RZ/A1H Group

USB Peripheral Human Interface Devices Class Driver (PHID)

R01AN3506EJ0120
Rev.1.20
Feb 28, 2017

Introduction

This application note describes USB Peripheral Human Interface Devices Class Driver (PHID). This driver operates in combination with the USB Basic Peripheral Driver (USB-BASIC-F/W). It is referred to below as the PHID.

Target Device

RZ/A1H Group

Related Documents

1. Universal Serial Bus Revision 2.0 specification
2. RZ/A1H Group, RZ/A1M Group User's Manual: Hardware (Document No.R01UH0403EJ)
3. RZ/A1H Group USB Host and Peripheral Interface Driver (Document No.R01AN3291EJ)
4. RZ/A1H Group Downloading Program to NOR Flash Memory Using ARM® Development Studio 5 (DS-5™) Semi hosting Function (for GENMAI) (Document No.R01AN1957EJ)
5. RZ/A1H Group I/O definition header file (Document No.R01AN1860EJ)
6. RZ/A1H Group Example of Initialization (for GENMAI) (Document No.R01AN1864EJ)

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

Contents

1. Overview

PHID, when used in combination with the USB-BASIC-F/W, operates as a USB peripheral human interface device class driver (PHID). The PHID conforms to the USB Human Interface Device class specifications (referred to here as HID) and implements communication with a HID host.

PHID supports the following functions.

1. Data transfer to and from a USB host
2. Response to HID class requests

1.1 Operating Confirmation Environment

The operation of the USB Driver module has been confirmed under the conditions listed in Table 1.1.

Table 1.1 Operation Confirmation Conditions

Item	Description
MCU	RZ/A1H
Operating frequency	CPU clock (Iφ): 400 MHz Image-processing clock (Gφ): 266.37 MHz Internal bus clock (Bφ): 133.33 MHz Peripheral clock 1 (P1φ): 66.67 MHz Peripheral clock 0 (P0φ): 33.33 MHz
Operating voltage	Power supply voltage (I/O): 3.3 V Power supply voltage (internal): 1.8 V
Integrated development environment	ARM Integrated Development Environment ARM Development Studio (DS-5^{TM}) Version 5.16
	IAR Integrated Development Environment IAR Embedded Workbench for ARM Version 7.40
Compiler	ARM C/C++ Compiler/Linker/Assembler Ver.5.03 [Build 102]
	KPIT GNUARM-RZ v14.01
	IAR C/C++ Compiler for ARM 7.40
Operating mode	Boot mode 0 (CS0-space 16-bit booting)
Board	GENMAI board R7S72100 CPU board (RTK772100BC00000BR)
Device (Functions used on the board)	USB1 connector, USB2 connector SW6

1.2 Sample Project

Table 1.2 shows sample projects stored in each folder.

Table 1.2 Sample project in folder

Folder	Integrated development environment	Compiler
armcc	ARM Development Studio (DS-5^{TM})	ARM C/C++ Compiler/Linker/Assembler
iccarm	IAR Embedded Workbench for ARM	IAR C/C++ Compiler for ARM
kpitgcc	ARM Development Studio (DS-5^{TM})	KPIT GNUARM-RZ

1.3 Limitations

This module is subject to the following restrictions.

1. Structures are composed of members of different types (Depending on the compiler, the address alignment of the structure members may be shifted).

1.4 Terms and Abbreviations

API	:	Application Program Interface
APL	:	Application program
HID	:	Human Interface Device class
USB Host	:	HID class USB Host
PHID	:	Peripheral Human Interface Devices
USB	:	Universal Serial Bus
USB-BASIC-FW	:	USB Basic Peripheral Driver
Data transfer	:	Control transfer, Bulk transfer, Interrupt transfer

2. Operating environment

Figure 2-1 shows an example operating environment for this F/W. Refer to the associated instruction manuals for details on setting up the evaluation board and using the emulator, etc.

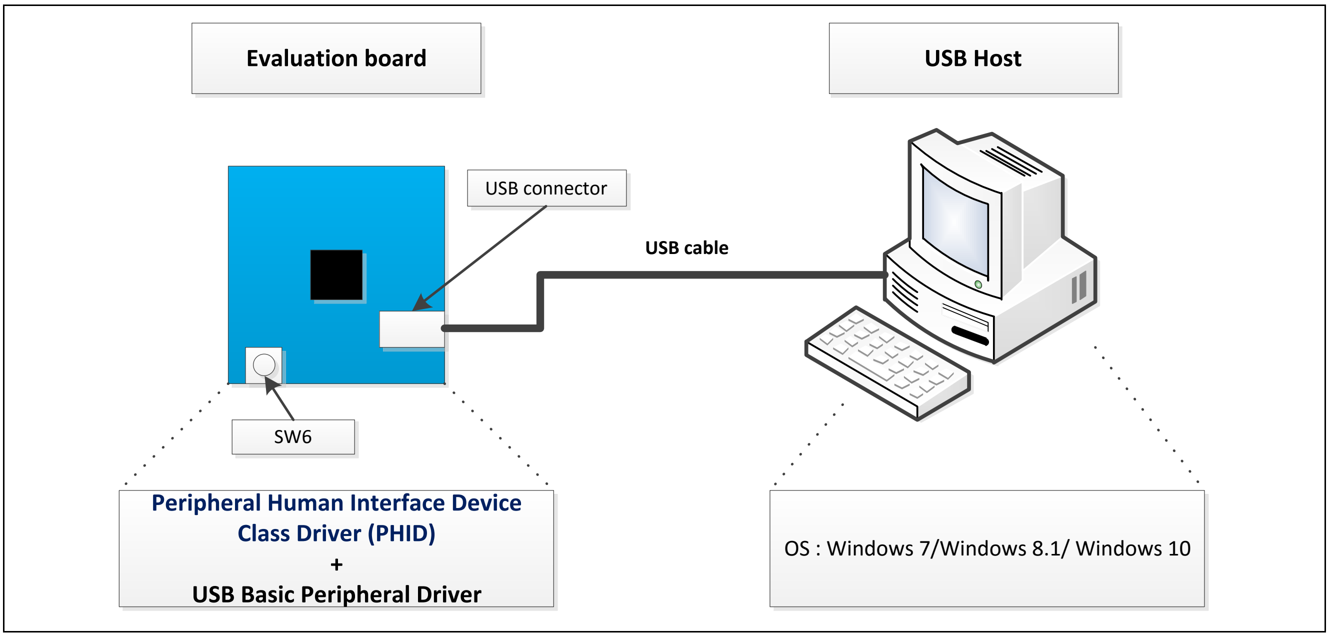
Evaluation board
USB Host
USB connector
USB cable
SW6
Peripheral Human Interface Device Class Driver (PHID)
+
USB Basic Peripheral Driver
OS : Windows 7/Windows 8.1/ Windows 10

Figure 2-1 Example Operating Environment

3. Software Configuration

Figure 3-1 shows the software configuration of this F / W, and Table 3.1 shows the functional overview of each hierarchy.

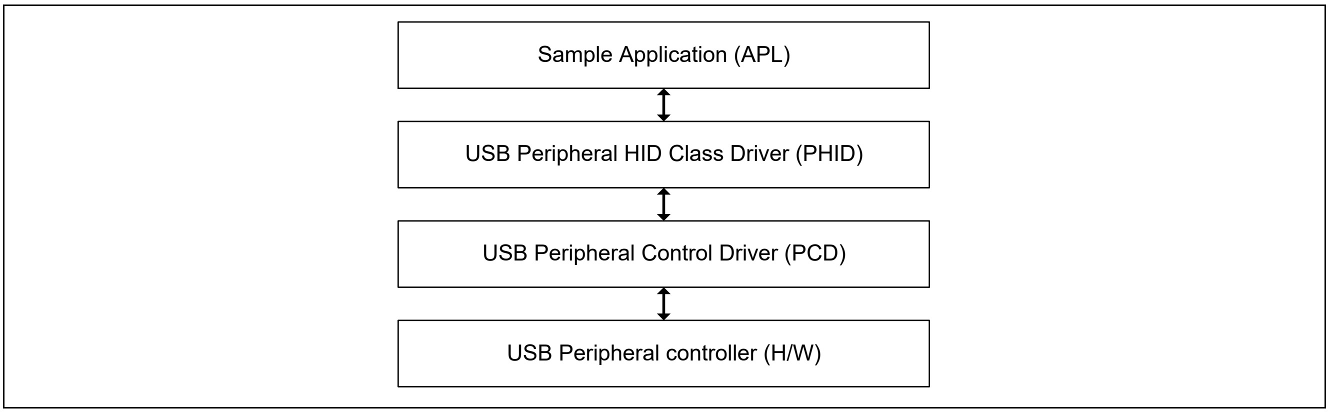
Sample Application (APL)
USB Peripheral HID Class Driver (PHID)
USB Peripheral Control Driver (PCD)
USB Peripheral controller (H/W)

Figure 3-1 Software Configuration

Table 3.1 Functional Overview of Each Hierarchy

Module	Description
APL	USB initial setting Data transfer request and result processing Class request processing
PHID	Data transfer result notification to APL
PCD	Interrupt processing Request analysis from USB Host Device state management Hardware control
H/W	hardware

4. Compile Setting

In order to use PHID, it is necessary to set the USB-BASIC-F/W FIT module as a peripheral.

Refer to USB Basic Firmware application note (Document No. R01AN3291JEJ) for information on USB-BASIC-F/W settings.

Please modify r_usb_phid_config.h when user sets the module configuration option.

The following table shows the option name and the setting value.

Configuration options in r_usb_phid_config.h	
USB_CFG_PHID_INT_IN	Specifies the pipe number which is used at the data transfer (Interrupt In). Specifies any one from USB_PIPE6 to USB_PIPE9
USB_CFG_PHID_INT_OUT	Specifies the pipe number which is used at the data transfer (Interrupt Out). Not used in this project. Specify the USB_NULL.

5. USB Peripheral Human Interface Devices Class Driver (PHID)

5.1 Basic Functions

The software presented here conforms to the HID Class specification

This software has the following main functions.

1. Response to class requests from the HID host
2. INPUT Report descriptor transfers to USB Host due to user input action on board.

5.2 HID Class Overview

5.2.1 Class Requests (Host to Peripheral)

Table 5.1 shows the class request correspondence status of this F / W.

Table 5.1 HID class requests

Request	Code	Description	Supported
Get_Report	0x01	Receives a report from the HID host	No
Set_Report	0x09	Sends a report to the HID host	Yes
Get_Idle	0x02	Receives a duration (time) from the HID host	Yes
Set_Idle	0x0A	Sends a duration (time) to the HID host	No
Get_Protocol	0x03	Reads a protocol from the HID host	No*
Set_Protocol	0x0B	Sends a protocol to the HID host	No*
Get_Descriptor Descriptor Type : Class Class Descriptor Type : Report	0x06 (Standard)	Transmits a report descriptor	Yes
Get_Descriptor Descriptor Type : Class Class Descriptor Type : HID	0x06 (Standard)	Transmits an HID descriptor	Yes

6. Sample Application (APL)

6.1 Application Specifications

The main functionalities of this F/W sample application (APL) are described below.

1. Keyboard Mode: Keyboard Functionality
   When the evaluation board connects to the USB host, the USB host recognizes the evaluation board as a keyboard. The evaluation board operates as a keyboard, transmitting keyboard data to the USB host using interrupt IN transfer. Also, in the suspend state, by pressing the switch on the evaluation board, a remote wakeup signal is transmitted to the USB host.

[Note]

1. Keyboard mode enable USB communication with PCs (USB hosts) supporting OSes such as Windows 7, Windows 8.1 and Windows 10.

6.2 Configuration File for the application program (r_usb_phid_apl_config.h)

The operation setting of the application is set in r_usb_phid_apl_config.h. Below are the setting items to be done in r_usb_phid_apl_config.h.

1. USE_USBIP
   Specify the module number of the USB module you are using. Specify one of the following settings for the USE_USBIP definition.
   Please set this setting to the setting of USB_CFG_USE_USBIP done in r_usb_basic_config.h.

```
#define     USE_USBIP              USE_IP0            // Specify USB_IP0
#define     USE_USBIP              USE_IP1            // Specify USB_IP1
```

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

```
#define     OPERATION_MODE     HID_KEYBOARD     // Keyboard mode
```

3. Note
   R_usb_phid_apl_config.h is a configuration setting for application programs. In addition to the above settings, configuration settings of the USB driver are required. For the USB driver configuration settings, refer to "4. Compile Setting".

6.3 Application Processing

The APL consists of two parts: processing of initial settings and the main loop. Figure 6-1 shows the outline of APL processing.

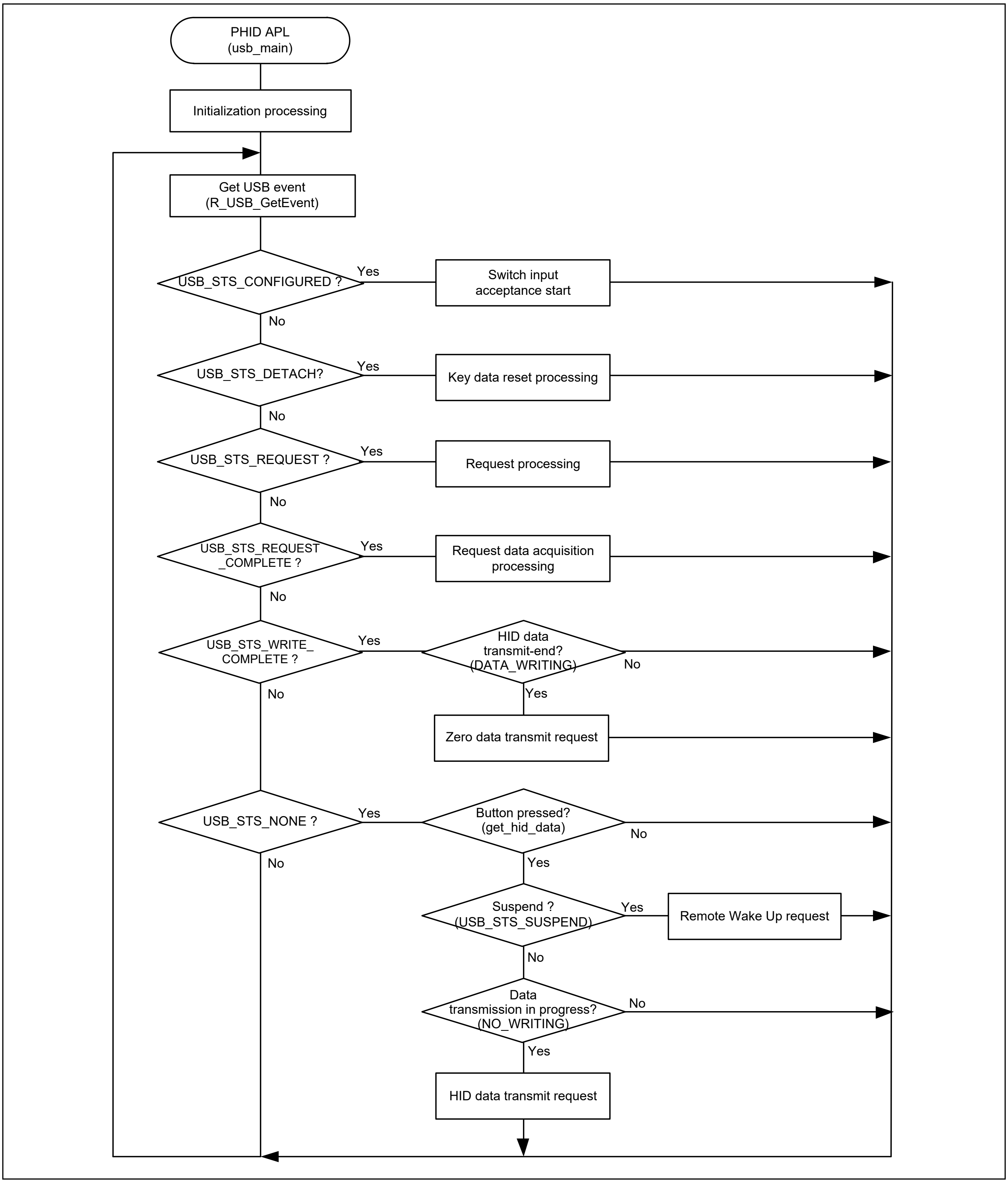
PHID APL
(usb_main)
Initialization processing
Get USB event
(R_USB_GetEvent)
USB_STS_CONFIGURED ?
Yes
Switch input acceptance start
No
USB_STS_DETACH?
Yes
Key data reset processing
No
USB_STS_REQUEST ?
Yes
Request processing
No
USB_STS_REQUEST _COMPLETE ?
Yes
Request data acquisition processing
No
USB_STS_WRITE_ COMPLETE ?
Yes
HID data transmit-end? (DATA_WRITING)
No
Yes
Zero data transmit request
No
USB_STS_NONE ?
Yes
Button pressed? (get_hid_data)
No
Yes
No
Suspend ? (USB_STS_SUSPEND)
Yes
Remote Wake Up request
No
Data transmission in progress? (NO_WRITING)
No
Yes
HID data transmit request

Figure 6-1 Overview of APL processing

6.3.1 Initial Setting

In the initial setting part, the initial setting of the USB controller and the initialization of the application program are performed.

6.3.2 Main Loop

In keyboard mode, processing is performed to send information about the switch (buttons) on the evaluation board to the USB host. When the evaluation board (HID device) is connected to the USB host (PC), the evaluation board is recognized as a keyboard, and the main loop consists of a main routine that sends the switch (button) information to the USB host as key input data. For details of the switch (button) information (key input data), refer to 6.4.1 Switch Specification.

1. USB_STS_CONFIGURED

   After enumeration with the USB host is completed, calling the R_USB_GetEvent function sets USB_STS_CONFIGURED as the return value.

   APL sets NO_WRITING as the state management variable for data transmission and accepts switch input.
2. USB_STS_DETACH

   USB_STS_DETACH is set in the return value when calling R_USB_GetEvent function after disconnecting from the USB Host.

   APL performs key data reset processing.
3. USB_STS_REQUEST

   After receiving the HID class request from the USB Host, calling the R_USB_GetEvent function sets USB_STS_REQUEST as the return value.

   Upon confirmation of USB_STS_REQUEST, APL analyzes the received class request and performs processing corresponding to that class request.
4. USB_STS_REQUEST_COMPLETE

   After class request processing is completed, calling the R_USB_GetEvent function sets USB_STS_REQUEST_COMPLETE as the return value.

   APL performs request information setting processing, etc.
5. USB_STS_NONE

   When the R_USB_GetEvent function is called with no USB related event occurring, USB_STS_NONE is set as the return value.

   APL confirms pressing of the switch on the evaluation board. If the switch is pressed, check the status of the HID device.

   When the HID device is suspended, it transmits a remote wakeup signal to the USB host.

   When the HID device is in the configuration guard state, refer to the state management variable and check if data transmission is in progress. If it is not transmitting, set DATA_WRITING to the state management variable, then call the R_USB_Write function and transmit the pressed switch as key data. If data is being transmitted, key information will not be sent until transmission is completed.
6. USB_STS_WRITE_COMPLETE

   When the R_USB_GetEvent function is called after the key data transmission is completed, USB_STS_WRITE_COMPLETE is set as the return value.

   APL checks the value of the state management variable.

   When the state management variable is DATA_WRITING, after setting ZERO_WRITING in the state management variable, call the R_USB_Write function to request the USB host that the key input has been released and request transmission of zero data (8 bytes) .

   If the state management variable is not DATA_WRITING, you can send key data again by setting NO_WRITING to the state management variable.

6.4 Keyboard operation

In APL the switch (buttons) on the evaluation board are used to make the evaluation board operate as an HID device. The switch (button) input information is used as keyboard key data.

6.4.1 Switch Specification

Table 6.1 shows the switch specifications used in APL. Note this switch specification is recognized when the switch is pressed, and generates the key code. After that, whenever released → pressed, pressing of the switch is recognized and the next key code is generated.

Table 6.1 Switch specification

Switch Number	Operation
Switch6 (SW6)	One of the key codes for characters “a” to “z” or “Enter” is reported to the host each time SW is pressed.

6.5 Data Format

Table 6.2 shows the format of the data transferred to the USB Host. This data format is set together with the contents of the HID report descriptor being transferred to the USB Host.

Table 6.2 Data Formats Used when Notifying the Host

offset	Keyboard Mode(8Bytes)
0	Modifier keys
1	Reserved
2	Keycode 1
3	Keycode 2
4	Keycode 3
5	Keycode 4
6	Keycode 5
7	Keycode 6

6.6 Descriptor

The PHID's descriptor information is contained in r_usb_phid_descriptor.c.

Note

1. Please be sure to use your vendor ID.

Table 6.3 shows the descriptor list of this F / W.

Table 6.3 Descriptor list

Descriptor	Variable
Device Descriptor	g_apl_device
Configuration Descriptor	g_apl_configuration (note 1)
Interface Descriptor	g_apl_configuration (note 1)
HID Descriptor	g_apl_configuration (note 1)
Endpoint Descriptor	g_apl_configuration (note 1)
String Descriptor	g_apl_string0
	g_apl_string1
	g_apl_string2
	g_apl_string3
	g_apl_string4
Report Descriptor	g_apl_report

note

1. g_apl_configuration includes Configuration Descriptor, Interface Descriptor, HID Descriptor, Endpoint Descriptor.

6.6.1 g_apl_device

Table 6.4 shows the Device Descriptor setting values.

Table 6.4 Device Descriptor

Offset	Field	Value	Remarks
0	bLength	USB_DD_BLENGTH	
1	bDescriptorType	USB_DT_DEVICE	
2	bcdUSB	USB_BCDNUM	Low : 1Byte
3		USB_BCDNUM	High : 1Byte
4	bDeviceClass	0x00	
5	bDeviceSubClass	0x00	
6	bDeviceProtocol	0x00	
7	bMAXPacketSize	USB_DCPMAXP	
8	idVendor	USB_VENDORID	Low : 1Byte
9		USB_VENDORID	High : 1Byte
10	idProduct	USB_PRODUCTID	Low : 1Byte
11		USB_PRODUCTID	High : 1Byte
12	bcdDevice	USB_RELEASE	Low : 1Byte
13		USB_RELEASE	High : 1Byte
14	iManufacturer	0x01	g_apl_string1[]
15	iProduct	0x02	g_apl_string2[]
16	iSerialNumber	0x03	g_apl_string3[]
17	bNumConfigurations	USB_CONFIGNUM	

6.6.2 g_apl_configuration

g_apl_configuration includes Configuration Descriptor, Interface Descriptor, HID Descriptor, Endpoint Descriptor. Table 6.5 shows the description of each descriptor.

Table 6.5 g_apl_configuration

Offset	Descriptor
0	Configuration Descriptor
9	Interface Descriptor
18	HID Descriptor
27	Endpoint Descriptor

Table 6.6 Configuration Descriptor

Offset	Field	Value	Remarks
0	bLength	USB_CD_BLENGTH	
1	bDescriptorType	USB_DT_CONFIGURATION	
2	wTotalLength	CD_LEN % 256	Low : 1Byte
3		CD_LEN / 256	High : 1Byte
4	bNumInterfaces	0x01	
5	bConfigurationValue	0x01	
6	iConfiguration	0x04	
7	bmAttributes	USB_CF_RESERVED | USB_CF_RWUPON	USB_CF_RESERVED: 1 fixed USB_CF_RWUPON: Remote wakeup enabled
8	bMaxPower	100 / 2	100mA (50 * 2)

Table 6.7 Interface Descriptor

Offset	Field	Value	Remarks
0	bLength	USB_ID_BLENGTH	
1	bDescriptorType	USB_DT_INTERFACE	
2	bInterfaceNumber	0x00	
3	bAlternateSetting	0x00	
4	bNumEndpoints	NUM_EP	
5	bInterfaceClass	USB_IFCLS_HID	HID
6	bInterfaceSubClass	USB_IFSUB_NOBOOT	NonBOOT
7	bInterfaceProtocol	USB_IFPROTOCOL	
8	iInterface	0x00	

Table 6.8 HID Descriptor

Offset	Field	Value	Remarks
0	bLength	0x09	
1	bDescriptorType	USB_DT_TYPE_HIDDESCRIPTOR	
2	bcdHID	0x00	Low : 1Byte
3		0x01	High : 1Byte
4	bCountryCode	0x00	
5	bNumDescriptors	0x01	
6	bDescriptorType	0x22	
7	wItemLength	ITEM_LEN	Low : 1Byte
8		ITEM_LEN	High : 1Byte

Table 6.9 Endpoint Descriptor

Offset	Field	Value	Remarks
0	bLength	USB_ED_BLENGTH	
1	bDescriptorType	USB_DT_ENDPOINT	
2	bEndpointAddress	USB_EP_IN | USB_EP1	b'7 : Direction b'6 – b'4: Reserved b'3 – b'0: Endpoint number
3	bmAttribute	USB_EP_INT	b'5 – b'4: Usage type b'3 – b'2: Synchronization Type b'1 – b'0: Transfer type
4	wMaxPacketSize	MXPS	Low : 1Byte
5		MXPS	High : 1Byte
6	bInterval	0x0A	

6.6.3 g_apl_stringX (X = 0 – 4)

The setting values of String Descriptor are shown in Table 6.10 to Table 6.14.

Table 6.10 String Descriptor Zero (g_apl_string0)

Offset	Field	Value	Remarks
0	bLength	4	
1	bDescriptorType	USB_DT_STRING	
2	wLANGID[0]	0x09	Low : 1Byte
3		0x04	High : 1Byte

Table 6.11 UNICODE String Descriptor (g_apl_string1)

Offset	Field	Value	Remarks
0	bLength	16	
1	bDescriptorType	USB_DT_STRING	
2	bString	'R'	
3		0x00	
4		'e'	
5		0x00	
6		'n'	
7		0x00	
8		'e'	
9		0x00	
10		's'	
11		0x00	
12		'a'	
13		0x00	
14		's'	
15		0x00	

Table 6.12 UNICODE String Descriptor (g_apl_string2)

Offset	Field	Value	Remarks
0	bLength	32	
1	bDescriptorType	USB_DT_STRING	
2	bString	'U'	
3		0x00	
4		'S'	
5		0x00	
6		'B'	
7		0x00	
8		' '	
9		0x00	
10		'P'	
11		0x00	
12		'e'	
13		0x00	
14		'r'	
15		0x00	
16		'i'	
17		0x00	
18		' '	
19		0x00	
20		'H'	
21		0x00	
22		'I'	
23		0x00	
24		'D'	
25		0x00	
26		' '	
27		0x00	
28		'F'	
29		0x00	
30		'W'	
31		0x00	

Table 6.13 UNICODE String Descriptor (g_apl_string3)

Offset	Field	Value	Remarks
0	bLength	10	
1	bDescriptorType	USB_DT_STRING	
2	bString	'P'	
3		0x00	
4		'0'	
5		0x00	
6		'0’	
7		0x00	
8		'1'	
9		0x00	

Table 6.14 UNICODE String Descriptor (g_apl_string4)

Offset	Field	Value	Remarks
0	bLength	24	
1	bDescriptorType	USB_DT_STRING	
2	bString	'F'	
3		0x00	
4		'S'	
5		0x00	
6		' '	
7		0x00	
8		'K'	
9		0x00	
10		'e'	
11		0x00	
12		'y'	
13		0x00	
14		'B'	
15		0x00	
16		'o'	
17		0x00	
18		'a'	
19		0x00	
20		'r'	
21		0x00	
22		'd'	
23		0x00	

6.6.4 g_apl_report

The setting values of Report Descriptor are shown in Table 6.15.

Table 6.15 Report Descriptor

Item	Value (Hex)
Usage Page (Generic Desktop)	0x05, 0x01,
Usage (Keyboard)	0x09, 0x06,
Collection (Application)	0xA1, 0x01,
Usage Page (Key Codes)	0x05, 0x07,
Usage Minimum (224)	0x19, 0xE0,
Usage Maximum (231)	0x29, 0xE7,
Logical Minimum (0)	0x15, 0x00,
Logical Maximum (1)	0x25, 0x01,
Report Size (1)	0x75, 0x01,
Report Count (8)	0x95, 0x08,
Input (Data , Variable , Absolute)	0x81, 0x02,
Report Count (1)	0x95, 0x01,
Report Size (8)	0x75, 0x08,
Input (Constant)	0x81, 0x01,
Report Count (5)	0x95, 0x05,
Report Size (1)	0x75, 0x01,
Usage Page (Page# for LEDs)	0x05, 0x08,
Usage Minimum (1)	0x19, 0x01,
Usage Maximum (5)	0x29, 0x05,
Output (Data , Variable , Absolute)	0x91, 0x02,
Report Count (1)	0x95, 0x01,
Report Size (3)	0x75, 0x03,
Output (Constant)	0x91, 0x01,
Report Count (6)	0x95, 0x06,
Report Size (8)	0x75, 0x08,
Logical Minimum (0)	0x15, 0x00,
Logical Maximum(101)	0x25, 0x65,
Usage Page (Key Codes)	0x05, 0x07,
Usage Minimum (0)	0x19, 0x00,
Usage Maximum (101)	0x29, 0x65,
Input (Data , Array)	0x81, 0x00,
Usage ID within this page (Vendor defined)	0x09, 0x00,
Logical Minimum (0)	0x15, 0x00,
Logical Maximum(255)	0x26, 0xFF, 0x00,
Size 8 Bits (Each Field will be 8bits)	0x75, 0x08,
Count (Number of fields(bytes) in OUTPUT report)	0x95, 0x01,
Output Report - type variable data	0x91, 0x02,
End Collection	0xC0,

Website and Support

Renesas Electronics Website

http://www.renesas.com/

Inquiries

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

Revision History

Rev.	Date	Description	
		Page	Summary
1.20	Feb.28.17	-	First edition issued

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

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, HAL II 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 6.0