To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

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

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. 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.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. 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.
5. When exporting the 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. You should not use Renesas Electronics products or the 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. 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.
6. 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.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. 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 categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. 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 an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "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.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. 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.
9. 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 system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. 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.
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.

SH7216 Group

USB Function Module: USB HID Class

Introduction

This application note describes how to use the USB function module of the SH7216 and examples of creation of firmware conforming to the USB Human Interface Devices (HID) Class.

The contents of this application note and the software are provided for describing application examples of the USB function module, but not for ensuring the contents.

Target Device

SH7216

Contents

1. Preface

1.1 Specifications

This application note describes how to use the USB function module of the SH7216 and examples of creation of firmware conforming to the USB Human Interface Devices (HID) Class.

1.2 Modules Used

- Interrupt controller (INTC)
- Multi function timer pulse unit 2 (MTU2)
- Pin function controller (PFC)
- USB function module (USB)

1.3 Applicable Conditions

MCU	SH7216
Operating frequency	Internal clock: 200 MHz Bus clock: 50 MHz Peripheral clock: 50 MHz
Integrated development environment	High-performance Embedded Workshop Ver.4.07.00 (from Renesas Technology Corp.)
C compiler	SuperH RISC engine Family C/C++ Compiler Package Ver.9.02 Release00, available from Renesas Technology
Compiler options	Default settings of the High-performance Embedded Workshop (-cpu=sh2afpu -pic=1 -object="$(CONFIGDIR)¥$(FILELEAF).obj" -debug -gbr=auto -chgincpath -errorpath -global_volatile=0 -opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1 -nologo)

1.4 Related Application Note

SH7216 Group Application Note: USB Function Module: USB Mass Storage Class (REJ06B0897)

2. Overview

This program performs control transfer, interrupt transfer, and processing for HID Class commands that use the USB function module (USB).

The features of the USB Function Module contained in the SH7216 are listed below.

- Automatic processing of USB protocol
- Automatic processing of USB standard commands for endpoint 0 (some commands need to be processed through the firmware)
- Full-speed transfer supported
- Interrupt request: Various interrupt signals needed for USB transmission and reception are generated.
- Clock: External input clock generated by the USB oscillator (48 MHz)
- Low power consumption mode provided.
- Internal USB transceiver
- Endpoint configurations: The configurations are indicated in table 1.

Table 1 Endpoint Configurations

Endpoint Name	Name	Transfer Type	Max. Packet Size	FIFO Buffer Capacity	DMA Transfer
Endpoint 0	EP0s	Setup	8 bytes	8 bytes	—
	EP0i	Control In	16 bytes	16 bytes	—
	EP0o	Control Out	16 bytes	16 bytes	—
Endpoint 1	EP1	Bulk-in	64 bytes	64 × 2 (128) bytes	Possible
Endpoint 2	EP2	Bulk-out	64 bytes	64 × 2 (128) bytes	Possible
Endpoint 3	EP3	Interrupt In	16 bytes	16 bytes	—
Endpoint 4	EP4	Bulk-in	64 bytes	64 × 2 (128) bytes	Possible
Endpoint 5	EP5	Bulk-out	64 bytes	64 × 2 (128) bytes	Possible
Endpoint 6	EP6	Interrupt In	16 bytes	16 bytes	—
Endpoint 7	EP7	Bulk-in	64 bytes	64 bytes	—
Endpoint 8	EP8	Bulk-out	64 bytes	64 bytes	—
Endpoint 9	EP9	Interrupt In	16 bytes	16 bytes	—

Figure 1 shows an example of a system configuration.

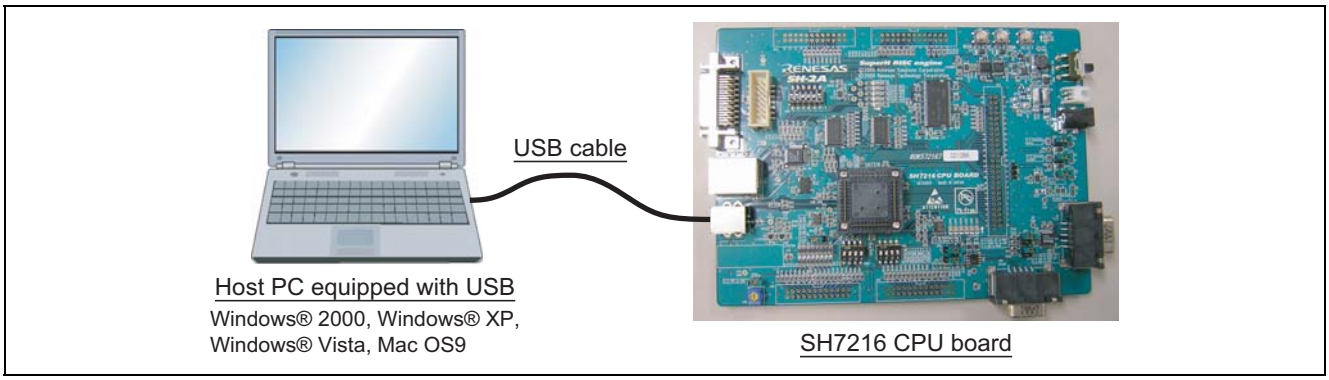


Figure 1 System Configuration Example

This system is configured of the SH7216 CPU Board made by Renesas Technology Corp. and a PC containing Windows® 2000/Windows® XP/Windows® Vista or Mac OS9 operating system.

This system is an HID class firmware that automatically generates pseudo mouse data on the SH7216 CPU Board and outputs the mouse data (hereafter called the HID data) to the host PC through the USB.

It is also possible to use the USB HID Class device driver that comes as an accessory with the operating systems listed above.

This system offers the following features.

1. The sample program can be used to evaluate the USB module of the SH7216.
2. The sample program supports USB control transfer and interrupt transport.
3. This system can be debugged with the E10A (USB Emulator).
4. Additional programs can be created to support bulk transfer.*

Note: * Bulk transfer programs are not provided, and will need to be created by the user.
The SH7216 does not support isochronous transfer.

3. Overview of the USB HID Class

This section describes the USB Human Interface Devices (HID) Class.

We hope that it will provide a convenient reference for use when developing USB HID class devices. For more detailed information on standards, please see (3) and (4) of the section 6 "Documents for Reference".

3.1 USB HID Class

USB HID class is a class of standards that apply to devices through which humans operate PCs. Typical examples include mouse devices, keyboards, and joysticks.

To notify the host PC of this class of function, the bInterfaceClass filed of the Interface descriptor must be H'03.

3.2 Sub-Class Code

Subclasses were intended to be used to identify the specific protocols of different HID class devices. However, as there are many types of devices used by humans, subclass protocol definitions are impractical, and subclasses are not used to define most protocols in the HID class. Instead, the protocol is identified by the Report descriptor in HID class devices.

As for BIOS-support devices (boot devices), a simple method to identify the protocol is needed. For this purpose, subclasses are used to indicate devices that support the predefined protocol (boot protocol) for mouse devices or keyboards (that is, devices that can be used for boot devices).

To notify the host PC that the device supports the boot protocol, the bInterfaceSubClass filed of the Interface descriptor must be H'01.

3.3 Protocol Code

When a device supports the boot protocol (subclass code other than 0), a protocol code is used to indicate the device type. The protocol code is H'01 for a keyboard, and H'02 for a mouse. Specifying the device type by the protocol code indicates that the device can use the protocol for the device type.

To notify the host PC of the device type, the bInterfaceProtocol filed of the Interface descriptor must be a value corresponding to the device type.

3.4 Descriptors for HID Class

HID class function devices need an HID descriptor, a Report descriptor, and a Physical descriptor (optional) in addition to descriptor information that other USB function devices need. Figure 2 shows the HID device descriptor configuration.

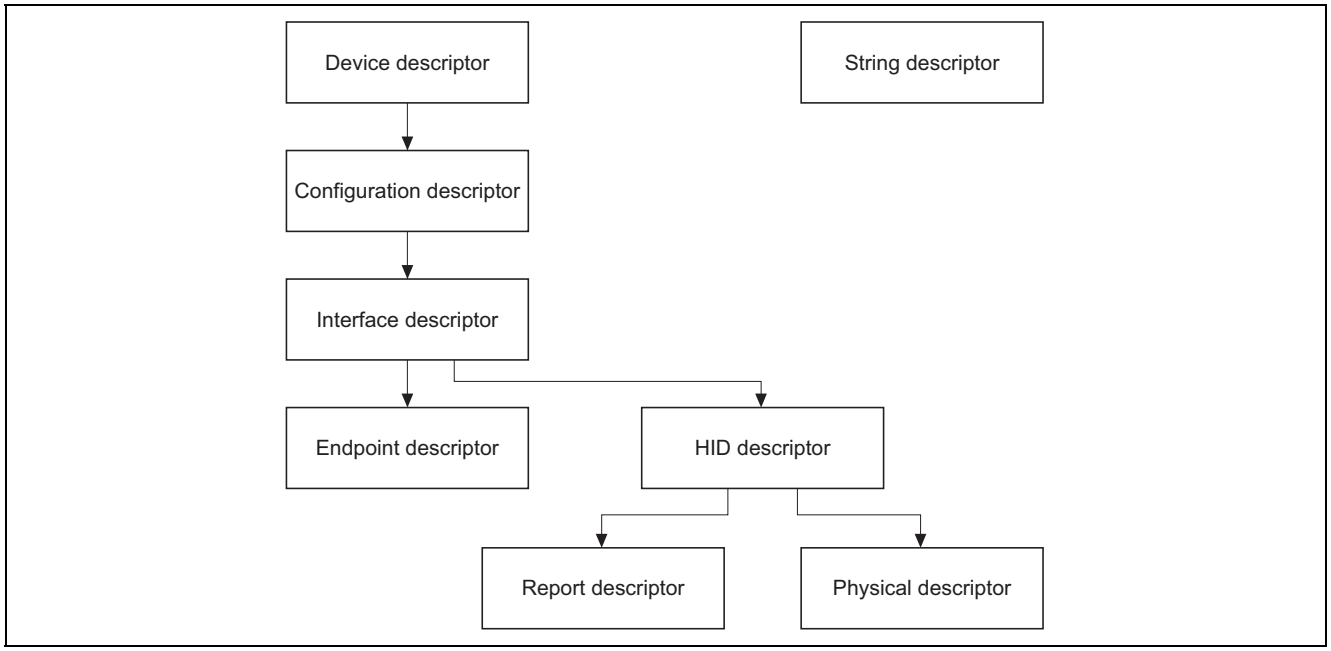


Figure 2 Descriptor Configuration

3.5 HID Descriptor

The HID descriptor combines the Report descriptor and Physical descriptor (optional). Table 2 shows the format of the HID descriptor.

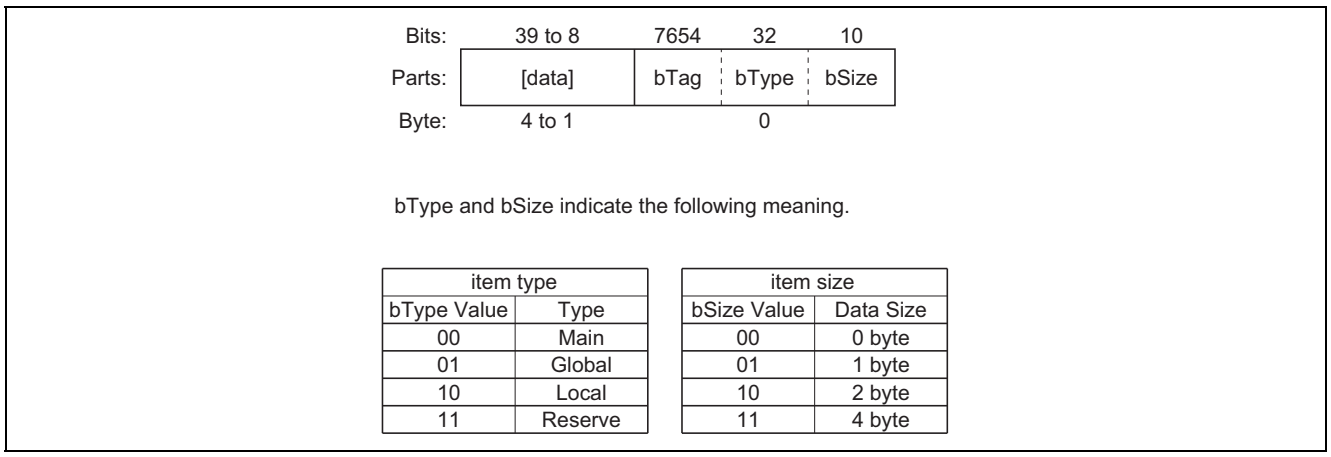
Table 2 HID Descriptor

Field	Size (bytes)	Description
bLength	1	Descriptor size (fixed to H'09)
bDescriptorType	1	Descriptor type (fixed to H'21)
bcdHID	2	HID version in BCD
bCountryCode	1	Country ID for devices specific to a particular country (0 unless necessary)
bNumDescriptors	1	Number of class descriptors
bDescriptorType	1	Type of class descriptor (H'22 for HIDREPORT)
wDescriptorLength	2	Size of Report descriptor

3.6 Report Descriptor

The Report descriptor specifies the format of data to be transferred between the host PC and the device. Unlike other descriptors, the Report descriptor has no standardized format, but the length and contents of the Report descriptor vary depending on the device's report or the number of data fields required for the device's report.

The Report descriptor consists of items that provide information about the device. There are two types of items, short and long items. The following describes the short item.



item type	
bType Value	Type
00	Main
01	Global
10	Local
11	Reserve

item size	
bSize Value	Data Size
00	0 byte
01	1 byte
10	2 byte
11	4 byte

Figure 3 Report Descriptor Item

An item consists of four fields: data, item tag, item type, and itemSize. The item uses these fields to indicate the information.

There are three item types: Main, Global, and Local. The Main item type (defining or grouping the data fields in a Report descriptor) has five types of item tags, the Global item type (describing data) has 12, and the Local item type (defining the characteristics) has ten.

By combining these item tags, the Report descriptor specifies the format of data to be transferred between the host PC and the device.

3.6.1 Main Items

Table 3 shows five item tags for the Main item type.

Table 3 Item Tags for Main Item Type

Item Tag	bTag	bType	bSize	Description
Input	1000	00	nn	Describes information about data provided by one or more physical controls
Output	1001	00	nn	Defines output data field
Feature	1011	00	nn	Describes device configuration information that can be sent to the device
Collection	1010	00	nn	Starts collecting relations between two or more data item tags (Input, Output, or Feature)
End Collection	1100	00	nn	Ends collecting relations between two or more data item tags (Input, Output, or Feature) in response to Collection

(1) Input Item Tag

The input item tag has eight parameters (data fields), which are set in 1-bit units, as shown in table 4.

Table 4 Input Item Tag Parameters

Bit	Value	Contents	Description
0	0	Data	The item reports data
	1	Constant	The item reports a constant
1	0	Array	The item reports an array data field
	1	Variable	The item reports a variable
2	0	Absolute	The item reports an absolute value
	1	Relative	The item reports a relative value from the last report
3	0	No Wrap	The value reported by the item does not roll over
	1	Wrap	The value reported by the item rolls over (for example, for a dial to output a value from 0 to 10, if dialing is continued, 0 is output after 10)
4	0	Linear	The item reports the state of the target control linearly
	1	Non Linear	The item processes raw data and does not report the state of the target linearly
5	0	Preferred State	The item has a state to which it returns when it is not controlled by the user
	1	No Preferred	The item does not have a state to which it returns when it is not controlled by the user
6	0	No Null position	The item has a state in which it does not send meaningful data
	1	Null state	The item does not have a state in which it does not send meaningful data
7	0	Reserved	Reserved
8	0	Bit Field	The item issues a bit field
	1	Buffered Bytes	The item issues a stream fixed to 1-byte size
9-31	0	Reserved	Reserved

(2) Output and Feature Item Tags

The output and feature item tags have nine parameters (data fields), which are the same as the input item tag except bit 7, as shown in table 5.

Table 5 Output and Feature Item Tag Parameters

Bit	Value	Contents	Description
1-6	—	—	Same as the input item tag
7	0	Non Volatile	The item value cannot change with or without host interactions
	1	Volatile	The item value can change with or without host interactions
8-31	—	—	Same as the input item tag

(3) Collection Item Tag

The collection item tag has eight parameters (data fields), which are set in one byte, as shown in table 6.

Table 6 Collection Item Tag Parameters

Value	Contents	Description
H'00	Physical	Used for data items collected into one. This is used for devices which need to associate correct or sensed data with a single point. It does not indicate that data comes from a single device such as a keyboard. It indicates that the device reports multiple sensor positions and data comes from different sensors.
H'01	Application	Identifies the Usage only used for the application level. It indicates that the collection is a functionally subordinate group of an HID device or a complex device. The operating system uses the Usage associated with this collection to link to the application or driver that controls the device.
H'02	Logical	Used when data items compose a composite data structure.
H'03	Report	Defines a logical collection that includes all fields. A report ID is included in this collection. An application can easily determine whether to support a certain function of the device.
H'04	Named Array	Used when data items compose a composite data structure and it is named.
H'05	Usage Switch	A logical collection that modifies the meaning of the included Usage. It identifies the Usage applied for logical collection to modify the purpose of the Usage being collected.
H'06	Usage Modifier	Modifies the meaning of the Usage attached to the including collection. The Usage typically defines a single operating mode for control, which enables the operating method of control to be expanded.
H'07-H'7F	Reserved	Reserved.
H'80-H'FF	Vendor-defined.	Defined by the vendor.

3.6.2 Global Items

Table 7 shows 12 item tags for the Global item type.

Table 7 Item Tags for Global Item Type

Item Tag	bTag	bType	bSize	Description
Usage Page	0000	01	nn	A value specifying the current Usage Page. It defines the index to the item usage.
Logical Minimum	0001	01	nn	The minimum value to be reported by a variable or array item. For example, the mouse that reports an X position value from 0 to 128 will have a minimum logical value of 0.
Logical Maximum	0010	01	nn	The maximum value to be reported by variable or array items. For example, the mouse that reports an X position value from 0 to 128 will have a maximum logical value of 128.
Physical Minimum	0011	01	nn	Minimum value of physical range for a variable item
Physical Maximum	0100	01	nn	Maximum value of physical range for a variable item
Unit Exponent	0101	01	nn	Unit exponent in base 10
Unit	0110	01	nn	Unit value
Report Size	0111	01	nn	Unsigned value that specifies the report field size in bits
Report ID	1000	01	nn	Unsigned value that specifies the report ID
Report Count	1001	01	nn	Specifies the number of data fields for the item. An unsigned integer specifies how many fields can be included in the report for the particular item (accordingly, how many bits are added to the report).
Push	1010	01	nn	Places a copy of the Global Item state table in the stack
Pop	1011	01	nn	Replaces the item state table with the top data in the stack.

3.6.3 Local Items

Table 8 shows 10 item tags for the Local item type.

Table 8 Item Tags for Local Item Type

Item Tag	bTag	bType	bSize	Description
Usage	0000	10	nn	A value specifying the current Usage. It defines the index to the items usage.
Usage Minimum	0001	10	nn	Defines the start of Usage associated with an array or a bitmap.
Usage Maximum	0010	10	nn	Defines the end of Usage associated with an array or a bitmap.
Designator Index	0011	10	nn	Determines the body part used for control.
Designator Minimum	0100	10	nn	Defines the start index to the designator associated with an array or a bitmap.
Designator Maximum	0101	10	nn	Defines the end index to the designator associated with an array or a bitmap.
String Index	0111	10	nn	Index to the String descriptor, which enables the string to be associated with a particular item or control
String Minimum	1000	10	nn	Specifies the first string index when associating a group of sequential strings to the control in an array or a bitmap.
String Maximum	1001	10	nn	Specifies the end string index when associating a group of sequential strings to the control in an array or a bitmap.
Delimiter	1010	10	nn	Defines the start or end of a set of Local items.

3.6.4 Sample Report Descriptor

Figure 4 shows the Report descriptor of this sample program.

```
Usage Page (Generic Desktop),                                   : 05  01
Usage (Mouse),                                                  : 09  02
Collection (Application),                                       : A1  01
    Usage (Pointer),                                            : 09  01
    Collection (Physical),                                      : A1  00
        Usage Page (Buttons),                                   : 05  09
        Usage Minimum (01),                                     : 19  01
        Usage Maximum (03),                                     : 29  03
        Logical Minimum (0),                                    : 15  00
        Logical Maximum (1),                                    : 25  01
        Report Count (3),                                       : 95  03
        Report Size (1),                                        : 75  01
        Input (Data, Variable, Absolute), ; 3 button bits       : 81  02
        Report Count (1),                                       : 95  01
        Report Size (5),                                        : 75  05
        Input (Constant), ; 5 bit padding                       : 81  01
        Usage Page (Generic Desktop),                           : 05  01
        Usage (X),                                              : 09  30
        Usage (Y),                                              : 09  31
        Usage (Wheel),                                          : 09  38
        Logical Minimum (-127),                                 : 15  81
        Logical Maximum (127),                                  : 25  7F
        Report Size (8),                                        : 75  08
        Report Count (3),                                       : 95  03
        Input (Data, Variable, Relative), ; 2 position bytes (X & Y) : 81  06
    End Collection,                                             : C0
End Collection                                                  : C0
```

Figure 4 Report Descriptor

3.6.5 Description of Report Descriptor

Table 9 shows the Report descriptor used by the sample program.

Table 9 Report Descriptor

Item	Value (hex.)	Item Classification	Description
Usage Page (Generic Desktop Control)	H'05 01	Global	A value specifying the Usage Page. H'01 indicates Generic Desktop Control.
Usage (Mouse)	H'09 02	Local	Index to the item Usage. H'02 indicates Mouse. The operating system links the device as a mouse to the active application or driver. The Usage type of Mouse is Collection Application.
Collection (Application)	H'A1 01	Main	Notifies the application of Pointer as a mouse.
Usage (Pointer)	H'09 01	Local	Index to the item Usage. H'01 indicates Pointer. The Usage type of Pointer is Collection Physical.
Collection (Physical)	H'A1 00	Main	Collects multiple sensor positions (button, X axis, Y axis, and rotary control) to one as a pointer.
Usage Page (Button)	H'05 09	Global	A value specifying the Usage Page. H'09 indicates Button.
Usage Minimum (1)	H'19 01	Local	Defines that the Usage associated with an array or a bitmap starts from 1.
Usage Maximum (3)	H'29 03	Local	Defines that the Usage associated with an array or a bitmap ends at 3.
Logical Minimum (0)	H'15 00	Global	The minimum value to be reported by the item is 0.
Logical Maximum (1)	H'25 01	Global	The maximum value to be reported by the item is 1.
Report Count (3)	H'95 03	Global	Indicates the number of data fields to be used for the item. This example indicates that three report fields are to be used.
Report Size (1)	H'75 01	Global	Indicates the report field size. This example indicates that 1-bit field is to be used.
Input (Data, Variable, Absolute)	H'81 02	Main	Indicates the type of input item. This example indicates that the input is variable data and reports an absolute value.
Report Count (1)	H'95 01	Global	Indicates the number of data fields to be used for the item. This example indicates that one report field is to be used.
Report Size (5)	H'75 05	Global	Indicates the report field size. This example indicates that 5-bit field is to be used.
Input (Constant)	H'81 01	Main	Indicates the type of input item. This example indicates that the input reports a constant.
Usage Page (Generic Desktop Control)	H'05 01	Global	A value specifying the Usage Page. H'01 indicates Generic Desktop Control.
Usage (X)	H'09 30	Local	Index to the item Usage. H'30 indicates X. The controller reports X-direction values, and when the controller moves from left to right from the user's viewpoint, a value increases linearly.
Usage (Y)	H'09 31	Local	Index to the item Usage. H'31 indicates Y. The controller reports Y-direction values, and when the controller moves from the far side to the near side from the user's viewpoint, a value increases linearly.
Usage (Wheel)	H'09 38	Local	Index to the item Usage. H'38 indicates Wheel. It is different from a dial; it is a rotary control that generates a variable value when rotated. When the controller rotates toward the front (the far side from the user), a value increases.
Logical Minimum (–127)	H'15 81	Global	The minimum value to be reported by the item is –127.
Logical Maximum (127)	H'25 7F	Global	The maximum value to be reported by the item is 127.
Report Size (8)	H'75 08	Global	Indicates the report field size. This example indicates that 8-bit field is to be used.
Report Count (3)	H'95 03	Global	Indicates the number of data fields to be used for the item. This example indicates that three report fields are to be used.
Input (Data, Variable, Relative)	H'81 06	Main	Indicates the type of input item. This example indicates that the input is variable data and reports the change from the last input.
End Collection	H'C0	Main	Indicates the end of collection of data set (physical).
End Collection	H'C0	Main	Indicates the end of collection of data set (application).

3.7 Physical Descriptor

The physical descriptor provides information about the human body (or a specific part of the human body) that is controlling the device. This descriptor is optional, and it is omitted in the sample program.

3.8 HID Data Transfer Format

HID data is transferred between the host PC and the USB function module mainly through interrupt transfers (control transfers are also available).

The boot device can use two types of protocols: report protocol and boot protocol. Other devices can only use one protocol: report protocol.

The format of data transfer used by the report protocol is described by a Report descriptor. The format used by the boot protocol is prescribed in the USB standard.

The default protocol for the boot device is the report protocol, but a class command can select either the boot or report protocol. Figure 5 shows the report protocol format used by the sample program.

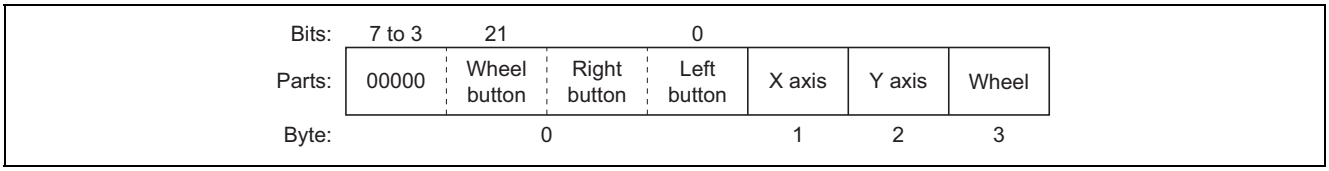


Figure 5 Report Protocol Format

3.9 Class Commands

Class commands are defined by each USB class. They use control transfer.

There are six commands for the USB HID class. Table 10 shows the class commands.

Table 10 Class Commands

bRequest Field Value	Command	Meaning of Command
H'01	GET_REPORT	Transfers HID data from the device to the host PC through control transfer
H'02	GET_IDLE	Returns the current value for the rate of time for which interrupt transfer stops
H'03	GET_PROTOCOL	Reports the current active protocol (boot protocol or report protocol)
H'09	SET_REPORT	Transfers HID data from the host PC to the device through control transfer
H'0A	SET_IDLE	Specifies the rate of time for which interrupt transfer stops
H'0B	SET_PROTOCOL	Specifies the active protocol (boot protocol or report protocol)

Notes: 1. All devices must support GET_REPORT.
2. Boot devices must support GET_PROTOCOL and SET_PROTOCOL.

When the GET_REPORT command is received, the function sends HID data to the host through the data stage of control transfer. The report type must be specified in the upper one byte of the wValue field in the setup data and the report ID in the lower one byte of the wValue field.

When the GET_IDLE command is received, the function returns the time for which interrupt transfer stops. The time should be expressed in time rate in 4-ms units. The host specifies the ID for the report that the host requests in the lower one byte of the wValue field in the setup data. If this value is 0, the time rates for all interrupt transfers of the target device are returned.

When the GET_PROTOCOL command is received, the function returns the current active protocol (boot protocol or report protocol) to the host through the data state of control transfer. Value 0 indicates the boot protocol, and value 1 indicates the report protocol.

When the SET_REPORT command is received, the function receives HID data through the data stage of control transfer. However, the function may ignore the command from the host.

When the SET_IDLE command is received, the function stops interrupt transfer for the time specified in the upper one byte of the wValue field in the setup data. The time is expressed in time rate in 4-ms units. The lower one byte of the wValue field specifies the report ID. If this value is not 0, the transfer of the specified report ID is stopped. If this value is 0, all interrupt transfers of the target device are stopped.

When the SET_PROTOCOL command is received, the function specifies the protocol (boot protocol or report protocol) to be used from that time on. The protocol is specified in the wValue filed in the setup data (value 0 indicates the boot protocol and value 1 indicates the report protocol). Note that the report protocol is the default protocol of the function.

4. Development Environment

This chapter looks at the development environment used to develop this system. The devices (tools) listed below were used when developing the system.

- SH7216 CPU Board (part number: R0K572167) manufactured by Renesas Technology Corp.
- E10A-USB Emulator manufactured by Renesas Technology Corp.
- E10A PC (Windows® 2000, Windows® XP)
- USB host PC (Windows® 2000/Windows® XP/Windows® Vista or Mac OS9)
- USB cable
- High-performance Embedded Workshop 4 (hereafter called the HEW4) manufactured by Renesas Technology Corp.

4.1 Hardware Environment

Figure 6 shows device connections.

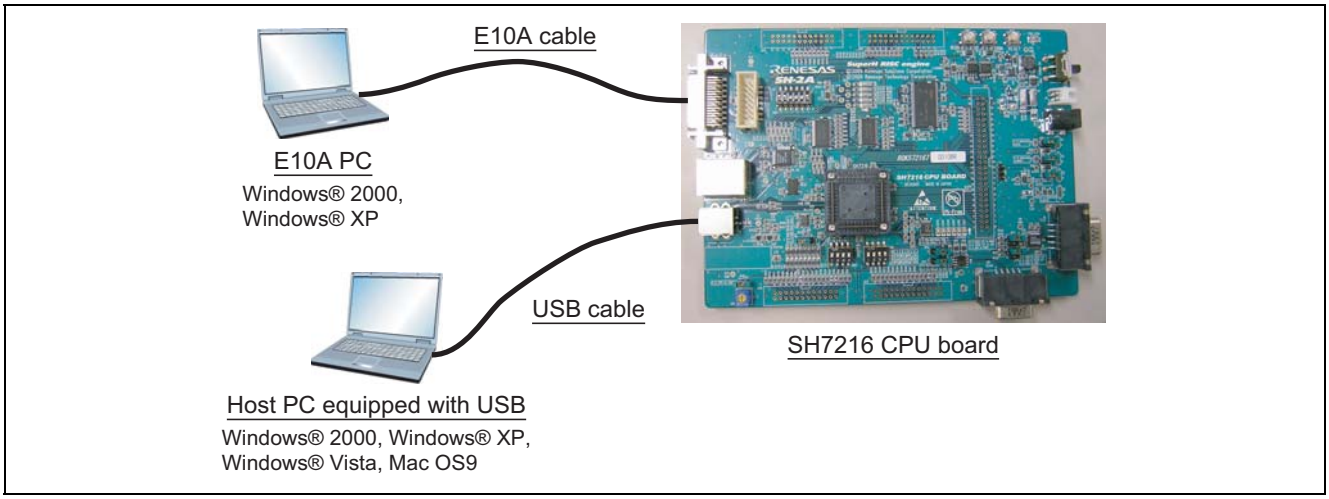


Figure 6 Device Connections

(1) SH7216 CPU Board

Because this system uses the on-chip ROM, the SH7216 CPU board must be operated in a mode that enables the onchip ROM. For example, the CPU board can be operated in MCU expansion mode 2 (both on-chip ROM and SDRAM enabled) by changing DIP switch SW1 on the SH7216 CPU board from the factory setting to the setting shown in table 11. Before turning on the power, ensure that the switches are set as follows. There is no need to change any other DIP switches.

Table 11 DIP Switch Settings

At Time of Shipment (Mode 6)	After Change (Mode 2)	DIP Switch Function
SW1-1 (FWE) OFF	SW1-1 (FWE) ON	On-chip flash memory write/erase protection
SW1-2 (MD1) OFF	SW1-2 (MD1) OFF	MD1 pin state
SW1-3 (MD0) ON	SW1-3 (MD0) ON	MD0 pin state

(2) USB host PC

A PC with Windows® 2000/Windows® XP/Windows® Vista or Mac OS9 installed, and with a USB port, is used as the USB host. This system uses USB HID Class device drivers installed as a standard part of the Windows® XP system, and so there is no need to install new drivers.

(3) E10A PC

A PC with Windows® 2000/Windows® XP installed, and with a USB port, is used as the E10A PC. Connect the E10A-USB emulator to the USB connector on the E10A-USB PC, and then connect the E10A-USB emulator to the CPU board with the cable. After connection, start the HEW4 and perform emulation.

4.2 Software Environment

Compile, link, and debug the source code with HEW4. To start HEW4, double-click "HID.hws" in this folder.

4.2.1 Sample Program

Files required for the sample program are all stored in the HID folder. When this entire folder with its contents is moved to a PC on which HEW4 have been installed, the sample program can be used immediately. Files included in the folder are indicated in figure 7 below.

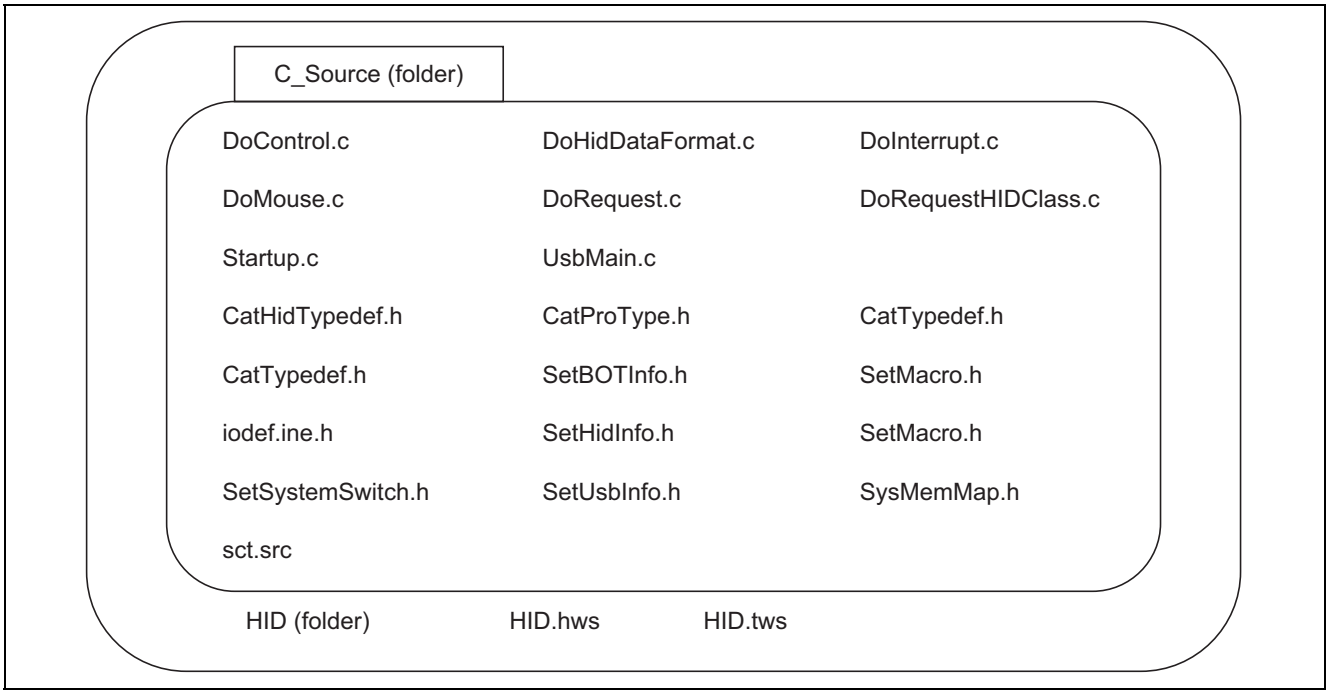


Figure 7 Files Included in the Folder

4.2.2 Compiling and Linking

Compile the source code with HEW4.

4.3 Loading and Executing the Program

The procedure for loading and executing the program is described below.

4.3.1 Loading the Program

In order to load the sample program into the SDRAM of the SH7216 CPU Board, the following procedure is used.

- Connect the E10A-USB to the E10A-USB PC with HEW4 installed.
- Connect the E10A-USB to the SH7216 CPU board with the user cable.
- Turn on the power to the SH7216 CPU Board.
- Execute "HID.hws" in the sh7216_usb_hid folder.
- Select "Debug > Connection."
- You are asked to select a mode of the emulator. Select "SH7216(R5F72167A)" or "E10A-USB Emulator."
- Press the reset switch on the SH7216 CPU board, and then click on the OK button.
- You are asked to enter an operating frequency. Enter the frequency (12.50 MHz) of the mounted crystal oscillator.
- You are asked to enter an ID code. Enter E10A.
- Select "Debug > Download > All Download Modules" to download the sample program.

4.3.2 Executing the Program

Select "Debug > Execute after Reset" to execute the sample program.

4.4 Demonstrating Mouse Pointer Movements

The sample program demonstrates movements of the host PC mouse pointer without a mouse connected.

While the program is running, connect series-B connector of the USB cable to the SH7216 CPU Board, and series-A connector to the USB host PC. After control transfer is completed, the human interface devices and USB human interface devices are displayed in the device manager window, and the host PC recognizes the SH7216 CPU Board as a mouse device.

After the SH7216 CPU Board is connected to the host PC, the system starts demonstrating mouse pointer movements. The SH7216 CPU Board sends data for mouse pointer movements to the host PC in response to interrupt-in transfer from the host PC. As a result, the mouse pointer on the USB host PC automatically starts moving.

5. Overview of the Sample Program

In this section, features of the sample program and its structure are explained. This sample program is an HID class firmware, which runs on the SH7216 CPU Board and generates data for mouse pointer movements to enable the movements to be emulated on the host PC. The sample program initiates USB transfers by means of tokens from the host PC. Of the interrupts from modules in the SH7216, there are six interrupts related to the USB function module: USI0, USI1, USBRXI0, USBTXI0, USBRXI1 and USBTXI1, but in this sample program, USI0 and USI1 are used.

5.1 State Transition Diagram

Figure 8 shows a state transition diagram for this sample program. In this sample program, as shown in figure 8, there are transitions between four states.

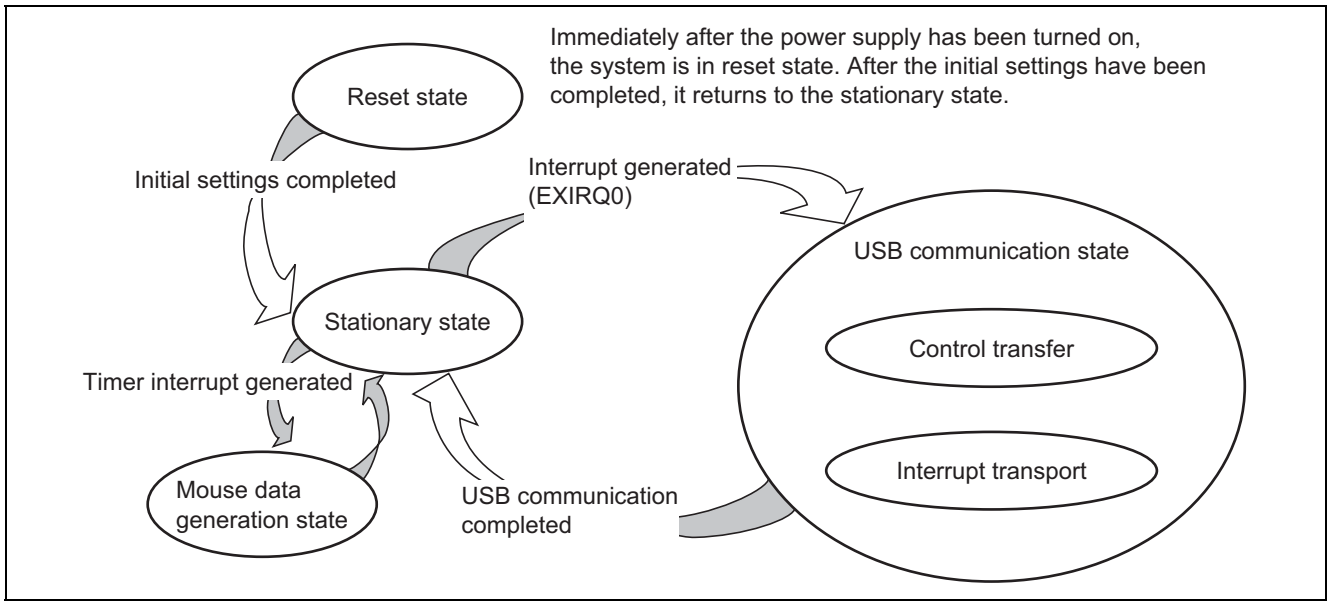


Figure 8 State Transition Diagram

- Reset state
  Upon power-on reset and manual reset, this state is entered. In the reset state, the SH7216 mainly performs initial settings.
- Stationary state
  When initial settings are completed, a stationary state is entered in the main loop.
- USB communication state
  In the stationary state, when an interrupt from the USB module occurs, this state is entered. In the USB communication state, data transfer is performed by a transfer method according to the type of interrupt. The interrupts used in this sample program are indicated by interrupt flag register 0, 1, 2, 3 and 4 (USBIFR0, 1, 2, 3 and 4). When an interrupt factor occurs, the corresponding bits in USBIFR0, 1, 2, 3 and 4 are set.
- Mouse Data Generation State
  In the stationary state, when an overflow interrupt from 16-bit timer MTU2 occurs, this state is entered. In the mouse data generation state, data of mouse pointer movements is automatically generated. A overflow interrupt occurs every 10 ms.

5.2 USB Communication State

The USB communication state can be further divided into three states according to the transfer type (see figure 9). When an interrupt occurs, first there is a transition to the USB communication state, and then there is further branching to a transfer state according to the interrupt type.

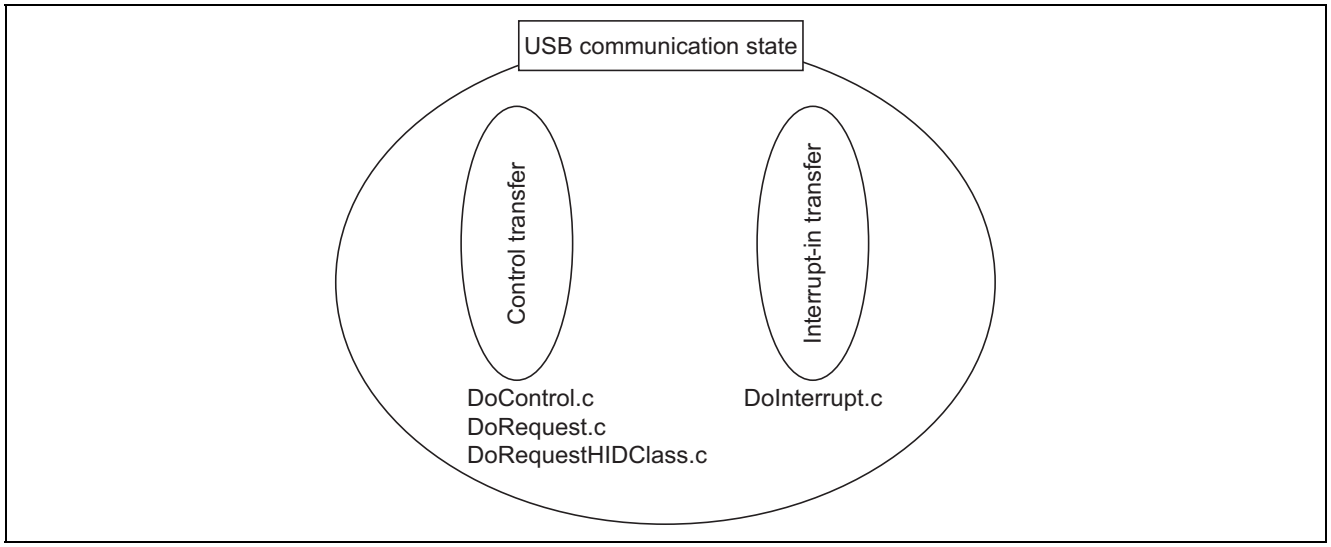


Figure 9 USB Communication State

5.2.1 Control Transfer

Control transfer is used mainly for functions such as obtaining device information and specifying device operating states. For this reason, when the function is connected to the host PC, control transfer is the first transport to be carried out.

Transport processing for control transfer is carried out in a series of two or three stages. These stages are a setup stage, a data stage, and a status stage.

5.2.2 Interrupt-in Transfer

Interrupt transfer is a system that transfers data at predetermined intervals, which ensures the data. The USB HID Class uses the interrupt transfer system to transfer mouse data and keyboard data between the host PC and the function.

5.3 File Structure

This sample program consists of nine source files and nine header files. The overall file structure is shown in table 12. Each function is arranged in one file by transfer method or function type.

Table 12 File Structure

File Name	Principle Role
Startup.c	USB function default settings
UsbMain.c	Judging the causes of interrupts Sending and receiving packets
DoRequest.c	Processing setup commands issued by the host
DoRequestHIDClass.c	Processing USB HID class commands
DoControl.c	Executing control transfer
DoInterrupt.c	Executing interrupt-in transfer
DoHidDataFormat.c	Formatting HID data to be transferred
DoMouse.c	Generating mouse data
sct.src	Transferring initial values of variables from the ROM to the RAM
CatHidTypedef.h	Defining types and structures specific to HID class
CatProType.h	Prototype declarations
CatTypedef.h	Defining the basic structures used in USB firmware
SetMacro.h	Defining macros
SetHidInfo.h	Default settings of variables needed to support HID class commands
SetSystemSwitch.h	System operation settings
SetUsbInfo.h	Default settings of variables used in USB firmware
SysMemMap.h	Defining memory map addresses
iodefine.h	Defining SH7216 registers

5.4 Purposes of Functions

Table 13 to 20 shows functions contained in each file and their purposes.

- Startup.c
  When a power-on reset or manual reset is carried out, the SetPowerOnSection of the Startup.c file is called. At this point, initial settings for the SH7216 registers or USB clock are performed.

Table 13 Startup.c

File in which Stored	Function Name	Purpose
Startup.c	SetPowerOnSection	Initializes module and memory, and shifts to the main loop
	_INITSCT	Copies variables that have default settings to the RAM work area
	InitSystem	Pull-up control of the USB bus
	Set_EPInfoR	Writes endpoint information

- UsbMain.c
  In UsbMain.c, interrupt factors are discriminated by the USB interrupt flag register, and functions are called according to the interrupt type. Also, packets are sent and received between the host controller and function modules.

Table 14 UsbMain.c

File in which Stored	Function Name	Purpose
UsbMain.c	BranchOfInt0	Performs a bus reset, connects the USB cable, determines endpoint 0 interrupt source, and calls a function corresponding to the interrupt.
	BranchOfInt1	Determines endpoint 1 to endpoint 9 interrupt sources, and calls a function corresponding to each interrupt
	GetPacket	Writes data transferred from the host controller to RAM
	GetPacket4	Writes data transferred from the host controller to RAM in longwords (ring buffer supported) (not used by this sample program)
	GetPacket4S	Writes data transferred from the host controller to RAM in longwords (ring buffer not supported, high-speed version)
	PutPacket	Writes data for transfer to the host controller to the USB module
	PutPacket4	Writes data for transfer to the host controller to the USB module in longwords (ring buffer supported) (not used by this sample program)
	PutPacket4S	Writes data for transfer to the host controller to the USB module in longwords (ring buffer not supported, high-speed version)
	SetControlOutContents	Overwrites data with that sent from the host
	SetUsbModule	Makes USB module initial settings
	ActBusReset	Clears FIFO on receiving bus reset
	ActBusVcc	Generates a USB cable connection interrupt
	ConvRealn	Reads data of a specified byte length from a specified address
	ConvReflexn	Reads data of a specified byte length from specified addresses, in reverse order

- DoRequest.c
  During control transfer, commands sent from the host controller are decoded and processed. In this sample program, a vendor ID of H'045B (vendor: Renesas) is used. When the customer develops a product, the customer should obtain a vendor ID at the USB Implementers' Forum. Because vendor commands are not used, DecVenderCommands does not perform any action. In order to use a vendor command, the customer should develop a program.

Table 15 DoRequest.c

File in which Stored	Function Name	Purpose
DoRequest.c	DecStandardCommands	Decodes command issued by host controller, and processes standard commands
	DecVenderCommands	Processes vendor commands

- DoRequestHIDClass.c
  These functions carry out processing according to the HID class commands (GET_REPORT, GET_IDLE, GET_PROTOCOL, SET_REPORT, SET_IDLE, and SET_PROTOCOL).
  — The GET_REPORT command sends HID data from the device to the host PC through control transfer.
  — The GET_IDLE command returns the rate for the time for which interrupt transfer stops.
  — The GET_PROTOCOL command returns the current active protocol (boot protocol or report protocol).
  — The SET_REPORT command sends HID data from the host PC to the device through control transfer, but this sample program does not support out-direction communications of HID data and only receives data.
  — The SET_IDLE command specifies the rate for the time for which interrupt transfer stops.
  — The SET_PROTOCOL command specifies the active protocol (boot protocol or report protocol).

Table 16 DoRequestHIDClass.c

File in which Stored	Function Name	Purpose
DoRequestHIDClass.c	DecHIDClassCommands	Processes USB HID Class commands
	ActIdleCount	This is called by an SOF interrupt, and counts the time for which interrupt transfer stops

- DoControl.c
  When control transfer interrupt SETUP TS is generated, ActControl obtains the command, and decoding is carried out by DecStandardCommands to determine the transfer direction. Next, when control transfer interrupt EP0o TS, EP0i TR, or EP0i TS is generated, ActControlInOut calls either ActControlIn or ActControlOut depending on the transfer direction, and the data stage and status stage are carried out by the called function.

Table 17 DoControl.c

File in which Stored	Function Name	Purpose
DoControl.c	ActControl	Controls the setup stage of control transfer
	ActControlIn	Controls the data stage and status stage of control IN transport (transport in which the data stage is in the IN direction)
	ActControlOut	Controls the data stage and status stage of control OUT transport (transport in which the data stage is in the OUT direction)
	ActControlInOut	Sorts the data stage and status stage of control transfers and direct them to ActControlIn and ActControlOut.

- DoInterrupt.c
  On receiving the in-token of the interrupt transfer from the host PC, this function prepares next data to be sent as soon as the interrupt transfer buffer becomes empty.

Table 18 DoInterrupt.c

File in which Stored	Function Name	Purpose
DoInterrupt.c	ActInterruptIn	On receiving the in-token of the interrupt transfer, gets data from the data transfer buffer as soon as FIFO has an empty space and prepares for interrupt transfer

- DoHidDataFormat.c
  These functions prepare HID data to be transmitted to the host PC.

Table 19 DoHidDataFormat.c

File in which Stored	Function Name	Purpose
DoHidDataFormat.c	ActMakeHidData	A program interface for HID data communications. Calls ActInterruptIn if interrupt transfer stops after ActReportProtocol is called.
	ActReportProtocol	Arranges transfer data according to the format specified by the Report descriptor, and writes the data to the transmit buffer.

- DoMouse.c
  This function uses a timer interrupt and generates data for mouse pointer movements.

Table 20 DoMouse.c

File in which Stored	Function Name	Purpose
DoMouse.c	MousePushedData Input2	This is initiated by a timer interrupt, and generates data for mouse pointer movements according to the elapsed time.

6. Documents for Reference

- Software Manual
  (1) SH-2A, SH2A-FPU Software Manual (REJ09B0051)
  The most up-to-date version of this document is available on the Renesas Technology Website.

- Hardware Manual
  (2) SH7216 Group Hardware Manual (REJ09B0543)
  The most up-to-date version of this document is available on the Renesas Technology Website.

- USB standard-related
  (3) Universal Serial Bus Specification
  (4) Device Class Definition for Human Interface Devices (HID)
  (5) HID Usage Tables
  — Website for USB developers
  http://www.usb.org/developers

Website and Support

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

Inquiries
http://www.renesas.com/inquiry
csc@renesas.com

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Jul.15.09	—	First edition issued
2.00	Feb.10.10	2	Operating frequency amended Version of integrated development environment amended
		15	Figure 7 amended
		16	4.3 description amended

Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
   (1) artificial life support devices or systems
   (2) surgical implantations
   (3) healthcare intervention (e.g., excision, administration of medication, etc.)
   (4) any other purposes that pose a direct threat to human life

   Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas 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 applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.