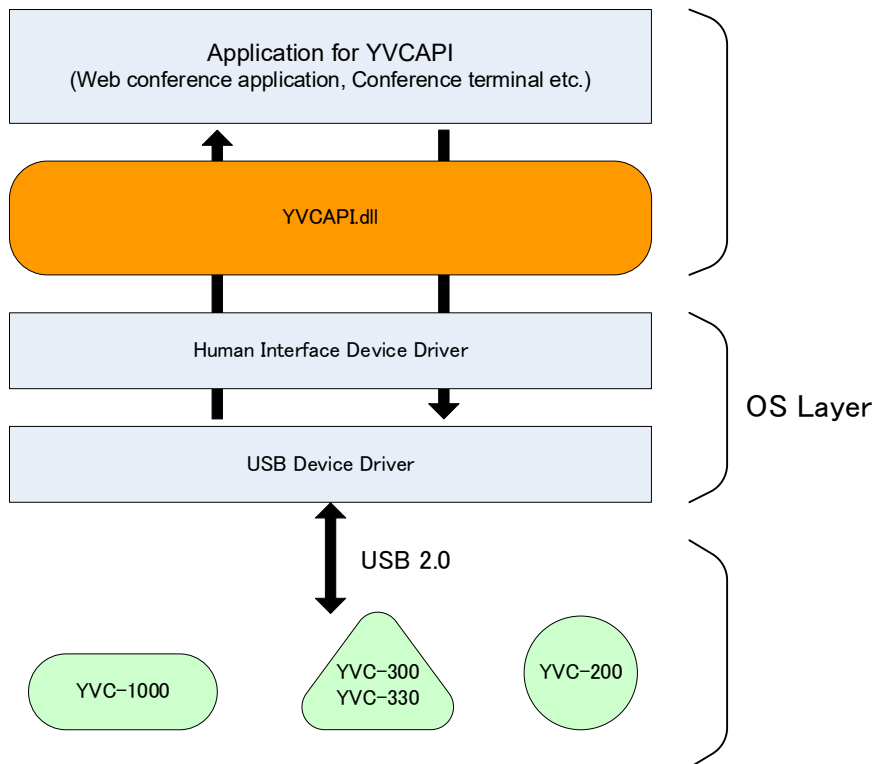
API version	YVC-1000 version	YVC-300 version	YVC-330 version	YVC-200 version
Ver. 1.00	Ver. 2.00 or later	Ver. 1.04 or later	-	-
Ver. 1.01	Ver. 2.00 or later	Ver. 1.04 or later	-	-
Ver. 1.02	Ver. 2.00 or later	Ver. 1.04 or later	-	Ver. 1.01 or later
Ver. 1.03	Ver. 2.00 or later	Ver. 1.04 or later	Ver. 1.02 or later	Ver. 1.01 or later

- **Mac**

API version	YVC-1000 version	YVC-300 version	YVC-330 version	YVC-200 version
Ver. 1.01	Ver. 2.00 or later	Ver. 1.04 or later	-	-
Ver. 1.02	Ver. 2.00 or later	Ver. 1.04 or later	-	Ver. 1.01 or later
Ver 1.03	Ver. 2.00 or later	Ver. 1.04 or later	Ver. 1.02 or later	Ver. 1.01 or later

3. Structure of the API

This chapter describes the structure of the API.



3.1 File components

Files include the binary file, a main component of the API for controlling the YVC-Series equipment, and the header file defining its interface.

- **Windows**

- YVCAPI.dll: DLL main component
- YVCAPI.h: Header file

- **Mac**

- YVCAPI.framework: Framework main component
- YVCAPI.h: Header file

4. API Specifications

This chapter describes the specifications of the API.

4.1 List of API functions

This following table lists and describes API functions:

Function	Description
<u>YVC_DeviceInfo</u>	Gets connection information of YVC-Series equipment.
<u>YVC_DeviceAttach</u>	Gets a handle to communicate with YVC-Series equipment via HID.
<u>YVC_DeviceControl</u>	Communicates with YVC-Series equipment via HID, controls the equipment, and gets and sets parameters.
<u>YVC_DeviceDetach</u>	Releases the handle for communication with the YVC-Series equipment via HID.
<u>YVC_APIInfo</u>	Gets the version information of the API.

4.2 API functions in detail

This section details the API functions.

4.2-1 YVC_DeviceInfo

[Format]

```
int __stdcall YVC_DeviceInfo(
    YVC_DEV_TYPE    type,
    YVC_DEV_INFO    *info
);
```

[Operational overview]

The API function gets the number of YVC-Series devices connected to a host PC.
It returns the number of connected devices of the model specified by *type* to *num* as the connection information.

[Limitations]

The function ignores the second and subsequent devices even if two or more devices of the same model are connected.
Therefore, it always returns the value of 1 or less as the number of connected devices. Also, when devices in NORMAL mode and FWUP mode are running, the NORMAL-mode device takes precedence.
For details on the operation mode, refer to the page on the [YVC_CMD_GetSystemMode](#) command.

[Parameters]

- [IN] [YVC_DEV_TYPE](#) type

This parameter specifies the model type of the YVC-Series device.

Parameter symbol name	Description
YVC_DEV_TYPE YVC1000	YVC-1000
YVC_DEV_TYPE YVC300	YVC-300
YVC_DEV_TYPE YVC200	YVC-200
YVC_DEV_TYPE YVC330	YVC-330

- [OUT] [YVC_DEV_INFO](#) *info

This parameter specifies a pointer for the buffer to store the connection information.

[Return value]

Error code	Description
YVC-API_ERROR_NOERROR	Successful completion
YVC-API_ERROR_INVALIDARGS	The argument is invalid.

4.2-2 YVC_DeviceAttach

[Format]

```
int __stdcall YVC_DeviceAttach(
    YVC_DEV_HANDLE      *handle,
    YVC_DEV_TYPE         type,
    YVC_EVENT_CALLBACK   callback,
    void                 *user
);
```

[Operational overview]

The API function gets the handle to communicate with the YVC-Series device via HID.

[Parameters]

- [OUT] [YVC_DEV_HANDLE](#) *handle

This parameter specifies a pointer for the variable that stores the handle obtained.

- [IN] [YVC_DEV_TYPE](#) type

This parameter specifies the model type of the YVC-Series device.

Parameter symbol name	Description
YVC_DEV_TYPE YVC1000	YVC-1000
YVC_DEV_TYPE YVC300	YVC-300
YVC_DEV_TYPE YVC200	YVC-200
YVC_DEV_TYPE YVC330	YVC-330

- [IN] [YVC_EVENT_CALLBACK](#) callback

This parameter specifies a callback function that notifies a notification destination of any event that occurs. If the callback function is not used, a developer can specify NULL for it. In addition, even if a callback function is specified, the destination is not notified of any event when the [YVC_CMD_SetNoticeEvent](#) command is running on firmware with a version that does not support the command. (Except the [YVC_EVENT_DEV_DISCONNECT](#) event)
For details, refer to the page on "[Event details](#)".

[Notes]

When a device is detached, for example by pulling out the USB cable, after the handle is obtained with this API, the handle obtained with the [YVC_DeviceAttach](#) function must be released immediately (by the time of the next USB connection).

The connection status of the device can also be checked by periodically calling the [YVC_DeviceInfo](#) function to monitor the connection information. However, by specifying the callback function with this parameter, the [YVC_EVENT_DEV_DISCONNECT](#) event notification can be received. We recommend that the handle obtained is released when this event is received.

Note that you do not perform a time-consuming operation or call [this API function](#) in the callback function.

- [IN] void *user

This parameter specifies the address of user management data. If the data is not needed, set the parameter to NULL.

[Return value]

Error code	Description
YVCAPI_ERROR_NOERROR	Successful completion
YVCAPI_ERROR_DEVNOTFOUND	The specified device is not found.
YVCAPI_ERROR_MEMEXHAUST	Memory is exhausted.
YVCAPI_ERROR_STATUS	A status error occurred.
YVCAPI_ERROR_INVALIDARGS	The argument is invalid.
YVCAPI_ERROR_UNSUPPORTEDVER	The firmware version is not supported.

4.2-3 YVC_DeviceDetach

[Format]

```
int __stdcall YVC_DeviceDetach(  
    YVC_DEV_HANDLE handle  
);
```

[Operational overview]

The API function releases the handle for communication with the YVC-Series device via HID.

[Parameters]

- [IN] [YVC_DEV_HANDLE](#) handle

This parameter specifies the handle obtained by the [YVC_DeviceAttach](#) function.

[Return value]

Error code	Description
YVCAPI_ERROR_NOERROR	Successful completion
YVCAPI_ERROR_INVALIDHANDLE	The handle is invalid.
YVCAPI_ERROR_STATUS	A status error occurred.

4.2-4 YVC_DeviceControl

[Format]

```
int __stdcall YVC_DeviceControl(
    YVC_DEV_HANDLE handle,
    YVC_CTRL_CMD cmd,
    YVC_CTRL_ARG *arg
);
```

[Operational overview]

The API function communicates with YVC-series devices via HID, controls the devices, and gets and sets parameters.

[Parameters]

- [IN] [YVC_DEV_HANDLE](#) handle

This parameter specifies the handle obtained by the [YVC_DeviceAttach](#) function.

- [IN] [YVC_CTRL_CMD](#) cmd

This parameter specifies the code for a control command. For the command codes, refer to "[Control command list](#)".

- [IN/OUT] [YVC_CTRL_ARG](#) *arg

This parameter specifies a pointer for the buffer that stores the parameters of the control command.

[Return value]

Error code	Description
YVCAPI_ERROR_NOERROR	Successful completion
YVCAPI_ERROR_PENDING	The request is accepted (running).
YVCAPI_ERROR_DEVNOTFOUND	The specified device is not found.
YVCAPI_ERROR_MEMEXHAUST	Memory is exhausted.
YVCAPI_ERROR_STATUS	A status error occurred.
YVCAPI_ERROR_INVALIDCOMMAND	The control command code is invalid.
YVCAPI_ERROR_INVALIDHANDLE	The handle is invalid.
YVCAPI_ERROR_INVALIDARGS	The argument is invalid.
YVCAPI_ERROR_CTRLEXEC	The API function failed to control the device.
YVCAPI_ERROR_HIDTIMEOUT	A time-out occurred during HID communication.
YVCAPI_ERROR_UNSUPPORTEDVER	The firmware version is not supported.
YVCAPI_ERROR_APPRUNNING	The API function cannot be executed because other related applications are running.
YVCAPI_ERROR_INVALIDPARAM	The parameter value of the device is invalid.

4.2-5 YVC_APIInfo

[Format]

```
int __stdcall YVC_APIInfo(
    YVC_API_INFO *info
);
```

[Operational overview]

This function gets the version information of the API.

version stores the version number in ASCII string format. The length of the string is fixed to four characters.

Example: In Ver. 1.00:

0	1	2	3	4
0x31	0x2E	0x30	0x30	0x00

[Parameters]

- [IN] [YVC_API_INFO](#) *info

This parameter specifies a pointer for the buffer to store the API information.

[Return value]

Error code	Description
YVCAPI_ERROR_NOERROR	Successful completion
YVCAPI_ERROR_INVALIDARGS	The argument is invalid.

4.3 Control commands

There are four types of control commands: Set-, Get-, Exe-, and Mon-prefixed commands.
The following table shows an overview of these commands:

Command type	Command name definition	Overview
Get-prefixed command	YVC_CMD_ Get <command-name>	Gets the information, status, and parameters of a device.
Set-prefixed command	YVC_CMD_ Set <command-name>	Sets the behavior and parameters of a device. The parameters can be divided into non-volatile and volatile ones.
Exe-prefixed command	YVC_CMD_ Exe <command-name>	Performs one of various functions of a device. These functions include the system restart, standby setting, factory settings, and automatic audio tuning.
Mon-prefixed command	YVC_CMD_ Mon <command-name>	Gets the operating status of a device. You can issue this command in a consecutive manner, but at limited time intervals. For the time intervals, refer to the page for each command.

4.4 Control command list

This section provides the lists of control commands.

* The following describes the symbols and expressions used in the tables below:

✓:	The command is supported.
✗:	The command is not supported.
Non-volatile:	The set values are maintained even after the power is off.
Volatile:	The set values are initialized upon system restart or power-off.
None:	In Set-prefixed commands, no parameters are available.
--:	No parameters are available because of a Get, Exe, or Mon-prefixed command.

4.4-1 Lists of commands by command type

1) Get-prefixed command

Defined command name	Overview	Setting value	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_CMD_GetSystemMode	Gets the operation mode of the system.	--	✓	✓	✓	✓
YVC_CMD_GetVersionInfo	Gets the version information.	--	✓	✓	✓	✓
YVC_CMD_GetModelName	Gets the model name.	--	✓	✓	✓	✓
YVC_CMD_GetSerialNumber	Gets the serial number.	--	✓	✓	✓	✓
YVC_CMD_GetMicNum	Gets the number of connected microphone units.	--	✓	✗	✗	✗
YVC_CMD_GetNoticeEvent	Gets the setting of the event to be notified of.	--	✓	✓	✓	✓
YVC_CMD_GetBTUse	Gets whether the Bluetooth function is enabled or disabled.	--	✓	✓	✓	✓
YVC_CMD_GetBTLocalName	Gets the Local Name setting for Bluetooth.	--	✓	✓	✓	✓
YVC_CMD_GetBTOperation	Gets the status of the Bluetooth function.	--	✓	✓	✓	✓
YVC_CMD_GetNoticeSoundUse	Gets whether the notification sound is enabled or disabled.	--	✓	✓	✓	✓
YVC_CMD_GetNoticeSoundVolume	Gets the volume setting of the notification sound.	--	✗	✓	✓	✓
YVC_CMD_GetVoiceGuideUse	Gets whether the voice guidance is enabled or disabled.	--	✓	✗	✗	✗
YVC_CMD_GetVoiceGuideLang	Gets the language setting of the voice guidance.	--	✓	✗	✗	✗
YVC_CMD_GetVoiceGuideVolume	Gets the volume setting of the voice guidance.	--	✓	✗	✗	✗
YVC_CMD_GetInitSpVolume	Gets the speaker volume setting when the device is connected via USB.	--	✓	✓	✓	✓
YVC_CMD_GetPowerOnState	Gets the operational setting at power-on.	--	✓	✗	✗	✗
YVC_CMD_GetMicMuteMode	Gets the mute mode setting for the microphone.	--	✓	✗	✗	✗
YVC_CMD_GetUsbSpeed	Gets the operating speed setting of the USB port.	--	✓	✗	✗	✗
YVC_CMD_GetExtTermMode	Gets the connection mode setting of the EXT terminal.	--	✗	✓	✓	✗
YVC_CMD_GetSpOutSelect	Gets the setting for which speaker is selected for output.	--	✓	✗	✗	✗
YVC_CMD_GetAudioInTerm	Gets the setting for the audio input terminal.	--	✓	✗	✗	✗
YVC_CMD_GetAudioVolume	Gets the volume setting.	--	✓	✓	✓	✓
YVC_CMD_GetAudioMute	Gets the mute setting.	--	✓	✓	✓	✓
YVC_CMD_GetMicAgc	Gets the AGC setting of the microphone.	--	✓	✓	✓	✓

YVC_CMD_GetMicUnitTrack	Gets the sound pickup mode setting for the microphone unit.	--	✓	X	X	X
YVC_CMD_GetMicCapsuleTrack	Gets the sound pickup mode setting for the microphone capsule.	--	X	✓	✓	X
YVC_CMD_GetMicFilter	Gets the frequency characteristic setting of the microphone.	--	✓	✓	✓	✓
YVC_CMD_GetSpFilter	Gets the frequency characteristic setting of the speaker.	--	✓	✓	✓	X
YVC_CMD_GetAATProgress	Gets the progress of automatic audio tuning.	--	✓	X	X	X
YVC_CMD_GetAATResult	Gets the measurement result of automatic audio tuning.	--	✓	X	X	X

2) Set-prefixed command

Defined command name	Overview	Setting value	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_CMD_SetNoticeEvent	Sets the event to be notified of.	Volatile	✓	✓	✓	✓
YVC_CMD_SetBTUse	Enables or disables the Bluetooth function.	Non-volatile	✓	✓	✓	✓
YVC_CMD_SetBTLocalName	Sets the Local Name for Bluetooth.	Non-volatile	✓	✓	✓	✓
YVC_CMD_SetBTOperation	Operates a Bluetooth function.	None	✓	✓	✓	✓
YVC_CMD_SetNoticeSoundUse	Enables or disables the notification sound.	Non-volatile	✓	✓	✓	✓
YVC_CMD_SetNoticeSoundVolume	Adjusts the volume of the notification sound.	Non-volatile	X	✓	✓	✓
YVC_CMD_SetVoiceGuideUse	Enables or disables the voice guidance.	Non-volatile	✓	X	X	X
YVC_CMD_SetVoiceGuideLang	Changes the language of the voice guidance.	Non-volatile	✓	X	X	X
YVC_CMD_SetVoiceGuideVolume	Adjusts the volume of the voice guidance.	Non-volatile	✓	X	X	X
YVC_CMD_SetInitSpVolume	Sets the speaker volume when the device is connected via USB.	Non-volatile	✓	✓	✓	✓
YVC_CMD_SetPowerOnState	Specifies an operation at power-on.	Non-volatile	✓	X	X	X
YVC_CMD_SetMicMuteMode	Sets the mute mode for the microphone.	Non-volatile	✓	X	X	X
YVC_CMD_SetUsbSpeed	Sets the operating speed of the USB port.	Non-volatile	✓	X	X	X
YVC_CMD_SetExtTermMode	Changes the connection mode setting of the EXT terminal.	Non-volatile	X	✓	✓	X
YVC_CMD_SetSpOutSelect	Sets which speaker is selected for output.	Non-volatile	✓	X	X	X
YVC_CMD_SetAudioInTerm	Changes the setting for the audio input terminal.	Non-volatile	✓	X	X	X
YVC_CMD_SetAudioVolume	Adjusts the volume setting.	Non-volatile (Note)	✓	✓	✓	✓
YVC_CMD_SetAudioMute	Changes the mute setting.	Non-volatile (Note)	✓	✓	✓	✓
YVC_CMD_SetMicAgc	Changes the AGC setting of the microphone.	Non-volatile	✓	✓	✓	✓
YVC_CMD_SetMicUnitTrack	Changes the sound pickup mode of the microphone unit.	Non-volatile	✓	X	X	X
YVC_CMD_SetMicCapsuleTrack	Changes the sound pickup mode of the microphone capsule.	Non-volatile	X	✓	✓	X
YVC_CMD_SetMicFilter	Changes the frequency characteristic of the microphone.	Non-volatile	✓	✓	✓	✓
YVC_CMD_SetSpFilter	Changes the frequency characteristic of the speaker.	Non-volatile	✓	✓	✓	X

(Note) Some volatile parameters are contained in the setting value.

3) Exe-prefixed command

Defined command name	Overview	Setting value	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_CMD_ExeSystemReboot	Restarts the system.	--	✓	✓	✓	✓
YVC_CMD_ExeFactoryReset	Resets the parameters to the factory settings.	--	✓	✓	✓	✓
YVC_CMD_ExeStandbyMode	Puts the device in the standby state.	--	✓	X	X	✓
YVC_CMD_ExeAAT	Controls execution of automatic audio tuning.	--	✓	X	X	X

4) Mon-prefixed command

Defined command name	Overview	Setting value	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_CMD_MonMicCapsule	Gets active microphone capsules.	--	✓	✓	✓	X
YVC_CMD_MonMicUnit	Gets the active microphone unit ID.	--	✓	X	X	X
YVC_CMD_MonHeadsetPortState	Gets the status of the headset function.	--	X	X	X	✓
YVC_CMD_MonBatteryLevel	Get the residual quantity of battery.	--	X	X	X	✓

4.4-2 Lists of commands by function

1) System control related commands

Defined command name	Overview	Setting value	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_CMD_GetSystemMode	Gets the operation mode of the system.	--	✓	✓	✓	✓
YVC_CMD_GetVersionInfo	Gets the version information.	--	✓	✓	✓	✓
YVC_CMD_GetModelName	Gets the model name.	--	✓	✓	✓	✓
YVC_CMD_GetSerialNumber	Gets the serial number.	--	✓	✓	✓	✓
YVC_CMD_GetMicNum	Gets the number of connected microphone units.	--	✓	X	X	X
YVC_CMD_SetNoticeEvent	Sets the event to be notified of.	Volatile	✓	✓	✓	✓
YVC_CMD_GetNoticeEvent	Gets the setting of the event to be notified of.	--	✓	✓	✓	✓
YVC_CMD_ExeSystemReboot	Restarts the system.	--	✓	✓	✓	✓
YVC_CMD_ExeFactoryReset	Resets the parameters to the factory settings.	--	✓	✓	✓	✓
YVC_CMD_ExeStandbyMode	Puts the device in the standby state.	--	✓	X	X	✓
YVC_CMD_SetBTUse	Enables or disables the Bluetooth function.	Non-volatile	✓	✓	✓	✓
YVC_CMD_GetBTUse	Gets whether the Bluetooth function is enabled or disabled.	--	✓	✓	✓	✓
YVC_CMD_SetBTLocalName	Sets the Local Name for Bluetooth.	Non-volatile	✓	✓	✓	✓
YVC_CMD_GetBTLocalName	Gets the Local Name setting for Bluetooth.	--	✓	✓	✓	✓
YVC_CMD_SetBTOperation	Operates a Bluetooth function.	None	✓	✓	✓	✓
YVC_CMD_GetBTOperation	Gets the status of the Bluetooth function.	--	✓	✓	✓	✓
YVC_CMD_SetNoticeSoundUse	Enables or disables the notification sound.	Non-volatile	✓	✓	✓	✓
YVC_CMD_GetNoticeSoundUse	Gets whether the notification sound is enabled or disabled.	--	✓	✓	✓	✓
YVC_CMD_SetVoiceGuideUse	Enables or disables the voice guidance.	Non-volatile	✓	X	X	X
YVC_CMD_GetVoiceGuideUse	Gets whether the voice guidance is enabled or disabled.	--	✓	X	X	X
YVC_CMD_SetVoiceGuideLang	Changes the language of the voice guidance.	Non-volatile	✓	X	X	X
YVC_CMD_GetVoiceGuideLang	Gets the language setting of the voice guidance.	--	✓	X	X	X
YVC_CMD_SetVoiceGuideVolume	Adjusts the volume of the voice guidance.	Non-volatile	✓	X	X	X
YVC_CMD_GetVoiceGuideVolume	Gets the volume setting of the voice guidance.	--	✓	X	X	X
YVC_CMD_SetNoticeSoundVolume	Adjusts the volume of the notification sound.	Non-volatile	X	✓	✓	✓
YVC_CMD_GetNoticeSoundVolume	Gets the volume setting of the notification sound.	--	X	✓	✓	✓
YVC_CMD_SetInitSpVolume	Sets the speaker volume when the device is connected via USB.	Non-volatile	✓	✓	✓	✓
YVC_CMD_GetInitSpVolume	Gets the speaker volume setting when the device is connected via USB.	--	✓	✓	✓	✓
YVC_CMD_SetPowerOnState	Specifies an operation at power-on.	Non-volatile	✓	X	X	X
YVC_CMD_GetPowerOnState	Gets the operational setting at power-on.	--	✓	X	X	X
YVC_CMD_SetMicMuteMode	Sets the mute mode for the microphone.	Non-volatile	✓	X	X	X
YVC_CMD_GetMicMuteMode	Gets the mute mode setting for the microphone.	--	✓	X	X	X
YVC_CMD_SetUsbSpeed	Sets the operating speed of the USB port.	Non-volatile	✓	X	X	X
YVC_CMD_GetUsbSpeed	Gets the operating speed setting of the USB port.	--	✓	X	X	X

YVC_CMD_SetExtTermMode	Changes the connection mode setting of the EXT terminal.	Non-volatile	X	✓	✓	X
YVC_CMD_GetExtTermMode	Gets the connection mode setting of the EXT terminal.	--	X	✓	✓	X

2) Audio control related commands

Defined command name	Overview	Setting value	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_CMD_SetSpOutSelect	Sets which speaker is selected for output.	Non-volatile	✓	X	X	X
YVC_CMD_GetSpOutSelect	Gets the setting for which speaker is selected for output.	--	✓	X	X	X
YVC_CMD_SetAudioInTerm	Changes the setting for the audio input terminal.	Non-volatile	✓	X	X	X
YVC_CMD_GetAudioInTerm	Gets the setting for the audio input terminal.	--	✓	X	X	X
YVC_CMD_SetAudioVolume	Adjusts the volume setting.	Non-volatile (Note)	✓	✓	✓	✓
YVC_CMD_GetAudioVolume	Gets the volume setting.	--	✓	✓	✓	✓
YVC_CMD_SetAudioMute	Changes the mute setting.	Non-volatile (Note)	✓	✓	✓	✓
YVC_CMD_GetAudioMute	Gets the mute setting.	--	✓	✓	✓	✓
YVC_CMD_SetMicAgc	Changes the AGC setting of the microphone.	Non-volatile	✓	✓	✓	✓
YVC_CMD_GetMicAgc	Gets the AGC setting of the microphone.	--	✓	✓	✓	✓
YVC_CMD_SetMicUnitTrack	Changes the sound pickup mode of the microphone unit.	Non-volatile	✓	X	X	X
YVC_CMD_GetMicUnitTrack	Gets the sound pickup mode setting for the microphone unit.	--	✓	X	X	X
YVC_CMD_SetMicCapsuleTrack	Changes the sound pickup mode of the microphone capsule.	Non-volatile	X	✓	✓	X
YVC_CMD_GetMicCapsuleTrack	Gets the sound pickup mode setting for the microphone capsule.	--	X	✓	✓	X
YVC_CMD_SetMicFilter	Changes the frequency characteristic of the microphone.	Non-volatile	✓	✓	✓	✓
YVC_CMD_GetMicFilter	Gets the frequency characteristic setting of the microphone.	--	✓	✓	✓	✓
YVC_CMD_SetSpFilter	Changes the frequency characteristic of the speaker.	Non-volatile	✓	✓	✓	X
YVC_CMD_GetSpFilter	Gets the frequency characteristic setting of the speaker.	--	✓	✓	✓	X

(Note) Some volatile parameters are contained in the setting value.

3) Automatic audio tuning control related commands

Defined command name	Overview	Setting value	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_CMD_ExecAAT	Controls execution of automatic audio tuning.	--	✓	X	X	X
YVC_CMD_GetAATProgress	Gets the progress of automatic audio tuning.	--	✓	X	X	X
YVC_CMD_GetAATResult	Gets the measurement result of automatic audio tuning.	--	✓	X	X	X

4) Device monitoring related commands

Defined command name	Overview	Setting value	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_CMD_MonMicCapsule	Gets active microphone capsules.	--	✓	✓	✓	X
YVC_CMD_MonMicUnit	Gets the active microphone unit ID.	--	✓	X	X	X
YVC_CMD_MonHeadsetPortState	Gets the status of the headset function.	--	X	X	X	✓

YVC_CMD_MonBattelyLevel	Get the residual quantity of battely.	--	X	X	X	✓
---	---------------------------------------	----	---	---	---	---

4.5 Control command in detail

4.5-1 YVC_CMD_GetSystemMode

[Operational overview]

This command gets the operation mode of the system.

There are two operation modes: NORMAL and FWUP.

In NORMAL mode, the system starts up as a microphone and speaker system (speakerphone). This is the default mode.

On the other hand, the FWUP mode is used to update the firmware of the device. In general, the system does not start up in this mode. It is when the firmware is updated or when the system cannot start up in NORMAL mode due to a system problem that the system starts up in this mode.

[Format]

```
typedef struct {
    YVC_SYSTEM_MODE mode;
} YVC_ARG_SystemMode;
```

[Parameters]

- [OUT] YVC_SYSTEM_MODE mode

The command returns the operation mode of the system.

Parameter symbol name	Description
YVC_SYSTEM_MODE_FWUP	The system is running in FWUP mode.
YVC_SYSTEM_MODE_NORMAL	The system is running in NORMAL mode.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✓
YVC-300	Ver. 1.04 or later	✓	✓
YVC-330	Ver. 1.02 or later	✓	✓
YVC-200	Ver. 1.01 or later	✓	✓

4.5-2 YVC_CMD_GetVersionInfo

[Operational overview]

This command gets the version information.

If the command succeeds, the *version* variable stores the version number in ASCII string format. The length of the string is fixed to four characters.

Example: In Ver. 1.06:

0	1	2	3	4
0x31	0x2E	0x30	0x36	0x00

[Format]

```
typedef struct {
    char    version[YVC_LEN_VERSION_INFO+1];
} YVC_ARG_VersionInfo;
```

[Parameters]

- [OUT] **char** **version[YVC_LEN_VERSION_INFO+1]**

The command returns the version information.

* It contains a terminating character (0x00).

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✓
YVC-300	Ver. 1.04 or later	✓	✓
YVC-330	Ver. 1.02 or later	✓	✓
YVC-200	Ver. 1.01 or later	✓	✓

4.5-3 YVC_CMD_GetModelName

[Operational overview]

This command gets the model name.

If the command succeeds, the *name* variable stores the model name in ASCII string format. The maximum length of the string is defined as 40 characters, but the command actually returns the strings as shown in the following table:

Model name	NORMAL mode	FWUP mode
YVC-1000	"YVC-1000"	"YVC-1000 Updater"
YVC-300	"YVC-300"	"YVC-300 Updater"
YVC-330	"YVC-330"	"YVC-330 Updater"
YVC-200	"YVC-200"	"YVC-200 Updater"

[Format]

```
typedef struct {
    char    name[YVC_LEN_MODEL_NAME+1];
} YVC_ARG_ModelName;
```

[Parameters]

- [OUT] **char** **name**[YVC_LEN_MODEL_NAME+1]

The command returns the model name.

* It contains a terminating character (0x00).

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✓
YVC-300	Ver. 1.04 or later	✓	✓
YVC-330	Ver. 1.02 or later	✓	✓
YVC-200	Ver. 1.01 or later	✓	✓

4.5-4 YVC_CMD_GetSerialNumber

[Operational overview]

This command gets the serial number.

If the command succeeds, the *serialNumber* variable stores the serial number in ASCII string format. The maximum length of the string is 16 characters.

In the YVC-1000 model, specifying a microphone unit ID of a microphone unit that is not connected returns an error ([YVC-API_ERROR_CTRLEXEC](#)). Also, in FWUP mode, it cannot get the serial number of a microphone unit. If this command is specified in this mode, an error ([YVC-API_ERROR_INVALIDARGS](#)) is returned.

[Format]

```
typedef struct {
    YVC_SERIALNUMBER_TYPE type;
    char serialNumber[YVC_LEN_SERIAL_NUMBER+1];
} YVC_ARG_SerialNumber;
```

[Parameters]

• [IN] YVC_SERIALNUMBER_TYPE type

In YVC-1000, this parameter specifies the unit to be obtained.

In YVC-300, YVC-330 and YVC-200, it is not required. (It is ignored if specified.)

Parameter symbol name	Description
YVC_SERIALNUMBER_TYPE SPUNIT	Specifies to select the Control Unit.
YVC_SERIALNUMBER_TYPE MICUNIT0	Specifies to select microphone unit 0.
YVC_SERIALNUMBER_TYPE MICUNIT1	Specifies to select microphone unit 1.
YVC_SERIALNUMBER_TYPE MICUNIT2	Specifies to select microphone unit 2.
YVC_SERIALNUMBER_TYPE MICUNIT3	Specifies to select microphone unit 3.
YVC_SERIALNUMBER_TYPE MICUNIT4	Specifies to select microphone unit 4.

• [OUT] char serialNumber[YVC_LEN_SERIAL_NUMBER+1]

The command returns the serial number.

* It contains a terminating character (0x00).

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✓
YVC-300	Ver. 1.04 or later	✓	✓
YVC-330	Ver. 1.02 or later	✓	✓
YVC-200	Ver. 1.01 or later	✓	✓

4.5-5 YVC_CMD_GetMicNum

[Operational overview]

This command gets the number of connected microphone units.

[Format]

```
typedef struct {  
    unsigned int    micNum;  
} YVC_ARG_MicNum;
```

[Parameters]

- [OUT] unsigned int micNum

The command returns the number of connected microphone units (0 to 5).

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-6 YVC_CMD_ExeSystemReboot

[Operational overview]

This command restarts the system.

[Format]

```
typedef struct {
    void (*callback)(YVC_CTRL_CMD cmd, int err, void *user);
    void *user;
} YVC_ARG_SystemReboot;
```

[Parameters]

- [IN] void (*callback)(YVC_CTRL_CMD cmd, int err, void *user)

This parameter specifies the address of the callback function for command completion notification.

If the parameter is set to NULL, the command operates as a completed type.

If the parameter is set to a value other than NULL, the command operates as an incomplete type.

The incomplete-type command returns [YVCAPI_ERROR_PENDING](#) when it is accepted. Then, the command is processed, and the destination is notified of the result through the callback function.

The parameters when the command operates as the incomplete type are as follows:

cmd returns [YVC_CMD_ExeSystemReboot](#).

err returns an [error code](#).

user returns the specified address as it is.

[Notes]

Do not perform a time-consuming operation or call [this API function](#) in the callback function.

- [IN] void *user

This parameter specifies the address of user management data.

This parameter setting is enabled only when the command is executed as the incomplete type. If the data is not needed, set the parameter to NULL.

It is ignored when the command is executed as the completed type.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✓
YVC-300	Ver. 1.04 or later	✓	✓
YVC-330	Ver. 1.02 or later	✓	✓
YVC-200	Ver. 1.01 or later	✓	✓

4.5-7 YVC_CMD_ExeFactoryReset

[Operational overview]

This command resets the parameters to the factory settings.

[Format]

```
typedef struct {
    void (*callback)(YVC_CTRL_CMD cmd, int err, void *user);
    void *user;
} YVC_ARG_FactoryReset;
```

[Parameters]

- [IN] void (*callback)(YVC_CTRL_CMD cmd, int err, void *user)

This parameter specifies the address of the callback function for command completion notification.

If the parameter is set to NULL, the command operates as a completed type.

If the parameter is set to a value other than NULL, the command operates as an incomplete type.

The incomplete-type command returns [YVCAPI_ERROR_PENDING](#) when it is accepted. Then, the command is processed, and the destination is notified of the result through the callback function.

The parameters when the command operates as the incomplete type are as follows:

cmd returns [YVC_CMD_ExeFactoryReset](#).

err returns an [error code](#).

user returns the specified address as it is.

[Notes]

Do not perform a time-consuming operation or call [this API function](#) in the callback function.

- [IN] void *user

This parameter specifies the address of user management data.

This parameter setting is enabled only when the command is executed as the incomplete type. If the data is not needed, set the parameter to NULL.

It is ignored when the command is executed as the completed type.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-8 YVC_CMD_ExeStandbyMode

[Operational overview]

This command puts a YVC-1000 or YVC-200 device in the standby state.

However, there is no command to cancel, and wake up the device from, the standby state. In YVC-1000, press the power button on the YVC-1000 Control Unit to cancel that state. In YVC-200, hold the power button on the YVC-200 or connecting a USB cable to the YVC-200 to cancel that state.

[Format]

No parameters. Specify NULL for the *arg* parameter.

[Parameters]

No parameters.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-9 YVC_CMD_SetNoticeEvent

[Operational overview]

This command specifies an event that a notification destination is notified of.

If the event notification function is used, a notification destination must be pre-registered by using the [YVC_DeviceAttach](#) function.

For details, refer to the page on "[Event details](#)".

[Notes]

The event setting is a volatile parameter. Restarting the device resets the setting, and therefore it must be set again as needed.

[Format]

```
typedef struct {
    unsigned int event;
} YVC_ARG_NoticeEvent;
```

[Parameters]

- [IN] unsigned int event

This parameter specifies the event that the destination is notified of. You can specify more than one event.

Parameter symbol name	Value	Description	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_PRM_NOTICE_EVENT_MIC_UNIT_NUM	0x1	Notifies the destination of the change in the connection status of a microphone unit.	✓	X	X	X
YVC_PRM_NOTICE_EVENT_MIC_MUTE_STATE	0x2	Notifies the destination of the mute status of a microphone. It is, however, limited to cases where the mute button is pressed. The destination is not notified of status changes due to a command setup or other causes.	✓	✓	✓	✓
YVC_PRM_NOTICE_EVENT_HOOK_BTN	0x4	Notifies the destination of the change in the status of the on/off-hook button when it is pressed. However, no notification is made when it is used as an incoming response on Bluetooth or call ending (call disconnect) function.	X	✓	✓	X

The value defaults to 0x0 (no event).

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.10 or later	✓	X
YVC-300	Ver. 1.09 or later	✓	X
YVC-330	Ver. 1.02 or later	✓	X
YVC-200	Ver. 1.01 or later	✓	X

4.5-10 YVC_CMD_GetNoticeEvent

[Operational overview]

This command gets the settings of the event to be notified of.

[Format]

```
typedef struct {  
    unsigned int event;  
} YVC_ARG_NoticeEvent;
```

[Parameters]

- [OUT] unsigned int event

The command returns the settings of the event to be notified of.
For details, refer to the page for [YVC_CMD_SetNoticeEvent](#).

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.10 or later	✓	✗
YVC-300	Ver. 1.09 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-11 YVC_CMD_SetBTUse

[Operational overview]

This command enables or disables the Bluetooth function.

[Format]

```
typedef struct {
    BOOL    onoff;
} YVC_ARG_BTUse;
```

[Parameters]

- [IN] **BOOL onoff**

This parameter specifies whether to enable or disable the Bluetooth function.

Value	Description	Default value
0	Disables the Bluetooth function.	
1	Enables the Bluetooth function.	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	X
YVC-300	Ver. 1.04 or later	✓	X
YVC-330	Ver. 1.02 or later	✓	X
YVC-200	Ver. 1.01 or later	✓	X

4.5-12 YVC_CMD_GetBTUse

[Operational overview]

This command gets whether the Bluetooth function is enabled or disabled.

[Format]

```
typedef struct {
    BOOL    onoff;
} YVC_ARG_BTUse;
```

[Parameters]

- [OUT] **BOOL onoff**

The command returns the setting of the Bluetooth function.

Value	Description	Default value
0	Indicates that the Bluetooth function is disabled.	
1	Indicates that the Bluetooth function is enabled.	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-13 YVC_CMD_SetBTLocalName

[Operational overview]

This command sets the Local Name for Bluetooth.

Setting the Local Name enables users to change the name of the device that is displayed on a Bluetooth-connected smartphone or tablet (hereinafter called a terminal). This is useful when a single terminal is connected to multiple devices. Possible character codes range from 0x20 to 0x7E in ASCII code. Also, the maximum number of possible letters is 11 characters ([YVC_LEN_BT_LOCAL_NAME](#)).

Note that the model name (for example, "YVC-1000") cannot be changed or removed.

Model	Default value	Name displayed on the device
YVC-1000	"Yamaha"	"YVC-1000 Yamaha"
YVC-300	"Yamaha"	"YVC-300 Yamaha"
YVC-330	"Yamaha"	"YVC-330 Yamaha"
YVC-200	"Yamaha"	"YVC-200 Yamaha"

[Notes]

Depending on the terminals, the name displayed may not be updated until the pairing information is removed.

[Format]

```
typedef struct {
    char    name[YVC_LEN_BT_LOCAL_NAME_TOTAL+1];
} YVC_ARG_BTLocalName;
```

[Parameters]

- [IN] **char name**[YVC_LEN_BT_LOCAL_NAME_TOTAL+1]

This parameter specifies the Local Name for Bluetooth.

Make sure that the string contains a terminating character (0x00).

Specify an empty character (0x00) to restore the default settings.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.09 or later	✓	✗
YVC-300	Ver. 1.08 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-14 YVC_CMD_GetBTLocalName

[Operational overview]

This command gets the Local Name setting for Bluetooth.

If the command succeeds, the *name* variable stores the Bluetooth Local Name in ASCII string format. The maximum length of the string is 20 characters ([YVC_LEN_BT_LOCAL_NAME_TOTAL](#)).

[Format]

```
typedef struct {
    char    name[YVC_LEN_BT_LOCAL_NAME_TOTAL+1];
} YVC_ARG_BTLocalName;
```

[Parameters]

- [OUT] **char** **name**[YVC_LEN_BT_LOCAL_NAME_TOTAL+1]

The command returns the Local Name setting for Bluetooth.

* It contains a terminating character (0x00).

The command returns either of the following strings by default.

Model	Default value
YVC-1000	"YVC-1000 Yamaha"
YVC-300	"YVC-300 Yamaha"
YVC-330	"YVC-330 Yamaha"
YVC-200	"YVC-200 Yamaha"

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.09 or later	✓	✗
YVC-300	Ver. 1.08 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-15 YVC_CMD_SetBTOperation

[Operational overview]

This command operates a Bluetooth function.

This operation is available only when the Bluetooth function is enabled. If the function is disabled, the command returns an error ([YVC-API_ERROR_CTRLEXEC](#)).

The ready state, the pairing, or reconnecting operation is canceled when 90 seconds elapse before a connection or pairing process is completed.

[Format]

```
typedef struct {
    union {
        YVC_BT_OPERATION_REQ req;
        YVC_BT_OPERATION_STAT stat;
    } u;
} YVC_ARG_BTOperation;
```

[Parameters]

- [IN] YVC_BT_OPERATION_REQ req

This parameter specifies a request for operation of the Bluetooth function.

Parameter symbol name	Description	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_BT_OPERATION_REQ_SLEEP	Used to enter the sleep state.	✓	✓	✓	X
YVC_BT_OPERATION_REQ_READY	Used to enter the ready state.	✓	✓	✓	X
YVC_BT_OPERATION_REQ_DISCONNECT	Used to disconnect the connection.	✓	✓	✓	✓
YVC_BT_OPERATION_REQ_PAIRING_ON	Used to perform a pairing operation.	✓	✓	✓	✓
YVC_BT_OPERATION_REQ_PAIRING_OFF	Used to cancel the pairing.	✓	✓	✓	✓
YVC_BT_OPERATION_REQ_LINKBACK	Used to reconnect to last connected device.	X	X	X	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	X
YVC-300	Ver. 1.04 or later	✓	X
YVC-330	Ver. 1.02 or later	✓	X
YVC-200	Ver. 1.01 or later	✓	X

4.5-16 YVC_CMD_GetBTOperation

[Operational overview]

This command gets the status of the Bluetooth function.

This operation is available only when the Bluetooth function is enabled. If the function is disabled, the command returns an error ([YVC-API_ERROR_CTRLEXEC](#)).

[Format]

```
typedef struct {
    union {
        YVC_BT_OPERATION_REQ req;
        YVC_BT_OPERATION_STAT stat;
    } u;
} YVC_ARG_BTOperation;
```

[Parameters]

- [OUT] YVC_BT_OPERATION_STAT stat

The command returns the operating status of the Bluetooth function.

Parameter symbol name	Description	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_BT_OPERATION_STAT_SLEEP	Indicates the sleep state.	✓	✓	✓	✓
YVC_BT_OPERATION_STAT_READY	Indicates the ready state.	✓	✓	✓	✓
YVC_BT_OPERATION_STAT_DISCONNECTING	Indicates that the disconnection process is in progress.	✓	✓	✓	✓
YVC_BT_OPERATION_STAT_PAIRING	Indicates that devices are paired.	✓	✓	✓	✓
YVC_BT_OPERATION_STAT_CONNECTED	Indicates that devices are connected together.	✓	✓	✓	✓
YVC_BT_OPERATION_STAT_CONNECTING	Indicates that devices are connecting.	X	X	X	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	X
YVC-300	Ver. 1.04 or later	✓	X
YVC-330	Ver. 1.02 or later	✓	X
YVC-200	Ver. 1.01 or later	✓	X

4.5-17 YVC_CMD_SetNoticeSoundUse

[Operational overview]

This command enables or disables the notification sound.

The notification sound beeps when:

In the YVC-1000 model, a user performs an operation, such as volume adjustment, speaker selection, automatic audio tuning, Bluetooth/NFC, and language selection.

In the YVC-300 and YVC-330 model, a user performs an operation, such as Bluetooth/NFC, setting/audio mode switching, connection mode selection, enabling or disabling Bluetooth, factory settings, and a daisy-chain connection with the main unit.

In the YVC-200 model, a user performs an operation, such as Bluetooth/NFC, volume adjustment, speaker and microphone mute switching and factory settings.

[Format]

```
typedef struct {
    BOOL    onoff;
} YVC_ARG_NoticeSoundUse;
```

[Parameters]

- [IN] **BOOL onoff**

This parameter specifies whether to enable or disable the notification sound.

Value	Description	Default value
0	Disables the notification sound.	
1	Enables the notification sound.	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-18 YVC_CMD_GetNoticeSoundUse

[Operational overview]

This command gets whether the notification sound is enabled or disabled.

[Format]

```
typedef struct {
    BOOL    onoff;
} YVC_ARG_NoticeSoundUse;
```

[Parameters]

- [OUT] **BOOL onoff**

The command returns the notification sound setting.

Value	Description	Default value
0	Indicates that the notification sound is disabled.	
1	Indicates that the notification sound is enabled.	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-19 YVC_CMD_SetNoticeSoundVolume

[Operational overview]

This command adjusts the volume of the notification sound.

[Format]

```
typedef struct {  
    short    volume;  
} YVC_ARG_NoticeSoundVolume;
```

[Parameters]

- [IN] **short volume**

This parameter specifies the volume value (in dB) of the notification sound.

Possible values (dB)	Default value (dB)	Setting value
-50 to 0	0	Non-volatile

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-20 YVC_CMD_GetNoticeSoundVolume

[Operational overview]

This command gets the volume setting of the notification sound.

[Format]

```
typedef struct {  
    short    volume;  
} YVC_ARG_NoticeSoundVolume;
```

[Parameters]

- [OUT] short volume

The command returns the volume setting (in dB) of the notification sound.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-21 YVC_CMD_SetVoiceGuideUse

[Operational overview]

This command enables or disables the voice guidance.

[Format]

```
typedef struct {
    BOOL    onoff;
} YVC_ARG_VoiceGuideUse;
```

[Parameters]

- [IN] **BOOL onoff**

This parameter specifies whether to enable or disable the voice guidance.

Value	Description	Default value
0	Disables the voice guidance.	
1	Enables the voice guidance.	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-22 YVC_CMD_GetVoiceGuideUse

[Operational overview]

This command gets whether the voice guidance is enabled or disabled.

[Format]

```
typedef struct {
    BOOL    onoff;
} YVC_ARG_VoiceGuideUse;
```

[Parameters]

- [OUT] **BOOL onoff**

The command returns the setting of the voice guidance.

Value	Description	Default value
0	Indicates that the voice guidance is disabled.	
1	Indicates that the voice guidance is enabled.	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-23 YVC_CMD_SetVoiceGuideLang

[Operational overview]

This command changes the language of the voice guidance.

[Format]

```
typedef struct {
    YVC_VOICE_GUIDE_LANG    type;
} YVC_ARG_VoiceGuideLang;
```

[Parameters]

- [IN] YVC_VOICE_GUIDE_LANG type

This parameter specifies the language of the voice guidance.

Parameter symbol name	Description	Default value
YVC_VOICE_GUIDE_LANG_EN	Sets the language to English.	✓
YVC_VOICE_GUIDE_LANG_JA	Sets the language to Japanese.	
YVC_VOICE_GUIDE_LANG_ZN	Sets the language to Chinese.	
YVC_VOICE_GUIDE_LANG_KO	Sets the language to Korean.	
YVC_VOICE_GUIDE_LANG_FR	Sets the language to French.	
YVC_VOICE_GUIDE_LANG_ES	Sets the language to Spanish.	
YVC_VOICE_GUIDE_LANG_DE	Sets the language to German.	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-24 YVC_CMD_GetVoiceGuideLang

[Operational overview]

This command gets the language setting of the voice guidance.

[Format]

```
typedef struct {
    YVC_VOICE_GUIDE_LANG    type;
} YVC_ARG_VoiceGuideLang;
```

[Parameters]

- [IN] YVC_VOICE_GUIDE_LANG type

The command returns the language setting of the voice guidance.

Parameter symbol name	Description	Default value
YVC_VOICE_GUIDE_LANG_EN	Indicates that the language is set to English.	✓
YVC_VOICE_GUIDE_LANG_JA	Indicates that the language is set to Japanese.	
YVC_VOICE_GUIDE_LANG_ZN	Indicates that the language is set to Chinese.	
YVC_VOICE_GUIDE_LANG_KO	Indicates that the language is set to Korean.	
YVC_VOICE_GUIDE_LANG_FR	Indicates that the language is set to French.	
YVC_VOICE_GUIDE_LANG_ES	Indicates that the language is set to Spanish.	
YVC_VOICE_GUIDE_LANG_DE	Indicates that the language is set to German.	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-25 YVC_CMD_SetVoiceGuideVolume

[Operational overview]

This command adjusts the volume of the voice guidance.

[Format]

```
typedef struct {
    short    volume;
} YVC_ARG_VoiceGuideVolume;
```

[Parameters]

- [IN] **short volume**

This parameter specifies the volume value (in dB) of the voice guidance.

Possible values (dB)	Default value (dB)	Setting value
-50 to 0	0	Non-volatile

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-26 YVC_CMD_GetVoiceGuideVolume

[Operational overview]

This command gets the volume setting of the voice guidance.

[Format]

```
typedef struct {  
    short    volume;  
} YVC_ARG_VoiceGuideVolume;
```

[Parameters]

- [OUT] **short** **volume**

The command returns the volume setting (in dB) of the voice guidance.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-27 YVC_CMD_SetInitSpVolume

[Operational overview]

This command sets the speaker volume when the device is connected via USB.

It specifies whether the speaker volume when the host is connected to the device via USB uses the value either held by the device or specified by the host.

The following examples show how the command works:

If *type* is set to YVC_INIT_SP_VOLUME_DEVICE:

State of the USB connection	Host (PC)	Device
Before the USB connection	-4.5 dB	-10.0 dB
After the USB connection	-10.0 dB	-10.0 dB

If *type* is set to YVC_INIT_SP_VOLUME_HOST:

State of the USB connection	Host (PC)	Device
Before the USB connection	-4.5 dB	-10.0 dB
After the USB connection	-4.5 dB	-4.5 dB

[Format]

```
typedef struct {
    YVC_INIT_SP_VOLUME    type;
} YVC_ARG_InitSpVolume;
```

[Parameters]

- [IN] YVC_INIT_SP_VOLUME type

This parameter specifies the speaker volume when the device is connected via USB.

Parameter symbol name	Description	Default value
YVC_INIT_SP_VOLUME_DEVICE	Specifies to use the speaker volume held by the device.	✓
YVC_INIT_SP_VOLUME_HOST	Specifies to use the speaker volume specified by the host (PC).	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	X
YVC-300	Ver. 1.04 or later	✓	X
YVC-330	Ver. 1.02 or later	✓	X
YVC-200	Ver. 1.01 or later	✓	X

4.5-28 YVC_CMD_GetInitSpVolume

[Operational overview]

This command gets the setting for the speaker volume when the device is connected via USB.

[Format]

```
typedef struct {
    YVC_INIT_SP_VOLUME    type;
} YVC_ARG_InitSpVolume;
```

[Parameters]

- [OUT] YVC_INIT_SP_VOLUME type

The command returns the setting for the speaker volume when the device is connected via USB.

Parameter symbol name	Description	Default value
YVC_INIT_SP_VOLUME_DEVICE	Indicates that the "speaker volume held by the device" is specified.	✓
YVC_INIT_SP_VOLUME_HOST	Indicates that the "speaker volume specified by the host (PC)" is specified.	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-29 YVC_CMD_SetPowerOnState

[Operational overview]

This command specifies an operation at power-on.

[Format]

```
typedef struct {
    YVC_POWER_ON_STATE    mode;
} YVC_ARG_PowerOnState;
```

[Parameters]

- [IN] YVC_POWER_ON_STATE mode

This parameter specifies the operation at power-on.

Parameter symbol name	Description	Default value
YVC_POWER_ON_STATE_HOLD	Specifies that the device transitions to the state at power-off.	
YVC_POWER_ON_STATE_STANDBY	Specifies that the device transitions to the standby state.	✓
YVC_POWER_ON_STATE_RUN	Specifies that the device transitions to the running state.	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-30 YVC_CMD_GetPowerOnState

[Operational overview]

This command gets the operational setting at power-on.

[Format]

```
typedef struct {
    YVC_POWER_ON_STATE    mode;
} YVC_ARG_PowerOnState;
```

[Parameters]

- [OUT] YVC_POWER_ON_STATE mode

The command returns the operational setting at power-on.

Parameter symbol name	Description	Default value
YVC_POWER_ON_STATE_HOLD	Indicates that the device is set to "transition to the state at power-off".	
YVC_POWER_ON_STATE_STANDBY	Indicates that the device is set to "transition to the standby state".	✓
YVC_POWER_ON_STATE_RUN	Indicates that the device is set to "transition to the running state".	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-31 YVC_CMD_SetMicMuteMode

[Operational overview]

This command sets the mute mode in which the microphone operates.

A microphone is unmuted if the mute mode is changed by this command while the microphone is muted.

If this operation mutes or unmutes the microphone, the state notification event for microphone muting ([YVC_EVENT_MIC_MUTE_STATE](#)) is not made even when the event notification is enabled (because the mute button is not pressed).

[Format]

```
typedef struct {
    YVC_MIC_MUTE_MODE    mode;
} YVC_ARG_MicMuteMode;
```

[Parameters]

- [IN] YVC_MIC_MUTE_MODE mode

This parameter specifies the mute mode of the microphone.

Parameter symbol name	Description	Default value
YVC_MIC_MUTE_MODE_GROUP	Indicates that the microphones operate in group mute mode (work together).	✓
YVC_MIC_MUTE_MODE_INDIVIDUAL	Indicates that the microphones operate in individual mute mode (work individually).	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.10 or later	✓	✗

4.5-32 YVC_CMD_GetMicMuteMode

[Operational overview]

This command gets the mute mode setting for the microphone.

[Format]

```
typedef struct {
    YVC_MIC_MUTE_MODE    mode;
} YVC_ARG_MicMuteMode;
```

[Parameters]

- [OUT] YVC_MIC_MUTE_MODE mode

The command returns the mute mode setting for the microphone.

Parameter symbol name	Description	Default value
YVC_MIC_MUTE_MODE_GROUP	Indicates that the microphones are "in group mute mode (working together)".	✓
YVC_MIC_MUTE_MODE_INDIVIDUAL	Indicates that the microphones are "in individual mute mode (working individually)".	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.10 or later	✓	✗

4.5-33 YVC_CMD_SetUsbSpeed

[Operational overview]

This command sets the operating speed of the USB port.

This speed setting is enabled at startup of the device. For the setting change to take effect, restart the device with the *callback* parameter specified or by using the [YVC_CMD_ExeSystemReboot](#) command. Alternatively, turn off and then on the device.

[Format]

```
typedef struct {
    YVC_USB_MODE mode;
    struct {
        void (*callback)(YVC_CTRL_CMD cmd, int err, void *user);
        void *user;
    } set;
} YVC_ARG_UsbSpeed;
```

[Parameters]

- [IN] YVC_USB_MODE mode

This parameter specifies the operating speed of the USB port.

Parameter symbol name	Description	Default value
YVC_USB_MODE_HI_SPEED	The USB port operates in Hi-Speed mode.	✓
YVC_USB_MODE_FULL_SPEED	The USB port operates in Full-Speed mode.	

- [IN] void (*callback)(YVC_CTRL_CMD cmd, int err, void *user)

This parameter specifies the address of the callback function for command completion notification.

If the parameter is set to NULL, only the speed is set, but the device is not restarted. The command operates as a completed type.

If the parameter is set to a value other than NULL, the device is restarted after the setting is done. The command operates as an incomplete type.

The incomplete-type command returns [YVC_API_ERROR_PENDING](#) when it is accepted. Then, the command is processed, and the destination is notified of the result through the callback function.

The parameters when the command operates as the incomplete type are as follows:

cmd returns [YVC_CMD_SetUsbSpeed](#).

err returns an [error code](#).

user returns the specified address as it is.

[Notes]

Do not perform a time-consuming operation or call [this API function](#) in the callback function.

- [IN] void *user

This parameter specifies the address of user management data.

This parameter setting is enabled only when the command is executed as the incomplete type. If the data is not needed, set the parameter to NULL.

It is ignored when the command is executed as the completed type.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.03 or later	✓	X

4.5-34 YVC_CMD_GetUsbSpeed

[Operational overview]

This command gets the operating speed setting for the USB port.

[Format]

```
typedef struct {
    YVC_USB_MODE mode;
    struct {
        BOOL        reboot;
        void        (*callback)(YVC_CTRL_CMD cmd, int err, void *user);
        void        *user;
    } set;
} YVC_ARG_UsbSpeed;
```

[Parameters]

- [OUT] YVC_USB_MODE mode

The command returns the operating speed setting for the USB port.

Parameter symbol name	Description	Default value
YVC_USB_MODE_HI_SPEED	Indicates that "Hi-Speed" is specified.	✓
YVC_USB_MODE_FULL_SPEED	Indicates that "Full-Speed" is specified.	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.03 or later	✓	✗

4.5-35 YVC_CMD_SetExtTermMode

[Operational overview]

This command changes the connection mode setting of the EXT terminal.
Either daisy-chain connection mode or videoconference system mode is available.

For details, refer to "[Audio block diagram](#)".

[Format]

```
typedef struct {
    YVC_EXTTERM_MODE    mode;
} YVC_ARG_ExtTermMode;
```

[Parameters]

- [IN] YVC_EXTTERM_MODE mode

This parameter specifies the connection mode of the EXT terminal.

Parameter symbol name	Description	Default value
YVC_EXTTERM_MODE_CASCADE	Specifies that the device operates in YVC-300 and YVC-330 daisy-chain connection mode.	✓
YVC_EXTTERM_MODE_CONFERENCE_SYSTEM	Specifies that the device operates in videoconference system mode.	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗

4.5-36 YVC_CMD_GetExtTermMode

[Operational overview]

This command gets the connection mode setting of the EXT terminal.

[Notes]

If the main unit is in menu mode, the command returns the value before it entered menu mode. Therefore, it may return a different value from the one displayed on the LED, while the user is selecting a mode in menu mode.

[Format]

```
typedef struct {
    YVC_EXTTERM_MODE    mode;
} YVC_ARG_ExtTermMode;
```

[Parameters]

- [OUT] YVC_EXTTERM_MODE mode

The command returns the connection mode setting of the EXT terminal.

Parameter symbol name	Description	Default value
YVC_EXTTERM_MODE_CASCADE	Indicates that YVC-300 and YVC-330 daisy-chain connection mode is specified.	✓
YVC_EXTTERM_MODE_CONFERENCE_SYSTEM	Indicates that "videoconference system mode" is specified.	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-300	Ver. 1.04 or later	✓	X
YVC-330	Ver. 1.02 or later	✓	X

4.5-37 YVC_CMD_SetSpOutSelect

[Operational overview]

This command sets which speaker is selected for output.

Set one of the following: "Output from the built-in speaker only", "output from the external speaker only", and "output from both built-in and external speakers". The table below shows the volume value of target channels for each output setting.

Note that this command overwrites the value specified by the [YVC_CMD_SetAudioVolume](#) command.

Which speaker is selected for output	VOLUME_CH_SP_BUILTIN	VOLUME_CH_SP_EXTERNAL
Output from the built-in speaker only	0 dB	-128 dB
Output from the external speaker only	-128 dB	0 dB
Output from both built-in and external speakers	0 dB	0 dB

For details, refer to "[Audio block diagram](#)".

[Format]

```
typedef struct {
    YVC_SP_OUT_SELECT select;
} YVC_ARG_SpOutSelect;
```

[Parameters]

- [IN] YVC_SP_OUT_SELECT select

This parameter specifies which speaker is selected for output.

Parameter symbol name	Description	Default value
YVC_SP_OUT_SELECT_BUILTIN	Specifies to use only the built-in speaker for output.	
YVC_SP_OUT_SELECT_EXTERNAL	Specifies to use only the external speaker for output.	
YVC_SP_OUT_SELECT_BOTH	Specifies to use both the built-in and external speakers for output.	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-38 YVC_CMD_GetSpOutSelect

[Operational overview]

This command gets the setting for which speaker is selected for output.

It usually returns one of "output from the built-in speaker only", "output from the external speaker only", and "output from both built-in and external speakers". However, if the return value is not one of these values, the command returns an error ([YVC_API_ERROR_INVALIDPARAM](#)). (The error occurs when the [YVC_CMD_SetAudioVolume](#) command is used to set each target channel to -128 dB.)

Which speaker is selected for output	VOLUME_CH_SP_BUILTIN	VOLUME_CH_SP_EXTERNAL
Output from the built-in speaker only	Value other than -128 dB	-128 dB
Output from the external speaker only	-128 dB	Value other than -128 dB
Output from both built-in and external speakers	Value other than -128 dB	Value other than -128 dB
Error	-128 dB	-128 dB

[Format]

```
typedef struct {
    YVC_SP_OUT_SELECT select;
} YVC_ARG_SpOutSelect;
```

[Parameters]

- [OUT] YVC_SP_OUT_SELECT select

The command returns the setting for which speaker is selected for output.

Parameter symbol name	Description	Default value
YVC_SP_OUT_SELECT_BUILTIN	Indicates that the "output from the built-in speaker only" option is specified.	
YVC_SP_OUT_SELECT_EXTERNAL	Indicates that the "output from the external speaker only" option is specified.	
YVC_SP_OUT_SELECT_BOTH	Indicates that the "output from both built-in and external speakers" option is specified.	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-39 YVC_CMD_SetAudioInTerm

[Operational overview]

This command changes the setting for the audio input terminal.

Videoconference system mode or external microphone mode is available for the setting of each channel (L and R channels).

For details, refer to "[Audio block diagram](#)".

[Format]

```
typedef struct {
    struct {
        YVC_AUDIO_IN_MODE    mode;
        struct {
            YVC_AUDIO_IN_LEVEL    level;
            YVC_AUDIO_IN_EQ      eq;
            YVC_AUDIO_IN_SP_OUT   spOut;
            short                 gain;
        } paMic;
    } Lch, Rch;
} YVC_ARG_AudioInTerm;
```

[Parameters]

- [IN] Lch, Rch

This parameter specifies a pointer for the buffer to store the setting information of the audio input terminal. The channels must be set up in order of L and then R channels.

- [IN] YVC_AUDIO_IN_MODE mode

Parameter symbol name	Description	Default value
YVC_AUDIO_IN_MODE_CONFERENCE	Specifies to enter "videoconference system" mode.	✓
YVC_AUDIO_IN_MODE_PA_MIC	Specifies to enter "external microphone" mode.	

The following *paMic* parameters (*level*, *eq*, *spOut*, *gain*) can be specified when an external microphone is selected (when *mode* is set to YVC_AUDIO_IN_MODE_PA_MIC).

- [IN] YVC_AUDIO_IN_LEVEL level

This parameter specifies the output level of the external microphone.

Parameter symbol name	Description
YVC_AUDIO_IN_LEVEL_LINE	Specifies to set the output level to the line level.
YVC_AUDIO_IN_LEVEL_MIC	Specifies to set the output level to the microphone level.

- [IN] YVC_AUDIO_IN_EQ eq

This parameter specifies the sound quality setting for the external microphone.

Parameter symbol name	Description
YVC_AUDIO_IN_EQ_THRU	Specifies that the sound quality is not changed.
YVC_AUDIO_IN_EQ_LOW_BOOST	Specifies to increase the bass level.
YVC_AUDIO_IN_EQ_LOW_CUT	Specifies to decrease the bass level.

- [IN] **YVC_AUDIO_IN_SP_OUT** **spOut**

This parameter specifies whether amplifying the sound from the external microphone in your location is enabled or disabled.

It is common to both L and R channels. Specifying YVC_AUDIO_IN_SP_OUT_OFF and YVC_AUDIO_IN_SP_OUT_ON for each channel causes both channels to be set as YVC_AUDIO_IN_SP_OUT_ON.

Parameter symbol name	Description
YVC_AUDIO_IN_SP_OUT_OFF	Disables amplifying the sound from the external microphone in your location.
YVC_AUDIO_IN_SP_OUT_ON	Enables amplifying the sound from the external microphone in your location.

- [IN] **short** **gain**

This parameter specifies the sensitivity of the external microphone. (-12 to +12 dB)

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-40 YVC_CMD_GetAudioInTerm

[Operational overview]

This command gets the setting for the audio input terminal.

[Format]

```
typedef struct {
    struct {
        YVC_AUDIO_IN_MODE    mode;
        struct {
            YVC_AUDIO_IN_LEVEL    level;
            YVC_AUDIO_IN_EQ      eq;
            YVC_AUDIO_IN_SP_OUT   spOut;
            short                 gain;
        } paMic;
    } Lch, Rch;
} YVC_ARG_AudioInTerm;
```

[Parameters]

• [OUT] Lch, Rch

The command returns the setting information of the audio input terminal.
The setting information is stored in order of L and then R channels.
When an external microphone is set up (*mode* is set to YVC_AUDIO_IN_MODE_PA_MIC), the *paMic* parameters (*level*, *eq*, *spOut*, *gain*) are enabled.
For each parameter, refer to the pages on the [YVC_CMD_SetAudioInTerm](#) command.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-41 YVC_CMD_SetAudioVolume

[Operational overview]

This command adjusts the volume setting.

The speaker master output channel and the USB output channel are changed through volume control by the host.

Therefore, these channels do not always have the value specified by this command. Also, if they are changed, they are inconsistent with the values held by the host.

In YVC-300 and YVC-330, the settings for the AudioIn input channel and AudioOut output channel are enabled only if the connection mode is set to "videoconference system" mode. However, the specified value is stored even if the mode is not set to "videoconference system" mode.

In YVC-1000, when the settings are changed in the built-in speaker output and external speaker output channels, the value specified by the [YVC_CMD_SetSpOutSelect](#) command may change.

For details, refer to "[Audio block diagram](#)".

[Notice]

In YVC-200, if a value exceeded 0dB are set, clipping may occur depending on input or output signal.

[Format]

```
typedef struct {
    YVC_AUDIO_VOLUME_CH  ch;
    short                volume;
} YVC_ARG_AudioVolume;
```

[Parameters]

• [IN] YVC_AUDIO_VOLUME_CH ch

This parameter specifies a channel.

Parameter symbol name	Description	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_AUDIO_VOLUME_CH_SP	Speaker master output channel	✓	✓	✓	✓
YVC_AUDIO_VOLUME_CH_MIC	Microphone master input channel	✓	✓	✓	✓
YVC_AUDIO_VOLUME_CH_USB_IN	USB input channel	✓	✓	✓	✗
YVC_AUDIO_VOLUME_CH_USB_OUT	USB output channel	✓	✓	✓	✓
YVC_AUDIO_VOLUME_CH_BT_IN	Bluetooth input channel	✓	✓	✓	✗
YVC_AUDIO_VOLUME_CH_BT_OUT	Bluetooth output channel	✓	✓	✓	✓
YVC_AUDIO_VOLUME_CH_AUDIO_IN	AudioIn input channel	✓	✓	✓	✗
YVC_AUDIO_VOLUME_CH_AUDIO_OUT	AudioOut output channel	✓	✓	✓	✗
YVC_AUDIO_VOLUME_CH_SP_BUILTIN	Built-in speaker output channel	✓	✗	✗	✗
YVC_AUDIO_VOLUME_CH_SP_EXTERNAL	External speaker output channel	✓	✗	✗	✗
YVC_AUDIO_VOLUME_CH_SP_EXMIC	External microphone amplification channel	✓	✗	✗	✗
YVC_AUDIO_VOLUME_CH_EXMIC	External microphone master input channel	✓	✗	✗	✗
YVC_AUDIO_VOLUME_CH_HS_IN	Headset input channel	✗	✗	✗	✓
YVC_AUDIO_VOLUME_CH_HP_OUT	Headphone output channel	✗	✗	✗	✓

• [IN] short volume

This parameter specifies the volume value (in dB).

Parameter symbol name	Possible values (dB)	Default value (dB)	Setting value			
			YVC-1000	YVC-300	YVC-330	YVC-200
YVC_AUDIO_VOLUME_CH_SP	-50 to 0	-10	Non-volatile	Non-volatile	Non-volatile	Non-volatile
YVC_AUDIO_VOLUME_CH_MIC	-128 to +12	0	Volatile	Volatile	Volatile	Non-volatile
YVC_AUDIO_VOLUME_CH_USB_IN	-128 to +12	0	Non-volatile	Non-volatile	Non-volatile	—
YVC_AUDIO_VOLUME_CH_USB_OUT	-20 to 0	0	Volatile	Volatile	Volatile	—

YVC_AUDIO_VOLUME_CH_BT_IN	-128 to +12	0	Non-volatile	Non-volatile	Non-volatile	—
YVC_AUDIO_VOLUME_CH_BT_OUT	-128 to +12	0	Non-volatile	Non-volatile	Non-volatile	Non-volatile
YVC_AUDIO_VOLUME_CH_AUDIO_IN	-128 to +12	0	Non-volatile	Non-volatile	Non-volatile	—
YVC_AUDIO_VOLUME_CH_AUDIO_OUT	-128 to +12	0	Non-volatile	Non-volatile	Non-volatile	—
YVC_AUDIO_VOLUME_CH_SP_BUILTIN	-128 to 0	0	Non-volatile	—	—	—
YVC_AUDIO_VOLUME_CH_SP_EXTERNAL	-128 to 0	0	Non-volatile	—	—	—
YVC_AUDIO_VOLUME_CH_SP_EXMIC	-128 to 0	0	Non-volatile	—	—	—
YVC_AUDIO_VOLUME_CH_EXMIC	-128 to +12	0	Non-volatile	—	—	—
YVC_AUDIO_VOLUME_CH_HS_IN	-128 to +12	0	—	—	—	Non-volatile
YVC_AUDIO_VOLUME_CH_HP_OUT	-128 to +12	0	—	—	—	Non-volatile

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-300	Ver. 1.09 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-42 YVC_CMD_GetAudioVolume

[Operational overview]

This command gets the volume setting.

[Format]

```
typedef struct {
    YVC_AUDIO_VOLUME_CH ch;
    short volume;
} YVC_ARG_AudioVolume;
```

[Parameters]

- **[IN]** YVC_AUDIO_VOLUME_CH ch

This parameter specifies a channel.

Parameter symbol name	Description	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_AUDIO_VOLUME_CH_SP	Speaker master output channel	✓	✓	✓	✓
YVC_AUDIO_VOLUME_CH_MIC	Microphone master input channel	✓	✓	✓	✓
YVC_AUDIO_VOLUME_CH_USB_IN	USB input channel	✓	✓	✓	X
YVC_AUDIO_VOLUME_CH_USB_OUT	USB output channel	✓	✓	✓	✓
YVC_AUDIO_VOLUME_CH_BT_IN	Bluetooth input channel	✓	✓	✓	X
YVC_AUDIO_VOLUME_CH_BT_OUT	Bluetooth output channel	✓	✓	✓	✓
YVC_AUDIO_VOLUME_CH_AUDIO_IN	AudioIn input channel	✓	✓	✓	X
YVC_AUDIO_VOLUME_CH_AUDIO_OUT	AudioOut output channel	✓	✓	✓	X
YVC_AUDIO_VOLUME_CH_SP_BUILTIN	Built-in speaker output channel	✓	X	X	X
YVC_AUDIO_VOLUME_CH_SP_EXTERNAL	External speaker output channel	✓	X	X	X
YVC_AUDIO_VOLUME_CH_SP_EXMIC	External microphone amplification channel	✓	X	X	X
YVC_AUDIO_VOLUME_CH_EXMIC	External microphone master input channel	✓	X	X	X
YVC_AUDIO_VOLUME_CH_HS_IN	Headset input channel	X	X	X	✓
YVC_AUDIO_VOLUME_CH_HP_OUT	Headphone output channel	X	X	X	✓

- **[OUT]** short volume

The command returns the volume setting.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	X
YVC-300	Ver. 1.04 or later	✓	X
YVC-330	Ver. 1.02 or later	✓	X
YVC-200	Ver. 1.01 or later	✓	X

4.5-43 YVC_CMD_SetAudioMute

[Operational overview]

This command changes the mute setting.

In YVC-1000, YVC-300 and YVC-330 the speaker master output channel and the USB output channel are changed through mute control by the host. In YVC-200, the speaker master channel and the microphone master input channel are changed through mute control by the host. Therefore, these channels do not always have the value specified by this command. Also, if they are changed, they are inconsistent with the values held by the host.

If the microphone master input channel is muted or unmuted, the state notification event for microphone muting (YVC_EVENT_MIC_MUTE_STATE) is not made even when the event notification is enabled (because the mute button is not pressed).

For details, refer to "[Audio block diagram](#)".

[Format]

```
typedef struct {
    YVC_AUDIO_MUTE_CH  ch;
    BOOL               mute;
} YVC_ARG_AudioMute;
```

[Parameters]

- [IN] YVC_AUDIO_MUTE_CH ch

This parameter specifies a channel.

Parameter symbol name	Description	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_AUDIO_MUTE_CH_SP	Speaker master output channel	✓	✓	✓	✓
YVC_AUDIO_MUTE_CH_MIC	Microphone master input channel	✓	✓	✓	✓
YVC_AUDIO_MUTE_CH_USB_IN	USB input channel	✓	✓	✓	X
YVC_AUDIO_MUTE_CH_USB_OUT	USB output channel	✓	✓	✓	✓
YVC_AUDIO_MUTE_CH_BT_IN	Bluetooth input channel	✓	✓	✓	X
YVC_AUDIO_MUTE_CH_BT_OUT	Bluetooth output channel	✓	✓	✓	✓
YVC_AUDIO_MUTE_CH_AUDIO_IN	AudioIn input channel	✓	✓	✓	X
YVC_AUDIO_MUTE_CH_AUDIO_OUT	AudioOut output channel	✓	✓	✓	X
YVC_AUDIO_MUTE_CH_SP_BUILTIN	Built-in speaker output channel	✓	X	X	X
YVC_AUDIO_MUTE_CH_SP_EXTERNAL	External speaker output channel	✓	X	X	X
YVC_AUDIO_MUTE_CH_EXMIC	External microphone master input channel	✓	X	X	X
YVC_AUDIO_MUTE_CH_HS_IN	Headset input channel	X	X	X	✓
YVC_AUDIO_MUTE_CH_HP_OUT	Headphone output channel	X	X	X	✓

- [IN] BOOL mute

This parameter specifies whether to mute or unmute the device.

Value	Description
0	Specifies to unmute the device.
1	Specifies to mute the device.

The following table lists the default value and attribute for the mute setting of each channel:

Parameter symbol name	Default value	Setting value			
		YVC-1000	YVC-300	YVC-330	YVC-200
YVC_AUDIO_MUTE_CH_SP	0	Volatile	Volatile	Volatile	Volatile
YVC_AUDIO_MUTE_CH_MIC	0	Volatile	Volatile	Volatile	Volatile

YVC_AUDIO_MUTE_CH_USB_IN	0	Non-volatile	Non-volatile	Non-volatile	—
YVC_AUDIO_MUTE_CH_USB_OUT	0	Volatile	Volatile	Volatile	Non-volatile
YVC_AUDIO_MUTE_CH_BT_IN	0	Non-volatile	Non-volatile	Non-volatile	—
YVC_AUDIO_MUTE_CH_BT_OUT	0	Non-volatile	Non-volatile	Non-volatile	Non-volatile
YVC_AUDIO_MUTE_CH_AUDIO_IN	0	Non-volatile	Non-volatile	Non-volatile	—
YVC_AUDIO_MUTE_CH_AUDIO_OUT	0	Non-volatile	Non-volatile	Non-volatile	—
YVC_AUDIO_MUTE_CH_SP_BUILTIN	0	Non-volatile	—	—	—
YVC_AUDIO_MUTE_CH_SP_EXTERNAL	0	Non-volatile	—	—	—
YVC_AUDIO_MUTE_CH_EXMIC	0	Non-volatile	—	—	—
YVC_AUDIO_MUTE_CH_HS_IN	0	—	—	—	Non-volatile
YVC_AUDIO_MUTE_CH_HP_OUT	0	—	—	—	Non-volatile

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-44 YVC_CMD_GetAudioMute

[Operational overview]

This command gets the mute setting.

[Format]

```
typedef struct {
    YVC_AUDIO_MUTE_CH ch;
    BOOL mute;
} YVC_ARG_AudioMute;
```

[Parameters]

- **[IN]** YVC_AUDIO_MUTE_CH ch

This parameter specifies a channel.

Parameter symbol name	Description	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_AUDIO_MUTE_CH_SP	Speaker master output channel	✓	✓	✓	✓
YVC_AUDIO_MUTE_CH_MIC	Microphone master input channel	✓	✓	✓	✓
YVC_AUDIO_MUTE_CH_USB_IN	USB input channel	✓	✓	✓	X
YVC_AUDIO_MUTE_CH_USB_OUT	USB output channel	✓	✓	✓	✓
YVC_AUDIO_MUTE_CH_BT_IN	Bluetooth input channel	✓	✓	✓	X
YVC_AUDIO_MUTE_CH_BT_OUT	Bluetooth output channel	✓	✓	✓	✓
YVC_AUDIO_MUTE_CH_AUDIO_IN	AudioIn input channel	✓	✓	✓	X
YVC_AUDIO_MUTE_CH_AUDIO_OUT	AudioOut output channel	✓	✓	✓	X
YVC_AUDIO_MUTE_CH_SP_BUILTIN	Built-in speaker output channel	✓	X	X	X
YVC_AUDIO_MUTE_CH_SP_EXTERNAL	External speaker output channel	✓	X	X	X
YVC_AUDIO_MUTE_CH_EXMIC	External microphone master input channel	✓	X	X	X
YVC_AUDIO_MUTE_CH_HS_IN	Headset input channel	X	X	X	✓
YVC_AUDIO_MUTE_CH_HP_OUT	Headphone output channel	X	X	X	✓

- **[OUT]** BOOL mute

The command returns the mute setting.

Value	Description
0	Indicates that the device is unmuted.
1	Indicates that the device is muted.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	X
YVC-300	Ver. 1.04 or later	✓	X
YVC-330	Ver. 1.02 or later	✓	X
YVC-200	Ver. 1.01 or later	✓	X

4.5-45 YVC_CMD_SetMicAgc

[Operational overview]

This command changes the AGC setting of the microphone.

[Format]

```
typedef struct {
    YVC_MIC_AGC_MODE    mode;
} YVC_ARG_MicAgc;
```

[Parameters]

- [IN] YVC_MIC_AGC_MODE mode

This parameter specifies the AGC setting of the microphone.

Parameter symbol name	Description	Default value
YVC_MIC_AGC_MODE_ON	Enables AGC of the microphone.	✓
YVC_MIC_AGC_MODE_OFF	Disables AGC of the microphone.	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-46 YVC_CMD_GetMicAgc

[Operational overview]

This command gets the AGC setting of the microphone.

[Format]

```
typedef struct {
    YVC_MIC_AGC_MODE    mode;
} YVC_ARG_MicAgc;
```

[Parameters]

- [OUT] YVC_MIC_AGC_MODE mode

The command returns the AGC setting of the microphone.

Parameter symbol name	Description	Default value
YVC_MIC_AGC_MODE_ON	Indicates that AGC of the microphone is enabled.	✓
YVC_MIC_AGC_MODE_OFF	Indicates that AGC of the microphone is disabled.	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

4.5-47 YVC_CMD_SetMicUnitTrack

[Operational overview]

This command changes the sound pickup mode of the microphone unit.

There are two types of sound pickup modes: TRACK and MIX modes.

In TRACK mode, the system automatically selects signals with the highest sound pickup level from a microphone unit specified by *unitSelect* and outputs them.

In MIX mode, the system mixes all signals from a microphone unit specified by *unitSelect* and outputs them.

[Format]

```
typedef struct {
    YVC_MIC_MUX_MODE    mode;
    unsigned char        unitSelect;
} YVC_ARG_MicUnitTrack;
```

[Parameters]

- [IN] YVC_MIC_MUX_MODE mode

This parameter specifies the multiplexer setting for the microphone.

Parameter symbol name	Description	Default value
YVC_MIC_MUX_MODE_TRACK	The microphone operates in TRACK mode.	✓
YVC_MIC_MUX_MODE_MIX	The microphone operates in MIX mode.	

- [IN] unsigned char unitSelect

This parameter specifies which microphone unit is selected.

At least one microphone unit must be selected. If nothing is selected, an error ([YVC-API_ERROR_INVALIDARGS](#)) is returned.

Parameter symbol name	Value	Description	Default value
YVC_PRM_MIC_UNIT0	0x01	Specifies to select microphone unit 0.	✓
YVC_PRM_MIC_UNIT1	0x02	Specifies to select microphone unit 1.	✓
YVC_PRM_MIC_UNIT2	0x04	Specifies to select microphone unit 2.	✓
YVC_PRM_MIC_UNIT3	0x08	Specifies to select microphone unit 3.	✓
YVC_PRM_MIC_UNIT4	0x10	Specifies to select microphone unit 4.	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-48 YVC_CMD_GetMicUnitTrack

[Operational overview]

This command gets the sound pickup mode setting for the microphone unit.

[Format]

```
typedef struct {
    YVC_MIC_MUX_MODE    mode;
    unsigned char        unitSelect;
} YVC_ARG_MicUnitTrack;
```

[Parameters]

- [OUT] **YVC_MIC_MUX_MODE** **mode**

The command returns the multiplexer setting for the microphone.

Parameter symbol name	Description	Default value
YVC_MIC_MUX_MODE_TRACK	Indicates that "TRACK mode" is specified.	✓
YVC_MIC_MUX_MODE_MIX	Indicates that "MIX mode" is specified.	

- [OUT] **unsigned char** **unitSelect**

The command returns which microphone unit is selected.

Parameter symbol name	Value	Description	Default value
YVC_PRM_MIC_UNIT0	0x01	Indicates that "microphone unit 0" is specified.	✓
YVC_PRM_MIC_UNIT1	0x02	Indicates that "microphone unit 1" is specified.	✓
YVC_PRM_MIC_UNIT2	0x04	Indicates that "microphone unit 2" is specified.	✓
YVC_PRM_MIC_UNIT3	0x08	Indicates that "microphone unit 3" is specified.	✓
YVC_PRM_MIC_UNIT4	0x10	Indicates that "microphone unit 4" is specified.	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗

4.5-49 YVC_CMD_SetMicCapsuleTrack

[Operational overview]

This command changes the sound pickup mode of the microphone capsule.

There are two types of sound pickup modes: TRACK and MIX modes.

In TRACK mode, the system automatically selects signals with the highest sound pickup level from a microphone capsule specified by *capsuleSelect* and outputs them.

In MIX mode, the system mixes all signals from all microphone capsules and outputs them.

[Format]

```
typedef struct {
    YVC_MIC_MUX_MODE    mode;
    unsigned char        capsuleSelect;
} YVC_ARG_MicCapsuleTrack;
```

[Parameters]

- [IN] YVC_MIC_MUX_MODE mode

This parameter specifies the multiplexer setting for the microphone.

Parameter symbol name	Description	Default value
YVC_MIC_MUX_MODE_TRACK	Indicates that "TRACK mode" is specified.	✓
YVC_MIC_MUX_MODE_MIX	Indicates that "MIX mode" is specified.	

- [IN] unsigned char capsuleSelect

A microphone capsule must be specified only when TRACK mode is selected.

At least one microphone capsule must be selected. If nothing is selected, an error ([YVC-API_ERROR_INVALIDARGS](#)) is returned.

This parameter is ignored when MIX mode is selected.

Parameter symbol name	Value	Description	Default value
YVC_PRM_MIC_CAPSULE_CENTER	0x01	Specifies to select the microphone capsule (center).	✓
YVC_PRM_MIC_CAPSULE_LEFT	0x02	Specifies to select the microphone capsule (left).	✓
YVC_PRM_MIC_CAPSULE_RIGHT	0x04	Specifies to select the microphone capsule (right).	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-300	Ver. 1.09 or later	✓	X
YVC-330	Ver. 1.02 or later	✓	X

4.5-50 YVC_CMD_GetMicCapsuleTrack

[Operational overview]

This command gets the setting for the sound pickup mode of the microphone capsule.

[Format]

```
typedef struct {
    YVC_MIC_MUX_MODE    mode;
    unsigned char        capsuleSelect;
} YVC_ARG_MicCapsuleTrack;
```

[Parameters]

- [OUT] YVC_MIC_MUX_MODE mode

The command returns the multiplexer setting for the microphone.

Parameter symbol name	Description	Default value
YVC_MIC_MUX_MODE_TRACK	Indicates that "TRACK mode" is specified.	✓
YVC_MIC_MUX_MODE_MIX	Indicates that "MIX mode" is specified.	

- [OUT] unsigned char capsuleSelect

The command returns the setting of the microphone capsule.

Parameter symbol name	Value	Description	Default value
YVC_PRM_MIC_CAPSULE_CENTER	0x01	Indicates that the "microphone capsule (center)" is specified.	✓
YVC_PRM_MIC_CAPSULE_LEFT	0x02	Indicates that the "microphone capsule (left)" is specified.	✓
YVC_PRM_MIC_CAPSULE_RIGHT	0x04	Indicates that the "microphone capsule (right)" is specified.	✓

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-300	Ver. 1.09 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗

4.5-51 YVC_CMD_SetMicFilter

[Operational overview]

This command changes the frequency characteristic of the microphone (cuts low frequency components in microphone signals).

[Format]

```
typedef struct {
    YVC_MIC_FILTER_MODE    mode;
} YVC_ARG_MicFilter;
```

[Parameters]

- [IN] YVC_MIC_FILTER_MODE mode

This parameter specifies the frequency characteristic of the microphone.

Parameter symbol name	Description	Default value	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_MIC_FILTER_MODE_OFF	Does not cut the low frequency components.	✓	✓	✓	✓	✓
YVC_MIC_FILTER_MODE_LOW	Cuts frequencies of 200 Hz or less.		✓	✓	✓	X
YVC_MIC_FILTER_MODE_MID	Cuts frequencies of 315 Hz or less.		✓	✓	✓	✓
YVC_MIC_FILTER_MODE_HIGH	Cuts frequencies of 500 Hz or less.		✓	✓	✓	X

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	X
YVC-300	Ver. 1.09 or later	✓	X
YVC-330	Ver. 1.02 or later	✓	X
YVC-200	Ver. 1.01 or later	✓	X

4.5-52 YVC_CMD_GetMicFilter

[Operational overview]

This command gets the frequency characteristic setting of the microphone.

[Format]

```
typedef struct {
    YVC_MIC_FILTER_MODE    mode;
} YVC_ARG_MicFilter;
```

[Parameters]

- [OUT] YVC_MIC_FILTER_MODE mode

The command returns the frequency characteristic setting of the microphone.

Parameter symbol name	Description	Default value	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_MIC_FILTER_MODE_OFF	Indicates that the option "not to cut the low frequency components" is specified.	✓	✓	✓	✓	✓
YVC_MIC_FILTER_MODE_LOW	Indicates that the option "to cut frequencies of 200 Hz or less" is specified.		✓	✓	✓	X
YVC_MIC_FILTER_MODE_MID	Indicates that the option "to cut frequencies of 315 Hz or less" is specified.		✓	✓	✓	✓
YVC_MIC_FILTER_MODE_HIGH	Indicates that the option "to cut frequencies of 500 Hz or less" is specified.		✓	✓	✓	X

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	X
YVC-300	Ver. 1.09 or later	✓	X
YVC-330	Ver. 1.02 or later	✓	X
YVC-200	Ver. 1.01 or later	✓	X

4.5-53 YVC_CMD_SetSpFilter

[Operational overview]

This command changes the frequency characteristic of the speaker (cuts low frequency components in speaker signals).

[Format]

```
typedef struct {
    YVC_SP_FILTER_MODE mode;
} YVC_ARG_SpFilter;
```

[Parameters]

- [IN] YVC_SP_FILTER_MODE mode

This parameter specifies the frequency characteristic of the speaker.

Parameter symbol name	Description	Default value
YVC_SP_FILTER_MODE_OFF	Does not cut the low frequency components.	✓
YVC_SP_FILTER_MODE_LOW	Cuts frequencies of 200 Hz or less.	
YVC_SP_FILTER_MODE_MID	Cuts frequencies of 315 Hz or less.	
YVC_SP_FILTER_MODE_HIGH	Cuts frequencies of 500 Hz or less.	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-300	Ver. 1.09 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗

4.5-54 YVC_CMD_GetSpFilter

[Operational overview]

This command gets the frequency characteristic setting of the speaker.

[Format]

```
typedef struct {
    YVC_SP_FILTER_MODE    mode;
} YVC_ARG_SpFilter;
```

[Parameters]

- [OUT] YVC_SP_FILTER_MODE mode

The command returns the frequency characteristic setting of the speaker.

Parameter symbol name	Description	Default value
YVC_SP_FILTER_MODE_OFF	Indicates that the option "not to cut the low frequency components" is specified.	✓
YVC_SP_FILTER_MODE_LOW	Indicates that the option "to cut frequencies of 200 Hz or less" is specified.	
YVC_SP_FILTER_MODE_MID	Indicates that the option "to cut frequencies of 315 Hz or less" is specified.	
YVC_SP_FILTER_MODE_HIGH	Indicates that the option "to cut frequencies of 500 Hz or less" is specified.	

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	X
YVC-300	Ver. 1.09 or later	✓	X
YVC-330	Ver. 1.02 or later	✓	X

4.5-55 YVC_CMD_ExeAAT

[Operational overview]

This command controls (starts and stops) execution of automatic audio tuning.

The function automatically measures acoustic characteristics of the space around the location where a YVC-1000 device is installed, as well as acoustical conditions, such as where microphones and speakers are installed. It also adjusts the acoustic settings to the best conditions depending on the operating environments.

The following steps explain how to perform the function:

The measurement uses the speaker on the Control Unit and the external speaker to reproduce measured sounds, and pick up the sounds with the microphone unit. Therefore, a measurement error occurs if something blocks sound reproduced by the Control Unit's speaker or the external speaker, or picked up by the microphone unit. A measurement error also tends to occur if a large noise is picked up by the microphone unit.

For these reasons, stay away from the device and keep quiet while the measurement is being done.

- (1) Use this command with the *action* parameter set to YVC_AAT_ACTION_START specified to perform automatic audio tuning. When the start request is accepted, the following voice guidance is played back:
"Automatic audio tuning will start. Please move away from the devices and do not make any noise. The measurement sounds will play."
 After the voice guidance ends, the measurement will start.
 To stop automatic audio tuning, set the *action* parameter to YVC_AAT_ACTION_STOP. Note that if any factor other than the command causes automatic audio tuning to stop, the following voice guidance is played back:
"Automatic audio tuning has been canceled."
 *The message is not played back if the voice guidance is disabled through the [YVC_CMD_SetVoiceGuideUse](#) command.
- (2) Check whether automatic audio tuning is finished.
 Run the [YVC_CMD_AATProgress](#) command periodically to check the progress. When the *progress* value reaches 100, the tuning is finished.
- (3) Use the [YVC_CMD_GetAATResult](#) command to get the measurement result.
 The measurement result is stored in the *result* parameter, and the warning error information in the *warning* parameter.
 *The voice guidance for the warning error is not played back.

[Notes]

The sound I/Os of the AUDIO IN and AUDIO OUT terminals and USB and Bluetooth I/Fs are disabled (no sound is produced) while automatic audio tuning is in progress. In addition, some commands are restricted during automatic audio tuning. Refer to the list on the next page.

[Format]

```
typedef struct {
    YVC_AAT_ACTION    action;
} YVC_ARG_AAT;
```

[Parameters]

- [IN] YVC_AAT_ACTION *action*

This parameter specifies an action on automatic audio tuning.

Parameter symbol name	Description
YVC_AAT_ACTION_STOP	Stops (cancels) automatic audio tuning.
YVC_AAT_ACTION_START	Performs automatic audio tuning.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.10 or later	✓	✗

Which command can be run during automatic audio tuning

Command	Description	Can be run?
YVC_CMD_GetSystemMode	Gets the operation mode of the system.	✓
YVC_CMD_GetVersionInfo	Gets the version information.	✓
YVC_CMD_GetModelName	Gets the model name.	✓
YVC_CMD_GetSerialNumber	Gets the serial number.	✓ (Note)
YVC_CMD_GetMicNum	Gets the number of connected microphone units.	✓
YVC_CMD_ExeSystemReboot	Restarts the system.	✓
YVC_CMD_ExeFactoryReset	Resets the parameters to the factory settings.	X
YVC_CMD_ExeStandbyMode	Puts the device in the standby state.	✓
YVC_CMD_SetNoticeEvent	Sets the event to be notified of.	✓
YVC_CMD_GetNoticeEvent	Gets the setting of the event to be notified of.	✓
YVC_CMD_SetBTUse	Enables or disables the Bluetooth function.	X
YVC_CMD_GetBTUse	Gets whether the Bluetooth function is enabled or disabled.	X
YVC_CMD_SetBTLocalName	Sets the Local Name for Bluetooth.	X
YVC_CMD_GetBTLocalName	Gets the Local Name setting for Bluetooth.	X
YVC_CMD_SetBTOperation	Operates a Bluetooth function.	X
YVC_CMD_GetBTOperation	Gets the status of the Bluetooth function.	X
YVC_CMD_SetNoticeSoundUse	Enables or disables the notification sound.	✓
YVC_CMD_GetNoticeSoundUse	Gets whether the notification sound is enabled or disabled.	✓
YVC_CMD_SetVoiceGuideUse	Enables or disables the voice guidance.	✓
YVC_CMD_GetVoiceGuideUse	Gets whether the voice guidance is enabled or disabled.	✓
YVC_CMD_SetVoiceGuideLang	Changes the language of the voice guidance.	X
YVC_CMD_GetVoiceGuideLang	Gets the language setting of the voice guidance.	X
YVC_CMD_SetVoiceGuideVolume	Adjusts the volume of the voice guidance.	X
YVC_CMD_GetVoiceGuideVolume	Gets the volume setting of the voice guidance.	X
YVC_CMD_SetInitSpVolume	Sets the speaker volume when the device is connected via USB.	X
YVC_CMD_GetInitSpVolume	Gets the speaker volume setting when the device is connected via USB.	X
YVC_CMD_SetPowerOnState	Specifies an operation at power-on.	✓
YVC_CMD_GetPowerOnState	Gets the operational setting at power-on.	✓
YVC_CMD_SetMicMuteMode	Sets the mute mode for the microphone.	X
YVC_CMD_GetMicMuteMode	Gets the mute mode setting for the microphone.	X
YVC_CMD_SetUsbSpeed	Sets the operating speed of the USB port.	X
YVC_CMD_GetUsbSpeed	Gets the operating speed setting of the USB port.	X
YVC_CMD_SetSpOutSelect	Sets which speaker is selected for output.	X
YVC_CMD_GetSpOutSelect	Gets the setting for which speaker is selected for output.	X
YVC_CMD_SetAudioInTerm	Changes the setting for the audio input terminal.	X
YVC_CMD_GetAudioInTerm	Gets the setting for the audio input terminal.	X
YVC_CMD_SetAudioVolume	Adjusts the volume setting.	X
YVC_CMD_GetAudioVolume	Gets the volume setting.	X
YVC_CMD_SetAudioMute	Changes the mute setting.	X

YVC_CMD_GetAudioMute	Gets the mute setting.	X
YVC_CMD_SetMicAge	Changes the AGC setting of the microphone.	X
YVC_CMD_GetMicAge	Gets the AGC setting of the microphone.	X
YVC_CMD_SetMicUnitTrack	Changes the sound pickup mode of the microphone unit.	X
YVC_CMD_GetMicUnitTrack	Gets the sound pickup mode setting for the microphone unit.	X
YVC_CMD_SetMicFilter	Changes the frequency characteristic of the microphone.	X
YVC_CMD_GetMicFilter	Gets the frequency characteristic setting of the microphone.	X
YVC_CMD_SetSpFilter	Changes the frequency characteristic of the speaker.	X
YVC_CMD_GetSpFilter	Gets the frequency characteristic setting of the speaker.	X
YVC_CMD_ExecAAT	Controls execution of automatic audio tuning.	✓
YVC_CMD_GetAATProgress	Gets the progress of automatic audio tuning.	✓
YVC_CMD_GetAATResult	Gets the measurement result of automatic audio tuning.	X
YVC_CMD_MonMicCapsule	Gets active microphone capsules.	X
YVC_CMD_MonMicUnit	Gets the active microphone unit ID.	X

(Note) The command can get only the serial number of the Control Unit or main unit.

4.5-56 YVC_CMD_GetAATProgress

[Operational overview]

This command gets the progress of automatic audio tuning.

[Format]

```
typedef struct {  
    unsigned int progress;  
} YVC_ARG_AATProgress;
```

[Parameters]

- [OUT] unsigned int progress

The command returns the progress of automatic audio tuning in percentages. (0 to 100)

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.10 or later	✓	✗

4.5-57 YVC_CMD_GetAATResult

[Operational overview]

This command gets the measurement result of automatic audio tuning.

[Format]

```
typedef struct {
    YVC_AAT_RESULT result;
    unsigned int    warning;
} YVC_ARG_AATResult;
```

[Parameters]

- [OUT] YVC_AAT_RESULT result

The command returns the measurement result of automatic audio tuning.

Parameter symbol name	Description
YVC_AAT_RESULT_OK	The tuning is completed successfully. (There is no problem found.)
YVC_AAT_RESULT_CANCEL	The tuning was canceled. In addition to the cancellation from the command, the tuning is also canceled when: <ul style="list-style-type: none"> - The number of connected microphone units is changed during measurement. - The host (PC) changes the volume of the speaker. - The tuning fork button on the Control Unit is pressed. In these cases, the following voice guidance is played back: <i>"Automatic audio tuning has been canceled."</i> * The message is not played back if the voice guidance is disabled through the YVC_CMD_SetVoiceGuideUse command.
YVC_AAT_RESULT_WARNING	A warning error was detected. Details are stored in the <i>warning</i> parameter.
YVC_AAT_RESULT_NO_MIC	Automatic audio tuning was canceled due to no microphone being connected.
YVC_AAT_RESULT_SP_VOL_SMALL	Automatic audio tuning was canceled due to speaker volume being too low. The speaker volume must be set to -16 dB or more. With the YVC_CMD_SetAudioVolume command, specify the speaker master output channel to change the volume.
YVC_AAT_RESULT_EXT_SP_UNDETECT	Automatic audio tuning was canceled due to no external speaker being detected. (This is limited to when only external speakers are enabled.)

- [OUT] unsigned int warning

The command returns a warning error on automatic audio tuning. It is enabled when the *result* parameter is set to YVC_AAT_RESULT_WARNING.

Note that multiple warning errors are sometimes detected.

Parameter symbol name	Value	Description
YVC_PRM_AAT_WARNING_OK	0x0	Successful completion
YVC_PRM_AAT_WARNING_TOOCLOSE	0x1	An error was detected. "The distance between the microphone unit and the Control Unit is too close."
YVC_PRM_AAT_WARNING_TOOFAR	0x2	An error was detected. "The distance between the microphone unit and the Control Unit is too far."
YVC_PRM_AAT_WARNING_LONGDELAY	0x4	An error was detected. "External speaker delay is too long."
YVC_PRM_AAT_WARNING_DISTORTION	0x8	An error was detected. "The external speaker has excessively high distortion."

For details on the warning errors, refer to "Unified Communications Microphone & Speaker System YVC-1000 User's Manual".

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.10 or later	✓	✗

4.5-58 YVC_CMD_MonMicCapsule

[Operational overview]

This command gets active microphone capsules.

It returns values when microphones are muted. Mask them while microphones are muted, according to applications.

This command must be run at 500-ms or longer intervals. A request for running the command at intervals less than 500 ms returns an error ([YVC-API_ERROR_CTRLEXEC](#)).

For positions of microphone capsules for each device, refer to "[Microphone capsule positions](#)".

[Format]

```
typedef struct {
    unsigned char trackInfo[YVC_LEN_MIC_UNIT_MAX];
} YVC_ARG_MonMicCapsule;
```

[Parameters]

- [OUT] **unsigned char trackInfo[YVC_LEN_MIC_UNIT_MAX]**

The command returns active microphone capsules.

Parameter symbol name	Value	Description
YVC_PRM_MIC_CAPSULE_NONE	0x00	Indicates that all microphone capsules are inactive.
YVC_PRM_MIC_CAPSULE_CENTER	0x01	Indicates that the microphone capsule (center) is active.
YVC_PRM_MIC_CAPSULE_LEFT	0x02	Indicates that the microphone capsule (left) is active.
YVC_PRM_MIC_CAPSULE_RIGHT	0x04	Indicates that the microphone capsule (right) is active.

In YVC-1000:

The number of active bytes represents the number of connected microphone units.

The information on microphone unit 0 is stored in *trackInfo[0]*.

The information on microphone unit 1 is stored in *trackInfo[1]*.

The information on microphone unit 2 is stored in *trackInfo[2]*.

The information on microphone unit 3 is stored in *trackInfo[3]*.

The information on microphone unit 4 is stored in *trackInfo[4]*.

In YVC-300 and YVC-330:

Only the first one byte (*trackInfo[0]*) is enabled regardless of the daisy-chain connection state.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗

4.5-59 YVC_CMD_MonMicUnit

[Operational overview]

This command gets the microphone unit ID of an active microphone unit.

It returns a microphone unit ID value when microphones are muted. Mask it while microphones are muted, according to applications.

This command must be run at 500-ms or longer intervals. A request for running the command at intervals less than 500 ms returns an error ([YVCAPI_ERROR_CTRLEXEC](#)).

For positions of microphone capsules for each device, refer to "[Microphone capsule positions](#)".

[Format]

```
typedef struct {
    int    unitId;
} YVC_ARG_MonMicUnit;
```

[Parameters]

- [OUT] int unitId

The command returns the microphone unit ID (0 to 4) of an active microphone unit.

If no microphone unit is connected, it returns -1.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.10 or later	✓	✗

4.5-60 YVC_CMD_MonHeadsetPortState

[Operational overview]

This command gets the status of headset function.

When the headset or headphones are not connected to the headset jack, the microphone and speaker of YVC-200 will be enabled, but when the headset or headphones are connected to the headset jack, the microphone and speaker of the connected device will be enabled.

This command must be run at 500-ms or longer intervals. A request for running the command at intervals less than 500 ms returns an error ([YVCAPI_ERROR_CTRLEXEC](#)).

For details, refer to "[Audio block diagram](#)".

[Format]

```
typedef struct {
    unsigned int  state;
} YVC_ARG_MonHeadsetPortState;
```

[Parameters]

- [OUT] unsigned int state

The command returns the status of headset function.

Parameter symbol name	Value	Description
YVC_PRM_HEADSET_PORT_NONE	0x00	No device is connected to the headset jack. The microphone and speaker of YVC-200 is enabled.
YVC_PRM_HEADSET_PORT_HP	0x01	Headphone is connected to the headset jack. The microphone of YVC-200 and the speaker of headphone is enabled.
YVC_PRM_HEADSET_PORT_HS	0x02	Headset is connected to the headset jack. The microphone and speaker of headset is enabled.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-200	Ver. 1.01 or later	✓	✗

4.5-61 YVC_CMD_MonBatteryLevel

[Operational overview]

This command gets the residual quantity of battery.

This command must be run at 500-ms or longer intervals. A request for running the command at intervals less than 500 ms returns an error ([YVC-API_ERROR_CTRLXEC](#)).

[Format]

```
typedef struct {
    unsigned int  level;
    unsigned int  charging;
} YVC_ARG_MonBatteryLevel;
```

[Parameters]

- [OUT] unsigned int level

The command returns the residual quantity of battery (0 to 3).

Value	Description
0	The residual quantity of battery is less than 25%.
1	The residual quantity of battery is more than 25% and less than 40%.
2	The residual quantity of battery is more than 40% and less than 80%.
3	The residual quantity of battery is more than 80%.

- [OUT] unsigned int level

The command returns the status of battery charging (0 or 1).

Value	Description
0	The battery is being charged.
1	The battery is not charged.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-200	Ver. 1.01 or later	✓	✗

4.6 Event details

[Operational overview]

This command notifies a notification destination of an event.

To do so, the notification destination must be pre-registered by using the [YVC_DeviceAttach](#) function. Also, whether notifications on events are made is specified with the [YVC_CMD_SetNoticeEvent](#) command.

However, the YVC_EVENT_DEV_DISCONNECT is always enabled. The notification of it cannot be disabled.

[Notes]

Do not perform a time-consuming operation or call [this API function](#) in the callback function.

[Format]

```
typedef void (*YVC_EVENT_CALLBACK)(YVC_EVENT event, int data, void *user)
```

[Parameters]

- [IN] YVC_EVENT event

This parameter notifies the destination of the type of an event.

Event definition name	Description	YVC-1000	YVC-300	YVC-330	YVC-200
YVC_EVENT_DEV_DISCONNECT	Notifies the destination of the device being disconnected. [Notes] <u>When this event notification is received, perform the YVC_DeviceDetach function to release the handle obtained by the YVC_DeviceAttach function. However, do not call the above API function in the callback function.</u>	✓	✓	✓	✓
YVC_EVENT_MIC_UNIT_NUM	Notifies the destination of the change in the connection state of a microphone unit.	✓	X	X	X
YVC_EVENT_MIC_MUTE_STATE	Notifies the destination of the change in the mute mode of a microphone. It is, however, limited to cases where the mute button is pressed. The destination is not notified of status changes due to a command setup or other causes.	✓	✓	✓	✓
YVC_EVENT_HOOK_BTN	Notifies the destination of the change in the status of the on/off-hook button when it is pressed. However, no notification is made when it is used as an incoming response on Bluetooth or call ending (call disconnect) function.	X	✓	✓	X

- [IN] int data

This parameter is used to notify the destination of the event in detail.

YVC_EVENT_DEV_DISCONNECT event

Value	Description
0	Does not have any meaning. (Fixed to 0)

YVC_EVENT_MIC_UNIT_NUM event

Value	Description
0	Indicates that no microphone unit is now connected.
1	Indicates that one microphone unit is now connected.
2	Indicates that two microphone units are now connected.
3	Indicates that three microphone units are now connected.
4	Indicates that four microphone units are now connected.
5	Indicates that five microphone units are now connected.

YVC_EVENT_MIC_MUTE_STATE event

In YVC-300, YVC-330 and YVC-200, or when the YVC_MIC_MUTE_MODE parameter is set to YVC_MIC_MUTE_MODE_GROUP in YVC-1000:

Value	Description
0	Indicates that the microphone is unmuted.
1	Indicates that the microphone is muted.

When the YVC_MIC_MUTE_MODE parameter is set to YVC_MIC_MUTE_MODE_INDIVIDUAL in YVC-1000:

The event notification is made when any microphone unit is muted or unmuted. *data* indicates whether all connected microphone units are muted or unmuted.

Parameter symbol name	Bit	Value	Description
YVC_EVENT_DATA_MICMUTE_UNIT0	0x1	0	Indicates that microphone unit 0 is unmuted.
		1	Indicates that microphone unit 0 is muted.
YVC_EVENT_DATA_MICMUTE_UNIT1	0x2	0	Indicates that microphone unit 1 is unmuted.
		1	Indicates that microphone unit 1 is muted.
YVC_EVENT_DATA_MICMUTE_UNIT2	0x4	0	Indicates that microphone unit 2 is unmuted.
		1	Indicates that microphone unit 2 is muted.
YVC_EVENT_DATA_MICMUTE_UNIT3	0x8	0	Indicates that microphone unit 3 is unmuted.
		1	Indicates that microphone unit 3 is muted.
YVC_EVENT_DATA_MICMUTE_UNIT4	0x10	0	Indicates that microphone unit 4 is unmuted.
		1	Indicates that microphone unit 4 is muted.

For example, when two microphone units are connected:

If microphone unit 0 is muted, *data* has a value of 0x01.

Then, if microphone unit 1 is muted, *data* has a value of 0x03.

YVC_EVENT_HOOK_BTN event

Parameter symbol name	Value	Description
YVC_EVENT_DATA_HOOKBTN_SHORT	0	Indicates that the on/off-hook button was pressed.
YVC_EVENT_DATA_HOOKBTN_LONG	1	Indicates that the on/off-hook button was held.

• [IN] void *user

The command returns the address specified in the fourth parameter (*user*) of the [YVC_DeviceAttach](#) function without change.

[Supported model]

Model	Firmware	Supported mode	
		NORMAL	FWUP
YVC-1000	Ver. 2.00 or later	✓	✗
YVC-300	Ver. 1.04 or later	✓	✗
YVC-330	Ver. 1.02 or later	✓	✗
YVC-200	Ver. 1.01 or later	✓	✗

[Notes]

Although events other than YVC_EVENT_DEV_DISCONNECT must be registered by using the [YVC_CMD_SetNoticeEvent](#) command, limited versions of firmware support the command. For details, refer to the page on the [YVC_CMD_SetNoticeEvent](#) command.

4.7 Constants and type definitions

This section provides the type definitions and constants used by the API.

4.7-1 YVC_API_INFO

```
/* API information definition */
typedef struct {
    char    version[YVC_LEN_API_VERSION+1];
} YVC_API_INFO;
```

4.7-2 YVC_DEV_HANDLE

```
/* Handle definition for device control */
typedef void* YVC_DEV_HANDLE;
```

4.7-3 YVC_DEV_TYPE

```
/* Definition for the model types of the YVC series */
typedef enum {
    YVC_DEV_TYPE_YVC1000 = 0,
    YVC_DEV_TYPE_YVC300,
    YVC_DEV_TYPE_YVC200,
    YVC_DEV_TYPE_YVC330,
    YVC_DEV_TYPE_NUM
} YVC_DEV_TYPE;
```

4.7-4 YVC_DEV_INFO

```
/* Definition for the device connection information */
typedef struct {
    unsigned int num;
} YVC_DEV_INFO;
```

4.7-5 YVC_EVENT

```
/* Event type definition */
typedef enum {
    YVC_EVENT_DEV_DISCONNECT = 0,
    YVC_EVENT_MIC_UNIT_NUM,
    YVC_EVENT_MIC_MUTE_STATE,
    YVC_EVENT_HOOK_BTN,
    YVC_EVENT_NUM
} YVC_EVENT;
```

4.7-6 YVC_EVENT_CALLBACK

```
/* Definition of the callback function for event notification */
typedef void (*YVC_EVENT_CALLBACK)(YVC_EVENT event, int data, void *user);
```

4.7-7 YVC_CMD

```

/* Command type definition */
typedef enum {
    YVC_CMD_GetSystemMode = 0,
    YVC_CMD_GetVersionInfo,
    YVC_CMD_GetModelName,
    YVC_CMD_GetSerialNumber,
    YVC_CMD_GetMicNum,
    YVC_CMD_ExeSystemReboot,
    YVC_CMD_ExeFactoryReset,
    YVC_CMD_ExeStandbyMode,
    YVC_CMD_SetNoticeEvent,
    YVC_CMD_GetNoticeEvent,
    YVC_CMD_SetBTUse,
    YVC_CMD_GetBTUse,
    YVC_CMD_SetBTLocalName,
    YVC_CMD_GetBTLocalName,
    YVC_CMD_SetBTOperation,
    YVC_CMD_GetBTOperation,
    YVC_CMD_SetNoticeSoundUse,
    YVC_CMD_GetNoticeSoundUse,
    YVC_CMD_SetNoticeSoundVolume,
    YVC_CMD_GetNoticeSoundVolume,
    YVC_CMD_SetVoiceGuideUse,
    YVC_CMD_GetVoiceGuideUse,
    YVC_CMD_SetVoiceGuideLang,
    YVC_CMD_GetVoiceGuideLang,
    YVC_CMD_SetVoiceGuideVolume,
    YVC_CMD_GetVoiceGuideVolume,
    YVC_CMD_SetInitSpVolume,
    YVC_CMD_GetInitSpVolume,
    YVC_CMD_SetPowerOnState,
    YVC_CMD_GetPowerOnState,
    YVC_CMD_SetMicMuteMode,
    YVC_CMD_GetMicMuteMode,
    YVC_CMD_SetUsbSpeed,
    YVC_CMD_GetUsbSpeed,
    YVC_CMD_SetExtTermMode,
    YVC_CMD_GetExtTermMode,
    YVC_CMD_SetSpOutSelect,
    YVC_CMD_GetSpOutSelect,
    YVC_CMD_SetAudioInTerm,
    YVC_CMD_GetAudioInTerm,
    YVC_CMD_SetAudioVolume,
    YVC_CMD_GetAudioVolume,
    YVC_CMD_SetAudioMute,
    YVC_CMD_GetAudioMute,
    YVC_CMD_SetMicAgc,
    YVC_CMD_GetMicAgc,
    YVC_CMD_SetMicUnitTrack,
    YVC_CMD_GetMicUnitTrack,
    YVC_CMD_SetMicFilter,
    YVC_CMD_GetMicFilter,
    YVC_CMD_SetSpFilter,
    YVC_CMD_GetSpFilter,
    YVC_CMD_ExeAAT,
    YVC_CMD_GetAATProgress,
    YVC_CMD_GetAATResult,
    YVC_CMD_MonMicCapsule,
    YVC_CMD_MonMicUnit,
    YVC_CMD_Reserved1,
    YVC_CMD_Reserved2,
    YVC_CMD_MonHeadsetPortState,
    YVC_CMD_MonBatteryLevel,
    YVC_CMD_NUM
} YVC_CTRL_CMD;

```

4.7-8 YVC_CTRL_ARG

```
/* Type definition for the control parameter */
typedef void* YVC_CTRL_ARG;
```

The following lists the definition of ARG parameters for each command:

1) YVC_CMD_GetSystemMode

```
/* Parameter definition for system operation modes */
typedef enum {
    YVC_SYSTEM_MODE_FWUP = 0,
    YVC_SYSTEM_MODE_NORMAL,
    YVC_SYSTEM_MODE_NUM
} YVC_SYSTEM_MODE;

typedef struct {
    YVC_SYSTEM_MODE mode;
} YVC_ARG_SystemMode;
```

2) YVC_CMD_GetVersionInfo

```
typedef struct {
    char version[YVC_LEN_VERSION_INFO+1];
} YVC_ARG_VersionInfo;
```

3) YVC_CMD_GetModelName

```
typedef struct {
    char name[YVC_LEN_MODEL_NAME+1];
} YVC_ARG_ModelName;
```

4) YVC_CMD_GetSerialNumber

```
/* Definition for the unit selection list of serial numbers */
typedef enum {
    YVC_SERIALNUMBER_TYPE_SPUNIT = 0,
    YVC_SERIALNUMBER_TYPE_MICUNIT0,
    YVC_SERIALNUMBER_TYPE_MICUNIT1,
    YVC_SERIALNUMBER_TYPE_MICUNIT2,
    YVC_SERIALNUMBER_TYPE_MICUNIT3,
    YVC_SERIALNUMBER_TYPE_MICUNIT4,
    YVC_SERIALNUMBER_TYPE_NUM
} YVC_SERIALNUMBER_TYPE;

typedef struct {
    YVC_SERIALNUMBER_TYPE type;
    char serialNumber[YVC_LEN_SERIAL_NUMBER+1];
} YVC_ARG_SerialNumber;
```

5) YVC_CMD_GetMicNum

```
typedef struct {
    unsigned int micNum;
} YVC_ARG_MicNum;
```

6) YVC_CMD_ExeSystemReboot

```
typedef struct {
    void (*callback)(YVC_CTRL_CMD cmd, int err, void *user);
    void *user;
} YVC_ARG_SystemReboot;
```

7) YVC_CMD_ExeFactoryReset

```
typedef struct {
    void (*callback)(YVC_CTRL_CMD cmd, int err, void *user);
    void *user;
} YVC_ARG_FactoryReset;
```

8) YVC_CMD_SetNoticeEvent / YVC_CMD_GetNoticeEvent

```
typedef struct {
    unsigned int event;
} YVC_ARG_NoticeEvent;
```

9) YVC_CMD_SetBTUse / YVC_CMD_GetBTUse

```
typedef struct {
    BOOL onoff;
} YVC_ARG_BTUse;
```

10) YVC_CMD_SetBTLocalName / YVC_CMD_GetBTLocalName

```
typedef struct {
    char name[YVC_LEN_BT_LOCAL_NAME_TOTAL+1];
} YVC_ARG_BTLocalName;
```

11) YVC_CMD_SetBTOperation / YVC_CMD_GetBTOperation

```
/* Parameter definition for BT operations */
typedef enum {
    YVC_BT_OPERATION_REQ_SLEEP = 0,
    YVC_BT_OPERATION_REQ_READY,
    YVC_BT_OPERATION_REQ_DISCONNECT,
    YVC_BT_OPERATION_REQ_PAIRING_ON,
    YVC_BT_OPERATION_REQ_PAIRING_OFF,
    YVC_BT_OPERATION_REQ_LINKBACK,
    YVC_BT_OPERATION_REQ_NUM
} YVC_BT_OPERATION_REQ;

/* Parameter definition for BT operating states */
typedef enum {
    YVC_BT_OPERATION_STAT_SLEEP = 0,
    YVC_BT_OPERATION_STAT_READY,
    YVC_BT_OPERATION_STAT_DISCONNECTING,
    YVC_BT_OPERATION_STAT_PAIRING,
    YVC_BT_OPERATION_STAT_CONNECTED,
    YVC_BT_OPERATION_STAT_CONNECTING,
    YVC_BT_OPERATION_STAT_NUM
} YVC_BT_OPERATION_STAT;

typedef struct {
    union {
        YVC_BT_OPERATION_REQ req;
        YVC_BT_OPERATION_STAT stat;
    } u;
} YVC_ARG_BTOperation;
```

12) YVC_CMD_SetNoticeSoundUse / YVC_CMD_GetNoticeSoundUse

```
typedef struct {
    BOOL onoff;
} YVC_ARG_NoticeSoundUse;
```

13) YVC_CMD_SetNoticeSoundVolume / YVC_CMD_GetNoticeSoundVolume

```
typedef struct {
    short    volume;
} YVC_ARG_NoticeSoundVolume;
```

14) YVC_CMD_SetVoiceGuideUse / YVC_CMD_GetVoiceGuideUse

```
typedef struct {
    BOOL     onoff;
} YVC_ARG_VoiceGuideUse;
```

15) YVC_CMD_SetVoiceGuideLang / YVC_CMD_GetVoiceGuideLang

/* Parameter definition for the language selection list of the voice guidance */

```
typedef enum {
    YVC_VOICE_GUIDE_LANG_EN = 0,
    YVC_VOICE_GUIDE_LANG_JA,
    YVC_VOICE_GUIDE_LANG_ZN,
    YVC_VOICE_GUIDE_LANG_KO,
    YVC_VOICE_GUIDE_LANG_FR,
    YVC_VOICE_GUIDE_LANG_ES,
    YVC_VOICE_GUIDE_LANG_DE,
    YVC_VOICE_GUIDE_LANG_NUM
} YVC_VOICE_GUIDE_LANG;

typedef struct {
    YVC_VOICE_GUIDE_LANG    type;
} YVC_ARG_VoiceGuideLang;
```

16) YVC_CMD_SetVoiceGuideVolume / YVC_CMD_GetVoiceGuideVolume

```
typedef struct {
    short    volume;
} YVC_ARG_VoiceGuideVolume;
```

17) YVC_CMD_SetInitSpVolume / YVC_CMD_GetInitSpVolume

/* Parameter definition for the mode in which the speaker volume is specified when the device is connected via USB */

```
typedef enum {
    YVC_INIT_SP_VOLUME_DEVICE = 0,
    YVC_INIT_SP_VOLUME_HOST,
    YVC_INIT_SP_VOLUME_NUM
} YVC_INIT_SP_VOLUME;

typedef struct {
    YVC_INIT_SP_VOLUME    type;
} YVC_ARG_InitSpVolume;
```

18) YVC_CMD_SetPowerOnState / YVC_CMD_GetPowerOnState

/* Parameter definition for the operation modes at power-on */

```
typedef enum {
    YVC_POWER_ON_STATE_HOLD = 0,
    YVC_POWER_ON_STATE_STANDBY,
    YVC_POWER_ON_STATE_RUN,
    YVC_POWER_ON_STATE_NUM
} YVC_POWER_ON_STATE;

typedef struct {
    YVC_POWER_ON_STATE    mode;
} YVC_ARG_PowerOnState;
```

19) YVC_CMD_SetMicMuteMode / YVC_CMD_GetMicMuteMode

```

/* Parameter definition for mute modes of the microphone */
typedef enum {
    YVC_MIC_MUTE_MODE_GROUP = 0,
    YVC_MIC_MUTE_MODE_INDIVIDUAL,
    YVC_MIC_MUTE_MODE_NUM
} YVC_MIC_MUTE_MODE;

typedef struct {
    YVC_MIC_MUTE_MODE    mode;
} YVC_ARG_MicMuteMode;

```

20) YVC_CMD_SetUsbSpeed / YVC_CMD_GetUsbSpeed

```

/* Parameter definition for operating speeds of the USB port */
typedef enum {
    YVC_USB_MODE_HI_SPEED = 0,
    YVC_USB_MODE_FULL_SPEED,
    YVC_USB_MODE_NUM
} YVC_USB_MODE;

typedef struct {
    YVC_USB_MODE    mode;
    struct {
        void        (*callback)(YVC_CTRL_CMD cmd, int err, void *user);
        void        *user;
    } set;
} YVC_ARG_UsbSpeed;

```

21) YVC_CMD_SetExtTermMode / YVC_CMD_GetExtTermMode

```

/* Definition for changing the connection mode of the EXT terminal */
typedef enum {
    YVC_EXTTERM_MODE_CASCADE = 0,
    YVC_EXTTERM_MODE_CONFERENCE_SYSTEM,
    YVC_EXTTERM_MODE_NUM
} YVC_EXTTERM_MODE;

typedef struct {
    YVC_EXTTERM_MODE    mode;
} YVC_ARG_ExtTermMode;

```

22) YVC_CMD_SetSpOutSelect / YVC_CMD_GetSpOutSelect

```

/* Parameter definition for the list of which speaker is selected for output */
typedef enum {
    YVC_SP_OUT_SELECT_BUILTIN = 0,
    YVC_SP_OUT_SELECT_EXTERNAL,
    YVC_SP_OUT_SELECT_BOTH,
    YVC_SP_OUT_SELECT_NUM
} YVC_SP_OUT_SELECT;

typedef struct {
    YVC_SP_OUT_SELECT    select;
} YVC_ARG_SpOutSelect;

```

23) YVC_CMD_SetAudioInTerm / YVC_CMD_GetAudioInTerm

```

/* Parameter definition for modes of the audio input terminal */
typedef enum {
    YVC_AUDIO_IN_MODE_CONFERENCE = 0,
    YVC_AUDIO_IN_MODE_PA_MIC,
    YVC_AUDIO_IN_MODE_NUM
} YVC_AUDIO_IN_MODE;

```



```

/* Parameter definition for external microphone levels */
typedef enum {
    YVC_AUDIO_IN_LEVEL_LINE = 0,
    YVC_AUDIO_IN_LEVEL_MIC,
    YVC_AUDIO_IN_LEVEL_NUM
} YVC_AUDIO_IN_LEVEL;

/* Parameter definition for external microphone EQ modes */
typedef enum {
    YVC_AUDIO_IN_EQ_THRU = 0,
    YVC_AUDIO_IN_EQ_LOW_BOOST,
    YVC_AUDIO_IN_EQ_LOW_CUT,
    YVC_AUDIO_IN_EQ_NUM
} YVC_AUDIO_IN_EQ;

/* Parameter definition for speaker output settings of the external microphone */
typedef enum {
    YVC_AUDIO_IN_SP_OUT_OFF = 0,
    YVC_AUDIO_IN_SP_OUT_ON,
    YVC_AUDIO_IN_SP_OUT_NUM
} YVC_AUDIO_IN_SP_OUT;

typedef struct {
    struct {
        YVC_AUDIO_IN_MODE      mode;
        struct {
            YVC_AUDIO_IN_LEVEL  level;
            YVC_AUDIO_IN_EQ      eq;
            YVC_AUDIO_IN_SP_OUT  spOut;
            short                 gain;
        } paMic;
    } Lch, Rch;
} YVC_ARG_AudioInTerm;

```

24) YVC_CMD_SetAudioVolume / YVC_CMD_GetAudioVolume

```

/* Channel definition for volume operation */
typedef enum {
    YVC_AUDIO_VOLUME_CH_SP = 0,
    YVC_AUDIO_VOLUME_CH_MIC,
    YVC_AUDIO_VOLUME_CH_USB_IN,
    YVC_AUDIO_VOLUME_CH_USB_OUT,
    YVC_AUDIO_VOLUME_CH_BT_IN,
    YVC_AUDIO_VOLUME_CH_BT_OUT,
    YVC_AUDIO_VOLUME_CH_AUDIO_IN,
    YVC_AUDIO_VOLUME_CH_AUDIO_OUT,
    YVC_AUDIO_VOLUME_CH_SP_BUILTIN,
    YVC_AUDIO_VOLUME_CH_SP_EXTERNAL,
    YVC_AUDIO_VOLUME_CH_SP_EXMIC,
    YVC_AUDIO_VOLUME_CH_EXMIC,
    YVC_AUDIO_VOLUME_CH_HS_IN,
    YVC_AUDIO_VOLUME_CH_HP_OUT,
    YVC_AUDIO_VOLUME_CH_NUM
} YVC_AUDIO_VOLUME_CH;

typedef struct {
    YVC_AUDIO_VOLUME_CH  ch;
    short                 volume;
} YVC_ARG_AudioVolume;

```

25) YVC_CMD_SetAudioMute / YVC_CMD_GetAudioMute

```

/* Channel definition for mute operation */
typedef enum {
    YVC_AUDIO_MUTE_CH_SP = 0,
    YVC_AUDIO_MUTE_CH_MIC,
    YVC_AUDIO_MUTE_CH_USB_IN,
    YVC_AUDIO_MUTE_CH_USB_OUT,

```

```

YVC_AUDIO_MUTE_CH_BT_IN,
YVC_AUDIO_MUTE_CH_BT_OUT,
YVC_AUDIO_MUTE_CH_AUDIO_IN,
YVC_AUDIO_MUTE_CH_AUDIO_OUT,
YVC_AUDIO_MUTE_CH_SP_BUILTIN,
YVC_AUDIO_MUTE_CH_SP_EXTERNAL,
YVC_AUDIO_MUTE_CH_EXMIC,
YVC_AUDIO_MUTE_CH_HS_IN,
YVC_AUDIO_MUTE_CH_HP_OUT,
YVC_AUDIO_MUTE_CH_NUM
} YVC_AUDIO_MUTE_CH;

typedef struct {
    YVC_AUDIO_MUTE_CH    ch;
    BOOL                 mute;
} YVC_ARG_AudioMute;

```

26) YVC_CMD_SetMicAgc / YVC_CMD_GetMicAgc

```

/* Parameter definition for the selection list of a microphone's AGC operations */
typedef enum {
    YVC_MIC_AGC_MODE_ON = 0,
    YVC_MIC_AGC_MODE_OFF,
    YVC_MIC_AGC_MODE_NUM
} YVC_MIC_AGC_MODE;

typedef struct {
    YVC_MIC_AGC_MODE    mode;
} YVC_ARG_MicAgc;

```

27) YVC_CMD_SetMicUnitTrack / YVC_CMD_GetMicUnitTrack

```

/* Definition for the selection list of MUX (multiplexer) operations of the microphone */
typedef enum {
    YVC_MIC_MUX_MODE_TRACK = 0,
    YVC_MIC_MUX_MODE_MIX,
    YVC_MIC_MUX_MODE_NUM
} YVC_MIC_MUX_MODE;

typedef struct {
    YVC_MIC_MUX_MODE    mode;
    unsigned char        unitSelect;
} YVC_ARG_MicUnitTrack;

```

28) YVC_CMD_SetMicCapsuleTrack / YVC_CMD_GetMicCapsuleTrack

```

/* Definition for the selection list of MUX (multiplexer) operations of the microphone */
typedef enum {
    YVC_MIC_MUX_MODE_TRACK = 0,
    YVC_MIC_MUX_MODE_MIX,
    YVC_MIC_MUX_MODE_NUM
} YVC_MIC_MUX_MODE;

typedef struct {
    YVC_MIC_MUX_MODE    mode;
    unsigned char        capsuleSelect;
} YVC_ARG_MicCapsuleTrack;

```

29) YVC_CMD_SetMicFilter / YVC_CMD_GetMicFilter

```

/* Parameter definition for the selection list of microphone frequency characteristics */
typedef enum {
    YVC_MIC_FILTER_MODE_OFF = 0,
    YVC_MIC_FILTER_MODE_LOW,
    YVC_MIC_FILTER_MODE_MID,
    YVC_MIC_FILTER_MODE_HIGH,

```

```

    YVC_MIC_FILTER_MODE_NUM
} YVC_MIC_FILTER_MODE;

typedef struct {
    YVC_MIC_FILTER_MODE    mode;
} YVC_ARG_MicFilter;

```

30) YVC_CMD_SetSpFilter / YVC_CMD_GetSpFilter

```

/* Parameter definitions for the selection lit of speaker frequency characteristics */
typedef enum {
    YVC_SP_FILTER_MODE_OFF = 0,
    YVC_SP_FILTER_MODE_LOW,
    YVC_SP_FILTER_MODE_MID,
    YVC_SP_FILTER_MODE_HIGH,
    YVC_SP_FILTER_MODE_NUM
} YVC_SP_FILTER_MODE;

typedef struct {
    YVC_SP_FILTER_MODE    mode;
} YVC_ARG_SpFilter;

```

31) YVC_CMD_ExeAAT

```

/* Parameter definition for the list of operations in automatic audio tuning */
typedef enum {
    YVC_AAT_ACTION_STOP = 0,
    YVC_AAT_ACTION_START,
    YVC_AAT_NUM
} YVC_AAT_ACTION;

typedef struct {
    YVC_AAT_ACTION    action;
} YVC_ARG_AAT;

```

32) YVC_CMD_GetAATProgress

```

typedef struct {
    unsigned int    progress;
} YVC_ARG_AATProgress;

```

33) YVC_CMD_GetAATResult

```

/* Parameter definition for measurement results of automatic audio tuning */
typedef enum {
    YVC_AAT_RESULT_OK = 0,
    YVC_AAT_RESULT_CANCEL,
    YVC_AAT_RESULT_WARNING,
    YVC_AAT_RESULT_NO_MIC,
    YVC_AAT_RESULT_SP_VOL_SMALL,
    YVC_AAT_RESULT_EXT_SP_UNDETECT,
    YVC_AAT_RESULT_NUM
} YVC_AAT_RESULT;

typedef struct {
    YVC_AAT_RESULT    result;
    unsigned int        warning;
} YVC_ARG_AATResult;

```

34) YVC_CMD_MonMicCapsule

```

typedef struct {
    unsigned char    trackInfo[YVC_LEN_MIC_UNIT_MAX];
} YVC_ARG_MonMicCapsule;

```

35) YVC_CMD_MonMicUnit

```
typedef struct {  
    int    unitId;  
} YVC_ARG_MonMicUnit;
```

36) YVC_CMD_MonHeadsetPortState

```
typedef struct {  
    unsigned int    state;  
} YVC_ARG_MonHeadsetPortState
```

37) YVC_CMD_MonBatteryLevel

```
typedef struct {  
    unsigned int    level;  
    unsigned int    charging;  
} YVC_ARG_MonBatteryLevel
```

4.7-9 Constants

API version definition

Symbol name	Value	Description
YVC_LEN_API_VERSION	4	Number of characters for the API version

Definition common to the YVC series

Symbol name	Value	Description
YVC_EVENT_DATA_MICMUTE_UNIT0	0x1	Microphone unit 0
YVC_EVENT_DATA_MICMUTE_UNIT1	0x2	Microphone unit 1
YVC_EVENT_DATA_MICMUTE_UNIT2	0x4	Microphone unit 2
YVC_EVENT_DATA_MICMUTE_UNIT3	0x8	Microphone unit 3
YVC_EVENT_DATA_MICMUTE_UNIT4	0x10	Microphone unit 4
YVC_EVENT_DATA_HOOKBTN_SHORT	0	The on/off-hook button is pressed.
YVC_EVENT_DATA_HOOKBTN_LONG	1	The on/off-hook button is held.
YVC_PRM_NOTICE_EVENT_MIC_UNIT_NUM	0x1	Event for indicating the change in the connection state of a microphone unit
YVC_PRM_NOTICE_EVENT_MIC_MUTE_STATE	0x2	Event for indicating the change in the mute mode of a microphone
YVC_PRM_NOTICE_EVENT_HOOK_BTN	0x4	Event for indicating the change in the state of the on/off-hook button
YVC_PRM_MIC_UNIT0	0x1	Microphone unit 0
YVC_PRM_MIC_UNIT1	0x2	Microphone unit 1
YVC_PRM_MIC_UNIT2	0x4	Microphone unit 2
YVC_PRM_MIC_UNIT3	0x8	Microphone unit 3
YVC_PRM_MIC_UNIT4	0x10	Microphone unit 4
YVC_PRM_MIC_CAPSULE_NONE	0x0	Inactive microphone capsule
YVC_PRM_MIC_CAPSULE_CENTER	0x1	Microphone capsule (center)
YVC_PRM_MIC_CAPSULE_LEFT	0x2	Microphone capsule (left)
YVC_PRM_MIC_CAPSULE_RIGHT	0x4	Microphone capsule (right)
YVC_PRM_AAT_WARNING_OK	0x0	Automatic audio tuning successfully completed
YVC_PRM_AAT_WARNING_TOOCLOSE	0x1	Automatic audio tuning error: The distance between the microphone unit and the Control Unit is too close.
YVC_PRM_AAT_WARNING_TOOFAR	0x2	Automatic audio tuning error: The distance between the microphone unit and the Control Unit is too far.
YVC_PRM_AAT_WARNING_LONGDELAY	0x4	Automatic audio tuning error: External speaker delay is too long.
YVC_PRM_AAT_WARNING_DISTORTION	0x8	Automatic audio tuning error: The external speaker has excessively high distortion.
YVC_LEN_VERSION_INFO	4	Number of characters for the firmware version
YVC_LEN_MODEL_NAME	40	Maximum number of characters for the model name
YVC_LEN_SERIAL_NUMBER	16	Maximum number of characters for the serial number
YVC_LEN_BT_LOCAL_NAME	11	Number of characters of Local Name for Bluetooth
YVC_LEN_BT_LOCAL_NAME_TOTAL	20	Total number of characters of Local Name for Bluetooth (including the model name)
YVC_LEN_MIC_UNIT_MAX	5	Maximum number of connectable microphone units

Definition for YVC-1000

Symbol name	Value	Description
YVC1000_LEN_MIC_UNIT_MAX	5	Maximum number of connectable microphone units

Definition for YVC-300

Symbol name	Value	Description
YVC300_LEN_MIC_UNIT_MAX	2	Maximum number of connectable microphone units (with the main unit)

4.8 Error definition

This section defines API errors.

4.8-1 Error code

Error code	Value	Description
YVCAPI_ERROR_PENDING	1	The request is accepted (running).
YVCAPI_ERROR_NOERROR	0	Successful completion
YVCAPI_ERROR_UNKNOWN	-1	Abnormal termination (unknown error)
YVCAPI_ERROR_DEVNOTFOUND	-2	The specified device is not found.
YVCAPI_ERROR_MEMEXHAUST	-3	Memory is exhausted.
YVCAPI_ERROR_STATUS	-4	A status error occurred.
YVCAPI_ERROR_INVALIDCOMMAND	-5	The control command code is invalid.
YVCAPI_ERROR_INVALIDHANDLE	-6	The handle is invalid.
YVCAPI_ERROR_INVALIDARGS	-7	The argument is invalid.
YVCAPI_ERROR_CTRLEXEC	-8	The API function failed to control the device.
YVCAPI_ERROR_HIDTIMEOUT	-9	A time-out occurred during HID communication.
YVCAPI_ERROR_UNSUPPORTVER	-15	The firmware version is not supported.
YVCAPI_ERROR_APPRUNNING	-16	The API function cannot be executed because other related applications are running.
YVCAPI_ERROR_INVALIDPARAM	-17	The parameter value of the device is invalid.

4.9 Limitations

This section describes limitations when the API is used.

- API version 1.01 or earlier only supports one YVC-Series device that is connected to a host PC.
- When the host PC already runs a different YVC-Series application, calling the [YVC_DeviceAttach](#) function or the [YVC_DeviceControl](#) function returns an [error](#) (YVCAPI_ERROR_APPRUNNING).

•Windows

Model name	Application name	File name
YVC-1000	YVC-1000 Configurator	<i>YVC-1000 Configurator.exe</i>
	YVC-1000 FW-Updater	<i>YVC-1000 Firmware <version>.exe</i>
YVC-300	YVC-300 FW-Updater	<i>YVC-300 Firmware <version>.exe</i>
YVC-330	YVC-330 FW-Updater	<i>YVC-330 Firmware <version>.exe</i>
YVC-200	YVC-200 FW-Updater	<i>YVC-200 Firmware <version>.exe</i>

•Mac

Model name	Application name	File name
YVC-1000	YVC-1000 Configurator	<i>YVC-1000 Configurator.app</i>
	YVC-1000 FW-Updater	<i>YVC-1000 Firmware <version>.app</i>
YVC-300	YVC-300 FW-Updater	<i>YVC-300 Firmware <version>.app</i>
YVC-330	YVC-330 FW-Updater	<i>YVC-330 Firmware <version>.app</i>
YVC-200	YVC-200 FW-Updater	<i>YVC-200 Firmware <version>.app</i>

4.10 Notes

This section describes a note when the API is used.

- When daisy-chain-connected YVC-300 or YVC-330 devices are running, a command entered is applied only to the parent device.

5. API Usage Example

This chapter provides an example of using the API.

*In the case of Mac, it is necessary to add the YVCAPI.framework to the project beforehand.

```
#if (defined(_WIN32) || defined(_WIN64))
#include <windows.h>
#else /* MAC */
#import <Foundation/Foundation.h>
#endif
:
#include <YVCAPI.h>

#if (defined(_WIN32) || defined(_WIN64))
typedef int (__stdcall *FNC_YVC_DEVICE_INFO)(YVC_DEV_TYPE type,
                                             YVC_DEV_INFO *info);
typedef int (__stdcall *FNC_YVC_DEVICE_ATTACH)(YVC_DEV_HANDLE *handle,
                                              YVC_DEV_TYPE type,
                                              YVC_EVENT_CALLBACK callback,
                                              void *user);
typedef int (__stdcall *FNC_YVC_DEVICE_CONTROL)(YVC_DEV_HANDLE handle,
                                              YVC_CTRL_CMD cmd,
                                              YVC_CTRL_ARG *arg);
typedef int (__stdcall *FNC_YVC_DEVICE_DETACH)(YVC_DEV_HANDLE handle);
typedef int (__stdcall *FNC_YVC_API_INFO)(YVC_API_INFO *info);
HMODULE hModule
#else /* MAC */
typedef int (*FNC_YVC_DEVICE_INFO)(YVC_DEV_TYPE type,
                                   YVC_DEV_INFO *info);
typedef int (*FNC_YVC_DEVICE_ATTACH)(YVC_DEV_HANDLE *handle,
                                    YVC_DEV_TYPE type,
                                    YVC_EVENT_CALLBACK callback,
                                    void *user);
typedef int (*FNC_YVC_DEVICE_CONTROL)(YVC_DEV_HANDLE handle,
                                    YVC_CTRL_CMD cmd,
                                    YVC_CTRL_ARG *arg);
typedef int (*FNC_YVC_DEVICE_DETACH)(YVC_DEV_HANDLE handle);
typedef int (*FNC_YVC_API_INFO)(YVC_API_INFO *info);
#endif
FNC_YVC_DEVICE_INFO DeviceInfo;
FNC_YVC_DEVICE_ATTACH DeviceAttach;
FNC_YVC_DEVICE_CONTROL DeviceControl;
FNC_YVC_DEVICE_DETACH DeviceDetach;
FNC_YVC_API_INFO APIInfo;
```



```

/* Get the YVCAPI module handle (map it to the address space) */
int YVCAPISample1(void)
{
    #if (defined(_WIN32) || defined(_WIN64))
        hModule = LoadLibrary(_T("YVCAPI.dll"));
        if (hModule == NULL) {
            return -1;
        }
        DeviceInfo = (FNC_YVC_DEVICE_INFO)GetProcAddress(hModule, "YVC_DeviceInfo");
        if (DeviceInfo == NULL) {
            return -1;
        }
        DeviceAttach = (FNC_YVC_DEVICE_ATTACH)GetProcAddress(hModule, "YVC_DeviceAttach");
        if (DeviceAttach == NULL) {
            return -1;
        }
        DeviceDetach = (FNC_YVC_DEVICE_DETACH)GetProcAddress(hModule, "YVC_DeviceDetach");
        if (DeviceDetach == NULL) {
            return -1;
        }
        DeviceControl = (FNC_YVC_DEVICE_CONTROL)GetProcAddress(hModule, "YVC_DeviceControl");
        if (DeviceControl == NULL) {
            return -1;
        }
        APIInfo = (FNC_YVC_API_INFO)GetProcAddress(hModule, "YVC_APIInfo");
        if (APIInfo == NULL) {
            return -1;
        }
    #else /* MAC */
        DeviceInfo = YVC_DeviceInfo;
        DeviceAttach = YVC_DeviceAttach;
        DeviceDetach = YVC_DeviceDetach;
        DeviceControl = YVC_DeviceControl;
        APIInfo = YVC_APIInfo;
        if (DeviceInfo == NULL || DeviceAttach == NULL || DeviceDetach == NULL || DeviceControl == NULL || APIInfo ==
        NULL) {
            return -1;
        }
    #endif
    return 0; /* Successfully completed */
}

#if (defined(_WIN32) || defined(_WIN64))
/* Release the YVCAPI module handle (unmap it) */
int YVCAPISample2(void)
{
    if (hModule) {
        FreeLibrary(hModule);
    }
    return 0; /* Successfully completed */
}
#endif

```

```

YVC_DEV_HANDLE handle = NULL;

/* Start communication with YVC-1000 */
int YVCAPISample3(void)
{
    int ret;

    /* Get the API information (version) */
    YVC_API_INFO    apiInfo;
    ret = APIInfo(&apiInfo);
    if (ret != YVCAPI_ERROR_NOERROR) {
        return -1;
    }
    /* Show the version obtained */
    printf("API Version : %s\n", apiInfo.version);

    /* Get the device connection information */
    YVC_DEV_INFO devInfo;
    /* Get the number of connected YVC-1000 devices */
    ret = DeviceInfo(YVC_DEV_TYPE_YVC1000, &devInfo);
    if (ret != YVCAPI_ERROR_NOERROR) {
        return -1;
    }
    if (devInfo.num < 1) {
        return -1; /* Wait until YVC-1000 is connected */
    }

    /* Get the communication handle with YVC-1000 */
    ret = DeviceAttach(&handle, YVC_DEV_TYPE_YVC1000, YVCAPISample6, NULL);
    if (ret != YVCAPI_ERROR_NOERROR || handle == NULL) {
        return -1;
    }

    /* Get the firmware version of YVC-1000 */
    YVC_ARG_VersionInfo VersionInfo;
    ret = DeviceControl(handle, YVC_CMD_GetVersionInfo, (YVC_CTRL_ARG *)&VersionInfo);
    if (ret != YVCAPI_ERROR_NOERROR) {
        goto END;
    }
    /* Show the version obtained */
    printf("YVC-1000 FW Version : %s\n", VersionInfo.version);

    /* Get the mode */
    YVC_ARG_SystemMode    SystemMode;
    ret = DeviceControl(handle, YVC_CMD_GetSystemMode, (YVC_CTRL_ARG *)&SystemMode);
    if (ret != YVCAPI_ERROR_NOERROR) {
        goto END;
    }
    /* Show the mode obtained */
    printf("Mode : %s\n", SystemMode.mode == YVC_SYSTEM_MODE_NORMAL ? "NORMAL" : "FWUP");

    :

END:
    return ret;
}

/*End the communication with YVC-1000 */
int YVCAPISample4(void)
{
    int ret;

    /* End of communication */
    if (handle != NULL) {
        ret = DeviceDetach(handle);
    }
    return ret;
}

```

```

/* Set the voice guidance language of YVC-1000 to "Japanese" */
int YVCAPISample5(void)
{
    int ret;

    YVC_ARG_VoiceGuideLang    VoiceGuideLang;
    /* Get the setting for the voice guidance language */
    ret = DeviceControl(handle, YVC_CMD_GetVoiceGuideLang, (YVC_CTRL_ARG *)&VoiceGuideLang);
    if (ret != YVCAPI_ERROR_NOERROR) {
        return ret;
    }
    if (VoiceGuideLang.type != YVC_VOICE_GUIDE_LANG_JA) {
        /* Set the language to "Japanese" */
        VoiceGuideLang.type = YVC_VOICE_GUIDE_LANG_JA;
        ret = DeviceControl(handle, YVC_CMD_SetVoiceGuideLang, (YVC_CTRL_ARG *)&VoiceGuideLang);
        if (ret == YVCAPI_ERROR_NOERROR) {
            printf("The language has been set to Japanese.\n");
        }
    }

    return ret;
}

/* Callback function for event notification */
void YVCAPISample6(YVC_EVENT event, int data, void *user)
{
    WPARAM wParam;
    LPARAM lParam;

    switch(event) {
    case YVC_EVENT_DEV_DISCONNECT:
        /* The device was removed */
        break;
    case YVC_EVENT_MIC_UNIT_NUM:
        /* The connection state of the microphone unit changed */
        break;
    case YVC_EVENT_MIC_MUTE_STATE:
        /* The mute mode of the microphone changed */
        break;
    default:
        return;
    }

    #if (defined(_WIN32) || defined(_WIN64))
        wParam = (WPARAM)event;
        lParam = (LPARAM)data;
        /* Specify the window handle for the destination and post a message */
        ::PostMessage(hWnd, WM_APP, wParam, lParam);
    #else /* MAC */
        NSDictionary *dict = [NSDictionary dictionaryWithObjectsAndKeys:
                               [NSNumber numberWithInt: (int)event], @"event",
                               [NSNumber numberWithInt: data], @"data",
                               nil
                              ];
        /* Specify the destination and post a message */
        [[NSNotificationCenter defaultCenter] postNotificationName: @"SAMPLE"
                                                    object: observer
                                                    userInfo: dict];
    #endif
}

```

6. Appendices

6.1 Lists of modes and firmware versions supported by the commands

1) System control related commands

Defined command name	YVC-1000		YVC-300		YVC-330		YVC-200	
	NORMAL	FWUP	NORMAL	FWUP	NORMAL	FWUP	NORMAL	FWUP
YVC_CMD_GetSystemMode	2.00 or later	2.00 or later	1.04 or later	1.04 or later	1.02 or later	1.02 or Later	1.01 or later	1.01 or later
YVC_CMD_GetVersionInfo	2.00 or later	2.00 or later	1.04 or later	1.04 or later	1.02 or later	1.02 or Later	1.01 or later	1.01 or later
YVC_CMD_GetModelName	2.00 or later	2.00 or later	1.04 or later	1.04 or later	1.02 or later	1.02 or Later	1.01 or later	1.01 or later
YVC_CMD_GetSerialNumber	2.00 or later	2.00 or later	1.04 or later	1.04 or later	1.02 or later	1.02 or later	1.01 or later	1.01 or later
YVC_CMD_GetMicNum	2.00 or later							
YVC_CMD_ExecSystemReboot	2.00 or later	2.00 or later	1.04 or later	1.04 or later	1.02 or later	1.02 or later	1.01 or later	1.01 or later
YVC_CMD_ExecFactoryReset	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_ExecStandbyMode	2.00 or later						1.01 or later	
YVC_CMD_SetNoticeEvent	2.10 or later		1.09 or later		1.02 or later		1.01 or later	
YVC_CMD_GetNoticeEvent	2.10 or later		1.09 or later		1.02 or later		1.01 or later	
YVC_CMD_SetBTUse	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_GetBTUse	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_SetBTLocalName	2.09 or later		1.08 or later		1.02 or later		1.01 or later	
YVC_CMD_GetBTLocalName	2.09 or later		1.08 or later		1.02 or later		1.01 or later	
YVC_CMD_SetBTOperation	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_GetBTOperation	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_SetNoticeSoundUse	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_GetNoticeSoundUse	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_SetNoticeSoundVolume			1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_GetNoticeSoundVolume			1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_SetVoiceGuideUse	2.00 or later							
YVC_CMD_GetVoiceGuideUse	2.00 or later							
YVC_CMD_SetVoiceGuideLang	2.00 or later							
YVC_CMD_GetVoiceGuideLang	2.00 or later							
YVC_CMD_SetVoiceGuideVolume	2.00 or later							
YVC_CMD_GetVoiceGuideVolume	2.00 or later							
YVC_CMD_SetInitSpVolume	2.00 or		1.04 or		1.02 or		1.01 or	

	later		later		later		later	
YVC_CMD_GetInitSpVolume	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_SetPowerOnState	2.00 or later							
YVC_CMD_GetPowerOnState	2.00 or later							
YVC_CMD_SetMicMuteMode	2.10 or later							
YVC_CMD_GetMicMuteMode	2.10 or later							
YVC_CMD_SetUsbSpeed	2.03 or later							
YVC_CMD_GetUsbSpeed	2.03 or later							
YVC_CMD_SetExtTermMode			1.04 or later		1.02 or later			
YVC_CMD_GetExtTermMode			1.04 or later		1.02 or later			

2) Audio control related commands

Defined command name	YVC-1000		YVC-300		YVC-330		YVC-200	
	NORMAL	FWUP	NORMAL	FWUP	NORMAL	FWUP	NORMAL	FWUP
YVC_CMD_SetSpOutSelect	2.00 or later		1.04 or later		1.02 or later			
YVC_CMD_GetSpOutSelect	2.00 or later		1.04 or later		1.02 or later			
YVC_CMD_SetAudioInTerm	2.00 or later		1.04 or later		1.02 or later			
YVC_CMD_GetAudioInTerm	2.00 or later		1.04 or later		1.02 or later			
YVC_CMD_SetAudioVolume	2.00 or later		1.09 or later		1.02 or later		1.01 or later	
YVC_CMD_GetAudioVolume	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_SetAudioMute	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_GetAudioMute	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_SetMicAgc	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_GetMicAgc	2.00 or later		1.04 or later		1.02 or later		1.01 or later	
YVC_CMD_SetMicUnitTrack	2.00 or later							
YVC_CMD_GetMicUnitTrack	2.00 or later							
YVC_CMD_SetMicCapsuleTrack			1.09 or later		1.02 or later			
YVC_CMD_GetMicCapsuleTrack			1.09 or later		1.02 or later			
YVC_CMD_SetMicFilter	2.00 or later		1.09 or later		1.02 or later		1.01 or later	
YVC_CMD_GetMicFilter	2.00 or later		1.09 or later		1.02 or later		1.01 or later	
YVC_CMD_SetSpFilter	2.00 or later		1.09 or later		1.02 or later			

YVC_CMD_GetSpFilter	2.00 or later		1.09 or later		1.02 or later			
-------------------------------------	---------------	--	---------------	--	---------------	--	--	--

3) Automatic audio tuning control related commands

Defined command name	YVC-1000		YVC-300		YVC-330		YVC-200	
	NORMAL	FWUP	NORMAL	FWUP	NORMAL	FWUP	NORMAL	FWUP
YVC_CMD_ExcAAT	2.10 or later							
YVC_CMD_GetAATProgress	2.10 or later							
YVC_CMD_GetAATResult	2.10 or later							

4) Device monitoring related commands

Defined command name	YVC-1000		YVC-300		YVC-330		YVC-200	
	NORMAL	FWUP	NORMAL	FWUP	NORMAL	FWUP	NORMAL	FWUP
YVC_CMD_MonMicCapsule	2.00 or later		1.04 or later		1.02 or later			
YVC_CMD_MonMicUnit	2.10 or later							
YVC_CMD_MonHeadsetPortState							1.01 or later	
YVC_CMD_MonBatteryLevel							1.01 or later	

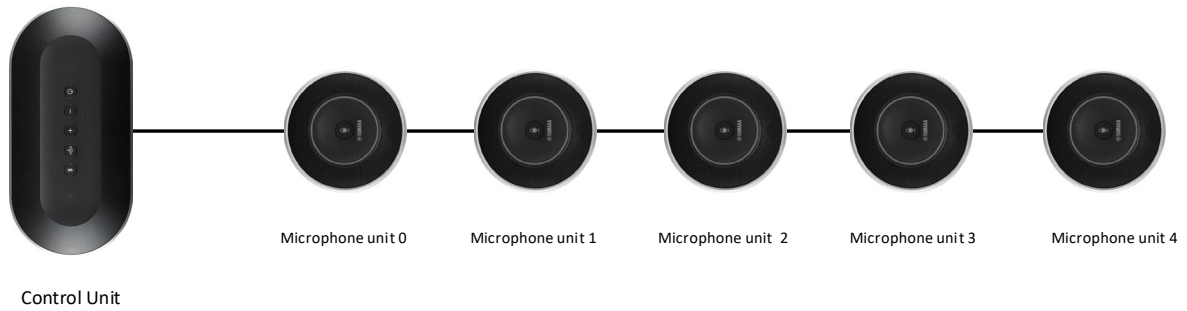
6.2 Microphone capsule positions

This section shows the positions of microphone capsules for each model.

1) YVC-1000

The figure below shows the microphone unit IDs for YVC-1000.

The ID starts with 0, beginning from the microphone unit connected closest to the Control Unit.



The following figure shows the positions of microphone capsules on the YVC-1000 microphone unit.



2) YVC-300

The following figure shows the positions of microphone capsules on YVC-300.



3) YVC-330

The following figure shows the positions of microphone capsules on YVC-330.



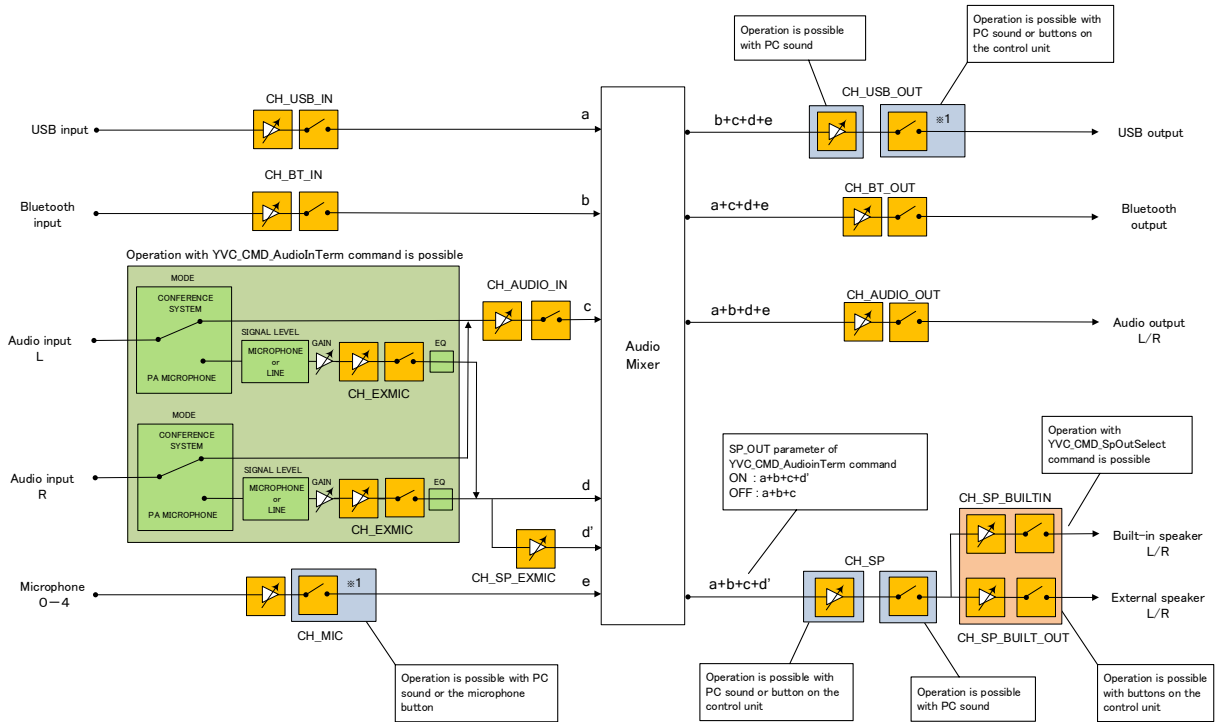
6.3 Audio block diagram

This section provides an audio block diagram of each model.

1) YVC-1000

The following figure shows the audio block diagram of YVC-1000.

【Version 3.00 or later】



Operation with YVC_CMD_AudioVolume command is possible

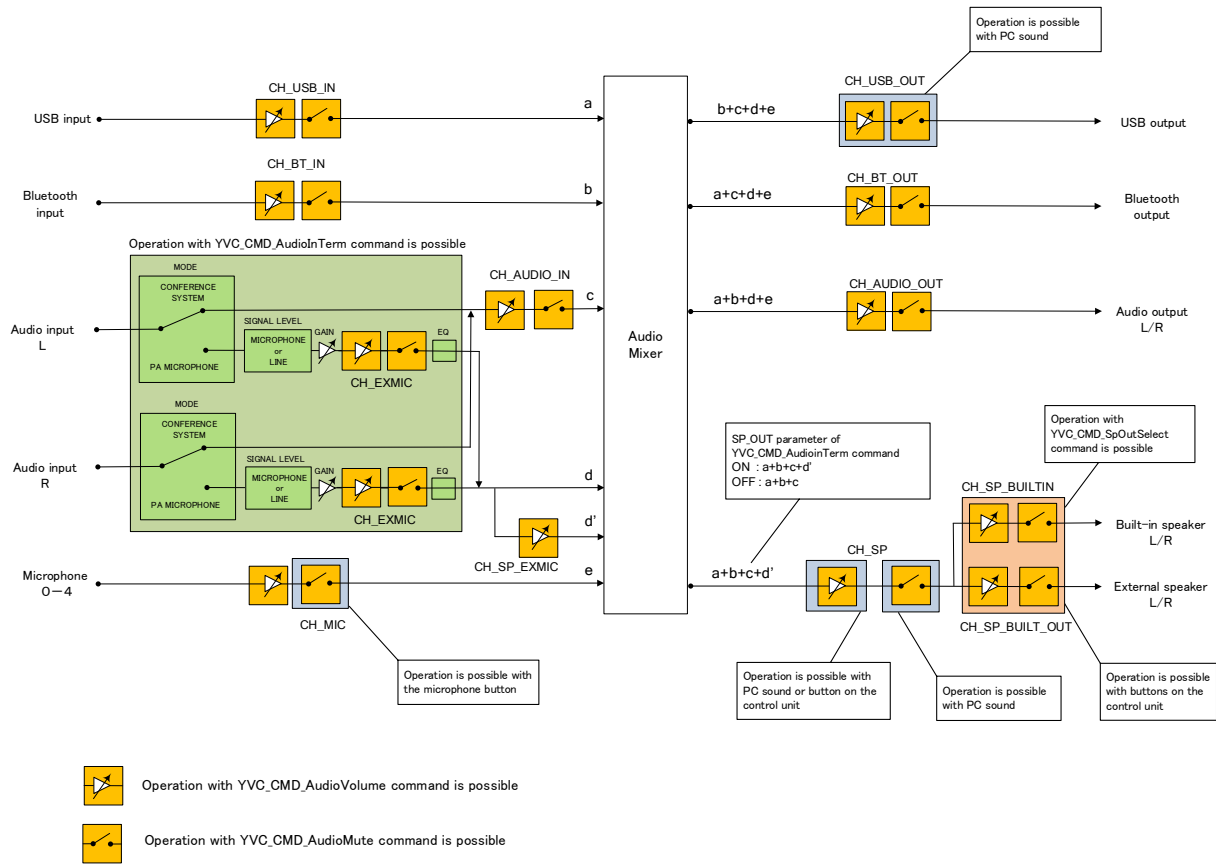


Operation with YVC_CMD_AudioMute command is possible

※1

The two microphone mutes are linked, but these are not linked if you use YVC_CMD_AudioMute command to control microphone mute. When non-linking is selected, the system operates the same as with firmware Ver.2.09 to Ver.2.11.

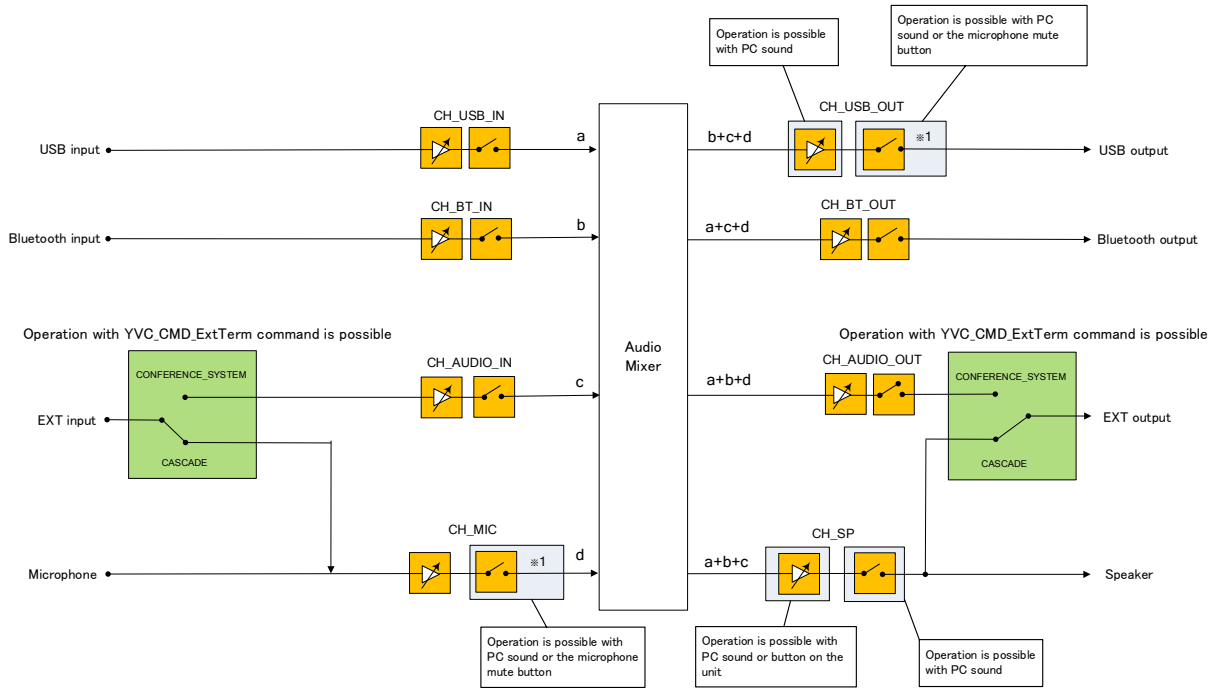
【Version 2.11 or earlier】



2) YVC-300

The following figure shows the audio block diagram of YVC-300.

【Version 2.00 or later】



Operation with YVC_CMD_AudioVolume command is possible

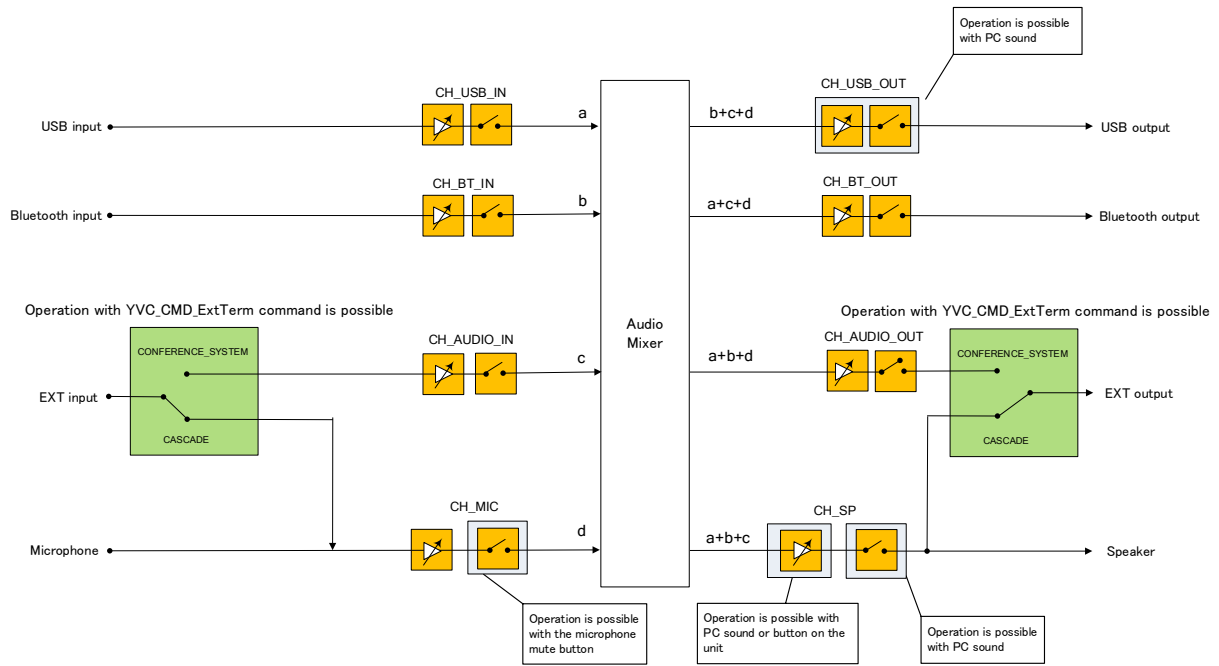


Operation with YVC_CMD_AudioMute command is possible

※1

The two microphone mutes are linked, but these are not linked if you use YVC_CMD_AudioMute command to control microphone mute.

【Version 1.09 or earlier】



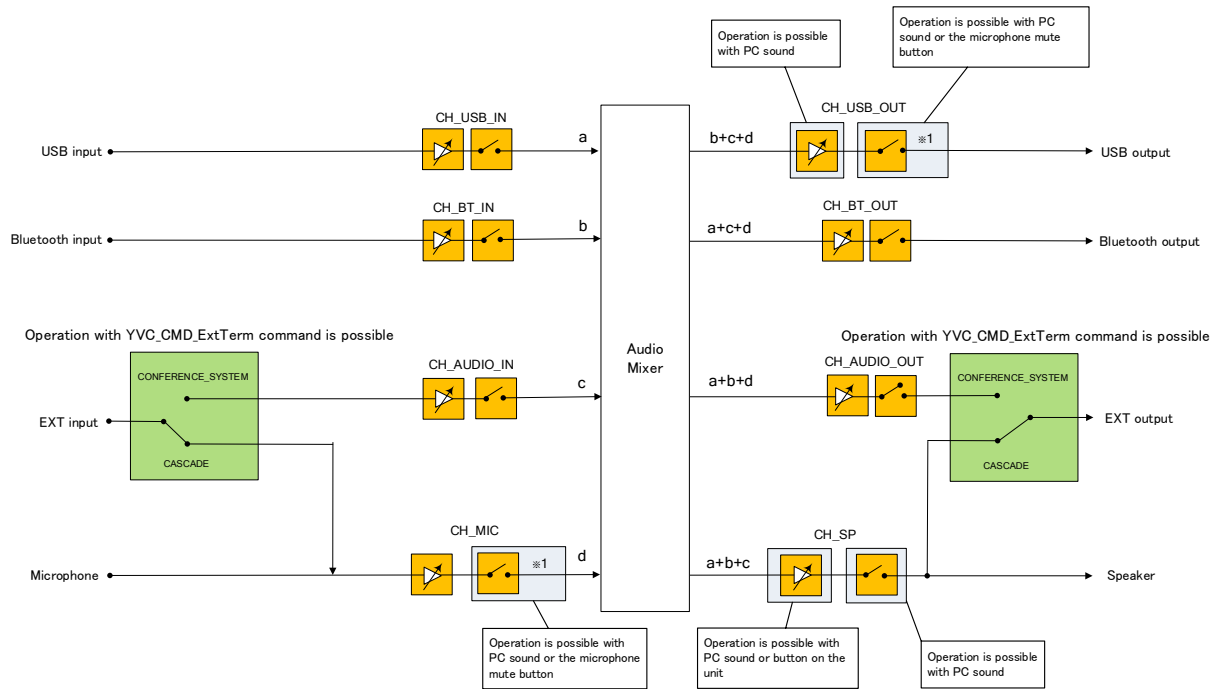
Operation with YVC_CMD_AudioVolume command is possible



Operation with YVC_CMD_AudioMute command is possible

3) YVC-330

The following figure shows the audio block diagram of YVC-330.



Operation with YVC_CMD_AudioVolume command is possible



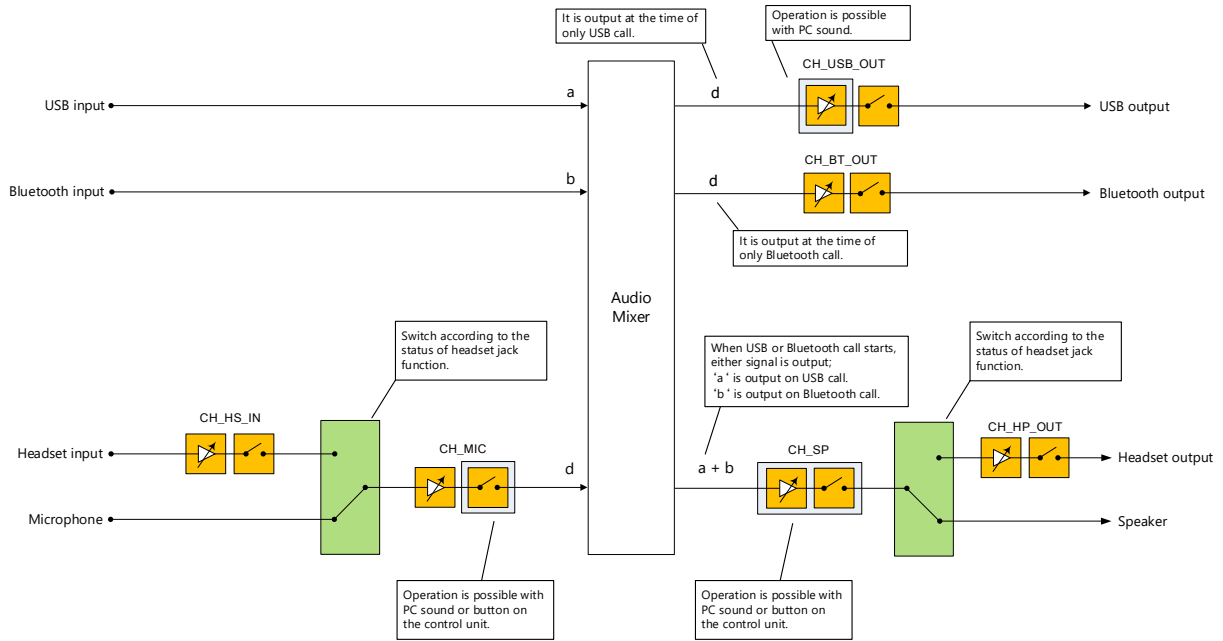
Operation with YVC_CMD_AudioMute command is possible

※1

The two microphone mutes are linked, but these are not linked if you use YVC_CMD_AudioMute command to control microphone mute.

4) YVC-200

The following figure shows the audio block diagram of YVC-200.



Operation with YVC_CMD_AudioVolume command is possible



Operation with YVC_CMD_AudioMute command is possible