RX Family

R01AN2294EJ0120
Rev.1.20
Jun 1, 2020

Sample Program using USB Host Human Interface Device Class Driver (HHID) for USB Mini Firmware to communicate via USB with HID device Firmware Integration Technology

Introduction

This document describes the following sample firmware: USB Host Human Interface Devices Class Driver using Firmware Integration Technology. The sample firmware is referred to below as the HHID.

When developing an actual software, be sure to use the “USB Basic Mini Host and Peripheral Driver (USB Mini Firmware) using Firmware Integration Technology Application Note” (Document number: R01AN2166) together with the user’s manual for each MCU (Hardware). In addition, also refer to the " USB Host Human Interface Device Class Driver for USB Mini Firmware using Firmware Integration Technology Application Note” (Document number: R01AN2168), if necessary. “USB Basic Mini Host and Peripheral Driver (USB Mini Firmwae) using Firmware Integration Technology Application Note” (Document number: R01AN2166) is located in the "reference_documents" folder within the package.

Target Device

RX111 Group
RX113 Group
RX231 Group
RX23W Group

The operation of this program has been confirmed using the Renesas Starter Kit (RSK) or the Renesas Solution Starter Kit (RSSK).

Contents

1. Introduction

1.1 Functions

The HHID performs communication with HID devices in conformance with the USB human interface devices class specification (HID).
It transfers HID class data when a mouse or keyboard is connected.

1.2 FIT Module Configuration

The HHID comprises the following FIT modules and a sample application:

Table 1-1 FIT Module Configuration

FIT Module	Folder Name
Board Support Package Module Firmware Integration Technology	r_bsp
RX Family USB Basic Mini Host and Peripheral Driver (USB Mini Firmware) using Firmware Integration Technology	r_usb_basic_mini
RX Family USB Host Human Interface Devices Class Driver for USB Mini Firmware using Firmware Integration Technology	r_usb_hhid_mini

Refer to the related documentation for details of each FIT module. Note that the latest versions of the FIT modules used by the sample firmware are available for download from the following website:

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

1.3 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.

1.4 Operating Confirmation Environment

The operating confirmation environment for the HHID is described below:

Table 1-2 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 4.08.04.201902 (The option "-std=gnu99" is added to the default setting of IDE)
	IAR C/C++ Compiler for Renesas RX version 4.12.01
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 RX23W

2. Software Configuration

2.1 Module Configuration

The HHID comprises an HID class driver as well as mouse and keyboard device drivers. When data is received from a connected HID device, the result is reported to the APL via the HCD. When the APL generates a data transfer request, it is reported to the HID device via the HCD.
Figure 2-1 shows the module configuration of the HHID, Table 2-1 lists the functions of the modules.

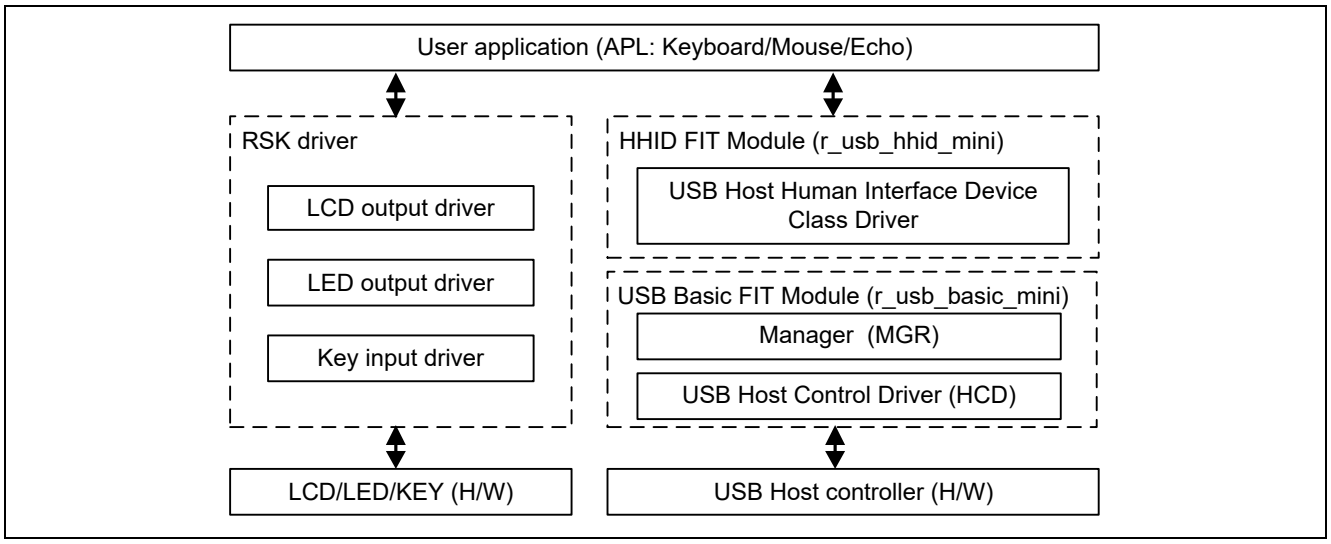


Figure 2-1 Moudle Configuration

Table 2-1 Functions of Modules

Module Name	Function
APL	Sample application program ・ Starts communication with the HID device and controls suspending and resuming by means of switch operations. ・ Displays on the LCD report information received from the HID device.
RSK driver	Sample application for using the peripheral functions of the RSK
HHID (r_usb_hhid_mini)	HID class driver ・ Interprets requests from the HID device. ・ Reports switch manipulation information from the APL to the HID device via the HCD.
HCD (r_usb_basic_mini)	USB Host Control Driver

3. Setup

3.1 Hardware

3.1.1 Example Operating Environment

Figure 3-1 shows an example operating environment for the HHID. Refer to the associated instruction manuals for details on setting up the evaluation board and using the emulator, etc.

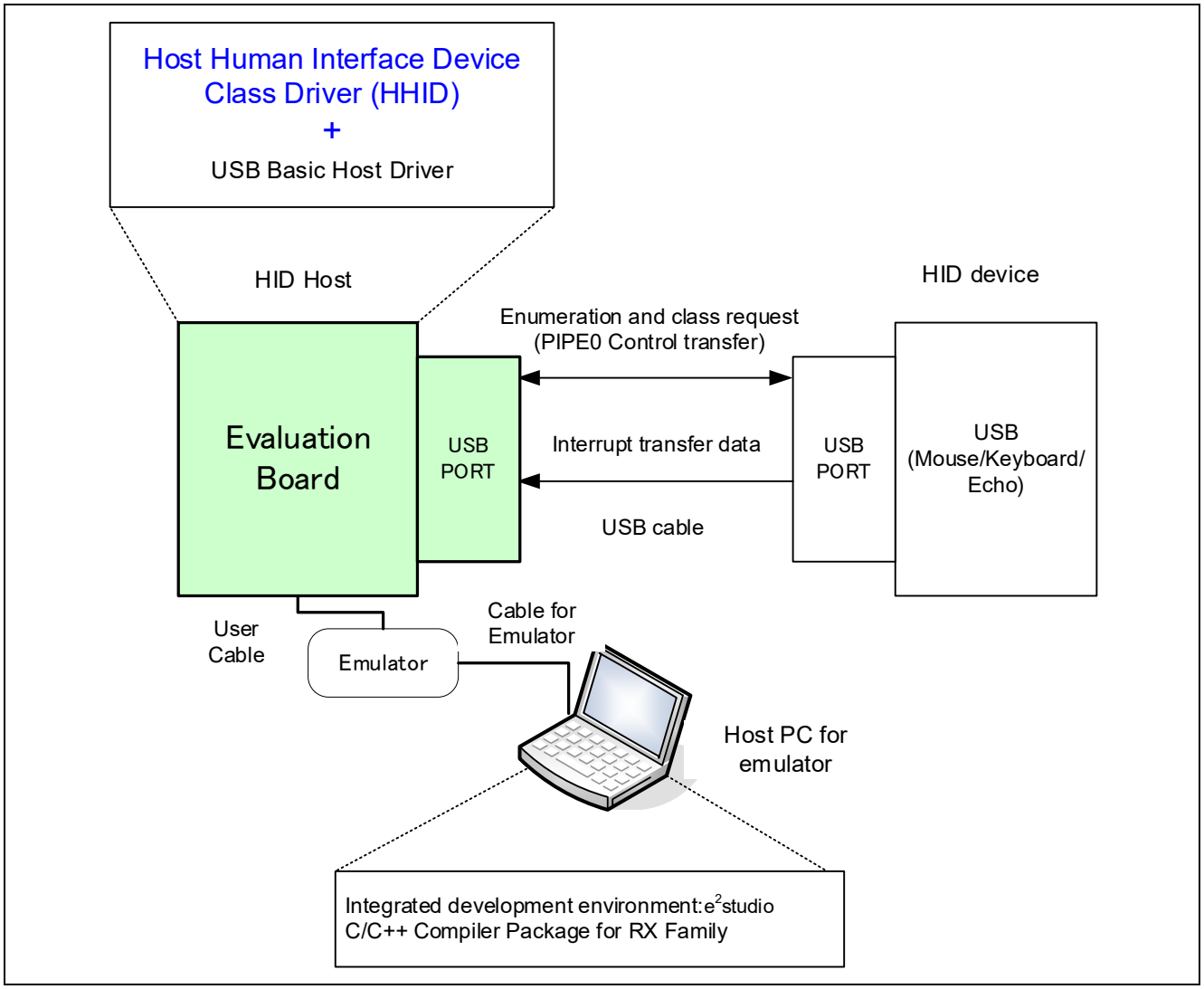


Figure 3-1 Example Operating Environment

Table 3-1 shows the evaluation board on which operation has been confirmed.

Table 3-1 Evaluation Board on which HHID Operation Has Been Verified

MCU	Evaluation Board
RX111	RSKRX111
RX113	RSKRX113
RX231	RSKRX231
RX23W	RSSKRX23W

3.1.2 RSK / RSSK Setting

It is necessary to set RSK/RSSK to operate in the host mode. Please refer to the following.

Table 3-2 **RSK / RSSK Setting**

RSK	Jumper Setting
RSKRX111	J12: Shorted Pin1-2
RSKRX113	J12: Shorted Pin1-2
RSKRX231	J15: Shorted Pin1-2
RSSKRX23W	J5: Shorted Pin2-3

Note:

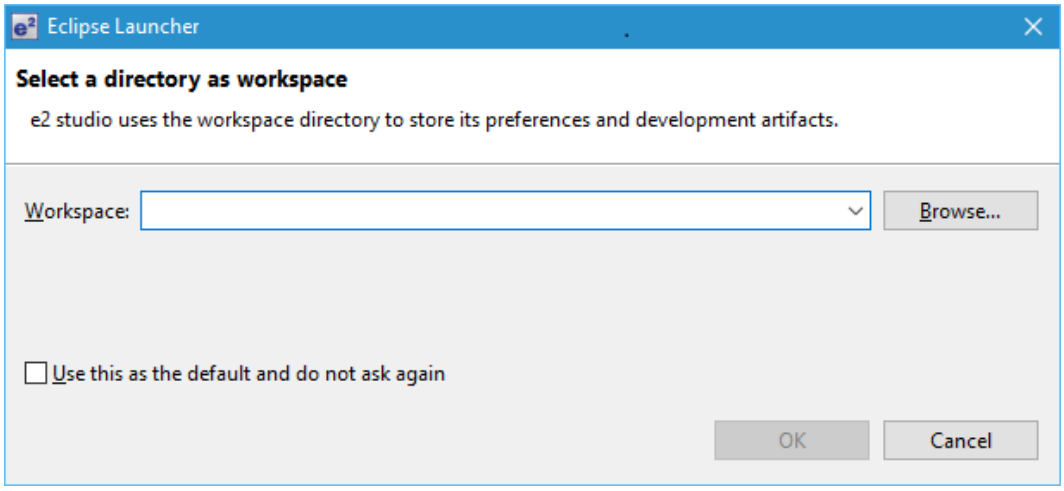
For the detail of RSK / RSSK setting, refer to the user's manual of RSK / RSSK.

3.2 Software

1) Setup e^2 studio

a) Start e^2 studio

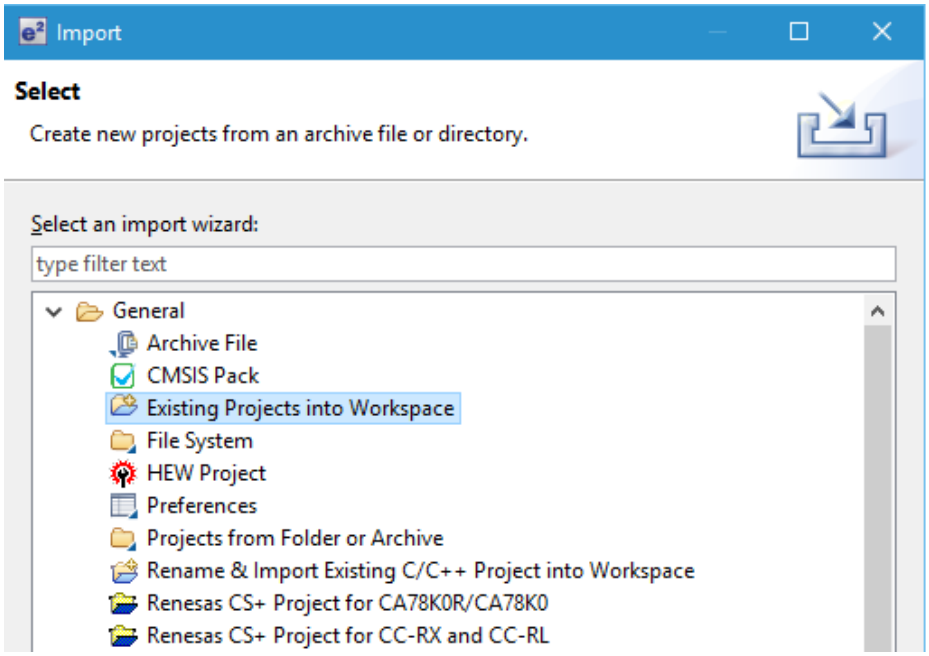
b) If you start up e^2 studio at first, the following dialog is displayed. Specify the folder to store the project in this dialog.



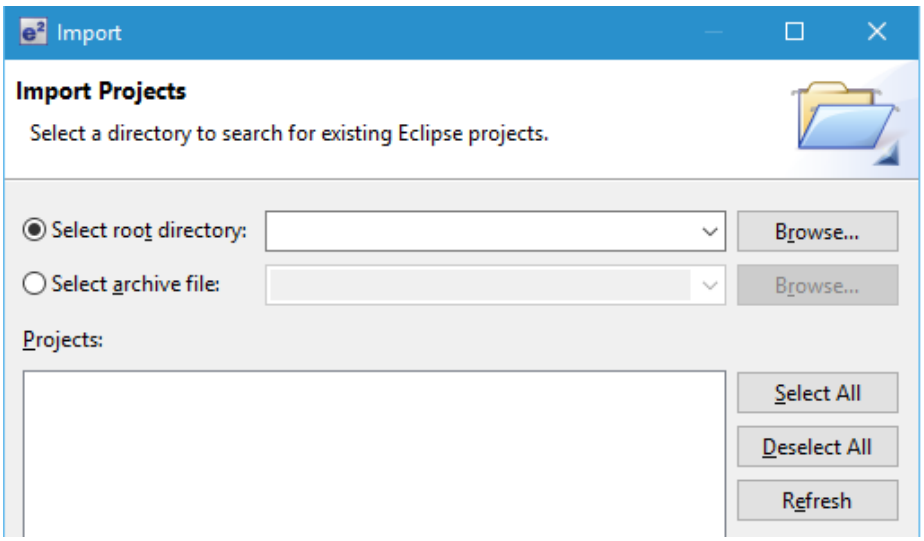
2) Import the project to the workspace

a) Select [File] > [Import]

b) Select [General] => [Existing Projects into Workspace]



c) Select the root directory of the project, that is, the folder containing the ".cproject" file.



d) Click "Finish".

You have now imported the project into the workspace. Note that you can import other projects into the same workspace.

3) Generate the binary target program by clicking the "Build" button.

4) Connect the target board to the debug tool and download the executable. The target is run by clicking the "Run" button.

4. Sample Application

4.1 Application Specifications

The following three application programs are provided:

4.1.1 Normal Mode Application (demo_src¥r_usb_hhid_apl.c)

Transfers data to and from an HID device (mouse or keyboard) connected to the RSK/RSSK. Data received from the HID device is read and discarded.

4.1.2 Demo Mode Application (demo_src¥r_usb_hhid_apl_demo.c)

Transfers data to and from an HID device (mouse or keyboard) connected to the RSK/RSSK. Data received from the HID device is displayed on an LCD. In addition, the processing is provided for sending of suspend and resume signals to the HID device.

4.1.3 Echo (Loopback) Mode Application (demo_src¥r_usb_hhid_apl_echo.c)

Performs echo(loopback) processing in which data received from an HID device connected to the RSK/RSSK is sent unmodified back to the HID device.

[Note]

Loopback processing is possible only when an HID device that supports interrupt out transfer is connected.

4.2 Application Processing (for Non-OS)

The application comprises two parts: initial settings and main loop. An overview of the processing in these two parts is provided below.

4.2.1 Initial Settings

Initial settings consist of MCU pin settings, USB driver settings, and initial settings to the USB controller.

4.2.2 Main Loop (Normal mode: demo_src¥r_usb_hhid_apl.c)

The main loop performs processing to receive data from the HID device as part of the main routine. An overview of the processing of the main loop is presented below.

1. When the *R_USB_GetEvent* function is called after an HID device attaches to the USB host (RSK/RSSK) and enumeration completes, *USB_STS_CONFIGURED* is set as the return value. When the APL confirms *USB_STS_CONFIGURED*, it calls the *R_USB_Write* function to request transmission of data to the HID device.
2. When the *R_USB_GetEvent* function is called after sending of class request *SET_PROTOCOL* to the HID device has completed, *USB_STS_REQUEST_COMPLETE* is set as the return value. When the APL confirms *USB_STS_REQUEST_COMPLETE*, it calls the *R_USB_Read* function to make a data receive request for data sent by the HID device.
3. When the *R_USB_GetEvent* function is called after reception of data from the HID device has completed, *USB_STS_READ_COMPLETE* is set as the return value. When the APL confirms *USB_STS_READ_COMPLETE*, it calls the *R_USB_Read* function to make a data receive request for data sent by the HID device.
4. The processing in step 3, above, is repeated.

An overview of the processing performed by the APL is shown below:

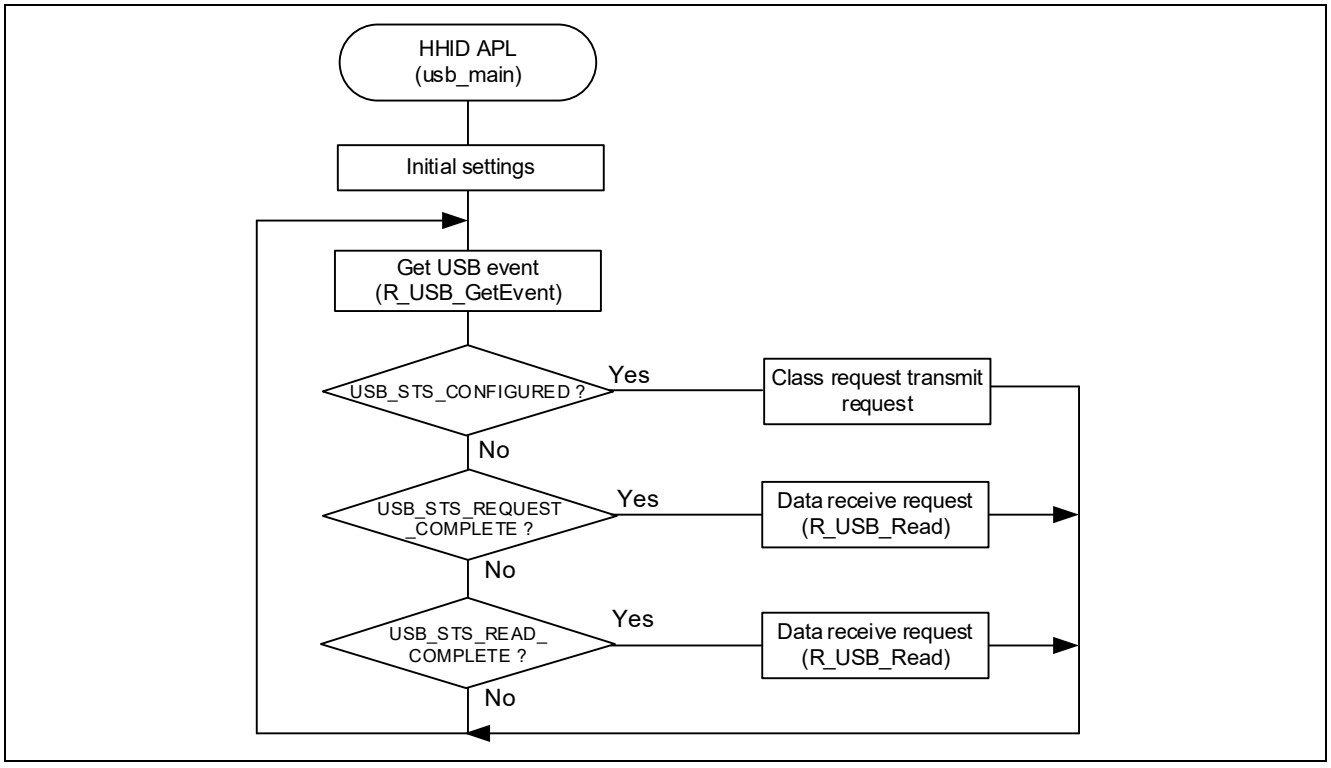


Figure 4-1 Main Loop (Normal mode)

4.2.3 Main Loop (Demo mode: demo_src¥r_usb_hhid_demo.c)

This application program performs the processing described below. For the data reception processing from the HID device, refer to 4.2.2, Main Loop (Normal mode: demo_src¥r_usb_hhid_apl.c).

1. The data reception processing from the HID device
2. LCD display based on data received from the HID device.
3. Processing to transmit a suspend or resume signal to the HID device.

[Note]

Pressing a switch(button) on the RSK/RSSK functions as the transmit trigger for sending a suspend or resume signal to the HID device. For LCD/LED indication of the reception data and the switch(button) specifications , refer to 4.4.1,Switch (button).

4.2.4 Main Loop (Echo(Loopback) mode: demo_src¥r_usb_hhid_echo.c)

In echo(loop-back) mode, loop-back processing in which data sent by the USB host is received and then transmitted unmodified back to the USB host takes place as part of the main routine. An overview of the processing of the main loop is presented below.

1. When the *R_USB_GetEvent* function is called after enumeration with the USB host completes, *USB_STS_CONFIGURED* is set as the return value. When the APL confirms *USB_STS_CONFIGURED*, it calls the *R_USB_Write* function to make the data transmission request to HID device.
2. When the *R_USB_GetEvent* function is called after the data transmission to HID device completes, *USB_STS_WRITE_COMPLETE* is set as the return value. When the APL confirms *USB_STS_WRITE_COMPLETE*, it calls the *R_USB_Read* function to make a data receive request for data sent by HID device.
3. When the *R_USB_GetEvent* function is called after reception of data from the USB device has completed, *USB_STS_READ_COMPLETE* is set as the return value. When the APL confirms *USB_STS_READ_COMPLETE*, it calls the *R_USB_Write* function to make a data transmit request to transmit the received data to the USB host.
4. The processing in steps 2 and 3, above, is repeated.

An overview of the processing performed by the APL is shown below:

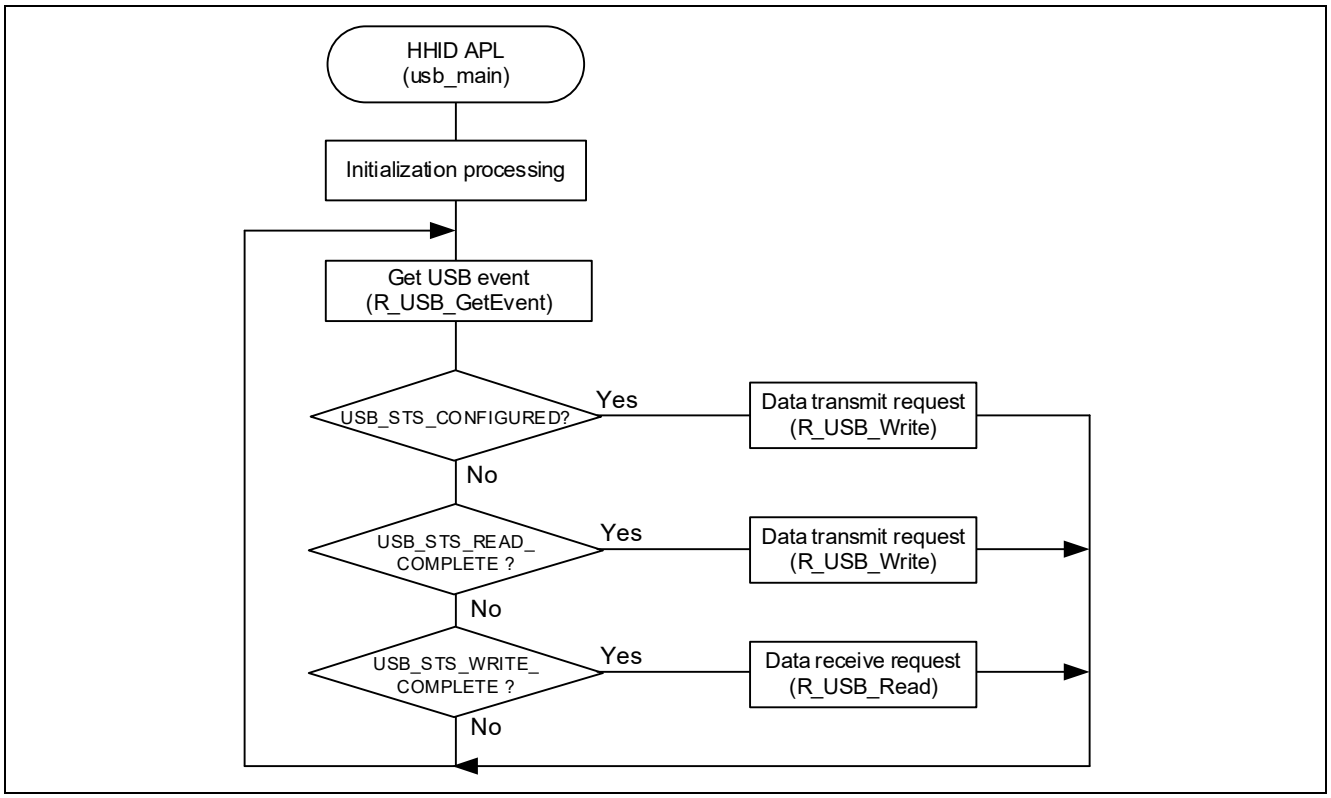


Figure 4-2 Main Loop (Echo mode)

4.3 Application Processing (for RTOS)

The application comprises two parts: initial settings and main loop. An overview of the processing in these two parts is provided below.

4.3.1 Initial Settings

Initial settings consist of MCU pin settings, USB driver settings, and initial settings to the USB controller.

4.3.2 Main Loop (Normal mode: demo_src¥r_usb_hhid_apl.c)

The loop performs processing to receive data from the HID device as part of the main routine. An overview of the processing performed by the loop is shown below.

1. When a USB-related event has completed, the USB driver calls the callback function (*usb_apl_callback*). In the callback function (*usb_apl_callback*), the application task (APL) is notified of the USB completion event using the real-time OS functionality.
2. In APL, information regarding the USB completion event was notified from the callback function is retrieved using the real-time OS functionality.
3. If the USB completion event (the *event* member of the *usb_ctrl_t* structure) retrieved in step 2 above is *USB_STS_CONFIGURED*, APL sends the class request (*SET_PROTOCOL*) to the HID device.
4. If the USB completion event (the *event* member of the *usb_ctrl_t* structure) retrieved in step 2 above is *USB_STS_REQUEST_COMPLETE*, APL performs a data reception request to receive data transmitted from the HID device by calling the *R_USB_Read* function.
5. The avove processing is repeated.

An overview of the processing performed by the APL is shown below:

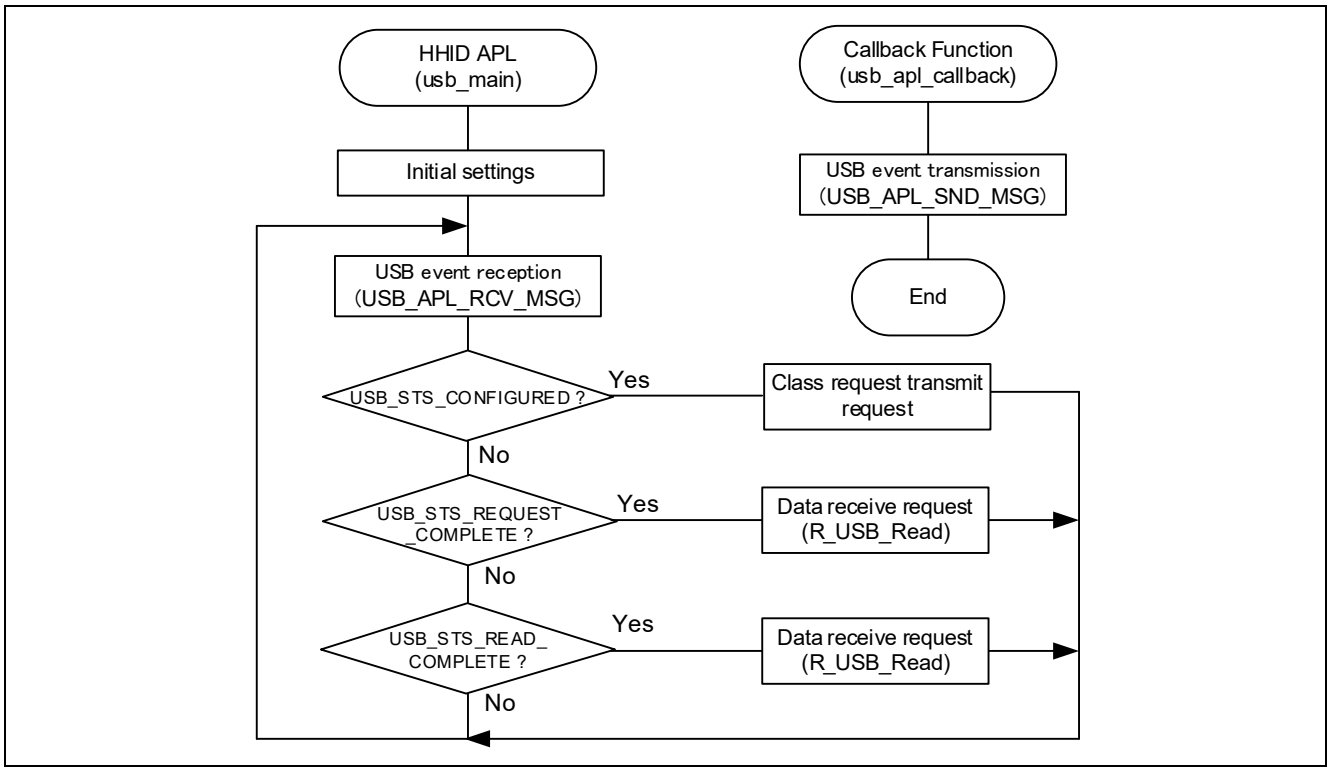


Figure 4-3 Main Loop (Normal mode)

4.3.3 Main Loop (Demo mode: demo_src¥r_usb_hhid_demo.c)

This application program performs the processing described below. For the data reception processing from the HID device, refer to **4.2.2, Main Loop (Normal mode: demo_src¥r_usb_hhid_apl.c)**.

1. The data reception processing from the HID device
2. LCD display based on data received from the HID device.
3. Processing to transmit a suspend or resume signal to the HID device.

[Note]

Pressing a switch(button) on the RSK functions as the transmit trigger for sending a suspend or resume signal to the HID device. For LCD/LED indication of the reception data and the switch(button) specifications, refer to **4.4, Switch(Button) Operations and LCD/LED Indications in Demo Mode.**

4.3.4 Main Loop (Echo(Loopback) mode: demo_src¥r_usb_hhid_echo.c)

In echo(loop-back) mode, loop-back processing in which data sent by the USB host is received and then transmitted unmodified back to the USB host takes place as part of the main routine. An overview of the processing performed by the loop is shown below.

1. When a USB-related event has completed, the USB driver calls the callback function (*usb_apl_callback*). In the callback function (*usb_apl_callback*), the application task (APL) is notified of the USB completion event using the real-time OS functionality.
2. In APL, information regarding the USB completion event was notified from the callback function is retrieved using the real-time OS functionality.
3. If the USB completion event (the *event* member of the *usb_ctrl_t* structure) retrieved in step 2 above is *USB_STS_CONFIGURED*, APL performs a data transmission request to the HID device by calling *R_USB_Read* function.
4. If the USB completion event (the *event* member of the *usb_ctrl_t* structure) retrieved in step 2 above is *USB_STS_WRITE_COMPLETE*, APL performs a data reception request to receive data transmitted from the HID device by calling the *R_USB_Read* function.
5. If the USB completion event (the *event* member of the *usb_ctrl_t* structure) retrieved in step 2 above is *USB_STS_READ_COMPLETE*, APL performs a data transmission request to send the recieved data to HID device by calling the *R_USB_Write* function.
6. The avove processing is repeated.

An overview of the processing performed by the APL is shown below:

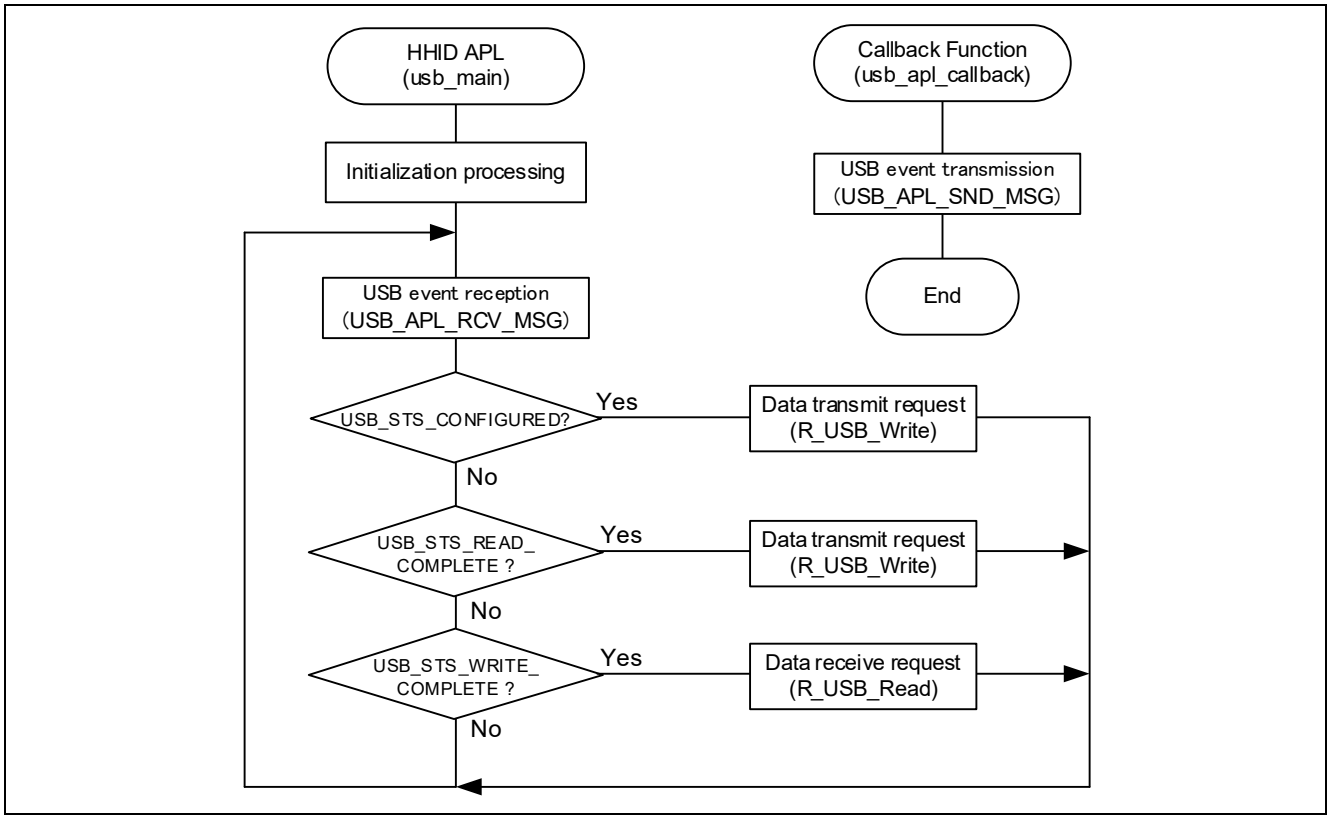


Figure 4-4 Main Loop (Echo mode)

4.4 Switch(Button) Operations and LCD/LED Indications in Demo Mode

4.4.1 Switch (button)

The APL starts data transfer after an HID device is connected. Pushing the switch while data transfer is in progress causes the following operations to occur:

1. When in the data transfer state, pressing SW2 transitions the HID device to the suspend state.
2. When the HID device is in the suspend state, pressing SW2 cancels the suspend state.

Table 4-1 shows the switch(button) specification.

Table 4-1 Switch Specification

Swich Name	Switch Number	Description
State Change Switch	Switch2(SW2)	Changes the status of the HID device connected to the USB0 module. 1. Data transfer state: Transitions to suspend state 2. Suspend state: Transitions to data transfer state

[Note]

For details of the switch and MCU pin connections on the RSK/RSSK, refer to the instruction manual of the RSK/RSSK and the user's manual of the MCU.

4.4.2 Display Information

The APL displays on the LCD screen the connection state of the HID device and data received from the connected HID device.

Mouse connected	:	Displays on the LCD the amount of movement on the X and Y axes (– 127 to 127). Lighting on LED0 when right clicking, Lighting on LED1 when left clicking, Lighting on LED2 when wheel button clicking.
Keyboard connected	:	Displays on the LCD the last input key data.

The LCD indication does not change when the data received from the HID device is NULL (no key on keyboard pressed, mouse not moved on X or Y axes).

4.5 Configuration File for the application program (r_usb_hhid_apl_config.h)

Make settings for the definitions listed below.

1. OPERATION_MODE Definition

Specify one of the following settings for the *OPERATION_MODE* definition.

```
#define   OPERATION_MODE   HID_NORMAL          // Normal Mode
#define   OPERATION_MODE   HID_DEMO            // Demo Mode
#define   OPERATION_MODE   HID_ECHO            // Echo Mode
```

2. USB_SUPPORT_RTOS Definition

Please specify *USB_APL_ENABLE* to *USB_SUPPORT_RTOS* definition when using the real-time OS.

```
#define   USB_SUPPORT_RTOS   USB_APL_DISABLE   // No use the real-time OS
#define   USB_SUPPORT_RTOS   USB_APL_ENABLE    // Use the real-time OS
```

3. Note

The above configuration settings apply to the application program. USB driver configuration settings are required in addition to the above settings. For information on USB driver configuration settings, refer to the application note *USB Basic Mini Host and Peripheral Driver (USB Mini Firmware) using Firmware Integration Technology* (Document number: R01AN2166).

5. Class Driver Overview

5.1 Class Request (Request from Host to Device)

Table 5-1 lists the class requests supported by the HHID.

Table 5-1 Supported Basic Requests and HID Class Requests

Request	Code	Description
GET_REPORT	0x01	Requests a report from the HID device.
SET_REPORT	0x09	Notifies the HID device of a report.
GET_IDLE	0x02	Requests the duration from the HID device.
SET_IDLE	0x0A	Notifies the HID device of the duration.
GET_PROTOCOL	0x03	Requests the protocol from the HID device.
SET_PROTOCOL	0x0B	Notifies the HID device of the protocol
GET_REPORT_DESCRIPTOR	Standard	Requests the report descriptor.
GET_HID_DESCRIPTOR	Standard	Requests the HID descriptor

5.2 Data Format

The boot protocol data format of data received from the keyboard or mouse through interrupt-IN transfers is shown below

Table 5-2 Receive Data Format (Boot Protocol)

offset	Keyboard (8 Bytes)	Mouse (3 Bytes)
0 (Top Byte)	Modifier keys	b0：Button 1 b1：Button 2 b2：Button 3 b3-b7：Reserved
+1	Reserved	X displacement
+2	Keycode 1	Y displacement
+3	Keycode 2	—
+4	Keycode 3	—
+5	Keycode 4	—
+6	Keycode 5	—
+7	Keycode 6	—

[Note]

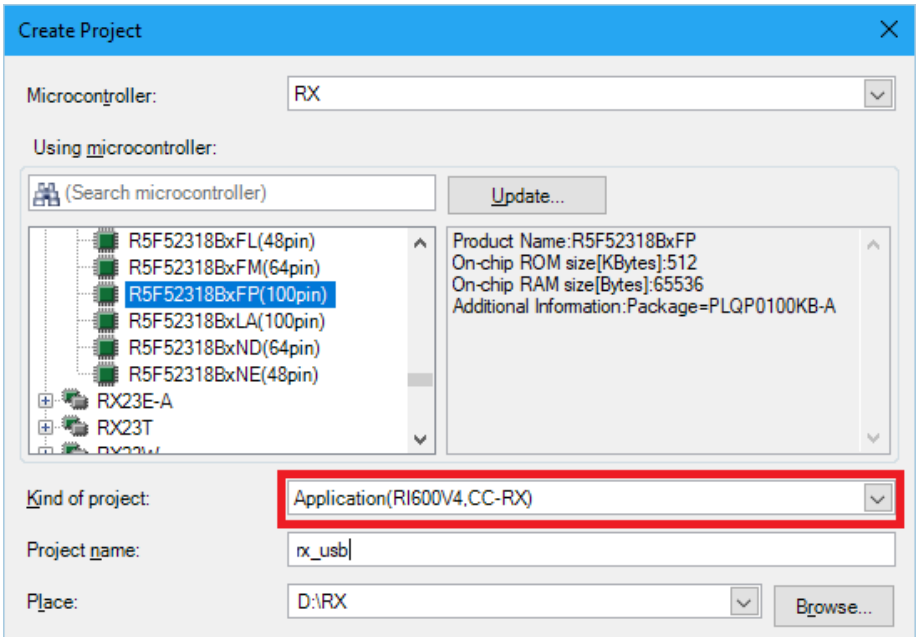
The data format used for report protocol data transfers must conform to the report descriptor. This driver does not acquire and analyze the report descriptor; it determines the report format according to whether the interface protocol code is “keyboard” or “mouse.” The user should make changes as necessary to match the system specifications.

6. Using RI600V4 project with CS+

The RI600V4 project in the package does not support CS+. The user needs to create a project for CS+ according to the following procedure when using RI600V4 project on CS+.

6.1 New Project Creation

Select "Application(RI600V4, CC-RX) for the Kind of project.

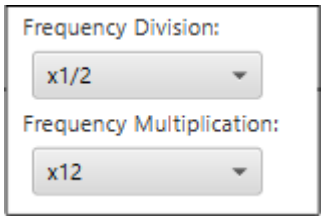


6.2 Launch Smart Configurator

1. Clock Setting (Select "Clocks" tab)

Set the related clock so that "48MHz" is set to UCLK (USB clock).

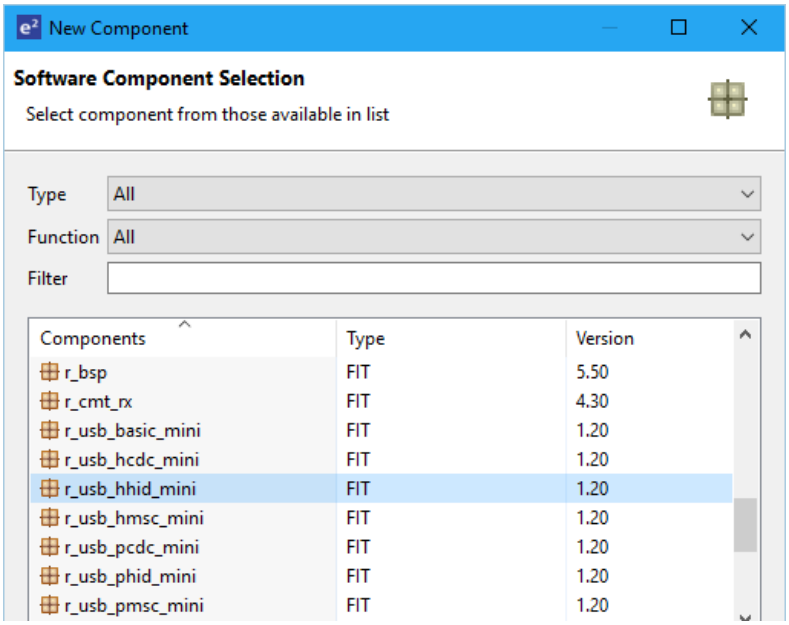
The following is a setting example when using the oscillator(8MHz).



2. Component Setting (Select "Components" tab)

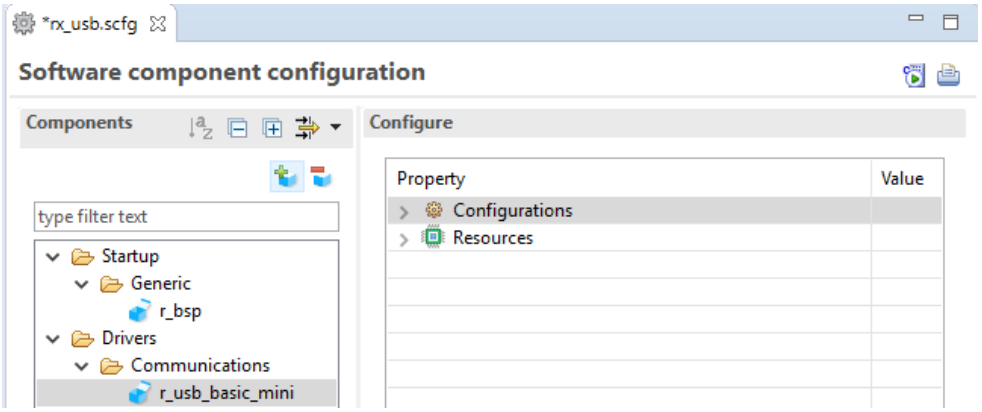
(1). Import the USB FIT module

Select the *r_usb_hhid_mini* module and press the "Finish" button. The *r_usb_basic_mini* module is imported at the same time.



(2). Configuration Setting

a. r_usb_basic_mini



(a). Configurations

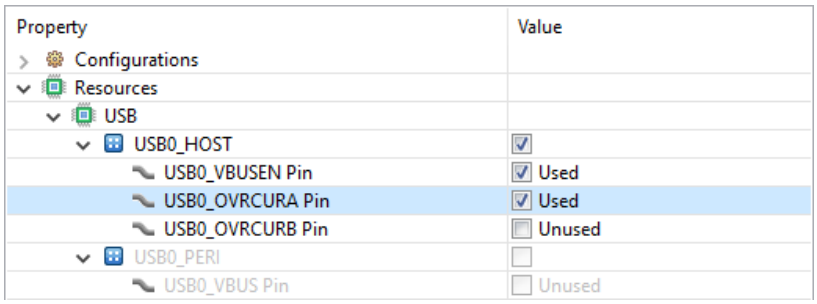
Set each item according to the user system.
For the detail of each item, refer to chapter "Configuration" in *USB Mini Basic Host and Peripheral Driver Firmware Integration Technology* application note (Document number: R01AN2166).

(b). Resources

Check the following check box.

i. *USBx_VBUSEN* pin

ii. *USBx_OVRCURA* pin or *USBx_OVRCURB* pin

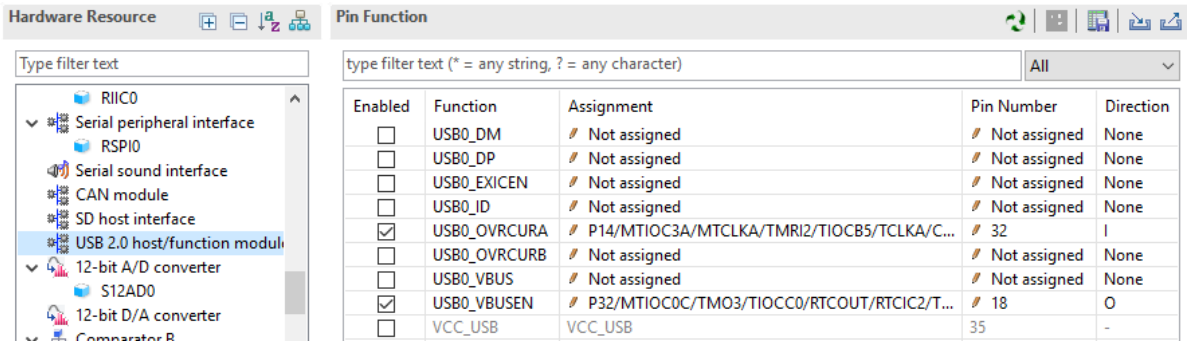


b. r_usb_hhid_mini

Refer to chapter "Configuration" in *USB Host Human Interface Devices Class Driver (HHID) for USB Mini Firmware Firmware Integration Technology* application note (Document number: R01AN2168).

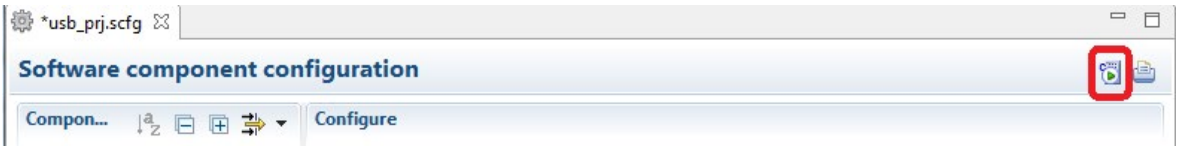
3. Pin Setting (Select "Pins" tab)

Select the port for USB pin match the user system.



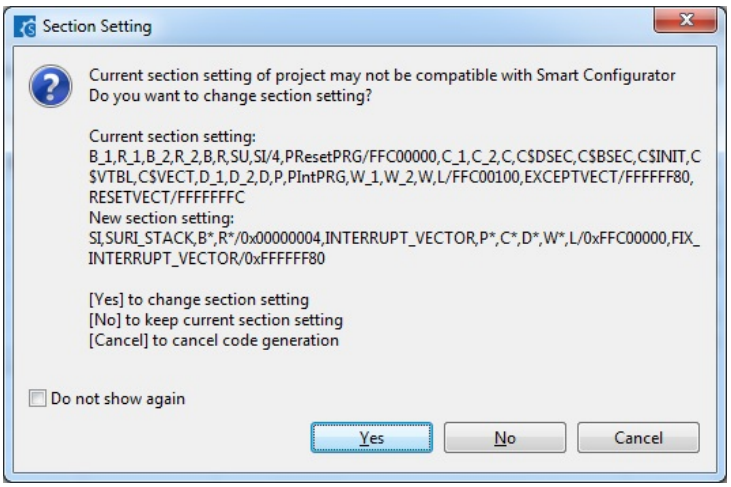
4. Generate Code

The Smart Configurator genrates source codes for USB FIT module and USB pin setting in "*<ProjectDir>¥src¥smc_gen*" folder by by clicking on the [(Generate Code)] button.



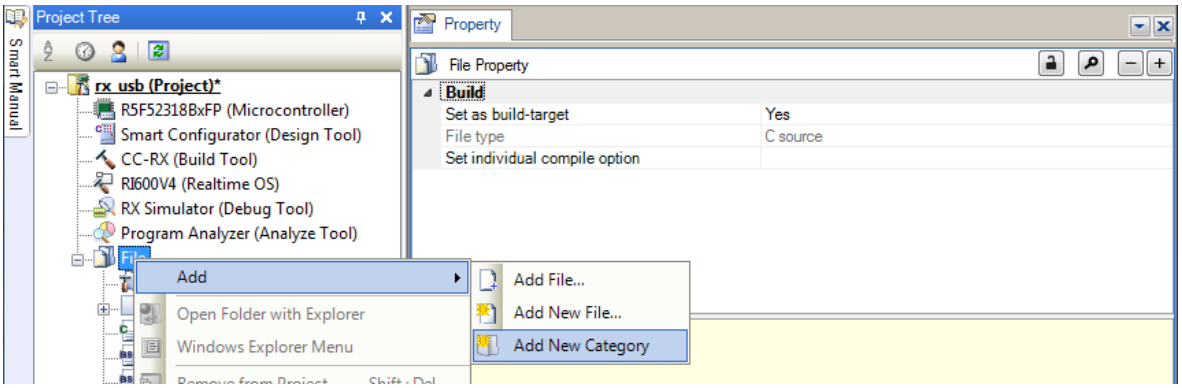
Note:

Select "Yes" if the following dialog box is displayed.



6.3 Add the application program and the configuration file

1. Copy the *demo_src* folder in this package to the *"<ProjectDir>¥src"* folder.
2. Copy the RI600V4 configuration file (.cfg file) to *"<ProjectDir>"* folder.
3. Select "File" in the "Project Tree" and click the right button. Select [Add] → [Add New Category] and create the category to store the application program. Then select [Add File] and register the application program and the configuration file which are copied at the above 2.



Note:

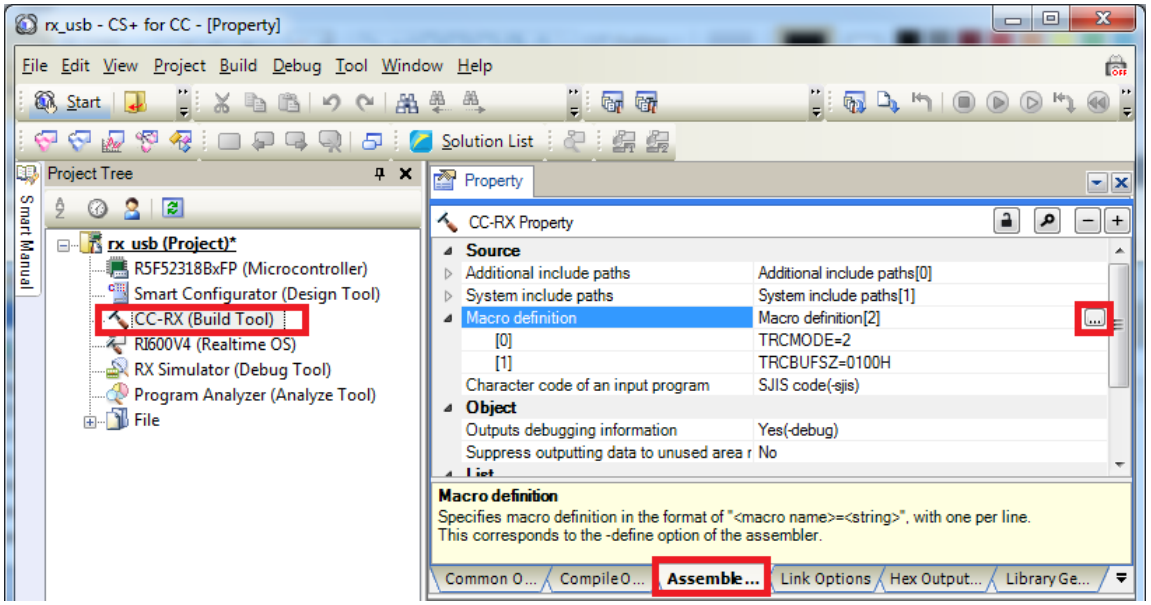
Remove the "task.c" file and "sample.cfg" created in "<*ProjectDir*>" folder by CS+.

6.4 Remote Macro Definition

Remove these macros since the following macros is defined in the new created project.

Select [CC-RX(Build Tool)] → [Assemble Options] tab, remove the following macros.

1. TRCMODE = 2
2. TRCBUFSZ = 0100H



6.5 Build Execution

Excecute the build and generate the binary target program.

7. Using the e² studio project with CS+

The HHID contains a project only for e² studio. When you use the HHID with CS+, import the project to CS+ by following procedures.

[Note]

1. Uncheck the checkbox Backup the project composition files after conversion in Project Convert Settings window.
2. The following method is not supported when using RI600V4. Refer to chapter **6, Using RI600V4 project with CS+**.

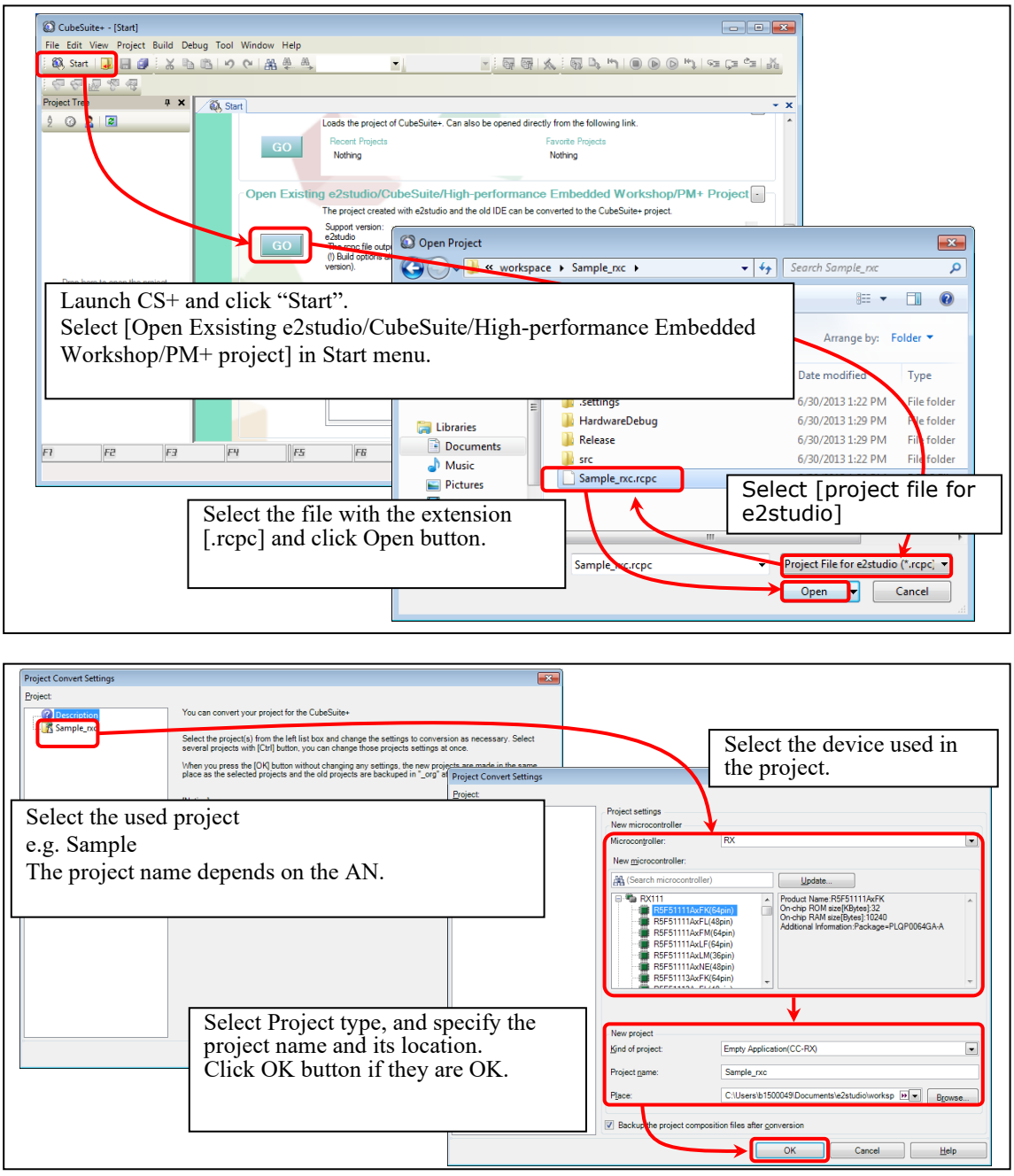


Figure 7-1 Using the e² studio project with CS+

Website and Support

Renesas Electronics Website

http://www.renesas.com/

Inquiries

http://www.renesas.com/inquiry/

Revision Record

Rev.	Date	Descriptions	
		Page	Summary
1.00	Dec 1, 2014	—	First Edition Issued.
1.01	Jun 1, 2015	—	RX231 is added in Target Device.
1.02	Dec 28, 2015	—	Upgrading of this USB driver by upgrading of "USB Basic Mini Firmware (R01AN2166)".
1.10	Nov 30, 2018	—	1. The following chapter has beed added. (1). 3.1.2 RSK/RSSK Setting 2. The following chapter has beed changed. (1). 4. Sample Application
1.12	Jun 30, 2019	—	RX23W is added in Target Device.
1.20	Jun 1, 2020	—	Supported 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.

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 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.
5. 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.
6. 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.
7. 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.
8. 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.
9. 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.
10. 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.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

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