R01AN2170EJ0120
Rev.1.20
Jun 1, 2020

RX Family

USB Peripheral Communications Device Class Driver for USB Mini Firmware Using Firmware Integration Technology

Introduction

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

Target Device

RX111 Group
RX113 Group
RX231 Group
RX23W 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. Universal Serial Bus Revision 2.0 specification
2. USB Class Definitions for Communications Devices Revision 1.2
3. USB Communications Class Subclass Specification for PSTN Devices Revision 1.2
   http://www.usb.org/developers/docs/
4. RX111 Group User's Manual: Hardware (Document number .R01UH0365)
5. RX113 Group User's Manual: Hardware (Document number.R01UH0448)
6. RX231 Group User's Manual: Hardware (Document number .R01UH0496)
7. RX23W Group User's Manual: Hardware (Document number .R01UH0823)
8. USB Basic Mini Host and Peripheral Driver (USB Mini Firmware) using Firmware Integration Technology Application Note (Document number.R01AN2166)

Renesas Electronics Website
http://www.renesas.com

USB Devices Page
http://www.renesas.com/prod/usb/

Contents

1. Overview

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

This module supports the following functions.

- Data transfer to and from a USB host
- Response to CDC class requests
- Communication device class notification transmit

1.1 Please be sure to read

Please refer to the document (Document number: R01AN2166) for *USB Basic Mini Host and Peripheral Driver (USB Mini Firmware) 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

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.

1.3 Terms and Abbreviations

Terms and abbreviations used in this document are listed below.

ACM	:	Abstract Control Model. This is the USB interface subclass used for virtual COM ports, based in the old V.250 (AT) command standard. See PSTN below.
APL	:	Application program
CDC	:	Communications Devices Class
H/W	:	Renesas USB device
IDE	:	Integrated Development Environment
PCD	:	Peripheral control driver of USB-BASIC-F/W
PCDC	:	Communications Devices Class for peripheral
PCDCD	:	Peripheral Communications Devices Class Driver
PSTN	:	Public Switched Telephone Network, contains the ACM (above) standard.
RTOS	:	USB Driver for the real-time OS
SCI	:	Serial Communication Interface
USB	:	Universal Serial Bus
USB-BASIC-FW	:	USB Basic Mini Host and Peripheral Driver

1.4 USB PCDC FIT Module

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

2. Software Configuration

Figure 2-1 shows the configuration of the modules related to PCDC.

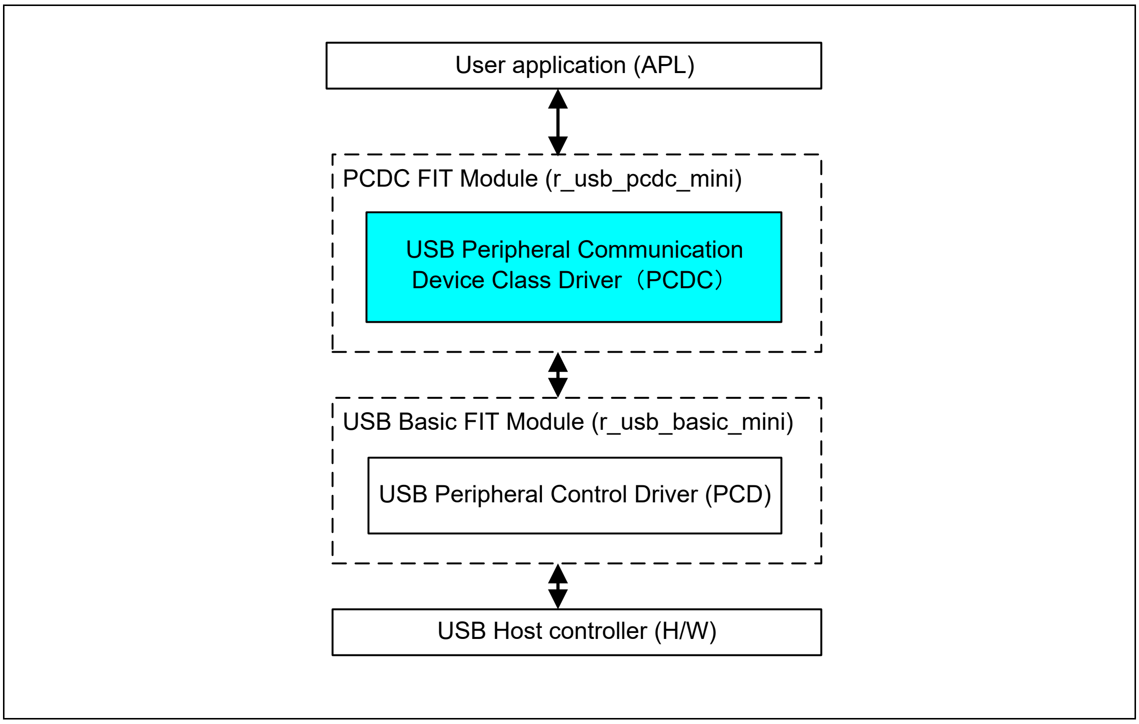

Figure 2-1 Source Module Structure

Table 2-1 Modules

Module	Description
APL	User application program.
PCDC	Sends requests from the APL for requests and data communication involving the CDC to the PCD.
PCD	USB peripheral hardware control driver. (Basic USB FW.)

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_mini

3.3 Operating Confirmation Environment

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

Table 3-1 Operation Confirmation Environment

Item	Contents
C compiler	Renesas Electronics C/C++ compiler for RX Family V.3.02.00 (The option "-lang=C99" is added to the default setting of IDE)
	GCC for Renesas RX 8.3.0.201904 (The option "-std=gnu99" is added to the default setting of IDE)
	IAR C/C++ Compiler for Renesas RX version 4.14.1
Real-Time OS	FreeRTOS V.10.0.0 RI600V4 V.1.06
Endian	Little Endian, Big Endian
USB Driver Revision Number	Rev.1.20
Using Board	Renesas Starter Kit for RX111 Renesas Starter Kit for RX113 Renesas Starter Kit for RX231 Renesas Solution Starter Kit for RX231
Host Environment	The operation of this USB Driver module connected to the following OSes has been confirmed. 1. Windows® 8.1 2. Windows® 10

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
RX111 RX113 RX231 RX23W	USBI0 Interrupt (Vector number: 36) / USBR0 Interrupt (Vector number: 90) USB D0FIFO0 Interrupt (Vector number: 36) / USB D1FIFO0 Interrupt (Vector number: 37)

3.5 Header Files

All API calls and their supporting interface definitions are located in *r_usb_basic_mini_if.h* and *r_usb_pcdc_mini_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 **6, Configuration (r_usb_pcdc_mini_config.h)** in this document and chapter "Configuration" in the document (Document number: R01AN2166) for *USB Basic Mini Host and Peripheral Driver (USB Mini Firmware) using Firmware Integration Technology Application Note.*

3.8 ROM / RAM Size

The follows shows ROM/RAM size of this driver.

1. CC-RX (Optimization Level: Default)

(1). Non-OS

	Checks arguments	Does not check arguments
ROM size	17.6K bytes (Note 3)	17.3K bytes (Note 4)
RAM size	3.3K bytes	3.3K bytes

(2). RI600V4

	Checks arguments	Does not check arguments
ROM size	34.1K bytes (Note 3)	33.8K bytes (Note 4)
RAM size	4.4K bytes	4.4K bytes

(3). FreeRTOS

	Checks arguments	Does not check arguments
ROM size	30.1K bytes (Note 3)	29.8K bytes (Note 4)
RAM size	14.2K bytes	14.2K bytes

2. GCC (Optimization Level: -O2)

	Checks arguments	Does not check arguments
ROM size	19.5K bytes (Note 3)	19.2K bytes (Note 4)
RAM size	3.2K bytes	3.2K bytes

3. IAR (Optimization Level: Medium)

	Checks arguments	Does not check arguments
ROM size	11.7K bytes (Note 3)	11.5K bytes (Note 4)
RAM size	2.6K bytes	2.6K 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_mini_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_mini_config.h* file.

3.9 Argument

For the structure used in the argument of API function, refer to chapter "**Structures**" in the document (Document number: R01AN2166) for *USB Basic Mini Host and Peripheral Driver (USB Mini Firmware) using Firmware Integration Technology Application Note.*

3.10 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) 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 "Smart Configurator" on 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. CDC, PSTN, and ACM (Abstract Control Model)

4.1 Basic Functions

This software conforms to the Abstract Control of the CDC PSTN Subclass. See.4.2 below.

4.2 Abstract Control Model Overview

The Abstract Control Model subclass of CDC 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. The class requests and class notifications supported are listed below.

4.2.1 Class Requests (Host to Peripheral)

This driver notifies to the application program when receiving the following class request.

For the class request processing, refer to chapter "**USB Class Requests**" in the document (Document number: R01AN2166) for *USB Basic Mini Host and Peripheral Driver (USB Mini Firmware) using Firmware Integration Technology Application Note*.

Table 4-1 CDC class requests

Request	Code	Description
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.

4.2.2 Data Format of Class Requests

The data format of the class requests supported by the class driver software is described below.

1. SetLineCoding

This is the class request the host transmits to perform the UART line setting.

The SetLineCoding data format is shown below.

Table 4-2 SetLineCoding Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0x21	SET_LINE_CODING (0x20)	0x00	0x00	0x07	Line Coding Structure See Table 4-3, Line Coding Format

Table 4-3 Line Coding 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
6	BdataBits	1	Number	Data bits (5, 6, 7, 8)

The following shows the setting that this S/W supports.

DwDTERate: 1200bps/2400bps/4800bps/9600bps/14400bps/19200bps/38400bps/57600bps/115200bps
BcharFormat: 1 Stop bit/2 Stop bit
BparityType: None/Odd/Even
BdataBits: 7bit/8bit

2. GetLineCoding

This is the class request the host transmits to request the UART line state.

The GetLineCoding data format is shown below.

Table 4-4 SetLineCoding Format

bmRequestType	bRequest	wValue	WIndex	wLength	Data
0xA1	GET_LINE_CODING (0x21)	0x00	0x00	0x07	Line Coding Structure See Table 4-2, SetLineCoding Format

3. SetControlLineState

This is a class request that the host sends to set up the signal for flow controls of UART.

This software does not support RTS/DTR control.

The SET_CONTROL_LINE_STATE data format is shown below.

Table 4-5 SET_CONTROL_LINE_STATE Format

bmRequestType	bRequest	WValue	wIndex	wLength	Data
0x21	SET_CONTROL_ LINE_STATE (0x22)	Control Signal Bitmap See Table 4-6, Control Signal Bitmap	0x00	0x00	None

Table 4-6 Control Signal Bitmap

Bit Position	Description	
D15 to D2	Reserved (reset to 0)	
D1	DCE transmit function control	0 - RTS Off 1 - RTS On
D0	Notification of DTE ready state	0 - DTR Off 1 - DTR On

4.2.3 Class Notifications (Peripheral to Host)

Whether or not a class notification is supported is shown in Table 4-7.

Table 4-7 CDC Class Notifications

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

1. Serial State

The host is notified of the serial state when a change in the UART port state is detected.

This software supports the detection of overrun, parity and framing errors. A state notification is performed when a change from normal state to error is detected. However, notification is not continually transmitted when an error is continually detected.

The SerialState data format is shown below.

Table 4-8 SerialState Format

bmRequestType	bRequest	wValue	wIndex	wLength	Data
0xA1	SERIAL_STATE (0x20)	0x00	0x00	0x02	UART State bitmap See Table 4-9, UART state bitmap format

Table 4-9 UART state bitmap format

Bits	Field	Description	Supported
D15～D7		Reserved	-
D6	b_over_run	Overrun error detected	Yes
D5	b_parity	Parity error detected	Yes
D4	b_framing	Framing error detected	Yes
D3	b_ring_signal	INCOMING signal (ring signal) detected	No
D2	b_break	Break signal detected	No
D1	btx_arrier	Data Set Ready: Line connected and ready for communication	No
D0	brx_carrier	Data Carrier Detect: Carrier detected on line	No

4.3 PC Virtual COM-port Usage

The CDC device can be used as a virtual COM port when operating in Windows OS.

Use a PC running Windows OS, and connect an RSK board. After USB enumeration, the CDC class requests *GetLineCoding* and *SetControlLineState* are executed by the target, and the CDC device is registered in Windows Device Manager as a virtual COM device.

Registering the CDC device as a virtual COM-port in Windows Device Manager enables data communication with the CDC device via a terminal app such as "HyperTerminal" which comes standard with Windows OS. When changing settings of the serial port in the Windows terminal application, the UART setting is propagated to the firmware via the class *request SetLineCoding*.

Data input (or file transmission) from the terminal app window is transmitted to the RSK board using endpoint 2 (EP2); data from the RSK board side is transmitted to the PC using EP1.

When the last packet of data received is the maximum packet size, and the terminal determines that there is continuous data, the received data may not be displayed in the terminal. If the received data is smaller than the maximum packet size, the data received up to that point is displayed in the terminal.

The received data is outputted on the terminal when the data less than Maximum packet size is received.

5. API Functions

For API used in the application program, refer to chapter "**API Functions**" in the document (Document number: R01AN2166) for *USB Basic Mini Host and Peripheral Driver (USB Mini Firmware) using Firmware Integration Technology Application Note.*

6. Configuration (r_usb_pcdc_mini_config.h)

Please set the following according to your system.

Note:

Be sure to set *r_usb_basic_mini_config.h* file as well. For *r_usb_basic_mini_config.h* file, refer to chapter "**Configuration**" in the document (Document number: R01AN2166) for *USB Basic Mini Host and Peripheral Driver (USB Mini Firmware) using Firmware Integration Technology Application Note.*

1. 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 for the definitions of *USB_CFG_PCDC_BULK_IN* and *USB_CFG_PCDC_BULK_OUT*.

```
#define    USB_CFG_PCDC_BULK_IN     Pipe number (USB_PIPE1 to USB_PIPE5)
#define    USB_CFG_PCDC_BULK_OUT    Pipe number (USB_PIPE1 to USB_PIPE5)
```

(2). Interrupt IN transfer

Set the pipe number (PIPE6 to PIPE9) to use for Interrupt IN transfer.

```
#define    USB_CFG_PCDC_INT_IN      Pipe number (USB_PIPE6 to USB_PIPE9)
```

7. CDC Driver Installation

If USB Host is PC(Windows®), a CDC driver must be installed in the PC. The wizard shown in Figure 7-1 will appear on your screen and prompt the CDC driver installation when you connect RSK.

(1). Select **Update Driver Software** from the device manager.

(2). Select “**Browse my computer for driver software**”.

Note:

(1). It is not necessary the following installation work for CDC driver when using Window® 10.

(2). The catalog file with the digital signature is required when using Windows® 8.1. The customer need to create this catalog file.

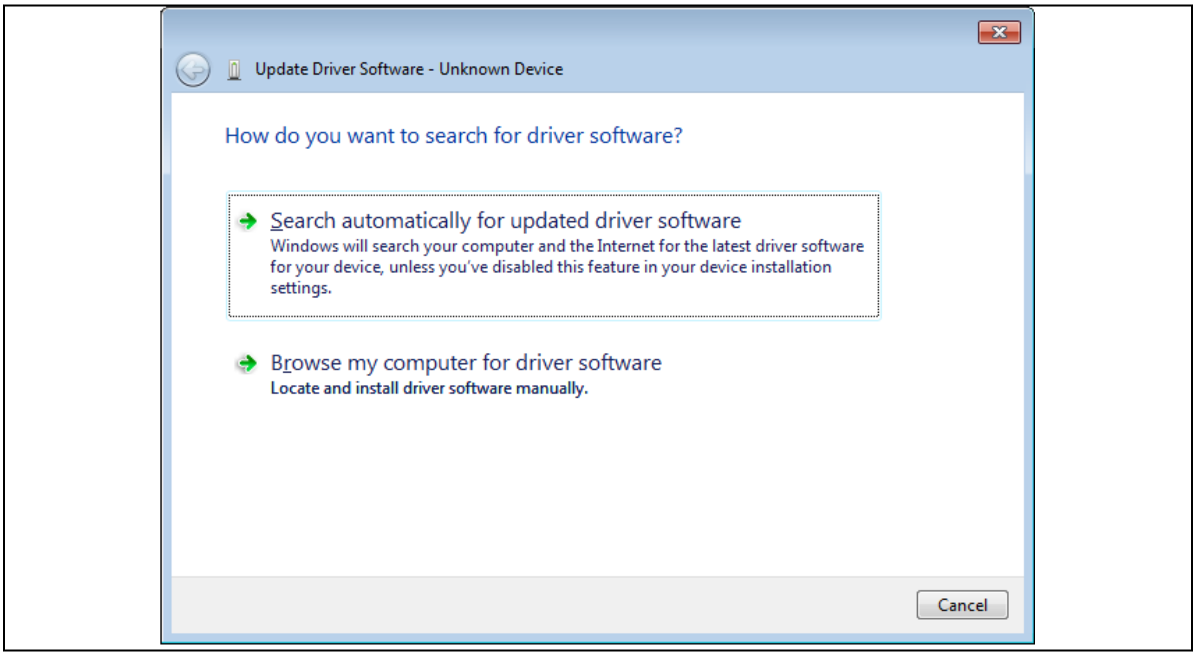

Figure 7-1 New Hardware Search Wizard

(3). Select **"Browse for driver software on your computer"**

Click **Browse**, specify the folder in which the *CDC_Demo.inf* is stored, then click **"Next"**

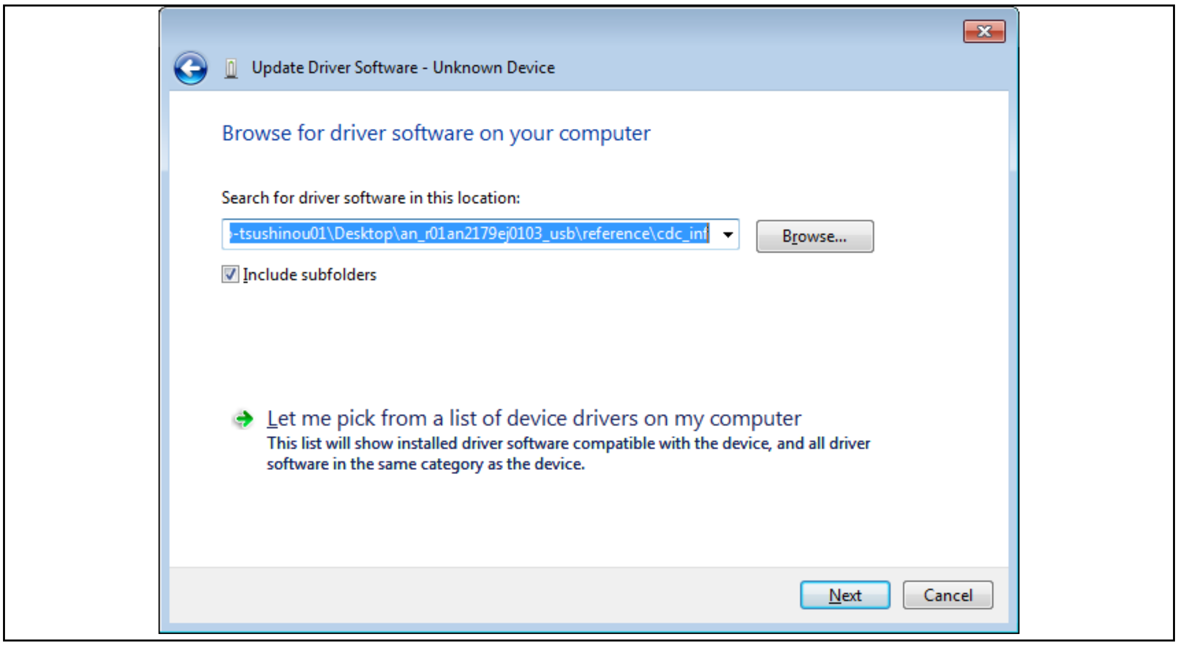

Figure 7-2 Select Driver Location

Note:

The *CDC_Demo.inf* file is stored in "*r_usb_pcdc_mini\utilities*" folder in the package.

(4). If the following installation confirmation screen appears, click **"Browse for driver software on your computer"**

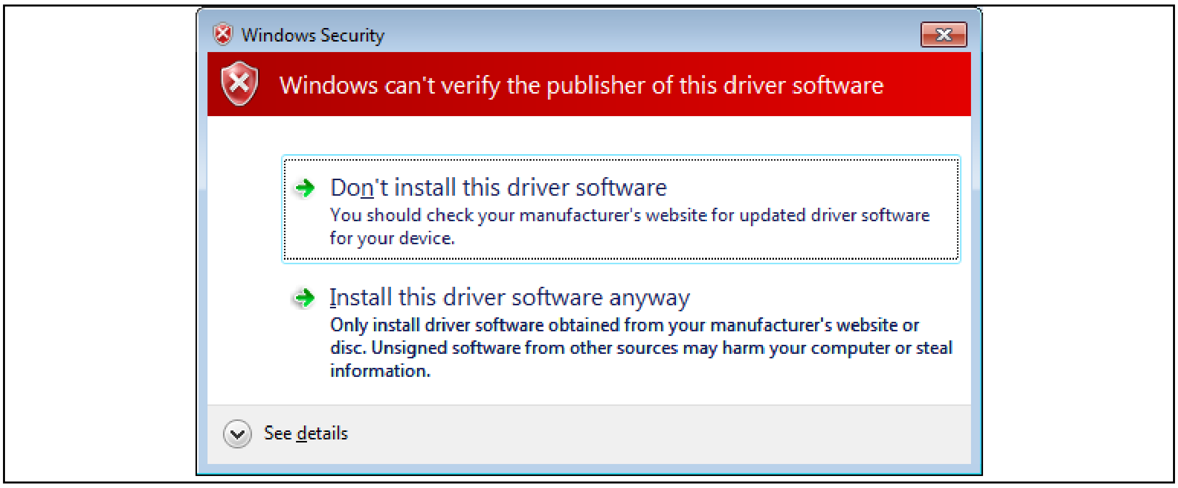

Figure 7-3 Installation Confirmation Screen

(5). When the following window appears, the CDC driver has been successfully installed. Click **"Close."**

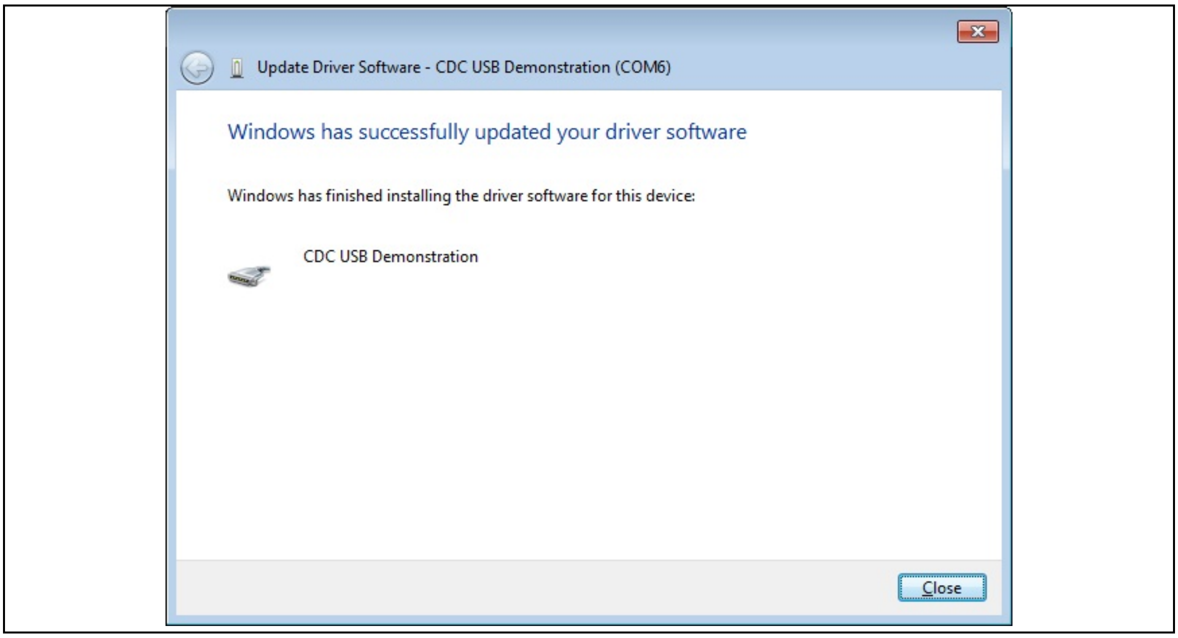

Figure 7-4 Installation Complete

8. Creating an Application

Refer to the chapter "**Creating an Application Program**" in the document (Document number: R01AN2166) for *USB Basic Mini Host and Peripheral Driver (USB Mini Firmware) 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	Dec 1, 2014	—	First edition issued
1.01	Jun 1, 2015	—	RX231 is added in the target device. The baud rate value which this driver supports is added.
1.02	Dec 28, 2015	—	Checked the operation with Windows® 10.
1.10	Nov 30, 2018	—	1. Supporting Smart Configurator. 2. The following chapters are added. (1). 5. API Functions (2). 6. Configuration (r_usb_pcdc_mini_config.h) (3). 7. CDC Driver Installation 3. The following chapters are changed. (1). 3. API Information (2). 8. Creating an Application 4. The following chapters are deleted. "How to Register Class Driver", "System Resources", "Task ID and Priority Setting", "USB Peripheral Communication Device Class Driver".
1.11	May 31, 2019	—	Support GCC compiler and IAR compiler.
1.12	Jun 30, 2019	—	RX23W is added in the target device.
1.20	Jun 1, 2020	—	Support the real time OS.

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. 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 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. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment 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 moment 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 moment 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 moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has 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. Moreover, 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.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the 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.

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics Corporation
TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

Renesas Electronics America Inc. Milpitas Campus
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics America Inc. San Jose Campus
6024 Silver Creek Valley Road, San Jose, CA 95138, USA
Tel: +1-408-284-8200, Fax: +1-408-284-2775

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 101-T01, Floor 1, Building 7, Yard No. 7, 8th Street, Shangdi, Haidian District, Beijing 100085, China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai 200333, China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, #06-02 Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit No 3A-1 Level 3A Tower 8 UOA Business Park, No 1 Jalan Pengaturcara U1/51A, Seksyen U1, 40150 Shah Alam, Selangor, Malaysia
Tel: +60-3-5022-1288, Fax: +60-3-5022-1290

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 9.0