Renesas USB MCU

R01AN1965EJ0103
Rev.1.03
Mar 28, 2016

USB Host Android Open Accessory (AOA) using Basic Mini Firmware

Introduction

This application note describes the use of USB Basic Mini Firmware for Renesas USB MCUs in USB Host Android Open Accessory.

This document is an application note describing the use of USB Host Android Open Accessory (AOA) with the USB Basic Firmware "Mini," a sample program for USB interface control using Renesas USB MCUs.

Confirmed Target Device

RL78/G1C

The program described here can be used with other microcontrollers (MCUs) that have the same USB module as the above target device. When using this code in an end product or other application, its operation must be tested and evaluated thoroughly by the user.

This program has been evaluated using the Renesas Starter Kit.

Contents

1. Overview

This document is an application note describing use of the USB Host Android Open Accessory (AOA) and with USB-BASIC-F/W (see section 1.2).

1.1 Functions and Features

USB Host Android Open Accessory conforms to the Universal Serial Bus Revision 2.0 specification (referred to as USB in this document) and enables communication with an Android device based on Android Open Accessory (AOA).

This class driver is intended to be used in combination with the USB Basic Firmware Mini from Renesas Electronics.

1.2 Related Documents

1. Universal Serial Bus Revision 2.0 specification
2. Android™(*) Open Accessory
   [http://developer.android.com/index.html]
   (*) Android is a trademark of Google Inc.
3. User's Manual: Hardware
4. Renesas USB MCU USB Basic Mini Firmware Application Note
   Available from the Renesas Electronics Website

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

1.3 Terms and Abbreviations

Terms and abbreviations used in this document are listed below.

AOA	:	Android Open Accessory
API	:	Application Program Interface
APL	:	Application program
cstd	:	Prefix of function and file for Host & Peripheral USB-BASIC-F/W
CS+	:	Renesas integrated development environment
Data Transfer	:	Generic name of Control transfer, Bulk transfer and Interrupt transfer
HCD	:	Host control driver of USB-BASIC-F/W
HDCD	:	Host device class driver (device driver and USB class driver)
HEW	:	High-performance Embedded Workshop
HID	:	Human interface device class
HM	:	Hardware Manual
hstd	:	Prefix for host function of USB-BASIC-F/W
HVEN	:	Host Vendor
KBD	:	Keyboard device
MGR	:	Peripheral device state manager of HCD
MSE	:	Mouse device
PP	:	Pre-process definition
RSK	:	Renesas Starter Kit
SW1/SW2/SW3	:	User switches on RSK
USB	:	Universal Serial Bus
USB-BASIC-FW	:	USB-BASIC-F/W (Host & Peripheral USB Basic Mini Firmware (USB low level) for Renesas USB MCU)
Task	:	Processing unit
Scheduler	:	For easy scheduling of task processing
Scheduler Macro	:	Used for calling scheduler functions

1.4 How to Use this Document

This book is not intended to be read from start to finish, first to last chapter. Use it first gain familiarity with the contents of the sample program, andthen refer to sections describing the functions or interface needed to create your own device solution.

Section 5.3 provides a list of source codes; MCU-specific source codes are available here: \devicename\src\HwResource. Use whichever files are needed for your specific application.

The application may require some modification to create the right solution for your device. Section 6 describes host vendor application operations as a reference.

All code modules are organized by task, and messages are passed between tasks. The scheduler determines the order in which functions (tasks) are executed. This allows the user to give priority to the most important tasks. The call-back mechanism registered in each task ensures parallel processing (non-blocking mode). The task mechanism is explained in Section 1.2 of the USB-BASIC-F/W Application Note**.**

2. How to Register a Device Class Driver

A user-created class driver functions as a USB device class driver when registered in the USB-BASIC-F/W. For instructions on how to register a driver, refer to function *usb_hsmpl_driver_registration()* in file *r_usb_vendor_hapl.c*. For details, see the USB-BASIC Application Note.

3. Android Open Accessory (AOA)

This software complies with Android Open Accessory 1.0 / 2.0 specifications. For details, see Section 1.2.

Android Open Accessory specification in interface specification established by Google Inc., developer of Android OS, and includes the following.

- Hardware connection specifications
- AOA Protocol definitions
- Android OS software API

This interface can be used for any Android device that supports Android Platform API 10 or later. In other words, this interface is applicable for Android OS version 2.3.4 or later (or version 3.1 or later for tablets). AOA version 2.0 can be used for devices that run on Android OS version 4.1 or later.

3.1 Basic Functions

The basic AOA functions are as follows.

1. Connected device verification
2. Inquiries concerning device capabilities and states
3. Output state and feature settings
4. Receiving/sending data from/to Android device

4. Operating Confirmation Environment

4.1 Compiler

The compilers which is used for the operating confirmation are follows.

a. CA78K0R Compiler V.1.71

b. CC-RL Compiler V.1.01

4.2 Evaluation Board

The evaluation boards which is used for the operating confirmation are follows.

a. Renesas Starter Kit for RL78/G1C (Product No: R0K5010JGC001BR)

5. Software Configuration

5.1 Module Configuration

AOA comprises a vendor class and device class.

AOA software modules are shown in Figure 5.1 and explained in Table 5-1.

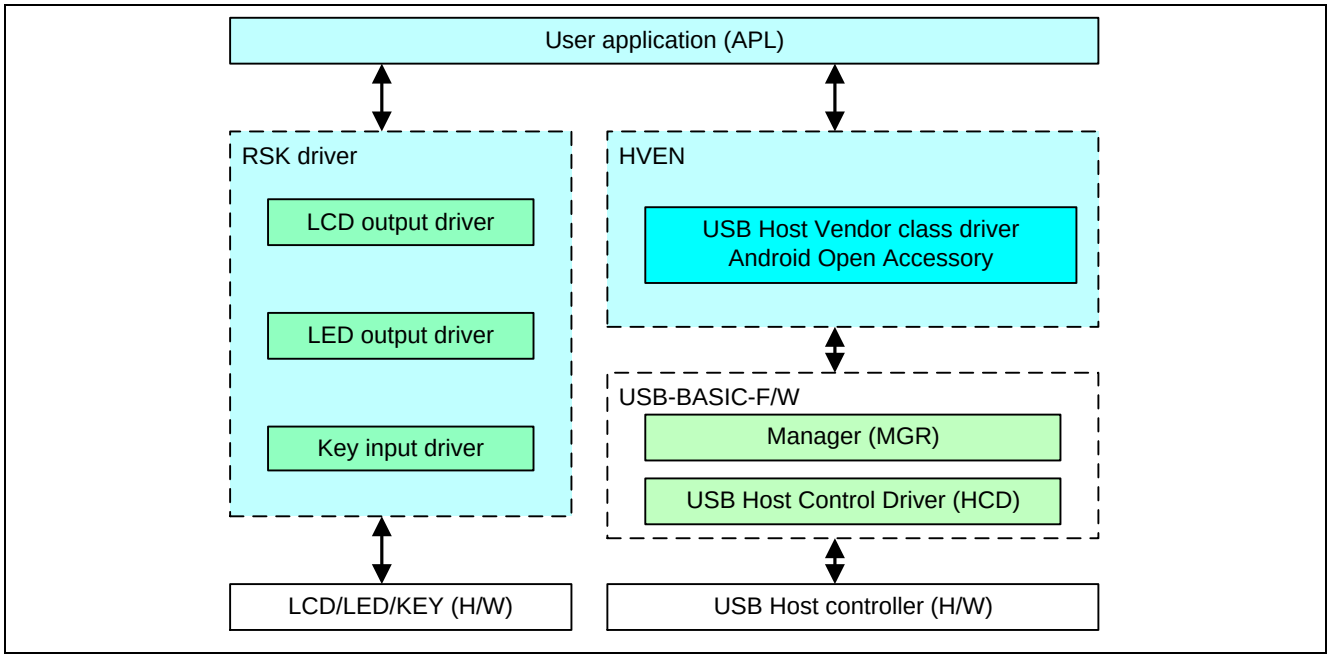

Figure 5.1 Module Configuration

Table 5-1 Module Functions

Module Name	Function Description	Notes
APL	User application program Communicates with the Android device based on switch operations	User-created
VENDOR	The registered device class driver checks the operations of the connected device. USB-BASIC-F/W confirms the compatibility of the connected device and the AOA through the APL. The following data transfer requests are sent to the USB-BASIC-F/W according to the APL requests. 1) Connected device controls based on AO protocol 2) Data transfer with connected device Results of the transmissions are notified to the APL with the callback function.	
USB-BASIC-F/W	USB host control driver (hardware control and device state management)	

5.2 Application Program Function Overview

The following are the main functions of the host demo APL.

1. Sends data to the connected Android device using AOA protocol, communicates with the Standard Android ADK Demo Kit application (referred to as Demo Kit application herein) running on the Android device.
   a) When the Demo Kit application is in “In” mode, use SW1-3 or RV1 on RSK to display the Demo Kit application status.
   b) When the Demo Kit application is in “Out” mode and the Servo1-3 gauge or LED1-3 RGB gauge is manipulated, LED0-2 on the RSK board turns on/off at the threshold value of the center of the gauge.
2. In AOA 2.0 mode, the mouse cursor will appear on the Android device screen and can be controlled with SW1-3 on the RSK board.

Table 5-2 and Table 5-3 show input/output specifications.

Table 5-2 Demo Kit In Mode Specifications

RSK Port Input	Operations (Implementations)
SW1	Turns button “B1” on/off
SW2	Turns button “B2” on/off Joystick Y axis direction input Repeatedly press the button to reverse the direction of the Y axis movement.
SW3	Turns button “B3” on/off Joystick X axis direction input Repeatedly press the button to reverse the direction of the X axis movement.
RV1	Turns Android icon on/off (RV1 AD input of 128 or higher turns LED on) Changes temp/light value (range: 0 to 255)

Table 5-3 Demo Kit Out Mode Specifications

Demo Kit Input	Operations (Implementations)
Servo1 Led1 (RGB)	Controls LED0 on RSK board 0~127: off 128~255: on
Servo2 Led (RGB)	Controls LED1 on RSK board 0~127: off 128~255: on
Servo3 Led (RGB)	Controls LED2 on RSK board 0~127: off 128~255: on
Relay1 & Relay2	No operation

5.3 Structure of Files and Folders

5.3.1 Folder Structure

The folder structure of the files supplied with this sample program is shown below.

The source codes dependent on each MCU and evaluation board are stored in each hardware resource folder (***devicename***src**HwResource*).

```
workspace
    + CS+/CCRL
      +RL78G1C                          Storage folder for all project files
        + AOA1.0                        Host build results (AOA1.0 spec)
        + AOA2.0                        Host build results (AOA2.0 spec)
        + src
          +――― VENDOR [Vendor Class driver ]      See Table 5-4
          |     +――― inc                 Vendor driver common header folder
          |     +――― src                 Vendor driver
          +―――SmplMain [Sample application ]      See Table 5-4
          |     +――― APL                 AOA communication application
          |     +――― inc                 AOA communication application common header
          +―――USBSTDFW [Basic firmware shared by all USB drivers ]
          |     +――― inc                 USB driver common header file
          |     +――― src                 USB driver
          +――― HwResource [Hardware access layer such as MCU initialization ]
                +――― inc                 Hardware resource header file
                +―――src                  Hardware resource
```

[Note]

a. The project for CA78K0R compiler is stored under the CS+ folder.

b. Refer to **10 Using the e2 studio project with CS+** section when using CC-RL compiler on CS+.

5.3.2 File List

Table 5-4 shows the file structure supplied with Vendor AOA.

Table 5-4 File Structure

Folder	File Name	Description
VENDOR/inc	r_usb_vendor_api.h	USB host Vendor AOA macro definitions
VENDOR/src	r_usb_vendor_hapi.c	USB host Vendor AOA API functions
VENDOR/src	r_usb_vendor_hdriver.c	USB host Vendor AOA driver functions
SmplMain	main.c	Main install process
SmplMain/inc	r_usb_aoa_common.h	USB host Vendor AOA common header
SmplMain/APL	r_usb_vendor_hapl.c	Sample application program

5.4 System Resources

5.4.1 System Resource Definitions

Table 5-5 lists task IDs and priority definitions for registering and using the vendor AOA in the scheduler.

These resources are defined in the header file *r_usb_ckernelid.h.*

Table 5-5 Scheduler Registration IDs

Scheduler Registration Task	Description	Note
USB_HVEN_TSK	**HVEN** (R_usb_hvndr_Task) Task ID: USB_HVEN_TSK Task priority: 2	
USB_HCD_TSK	**HCD** (R_usb_hstd_HcdTask) Task ID: USB_HCD_TSK Task priority: 0	
USB_MGR_TSK	**MGR** (R_usb_hstd_MgrTask) Task ID: USB_MGR_TSK Task priority: 1	
USB_HSMP_TSK	**HSMP** (usb_hsmpl_apl_task) Task ID: USB_HSMP_TSK Task priority: 3	
Mailbox IDs / Default Receive Task	**Message Name**	**Note**
USB_HVEN_MBX / USB_HVEN_TSK	HVEN -> HVEN / APL -> HVEN mailbox ID	
USB_HCD_MBX / USB_HCD_TSK	HCD task mailbox ID	
USB_MGR_MBX / USB_MGR_TSK	MGR task mailbox ID	
USB_HSMP_MBX / USB_HSMP_TSK	Host Sample task mailbox ID	

6. Host Vendor AOA Sample Application Program (APL)

The Host Demo Application communicates with the Demo Kit Application running on the Android device when the device is connected. The Vendor AOA Application operates according to the AOA Protocol as defined by Google Inc. See term 2 in section 1.2 for more details.

6.1 Operating Environment

Figure 6.1 shows the sample operating environment for the software.

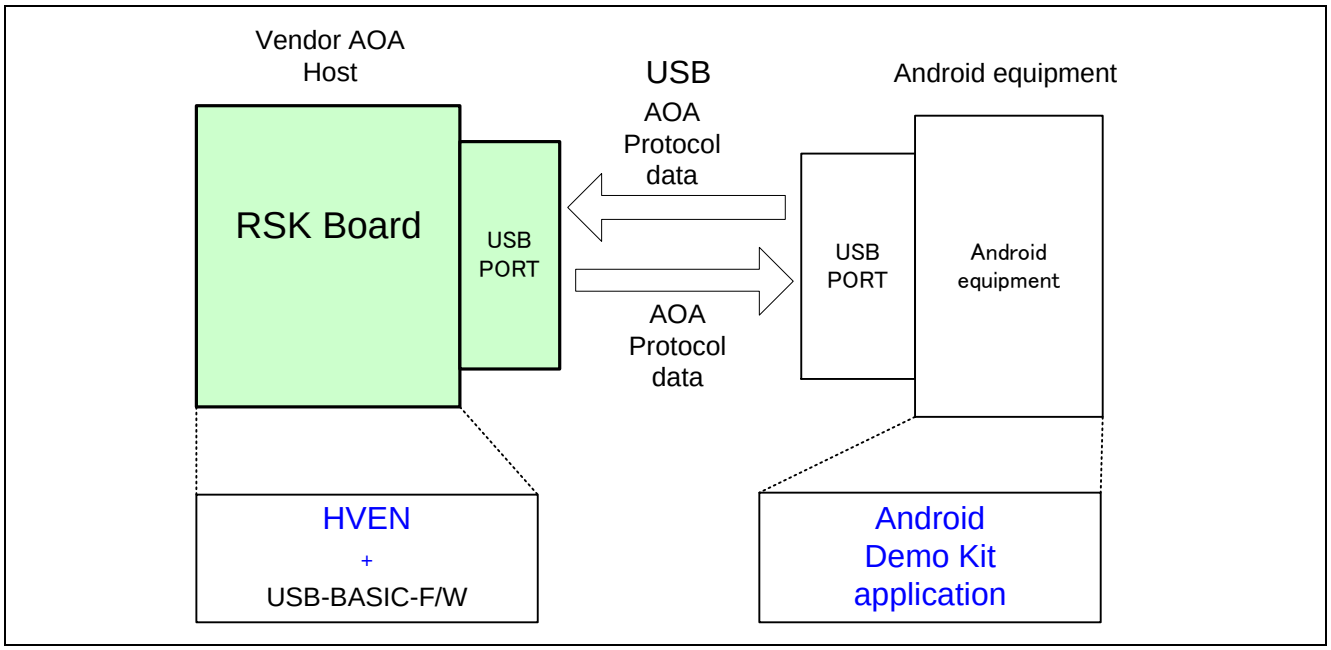

Figure 6.1 Sample Operating Environment

6.1.1 AOA 2.0 Communications

When you switch the build mode from AOA 1.0 to AOA 2.0, the ENABLE_AOA2_HID macro is enabled and the HID report can then be sent to the Android OS running on the Android device. At this time, the mouse cursor will appear on the Android device screen and can be moved by the user.

6.2 Application Program Overview

The following provides descriptions of all application operations.

- Android device connection

  The connected USB device automatically confirms whether the Android device supports AOA Protocol transmissions. If supported, the device re-enumerates, and then starts transmitting data as Bulk IN/OUT transfers. See Figure 6.2 for more details.

- Operations during data transmission (on RSK board side)
  Use SW1~3 and RV1 on the RSK board to generate and send data to the Android Demo Kit, such as mouse movement or temperature and light sensor readings.

  In AOA 2.0 operations, the Android OS also receives HID mouse movement data, which displays the cursor output on the Android terminal and enables the user to move control the HID mouse.

- Operations during data transmission (on Android Demo Kit side)
  When controlling operations on the Android Demo Kit side, you can create LED on/off settings on the RSK board and control the RSK board LEDs from the Vendor AOA side.

Table 6-1 Operations during Data Transmission

Target	Description
RSK board	Notifies HID mouse movements to Android Demo Kit Updates temperature/light data based on AD data transfer
Android Demo Kit side	Turns LED on RSK board on/off

6.3 Endpoint Specification

Endpoints used by vendor AOA are shown in Table 6-2.

Table 6-2 Endpoint Specification

EP No.	Pipe No.	Transfer Method	Description
0	0	Control In/Out	Standard request, class request
Based on received descriptor	4	Bulk In	Data transfer from device to host
Based on received descriptor	5	Bulk Out	Data transfer from host to device

The endpoint number corresponds to the endpoint descriptor of the device.

6.4 Connected Android Device

Android devices that support AOA Protocol communication can be connected to the RSK board, specifically those running on Android OS 2.3.4 or later (for tablets: OS 3.1 or later). AOA 2.0 Protocol communication supports Android OS 4.1 or later.

Devices integrated in a hub or complex devices cannot be connected to the RSK board.

6.5 APL Functions

Table 6-3 lists the sample application functions.

Table 6-3 Sample Application Functions

Function Name	Description
main	Main loop processing
usb_hsmpl_main_init	System initialization Various Host USB task startup processes
usb_hsmpl_apl_task	Vendor AOA sample application task
usb_hsmpl_driver_registration	Vendor AOA driver registration
usb_hsmpl_class_check	Connected device check
usb_hsmpl_device_state	Device state change detection callback
usb_hsmpl_AOA_SendData	Data send process (to Android device)
usb_hsmpl_AOA_SendData_result	Data send complete processing
usb_hsmpl_AOA_RecvData	Data receive processing (from Android device)
usb_hsmpl_AOA_RecvData_result	Data receive complete processing
usb_hsmpl_AOA_mouse_switch_operation	RSK board software processing

6.6 Sequence Chart

The following is the time sequence chart for the sample application.

After the RSK board completes the SET_CONFIGURATION transmission and sends the Vendor Request based on AOA Protocol, the RSK will be accepted as an Android accessory device.

Figure 6.2 shows the AOA Protocol.

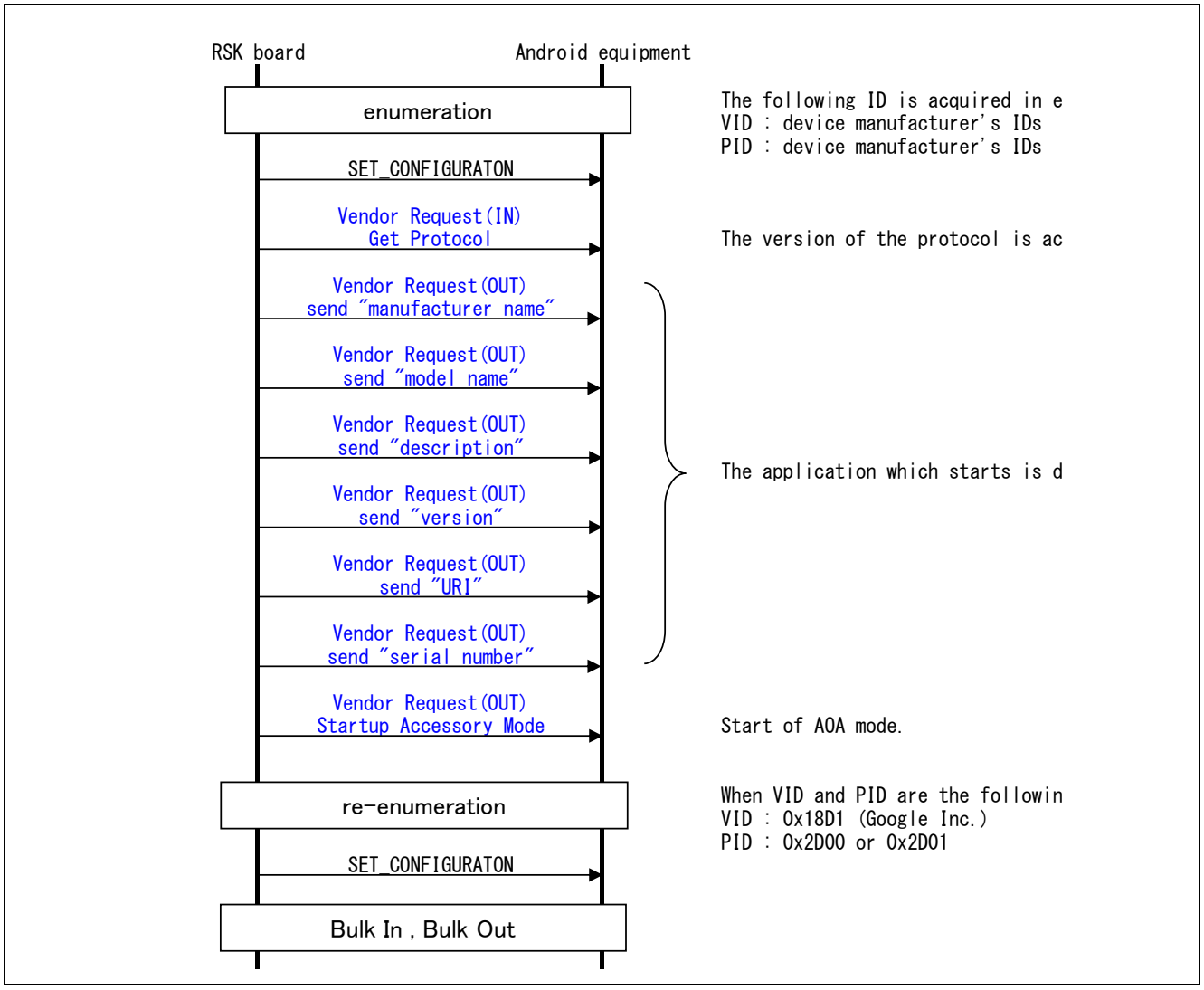

Figure 6.2 AOA Protocol Sequence

When the Android terminal is connected and normal enumeration is executed, the routine continues until it reaches the AOA Protocol, and then re-enumeration is executed. After this, the RSK can operate as an Android terminal accessory using Bulk In/Out transfers.

6.7 Sample Application Operational Flow

Figure 6.3 shows an overview of the USB-BASIC-F/W operational flow.

The USB-BASIC-F/W includes a task for executing control functions for USB data send/receive. When a hardware interrupt occurs, the task sends a message to notify the USB-BASIC-F/W. The firmware receives the message from the USB interrupt handler and executes the appropriate process based on the interrupt factor.

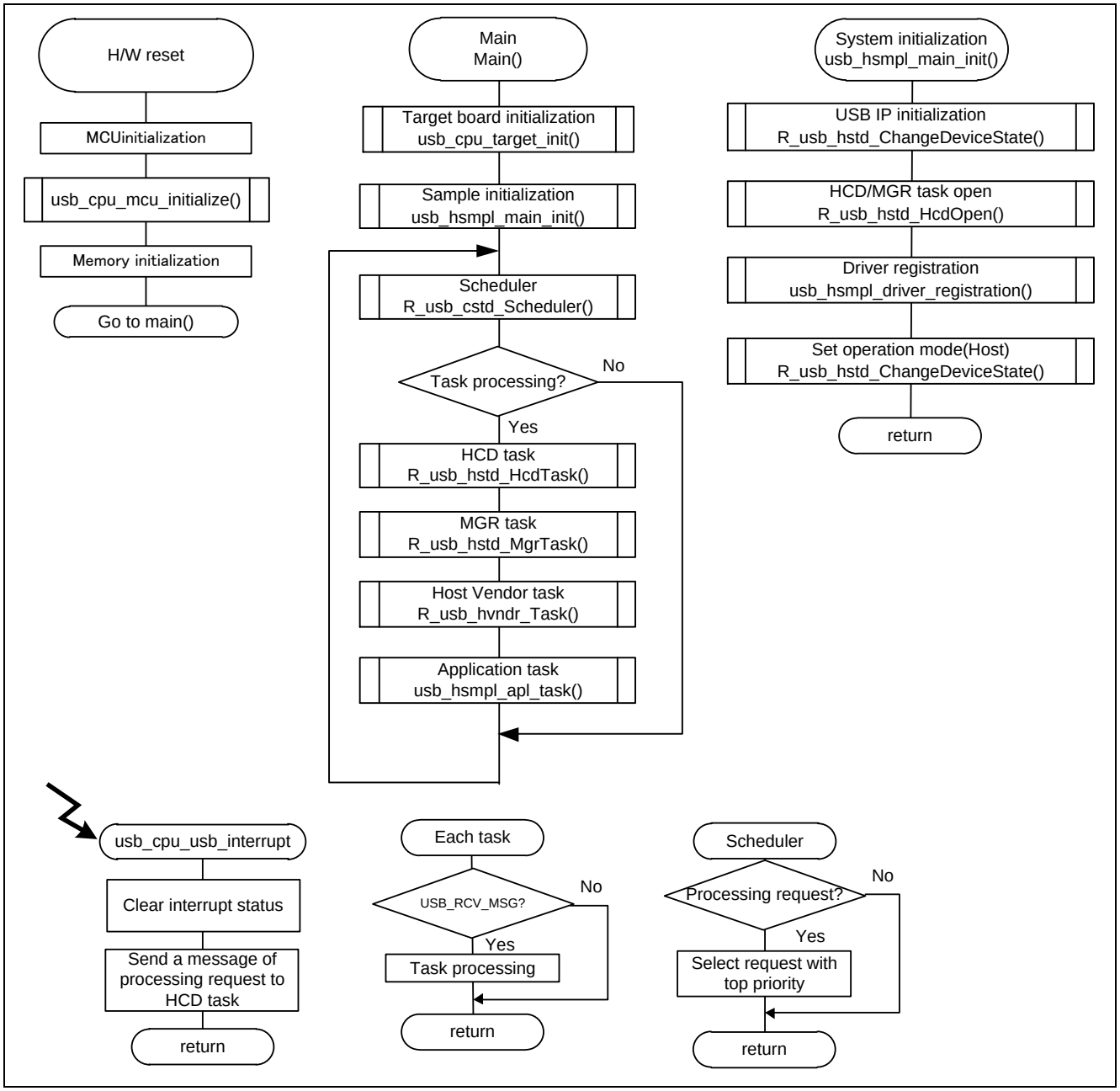

Figure 6.3 Flow Overview

7. USB Host Vendor Class Driver (HVEN)

7.1 Basic Functions

This software complies with the Android Open Accessory specifications, as shown in section 1.2.

The basic functions of HVEN are as follows.

1. Manage AOA 1.0/2.0 specification Protocol communications with the Android device
2. Send to and receive data from the Android device.

7.2 HVEN Task Description

This task executes processes according to message type received in USB_HVEN_MBX. The messages are described in Table 7-1.

Table 7-1 Message Types

Message Type	Process Description	Message Origin
USB_SMPL_CHECKREQUEST	Gets string descriptor or sets pipe according to the enumeration sequence.	USB-BASIC-F/W confirms operations of the connected device in the enumeration process with the callback function
USB_SMPL_AOA_ PROTOCOL_RESULT_CHECK	Starts Bulk In/Out transfer When data transfer is complete, notifies APL with callback function.	usb_hvndr_AOA_Protocol_result() API function is executed when AOA Protocol communication used for Bulk In/Out transfer is complete.
USB_SMPL_AOA_ HID_RESULT_CHECK	Sends HID report data with AOA Protocol. When data transfer is complete, notifies APL with the callback function.	usb_hvndr_AOA_HID_result() API function is executed (in AOA 2.0 mode) when AOA Protocol communication used for Bulk In/Out transfer is complete.

7.3 Structure

7.3.1 HVEN Structure

Table 7-2 to Table 7-4 describe the Vendor AOA class parameter structure.

Table 7-2 aoa_param_t structure

Type	Member Name	Description
input_t	Input	Storage structure member for input/sensor data from RSK board
id_t	dev_id	Storage structure member for Android device VID/PID

Table 7-3 input_t structure

Type	Member Name	Description
int8_t	mouse_x	X axis mouse movement
int8_t	mouse_y	Y axis mouse movement
int8_t	temp_c	Temperature data
uInt16_t	lux	Light data

Table 7-4 id_t structure

Type	Member Name	Description
uint8_t	v_hi	Android device Vendor ID upper 8 bits
uint8_t	v_lo	Android device Vendor ID lower 8 bits
uint8_t	p_hi	Android device Product ID upper 8 bits
uint8_t	p_lo	Android device Product ID lower 8 bits

7.3.2 Vendor Class Request

Table 7-5 shows the supported vendor class requests (bRequest). Request codes are issued according to the AOA Protocol. For more details, see Figure 6.2.

Table 7-5 Vendor Class Request

Request	bRequest	Description
ACCESSORY_GET_PROTOCOL	51	Get AOA protocol version
ACCESSORY_SEND_STRING	52	ASend Accessory Identification code
ACCESSORY_START	53	Start accessory mode

Table 7-6 shows index (wIndex) when the ACCESSORY_SEND_STRING has been sent.

Table 7-6 Accessory IdentificationCode Index

Request	wIndex	Description
ACCESSORY_STRING_MANUFACTURER	0	Manufacturer name
ACCESSORY_STRING_MODEL	1	Model name
ACCESSORY_STRING_DESCRIPTION	2	Description
ACCESSORY_STRING_VERSION	3	Version
ACCESSORY_STRING_URI	4	URI
ACCESSORY_STRING_SERIAL	5	Serial number

7.4 HVEN API

Table 7-7 provides a list of HVEN API.

Table 7-7 List of HVEN API Functions

Function Name	Description	Note
R_usb_hvndr_Task	HVEN task process	
R_usb_hvndr_ClassCheck	Message notifying connected device operations confirmation	
R_usb_hvndr_PipeRegistration	Pipe control registration process	

R_usb_hvndr_Task

HVEN task

Format

void	R_usb_hvndr_Task(void)

Argument

—	—

Return Value

—	—

Description

This task calls the *g_usb_hvndr_task()* function.

The HVEN task processes requests from the application and notifies the application of the results.

Note

Refer to the USB-BASIC-F/W Application Notes for more information concerning the corresponding loop.

Example

```
void usb_apl_task_switch(void)
{
    /* Main Loop  */
    while( 1 )
    {
        if( R_usb_cstd_Scheduler() == USB_FLGSET )
        {
            R_usb_hstd_HcdTask();       /* HCD Task */
            R_usb_hstd_MgrTask();       /* MGR Task */
            R_usb_hvndr_Task();         /* HVEN Task */
            usb_hsmpl_apl_task();       /* HSMPL Task */
        }
    }
}
```

R_usb_hvndr_ClassCheck

Descriptor check

Format

void	R_usb_hvndr_ClassCheck(uint8_t **table)

Argument

**table	Address array of the device information table
	[0] : Address of Device Descriptor
	[1] : Address of Configuration Descriptor
	[2] : Address of global variable the Device Address

Return Value

—	—

Description

This function requests the HVEN task to execute the process that determines connected device operability. Call this function when the USB-BASIC-F/W executes a *classcheck* callback.

The HVEN task references the endpoint descriptor from the configuration descriptor of the peripheral device, and then edits the pipe information table and checks the information of the pipe to be used.

Note

Example

```
USB_STATIC void usb_hhid_class_check(uint8_t **table)
{
       R_usb_hvndr_ClassCheck(table);
       usb_shven_smpl_devaddr = (usb_addr_t)(*table[2]);
}
```

R_usb_hvndr_PipeRegistration

Pipe registration

Format

void	R_usb_hvndr_PipeRegistration(void)

Argument

—	—

Return Value

—	—

Description

This function updates the address field of the pipe information table. Set the pipe to be used for AOA Protocol communication in the hardware.

Note

1. Refer to the USB-BASIC-F/W Application Note (pipe information)
2. Pre-set all fields of the pipe information table that cannot reference the endpoint descriptor.

Example

```
void  usb_smp_task( void )
{
    :
  R_usb_hvndr_PipeRegistration();
    :
}
```

7.5 HVEN Functions

Table 7-8 lists the main AOA-related functions in the HVEN.

Table 7-8 List of HVEN Internal Functions

Function Name	Description	Note
usb_hvndr_AOA_Protocol	AOA Protocol processing function	
usb_hvndr_AOA_HID	HID report data transmission processing function	

usb_hvndr_AOA_Protocol

AOA Protocol processing function

Format

void usb_hvndr_AOA_Protocol(usb_addr_t addr, usb_cb_t complete)

Argument

addr	Device Address
complete	Callback function address

Return Value

—	—

Description

Creates AOA Protocol-formatted data and executes data transmissions.

This function is repeatedly called by the HVEN task to create and transmit data for the first sequence in the AOA Protocol. See Figure 6.2 for more details.

Note

—

Example

```
void    g_usb_hvndr_task(void)
{
    :
  usb_hvndr_AOA_Protocol(g_usb_hvndr_DevAddr,(usb_cb_t)&usb_hvndr_AOA_Protocol_
result);
    :
}
```

usb_hvndr_AOA_HID

HID Processing Function

Format

void	usb_hvndr_AOA_HID(usb_addr_t addr, usb_cb_t complete)

Argument

addr	Device Address
complete	Callback function address

Return Value

—	—

Description

Executes the HID Accessory process according to the AOA Protocol.

This function is repeatedly called by the HVEN task to transfer the HID report descriptor and HID report data.

Note

This function is used in the AOA 1.0 mode.

Example

```
void    g_usb_hvndr_task(void)
{
    :
  usb_hvndr_AOA_HID( g_usb_hvndr_DevAddr,(usb_cb_t)&usb_hvndr_AOA_HID_result );
    :
}
```

8. Limitations

The following limitations apply to Vendor AOA.

1. This sample does not support operations in the audio mode.
2. Only one device can be connected to the Vendor AOA driver. Do not connect more than one device at any time.

9. Setup for the e² studio project

(1). Start up e² studio.

* If starting up e² studio for the first time, the Workspace Launcher dialog box will appear first. Specify the folder which will store the project.

(2). Select [File] → [Import]; the import dialog box will appear.

(3). In the Import dialog box, select [Existing Projects into Workspace].

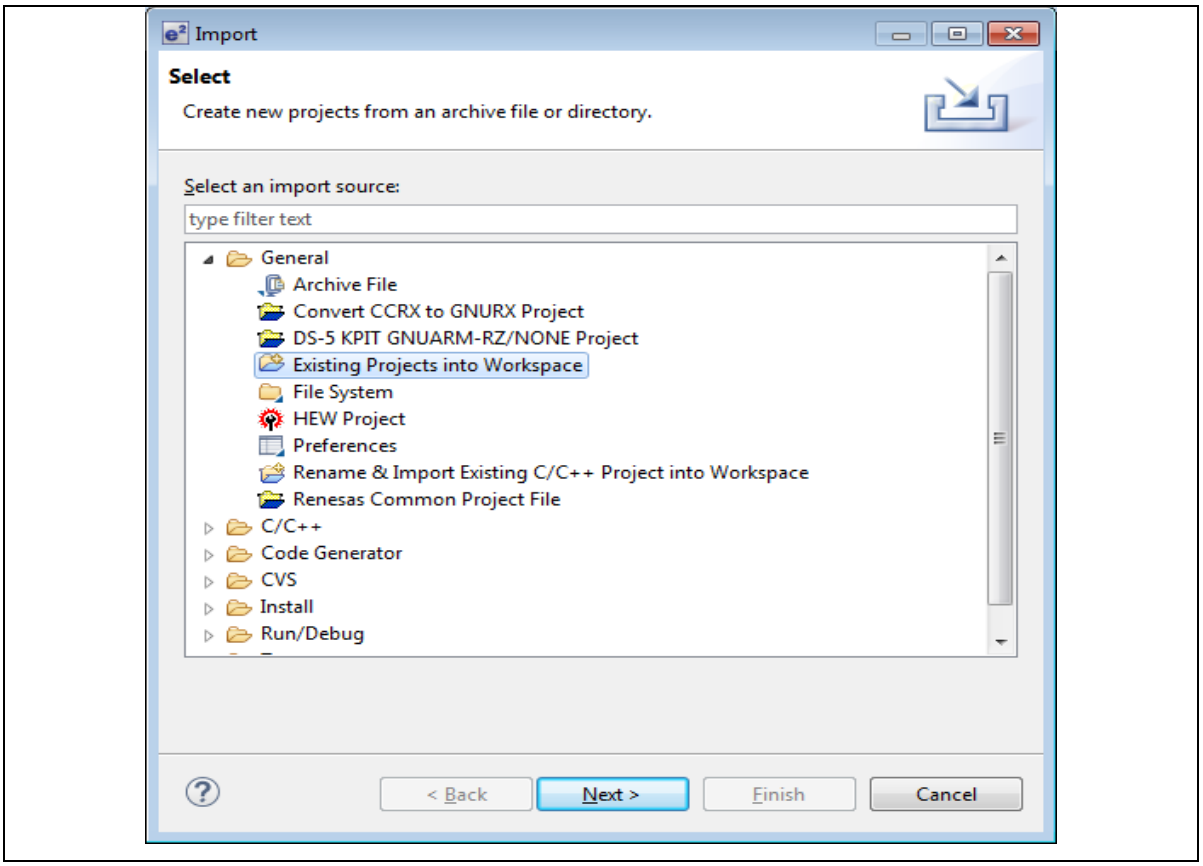

Figure 9-1 Select Import Source

(4). Press [Browse] for [Select root directory]. Select the folder in which [.cproject] (project file) is stored.

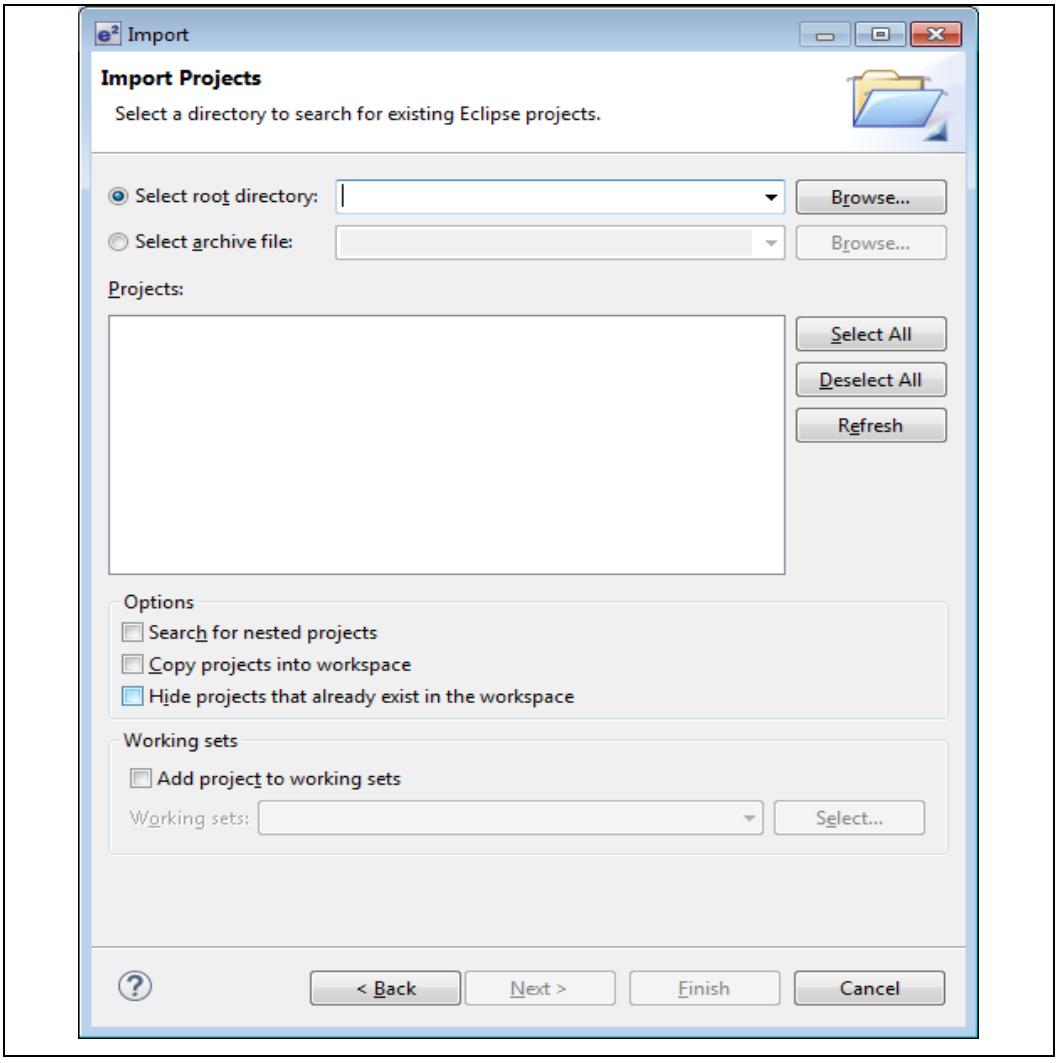

Figure 9-2 Project Import Dialog Box

(5). Click [**Finish**].

This completes the step for importing a project to the project workspace.

10. Using the e² studio project with CS+

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

Note:

The *rcpc* file is stored in "workspace\CCRL*devicename*" folder.

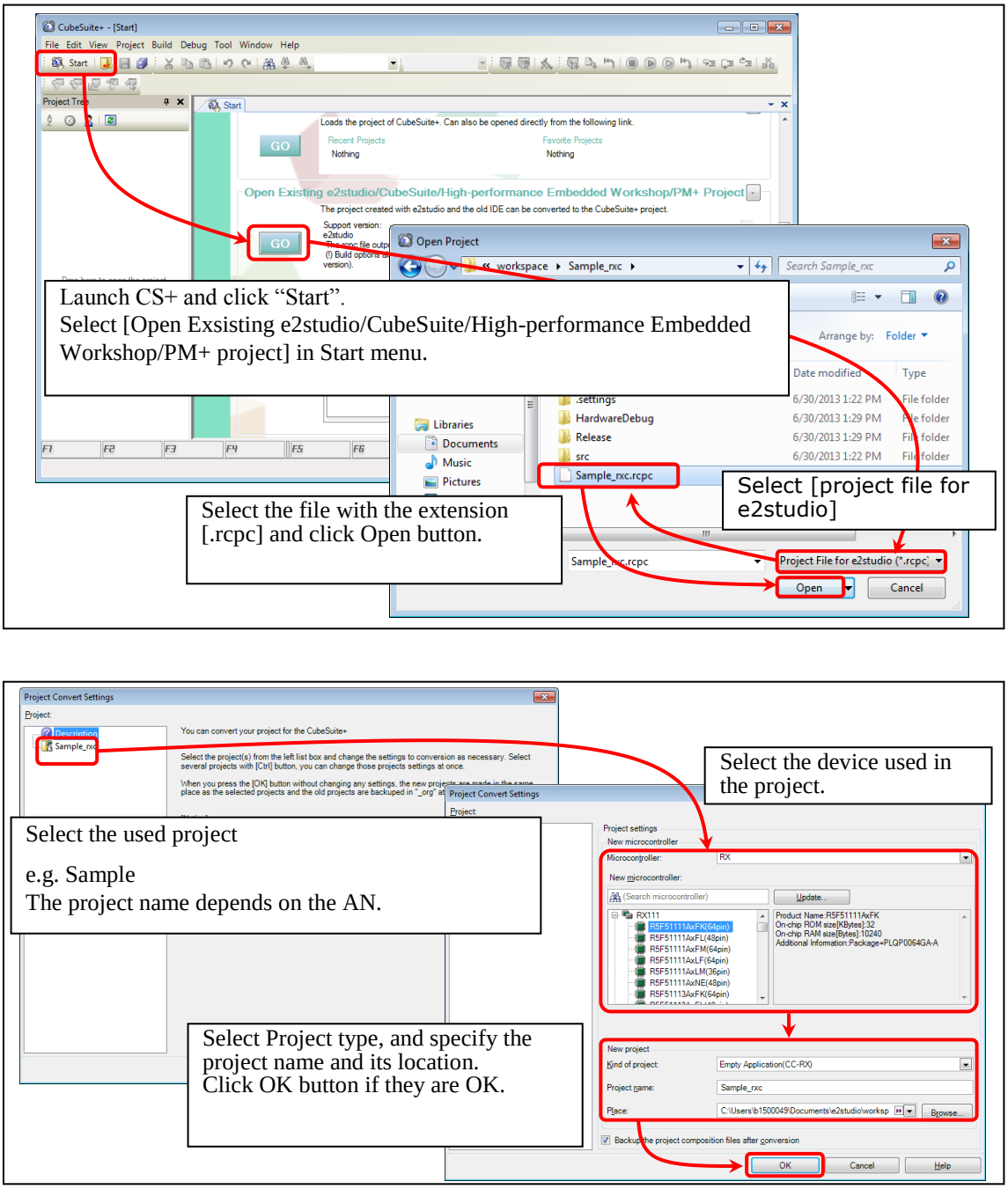

Figure 10-1 Using the e² studio project with CS+

Website and Support

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

Inquiries
http://www.renesas.com/contact/

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Mar.31.14	—	First edition issued
1.01	Mar.16.15	—	The documentation in “reference” folder is updated.
1.02	Jan.18.16	—	Supported Technical Update (Document No. TN-RL*-A055A/E)
1.03	Mar.28.16	—	CC-RL compiler is supported.

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU 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 an MPU or MCU 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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of 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. 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 should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended 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; and industrial robots etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
   Renesas Electronics products are neither intended nor 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 damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, please evaluate 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. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may 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 should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be 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, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

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

RENESAS

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

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

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

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

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

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 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.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, P. R. China 200333
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, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HALII Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 5.0