RX Family

R01AN2027EJ0144
Rev.1.44
Mar 01, 2025

USB Host Communications Device Class Driver (HCDC) using Firmware Integration Technology

Introduction

This application note describes USB Host Communication Device Class Driver (HCDC), which utilizes Firmware Integration Technology (FIT). This module operates in combination with the USB Basic Host and Peripheral Driver (USB-BASIC-FW FIT module). It is referred to below as the USB HCDC FIT module.

Target Device

RX65N/RX651 Group
RX64M Group
RX71M Group
RX66T Group
RX72T Group
RX72M Group
RX66N Group
RX72N Group
RX671 Group

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Related Documents

1. USB Basic Host and Peripheral Driver using Firmware Integration Technology Application Note (Document number.R01AN2025)
2. Universal Serial Bus Revision 2.0 specification
   http://www.usb.org/developers/docs/
3. USB Class Definitions for Communications Devices Revision 1.2
4. USB Communications Class Subclass Specification for PSTN Devices Revision 1.2
   http://www.usb.org/developers/docs/
5. RX64M Group User's Manual: Hardware (Document number.R01UH0377)
6. RX71M Group User's Manual: Hardware (Document number .R01UH0493)
7. RX65N/RX651 Group User's Manual: Hardware (Document number .R01UH0590)
8. RX65N/RX651-2M Group User's Manual: Hardware (Document number .R01UH0659)
9. RX66T User's Manual: Hardware (Document number. R01UH0749)
10. RX72T User's Manual: Hardware (Document number. R01UH0803)
11. RX72M User's Manual: Hardware (Document number. R01UH0804)
12. RX66N User's Manual: Hardware (Document number. R01UH0825)
13. RX72N User's Manual: Hardware (Document number. R01UH0824)
14. RX671 User's Manual: Hardware (Document number. R01UH0899)

- Renesas Electronics Website
  http://www.renesas.com/
- USB Devices Page
  http://www.renesas.com/prod/usb/

Contents

1. Overview 3
2. Software Configuration 4
3. API Information 5
4. Target Peripheral List（TPL） 9
5. Communication Device Class (CDC), PSTN and ACM 10
6. USB Host Communication Device Class Driver (HCDC) 15
7. API Functions 18
8. Configuration (r_usb_hcdc_config.h) 19
9. Creating an Application 20

1. Overview

The USB HCDC FIT module, when used in combination with the USB-BASIC-FW FIT module, operates as a USB host communications device class driver (HCDC). The HCDC conforms to the PSTN device subclass abstract control model of the USB communication device class specification (CDC) and enables communication with a CDC peripheral device.

This module supports the following functions.

1. Checking of connected devices
2. Implementation of communication line settings
3. Acquisition of the communication line state
4. Data transfer to and from a CDC peripheral device
5. HCDC can connect maximum 2 CDC devices to 1 USB module by using USB Hub.

1.1 Please be sure to read

Please refer to the document (Document number: R01AN2025) for *USB Basic Host and Peripheral Driver using Firmware Integration Technology Application Note* when creating an application program using this driver.

This document is located in the "**reference_documents**" folder within this package.

1.2 Note

1. This driver is not guaranteed to provide USB communication operation. The customer should verify operation when utilizing it in a system and confirm the ability to connect to a variety of different types of devices.
2. It is not necessary to use this module since USBX driver is used when using Azure RTOS.
3. For Azure RTOS API and FileX API, please refer to Azure RTOS and USBX documentations.

1.3 Limitations

This driver is subject to the following limitations.

1. Only one stage of the USB hub can be used. (USB Hub is not supported in Azure RTOS(USBX))
2. Suspend and resume are not supported for CDC devices connected to the USB hub and USB hub downstream ports.
3. Suspend is not supported when data transfer is in progress. Confirm that data transfer has completed before executing suspend.
4. Use of compound USB devices with CDC class support is not supported.
5. In the Azure RTOS version, the number of CDC devices that can be connected is 1 device.

1.4 Terms and Abbreviations

APL	:	Application program
CDC	:	Communications Devices Class
CDCC	:	Communications Devices Class － Communications Class Interface
CDCD	:	Communications Devices Class － Data Class Interface
HCD	:	Host Control Driver for USB-BASIC-FW
HCDC	:	Host Communication Devices Class
HDCD	:	Host Device Class Driver (Device Driver and USB Class Driver)
HUBCD	:	Hub Class Driver
IDE	:	Integrated Development Environment
MGR	:	Peripheral Device State Manager for HCD
Non-OS	:	USB Driver for OS-less
RSK	:	Renesas Starter Kits
RTOS	:	USB Driver for the real-time OS
USB-BASIC-FW	:	USB Basic Host and Peripheral Driver

1.5 USB HCDC FIT module

User needs to integrate this module to the project using r_usb_basic. User can control USB H/W by using this module API after integrating to the project.

2. Software Configuration

Table 2-1 lists the modules, and Figure 2.1 shows a block diagram of HCDC.

Table 2-1 Modules

Module	Description
APL	User application program. Created by customer.
HCDC	Requests CDC requests command and the data transfer from APL to HCD .
MGR / HUB	Enumerates the connected devices and starts HCDC. Also performs device state management.
HCD	USB host H/W control driver.

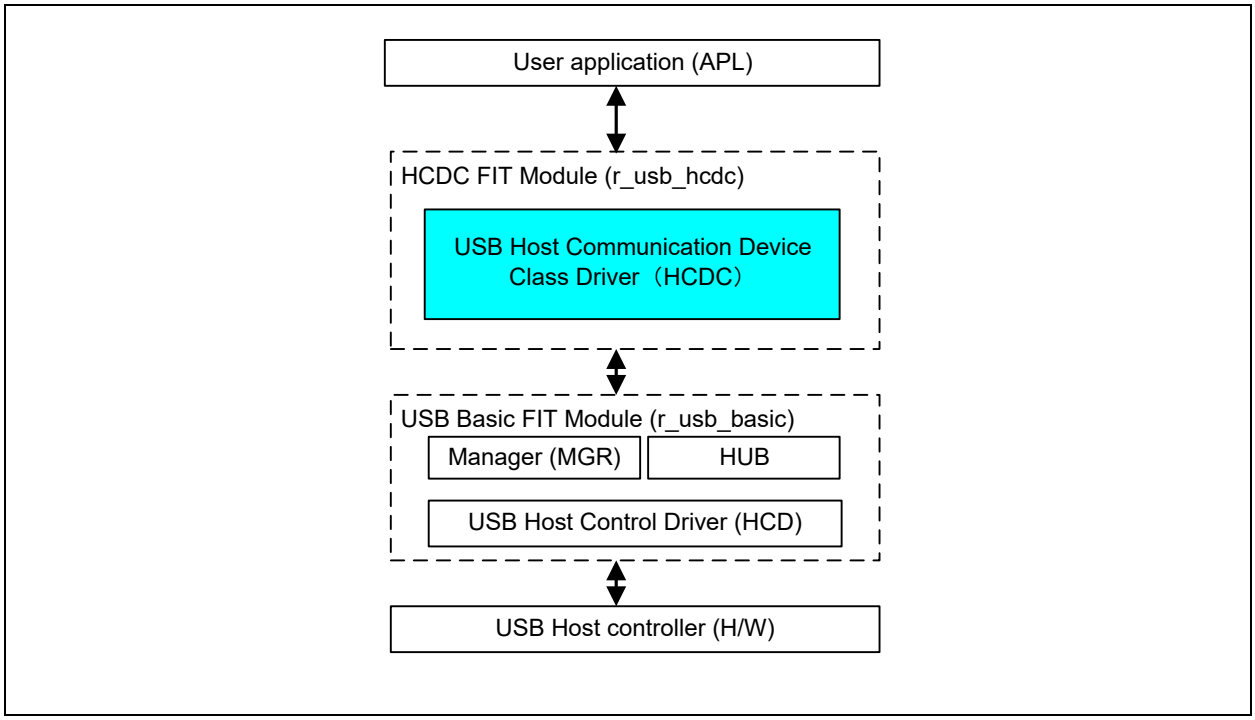

Figure 2.1 Software Module Structure

3. API Information

This Driver API follows the Renesas API naming standards.

3.1 Hardware Requirements

This driver requires your MCU support the following features:

- USB

3.2 Software Requirements

This driver is dependent upon the following packages:

- r_bsp
- r_usb_basic

3.3 Operating Confirmation Environment

Table 3-1 shows the operating confirmation environment of this driver.

Table 3-1 Operating Confirmation Environment

Item	Contents
C compiler	Renesas Electronics C/C++ compiler for RX Family V.3.07.00 (The option "-lang=C99" is added to the default setting of IDE)
	GCC for Renesas RX 8.3.0.202411 (The option "-std=gnu99" is added to the default setting of IDE)
	IAR C/C++ Compiler for Renesas RX version 5.10.1
Real-Time OS	FreeRTOS V.10.0.0 RI600V4 Azure RTOS (USBX) 6.1.12
Endian	Little Endian, Big Endian
USB Driver Revision Number	Rev 1.44
Using Board	Renesas Starter Kits for RX64M Renesas Starter Kits for RX71M Renesas Starter Kits for RX65N, Renesas Starter Kits for RX65N-2MB Renesas Starter Kits for RX72T Renesas Starter Kits for RX72M Renesas Starter Kits for RX72N Renesas Starter Kits for RX671

3.4 Usage of Interrupt Vector

Table 3-2 shows the interrupt vector which this driver uses.

Table 3-2 List of Usage Interrupt Vectors

Device	Contents
RX64M RX71M	USBI0 Interrupt (Vector number: 189, Interrupt source number : 62, Software Configurable Interrupt B) USB D0FIFO0 Interrupt (Vector number: 34) / USB D1FIFO0 Interrupt (Vector number: 35) USBR0 Interrupt (Vector number:90)
	USBAR Interrupt (Vector number: 94) USB D0FIFO2 Interrupt (Vector number: 32) / USB D1FIFO2 Interrupt (Vector number: 33)
RX65N RX651 RX72M RX66N RX72N	USBI0 Interrupt (Vector number: 185, Interrupt source number : 62, Software Configurable Interrupt B) USB D0FIFO0 Interrupt (Vector number: 34) / USB D1FIFO0 Interrupt (Vector number: 35) USBR0 Interrupt (Vector number:90)
RX66T RX72T	USBI0 Interrupt (Vector number: 174) / USBR0 Interrupt (Vector number: 90) USB D0FIFO0 Interrupt (Vector number: 34) / USB D1FIFO0 Interrupt (Vector number: 35)
RX671	USBI0 Interrupt (Vector number: 185, Interrupt source number : 62, Software Configurable Interrupt B) USB D0FIFO0 Interrupt (Vector number: 34) / USB D1FIFO0 Interrupt (Vector number: 35) USBR0 Interrupt (Vector number:90)
	USBI1 Interrupt (Vector number: 182, Interrupt source number : 63, Software Configurable Interrupt B) USB D0FIFO1 Interrupt (Vector number: 36) / USB D1FIFO1 Interrupt (Vector number: 37)

3.5 Header Files

All API calls and their supporting interface definitions are located in r_usb_basic_if.h and r_usb_hcdc_if.h.

3.6 Integer Types

This project uses ANSI C99 “Exact width integer types” in order to make the code clearer and more portable. These types are defined in *stdint.h*.

3.7 Compile Setting

For compile settings, refer to chapter **8, Configuration** in this document and chapter "Configuration" in the document (Document number: R01AN2025) for *USB Basic Host and Peripheral Driver using Firmware Integration Technology Application Note.*.

3.8 ROM / RAM Size

The follows show ROM/RAM size of this driver.

1. CC-RX (Optimization Level: Default)

(1). Non-OS

	Checks arguments	Does not check arguments
ROM size	37.8K bytes (Note 3)	37.3K bytes (Note 4)
RAM size	13.2K bytes	13.2K bytes

(2). RTOS

a. FreeRTOS

	Checks arguments	Does not check arguments
ROM size	48.5K bytes (Note 3)	48.0K bytes (Note 4)
RAM size	34.7K bytes	34.7K bytes

b. RI600V4

	Checks arguments	Does not check arguments
ROM size	50.5K bytes (Note 3)	50.0K bytes (Note 4)
RAM size	16.8K bytes	16.8K bytes

c. Azure RTOS

ROM size	67.5K bytes
RAM size	20.6K bytes

2. GCC (Optimization Level: -O2)

a. Non-OS

	Checks arguments	Does not check arguments
ROM size	44.2K bytes (Note 3)	43.6K bytes (Note 4)
RAM size	13.1K bytes	13.1K bytes

b. Azure RTOS

ROM size	78.0K bytes
RAM size	15.2K bytes

3. IAR (Optimization Level: Medium)

a. Non-OS

	Checks arguments	Does not check arguments
ROM size	38.0K bytes (Note 3)	37.4K bytes (Note 4)
RAM size	12.0K bytes	12.0K bytes

b. Azure RTOS

ROM size	46.5K bytes
RAM size	19.0K bytes

Note:

1. ROM/RAM size for BSP and USB Basic Driver is included in the above size.
2. The above is the size when specifying RX V2 core option.
3. The ROM size of “Checks arguments” is the value when *USB_CFG_ENABLE* is specified to *USB_CFG_PARAM_CHECKING* definition in *r_usb_basic_config.h* file.
4. The ROM size of “Does not check arguments” is the value when *USB_CFG_DISABLE* is specified to *USB_CFG_PARAM_CHECKING* definition in *r_usb_basic_config.h* file.
5. The result of RTOS includes the ROM/RAM size of the real-time OS.
6. The result of Azure RTOS includes the ROM/RAM size of Azure RTOS and USBX.

3.9 Argument

For the structure used in the argument of API function, refer to chapter "**Structures**" in the document (Document number: R01AN2025) for *USB Basic Host and Peripheral Driver using Firmware Integration Technology Application Note*.

3.10 “for”, “while” and “do while” statements

In FIT module, when using “for”, “while” and “do while” statements (loop processing) in register reflection waiting processing, etc., write comments with “WAIT_LOOP” as a keyword for these loop processing. Also, write in the FIT documentation that “WAIT_LOOP” is written as a comment in these loop processes.

3.11 Adding the FIT Module to Your Project

This module must be added to each project in which it is used. Renesas recommends the method using the Smart Configurator described in (1) or (3) or below. However, the Smart Configurator only supports some RX devices. Please use the methods of (2) or (4) for RX devices that are not supported by the Smart Configurator.

(1) Adding the FIT module to your project using the Smart Configurator in e^2 studio

By using the Smart Configurator in e^2 studio, the FIT module is automatically added to your project. Refer to “Renesas e^2 studio Smart Configurator User Guide (R20AN0451)” for details.

(2) Adding the FIT module to your project using the FIT Configurator in e^2 studio

By using the FIT Configurator in e^2 studio, the FIT module is automatically added to your project. Refer to “Adding Firmware Integration Technology Modules to Projects (R01AN1723)” for details.

(3) Adding the FIT module to your project using the Smart Configurator in CS+

By using the Smart Configurator Standalone version in CS+, the FIT module is automatically added to your project. Refer to “Renesas e^2 studio Smart Configurator User Guide (R20AN0451)” for details.

(4) Adding the FIT module to your project on CS+

In CS+, please manually add the FIT module to your project. Refer to “Adding Firmware Integration Technology Modules to CS+ Projects (R01AN1826)” for details.

4. Target Peripheral List（TPL）

For the structure used in the argument of API function, refer to chapter " **How to Set the Target Peripheral List (TPL)**" in the document (Document number: R01AN2025) for *USB Basic Host and Peripheral Driver using Firmware Integration Technology Application Note*.

5. Communication Device Class (CDC), PSTN and ACM

This software conforms to the Abstract Control Model (ACM) subclass of the Communication Device Class specification, as specified in detail in the PSTN Subclass document listed in “Related Documents”.

The Abstract Control Model subclass is a technology that bridges the gap between USB devices and earlier modems (employing RS-232C connections), enabling use of application programs designed for older modems.

5.1 Basic Functions

The main functions of HCDC are as follows.

1. Verify connected devices
2. Make communication line settings
3. Acquire the communication line state
4. Transfer data to and from the CDC peripheral device

5.2 Abstract Control Model Class Requests - Host to Device

This driver supports the following class requests.

For the class request processing, refer to chapter "**USB Class Requests**" in the document (Document number: R01AN2025) for *USB Basic Host and Peripheral Driver using Firmware Integration Technology Application Note.*

Table 5-1 CDC Class Requests

Request	Code	Description
SendEncapsulatedCommand	0x00	Transmits an AT command as defined by the protocol used by the device (normally 0 for USB).
GetEncapsulatedResponse	0x01	Requests a response to a command transmitted by SendEncapsulatedCommand.
SetCommFeature	0x02	Enables or disables features such as device-specific 2-byte code and country setting.
GetCommFeature	0x03	Acquires the enabled/disabled state of features such as device-specific 2-byte code and country setting.
ClearCommFeature	0x04	Restores the default enabled/disabled settings of features such as device-specific 2-byte code and country setting.
SetLineCoding	0x20	Makes communication line settings (communication speed, data length, parity bit, and stop bit length).
GetLineCoding	0x21	Acquires the communication line setting state.
SetControlLineState	0x22	Makes communication line control signal (RTS, DTR) settings.
SendBreak	0x23	Transmits a break signal.

For details concerning the Abstract Control Model requests, refer to Table 11, “Requests - Abstract Control Model” in “USB Communications Class Subclass Specification for PSTN Devices”, Revision 1.2.

The following describes the class request data formats supported by this class driver software.

5.2.1 SendEncapsulatedCommand

The SendEncapsulatedCommand data format is shown in Table 5-2.

Table 5-2 SendEncapsulatedCommand Data Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0x21	SEND_ENCAPSULATED _COMMAND(0x00)	0x0000	0x0000	Data length	Control protocol command

Note: Items such as AT commands for modem control are set as Data, and wLength is set to match the length of the data.

5.2.2 GetEncapsulatedResponse

The GetEncapsulatedResponse data format is shown Table 5-3.

Table 5-3 GetEncapsulatedResponse Data Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0x21	GET_ENCAPSULATED_ RESPONSE (0x01)	0x0000	0x0000	Data length	The data depends on the protocol.

Note: The response data to SendEncapsulatedCommand is set as Data, and wLength is set to match the length of the data.

5.2.3 SetCommFeature

The SetCommFeature data format is shown Table 5-4.

Table 5-4 SetCommFeature Data Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0x21	SET_COMM_FEATURE (0x02)	Feature Selector Note	0x0000	Data length	Status Either the country code or the Abstract Control Model idle setting/multiplexing setting for Feature Selector.

Note: Shown in Table 4.6 Feature selector Settings.

5.2.4 GetCommFeature Data Format

The GetCommFeature data format is shown below.

Table 5-5 GetCommFeature Data Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0x21	GET_COMM_FEATURE (0x03)	Feature Selector Note	0x0000	Data length	Status Either the country code or the Abstract Control Model idle setting/multiplexing setting for Feature Selector.

Note: Shown in Table 4.6 Feature selector Settings.

A Feature selector setup is shown in Table 5-6. The Status format at the time of ABSTRACT_STATE is shown in Table 5-7.

Table 5-6 Feature Selector Settings

Feature Selector	Code	Targets	Length of Data	Description
RESERVED	0x00	None	None	Reserved
ABSTRACT_STATE	0x01	Interface	2	Selects the setting for Abstract Control Model idle state and signal multiplexing.
COUNTRY_SETTING	0x02	Interface	2	Selects the country code in hexadecimal format, as defined by ISO 3166.

Table 5-7 Status Format when ABSTRACT_STATE Selected

Bit Position	Description
D15 to D2	Reserved
D1	Data multiplexing setting 1: Multiplexing of call management commands is enabled for the Data class. 0: Multiplexing is disabled.
D0	Idle setting 1: No endpoints of the target interface accept data from the host, and data is not supplied to the host. 0: Endpoints continue to accept data and it is supplied to the host.

5.2.5 ClearCommFeature

The ClearCommFeature data format is shown Table 5-8.

Table 5-8 ClearCommFeature Data Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0x21	CLEAR_COMM_FEATURE (0x04)	Feature Selector Note	0x0000	0x0000	None

Note: Shown in Table 4.6 Feature selector Settings.

5.2.6 SetLineCoding

The SetLineCoding data format is shown Table 5-9.

Table 5-9 SetLineCoding Data Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0x21	SET_LINE_CODING (0x20)	0x0000	0x0000	0x0000	Line Coding Structure See Table 5-10, Line Coding Structure Format

Line Coding Structure Format is shown Table 5-10.

Table 5-10 Line Coding Structure Format

Offset	Field	Size	Value	Description
0	dwDTERate	4	Number	Data terminal speed (bps)
4	bCharFormat	1	Number	Stop bits 0 - 1 stop bit 1 - 1.5 stop bits 2 - 2 stop bits
5	bParityType	1	Number	Parity 0 - None 1 - Odd 2 - Even 3 - Mask 4 - Space
6	bDataBits	1	Number	Data bits (5, 6, 7, 8)

5.2.7 GetLineCoding

The GetLineCoding data format is shown Table 5-11.

Table 5-11 GetLineCoding Data Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0xA1	GET_LINE_CODING (0x21)	0x0000	0x0000	0x0007	Line Coding Structure See Table 5-10 , Line Coding Structure Format

5.2.8 SetControlLineState

The SetControlLineState data format is shown below.

Table 5-12 SetControlLineState Data Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0x21	SET_CONTROL_LINE _STATE (0x22)	Control Signal Bitmap See Table 5-13, Control Signal Bitmap Format	0x0000	0x0000	None

Table 5-13 Control Signal Bitmap

Bit Position	Description
D15 to D2	Reserved
D1	DCE transmit function control 0 - RTS OFF 1 - RTS ON
D0	Notification of DTE ready state 0 - DTR OFF 1 - DTR ON

5.2.9 SendBreak

The SendBreak data format is shown below.

Table 5-14 SendBreak Data Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0x21	SEND_BREAK (0x23)	Break signal output duration	0x0000	0x0000	None

5.3 ACM Notifications from Device to Host

The class notifications supported and not supported by the software are shown Table 5-15.

Table 5-15 CDC Class Notifications

Notification	Code	Description	Supported
NETWORK_CONNECTION	0x00	Notification of network connection state	No
RESPONSE_AVAILABLE	0x01	Response to GET_ENCAPSLATED_RESPONSE	Yes
SERIAL_STATE	0x20	Notification of serial line state	Yes

5.3.1 SerialState

The SerialState data format is shown below.

Table 5-16 SerialState Data Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0xA1	SERIAL_STATE (0x20)	0x0000	0x0000	0x0000	UART State bitmap See Table 5-17, UART State bitmap Format

UART State bitmap format is shown Table 5-17.

Table 5-17 UART State bitmap Format

Bits	Field	Description
D15 to D7		Reserved
D6	bOverRun	Overrun error detected
D5	bParity	Parity error detected
D4	bFraming	Framing error detected
D3	bRingSignal	INCOMING signal (ring signal) detected
D2	bBreak	Break signal detected
D1	bTxCarrier	Data Set Ready: Line connected and ready for communication
D0	bRxCarrier	Data Carrier Detect: Carrier detected on line

5.3.2 ResponseAvailable

The ResponseAvailable data format is shown below.

Table 5-18 ResponseAvailable Data Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0xA1	RESPONSE_AVAILABLE (0x01)	0x0000	0x0000	0x0000	None

6. USB Host Communication Device Class Driver (HCDC)

6.1 Basic Functions

This software conforms to the Abstract Control Model subclass of the communication device class specification.

The main functions of HCDC are to:

1. Send class requests to the CDC peripheral
2. Transfer data to and from the CDC peripheral
3. Receive communication error information from the CDC peripheral

6.2 Structure / Union

The following structure or union is defined in *r_usb_hcdc_if.h*.

6.2.1 HCDC Request Structure

Table 6-1 describes the “UART settings” parameter structure used for the CDC requests *SetLineCoding* and *GetLineCoding*.

Table 6-1 usb_hcdc_linecoding_t Structure

Type	Member	Description	Remarks
uint32_t	dwdte_rate	Line speed	Unit: bps
uint8_t	bchar_ormat	Stop bits setting	
uint8_t	bparity_type	Parity setting	
uint8_t	bdata_bits	Data bit length	

Table 6-2 describes the “UART settings” parameter structure used for the CDC requests *SetControlLineState*.

Table 6-2 usb_hcdc_controllinestate_t Structure

Type	Member	Description	Remarks
uint16_t (D1)	brts:1	Carrier control for half duplex modems 0 - Deactivate carrier, 1 - Activate carrier	
uint16_t (D0)	bdtr:1	Indicates to DCE if DTE is present or not 0 - Not Present, 1 - Present	

Table 6-3 describes the “AT command” parameter structure used for the CDC requests *SendEncapsulatedCommand* and *GetEncapsulatedResponse*.

Table 6-3 usb_hcdc_encapsulated_t Structure

Type	Member	Description	Remarks
uint8_t	*p_data	Area where AT command data is stored	
uint16_t	wlength	Size of AT command data	Unit: byte

Table 6-4 describes the “Break signal” parameter structure used for the CDC requests *SendBreak*.

Table 6-4 usb_hcdc_breakduation_t Structure

Type	Member	Description	Remarks
uint16_t	wtime_ms	Duration of Break	Unit: ms

6.2.2 CommFeature Function Selection Union

Table 6-5 and Table 6-6 describe the “Feature Selector” parameter structure used for the CDC requests *SetCommFeature* and *GetCommFeature*, and Table 6-7 describes the parameter union.

Table 6-5 usb_hcdc_abstractstate_t Structure

Type	Member	Description	Remarks
uint16_t	rsv1:14	Reserved	
uint16_t	bdms:1	Data Multiplexed State	
iomt16_t	bis:1	Idle Setting	

Table 6-6 usb_hcdc_countrysetting_t Structure

Type	Member	Description	Remarks
uint16_t	country_code	Country code in hexadecimal format as defined in [ISO3166],	

Table 6-7 usb_hcdc_commfeature_t Union

Type	Member	Description	Remarks
usb_hcdc_abstractstate_t	abstract_state	Parameter when selecting Abstract Control Model	
usb_hcdc_countrysetting_t	country_setting	Parameter when selecting Country Setting	

6.2.3 CDC Notification Format

Table 6-8 and Table 6-9 describe the data format of the CDC notification.

Table 6-8 Response_Available notification format

Type	Member	Description	Remarks
uint8_t	bmRequestType	0xA1	
uint8_t	bRequest	RESPONSE_AVAILABLE(0x01)	
uint16_t	wValue	0x0000	
uint16_t	wIndex	Interface	
uint16_t	wLength	0x0000	
uint8_t	Data	none	

Table 6-9 Serial_State notification format

Type	Member	Description	Remarks
uint8_t	bmRequestType	0xA1	
uint8_t	bRequest	SERIAL_STATE(0x20)	
uint16_t	wValue	0x0000	
uint16_t	wIndex	Interface	
uint16_t	wLength	0x0002	
uint16_t	Data	UART State bitmap	Refer to Table 6-10

The host is notified of the "*SerialState*" when a change in the UART port state is detected. Table 6-10 describes the structure of the UART State bitmap.

Table 6-10 usb_hcdc_serialstate_t Structure

Type	Member	Description	Remarks
uint16_t (D15-D7)	rsv1:9	Reserved	
uint16_t (D6)	bover_run:1	Overrun error detected	
uint16_t (D5)	bparity:1	Parity error detected	
uint16_t (D4)	bframing:1	Framing error detected	
uint16_t (D3)	bring_signal:1	Incoming signal (Ring signal) detected	
uint16_t (D2)	bbreak:1	Break signal detected	
uint16_t (D1)	btx_carrier:1	Line connected and ready for communication	Data Set Ready
uint16_t (D0)	brx_carrier:1	Carrier detected on line	Data Carrier Detect

7. API Functions

For API used in the application program, refer to chapter "**API Functions**" in the document (Document number: R01AN2025) for *USB Basic Host and Peripheral Driver using Firmware Integration Technology Application Note*.

8. Configuration (r_usb_hcdc_config.h)

Please set the following according to your system.

Note:

Be sure to set *r_usb_basic_config.h* file as well. For *r_usb_basic_config.h* file, refer to chapter "**Configuration**" in the document (Document number: R01AN2025) for *USB Basic Host and Peripheral Driver using Firmware Integration Technology Application Note*.

1. Setting connection of multiple CDC devices

To simultaneously connect multiple CDC devices and perform USB communication, set *USB_CFG_ENABLE* as the definition below. If multiple CDC devices are not connected simultaneously, then set *USB_CFG_DISABLE*.

```
#define   USB_CFG_HCDC_MULTI   USB_CFG_ENABLE    // Multiple connection supported
#define   USB_CFG_HCDC_MULTI   USB_CFG_DISABLE   // Multiple connection not supported
```

2. Setting CDC class

Specify the device class ID of the CDC device to be connected.

```
#define   USB_CFG_HCDC_IFCLS   USB_CFG_CDC       // CDC class supported device
#define   USB_CFG_HCDC_IFCLS   USB_CFG_VEN       // Vendor class device
```

Note:

With regard to the USB serial conversion device in the marketplace, the device class ID may be the Vendor class. Check the CDC device specifications before use. If the device class is Vendor class, then set *USB_CFG_VEN*.

3. Setting pipe to be used

Set the pipe number to use for data transfer.

(1). Bulk IN/OUT transfer

Set the pipe number (PIPE1 to PIPE5) to use for Bulk IN/OUT transfer. Do not set the same pipe number.

```
#define   USB_CFG_HCDC_BULK_IN     Pipe number (USB_PIPE1 to USB_PIPE5)
#define   USB_CFG_HCDC_BULK_OUT    Pipe number (USB_PIPE1 to USB_PIPE5)
#define   USB_CFG_HCDC_BULK_IN2    Pipe number (USB_PIPE1 to USB_PIPE5)
#define   USB_CFG_HCDC_BULK_OUT2   Pipe number (USB_PIPE1 to USB_PIPE5)
```

(2). Interrupt IN transfer

Set the pipe number (PIPE6 to PIPE9) to use for Interrupt IN transfer. Do not set the same pipe number. If the USB Hub is being used, then PIPE9 cannot be set as the following definitions.

```
#define   USB_CFG_HCDC_INT_IN      Pipe number (USB_PIPE6 to USB_PIPE9)
#define   USB_CFG_HCDC_INT_IN2     Pipe number (USB_PIPE6 to USB_PIPE9)
```

Note:

a. Only if *USB_CFG_ENABLE* is set for the definition of *USB_CFG_HCDC_MULTI* in 1 above, set the pipe number for the definitions of *USB_CFG_HCDC_BULK_IN2*, *USB_CFG_HCDC_BULK_OUT2*, and *USB_CFG_HCDC_INT_IN2*.

b. If USB_CFG_DISABLE is set for the definition of *USB_CFG_HCDC_MULTI*, set *USB_NULL* for the definitions of *USB_CFG_HCDC_BULK_IN2*, *USB_CFG_HCDC_BULK_OUT2*, and *USB_CFG_HCDC_INT_IN2*.

9. Creating an Application

Refer to the chapter "**Creating an Application Program**" in the document (Document number: R01AN2025) for *USB Basic Host and Peripheral Driver using Firmware Integration Technology Application Note*.

Website and Support

Renesas Electronics Website

http://www.renesas.com/

Inquiries

http://www.renesas.com/inquiry

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Apr 1, 2014	—	First edition issued
1.10	Dec 26, 2014	—	1. RX71M is added as new device. 2. The multiple connecting of CDC device is supported. 3. The macro definition to set the multi device connecting is added in r_usb_hcdc_config.h.
1.11	Sep 30, 2015	—	RX63N and RX631 are added in Target Device.
1.20	Sep 30, 2016	—	1. RX65N and RX651 are added in Target Device. 2. Supporting DMA transfer. 3. Supporting USB Host and Peripheral Interface Driver application note(Document No.R01AN3293EJ)
1.21	Mar 31, 2017	—	1. Supported Technical Update (Document number. TN-RX*-A172A/E) 2. The chapter *API Functions* is moved to the document (Document number: R01AN2025) of *USB Basic Host and Peripheral Driver Firmware Integration Technology.*
1.22	Sep 30, 2017	—	Supporting RX65N/RX651-2M
1.23	Mar 31, 2018	—	Supporting the Smart Configurator.
1.24	Dec 28, 2018	—	Supporting RTOS.
1.25	Apr 16, 2019	—	Added RX66T/RX72T in Target Device.
1.26	May 31, 2019	—	1. Support GCC compiler and IAR compiler. 2. Remove RX63N from Target Device.
1.27	Jul 31, 2019	—	RX72M is added in Target Device.
1.30	Mar 1, 2020	—	1. Supported the real time OS (uITRON:RI600V4). 2. Added RX72N/RX66N in Target Device.
1.31	Mar 1, 2021	—	Added RX671 in Target Device.
1.31	Sep 29, 2023	—	Support Azure RTOS (USBX HCDC).
1.44	Mar 01, 2025	—	Change Disclaimer.

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on
   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins
   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.