

V850E2/ML4

R01AN1391EJ0100

Rev.1.00

Nov 08, 2012

MultiMediaCard Control Over USB Sample Program

Introduction

This application note describes a sample program that combines a MultiMediaCard driver created for the V850E2/ML4 microcontroller with an MSC (Mass Storage Class) sample driver created for the internal (on-chip) USB function controller.

This application note and its associated code are provided to explain this application example, and are not guaranteed in any way.

Target Device

V850E2/ML4 (μ PD70F4022)

Contents

1. Introduction.....	2
2. System Structure.....	3
3. V850E2/ML4 Sample Program	4
4. Descriptors	8
5. Function Specifications	11
6. CUSTOMIZATION.....	12
7. CubeSuite+ Environment Settings	13
8. Verifying Program Operation.....	14
9. Reference Documents.....	15

1. Introduction

1.1 Notes

The sample code used in the application note is only provided for explanatory purposes; its operation is not guaranteed by Renesas.

If this sample code is to be used in a user system, it must be thoroughly evaluated and tested in an actual user system before use.

1.2 Target Users

This application note is written for users who want to understand the functions of the V850E2/ML4 microcontroller and develop applications using those functions.

1.3 Specifications

The USB MultiMediaCard driver has the following features.

- It is recognized as a Mass Storage Class (MSC).
- It operates as a self-powered device.
- Data, such as files and folders, can be written to a MultiMediaCard.
- The files and folders written to a MultiMediaCard can be read.

1.4 Functions Used

- Interrupt functions
- USB function controller (USBF)
- Clock synchronous serial interface H (CSIH)

1.5 Conditions

MCU	V850E2/ML4
Operating frequency	Internal clock: 200 MHz
USB clock	Either an internal or an external clock may be selected. <ul style="list-style-type: none"> • Internal clock: external 9.6 MHz clock × internal 20x frequency multiplier ÷ divide by 4 circuit (48 MHz) or external 7.2 MHz clock × internal 20x frequency multiplier ÷ divide by 3 circuit (48 MHz). • External clock: input to the USBCLK pin (fUSB = 48 MHz)
MMC clock	Identification mode: 343.8 KHz Data transfer mode: 16.5 MHz
Integrated development environment	Renesas Electronics Corporation CubeSuite+ V1.02.01
C compiler	CX V1.21, which is provided with the Renesas Electronics Corporation CubeSuite+ product
Supported OS	Windows®7, Vista, XP

1.6 Reference Application Note(s)

- V850E2/ML4 Microcontrollers Example of USB Multifunction Operation (R01AN1037EJ)
- V850E2/ML4 MultiMediaCard SPI Mode Device Driver Adoption Guide (R01AN1026EJ)

2. System Structure

The figure below shows the system structure in which the sample program is used.

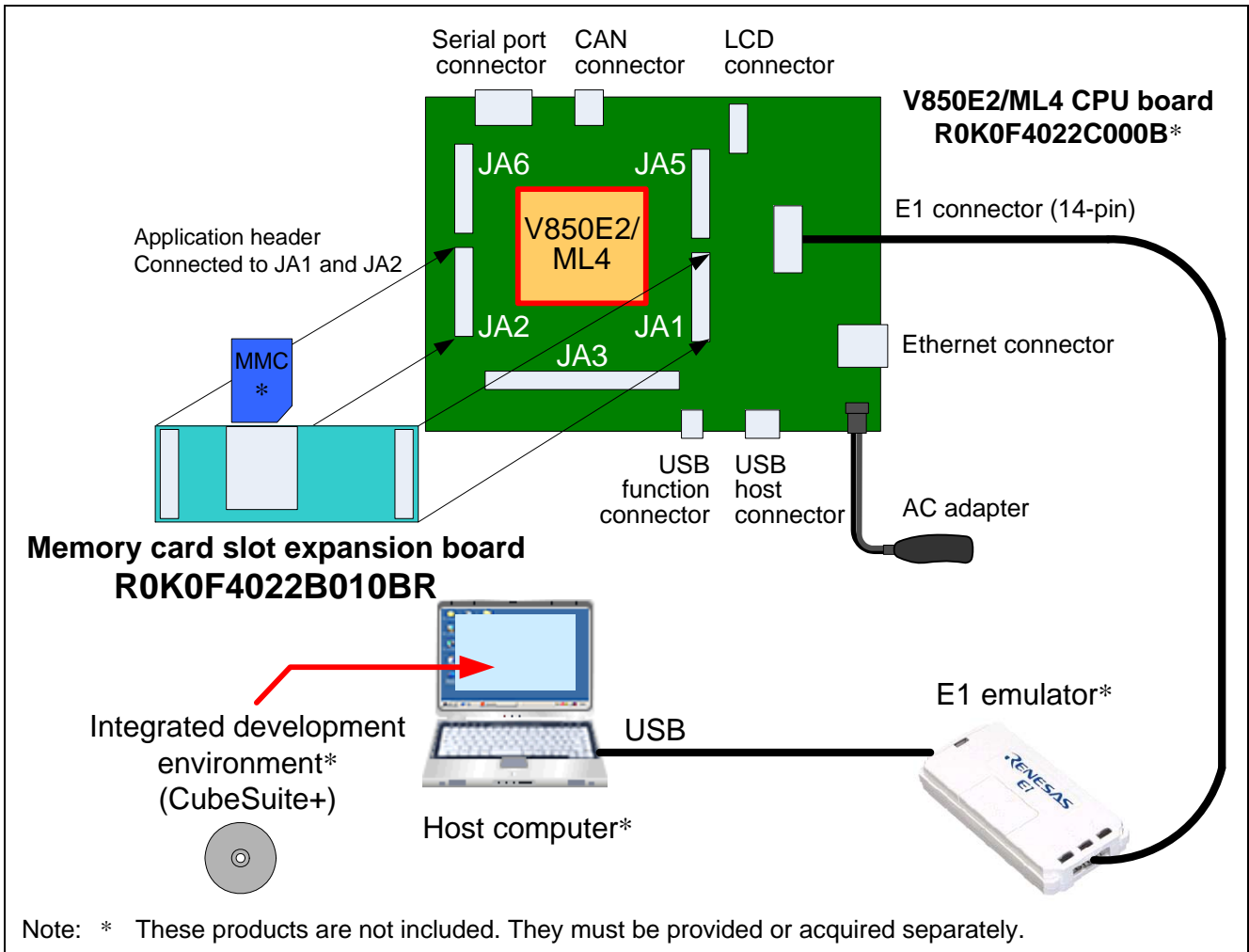


Figure 2.1 System Structure

3. V850E2/ML4 Sample Program

3.1 Folder Structure

The table below lists the folders used for the sample program.

Table 3.1 Folder Structure (CubeSuite+ version)

Folder	Description
prj	Holds the CubeSuite+ project
src	Holds the source files

The V850E2/ML4 sample program consists of a main() routine, a MultiMediaCard driver, and a USB MSC driver. Although the USB MSC driver uses internal memory as a data storage area, it is modified to use the MultiMediaCard as a data storage area.

The source code is stored in the src folder. The following table lists the file structure.

Table 3.2 Source Code File Structure (CubeSuite+ version)

Folder	File Name	Description
/	cstart.asm	Bootstrap
	main.c	main routine
	main.h	Function prototypes for main.c
	macrodriver.h	State control definitions header file
	port.h	CSIH port settings definitions
	serial.c	CSIH operation control module
	serial.h	CSIH operation control module definitions
	serial_user.c	CSIH user settings module
/usb	usbf850.c	USB initialization, endpoint control, bulk transfer, and control transfer
	usbf850.h	Function prototypes for usbf850.c
	scsi_cmd.c	SCSI command processing
	usbf850_storage.c	Dedicated MSC processing
	usbf850_storage.h	Function prototypes for usbf850_storage.c
	usbf850_desc.h	Descriptor definitions
	scsi.h	SCSI related macro definitions
	reg_v850e2ml4.h	USBF register definitions
	usbf850_errno.h	Error code definitions
	usbf850_types.h	User type declarations
	/mmc	r_mmc.h
r_mmc_io.c		MultiMediaCard driver SPI mode I/O module
r_mmc_io.h		
r_mmc_mmc.c		MultiMediaCard driver SPI mode card protocol module
r_mmc_sub.c		MultiMediaCard driver SPI mode card protocol sub-module
r_mmc_sub.h		
r_mmc_spi.c		MultiMediaCard driver SPI mode SPI communication module
r_mmc_usr.c		MultiMediaCard driver SPI mode user API
/mmc/V850E2_ML4	r_mmc_sfr.h	MultiMediaCard driver SFR definitions
	r_mmc_user_config.h	User definitions file
/common	r_mtl_com.c	Common functions (logging)
	r_mtl_com2.h	Common function definitions
	r_mtl_endi.c	Common functions (endian related)
	r_mtl_mem.c	Common functions (standard library functions)
	r_mtl_str.c	Common functions (standard library functions)
	r_mtl_tim.c	Common functions (software loop timer)
	r_mtl_tim.h	Common functions (software loop timer) definitions
	r_stdint.h	Type definitions file
/common/V850E2_66MHz	r_mtl_com.h	Common function header file

3.2 Processing Flow

Figure 3.1 shows the overall flow of control in this sample program. After performing various initializations, it enters the main loop.

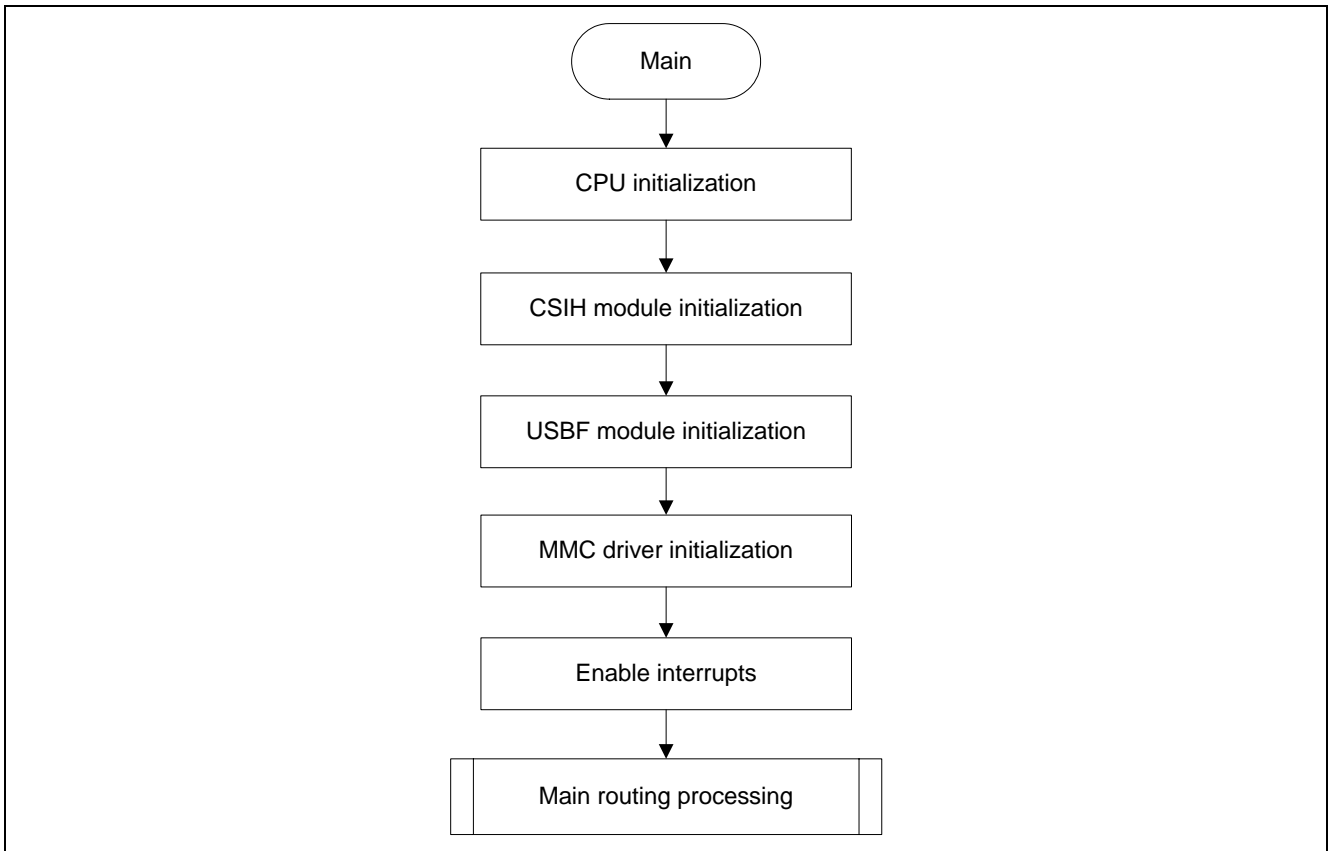


Figure 3.1 Overall Processing Flow

Figure 3.2 shows the main loop processing.

The main loop performs the following two operations.

- Media exchange processing
- USB MSC Bulk Only Transfer protocol processing

In the media exchange processing, the MultiMediaCard mount processing is performed when the state changes from the media not inserted state to the inserted state.

The USB MSC Bulk Only Transfer protocol consists of the following processing flow.

- (1) CBW reception
- (2) CBW command analysis
- (3) SCSI command processing (access to the MultiMediaCard)
- (4) CSW transmission

See the USB MSC driver application note for details on the individual processing operations.

The CBW reception processing (item (1), above) is implemented as interrupt handling. When a CBW is received, the flag is set by the interrupt handler (cbw_in_cbw = USB_CBW_PROCESS).

This flag is monitored in the main loop and the processing of items (2), (3), and (4) is performed when the flag is set.

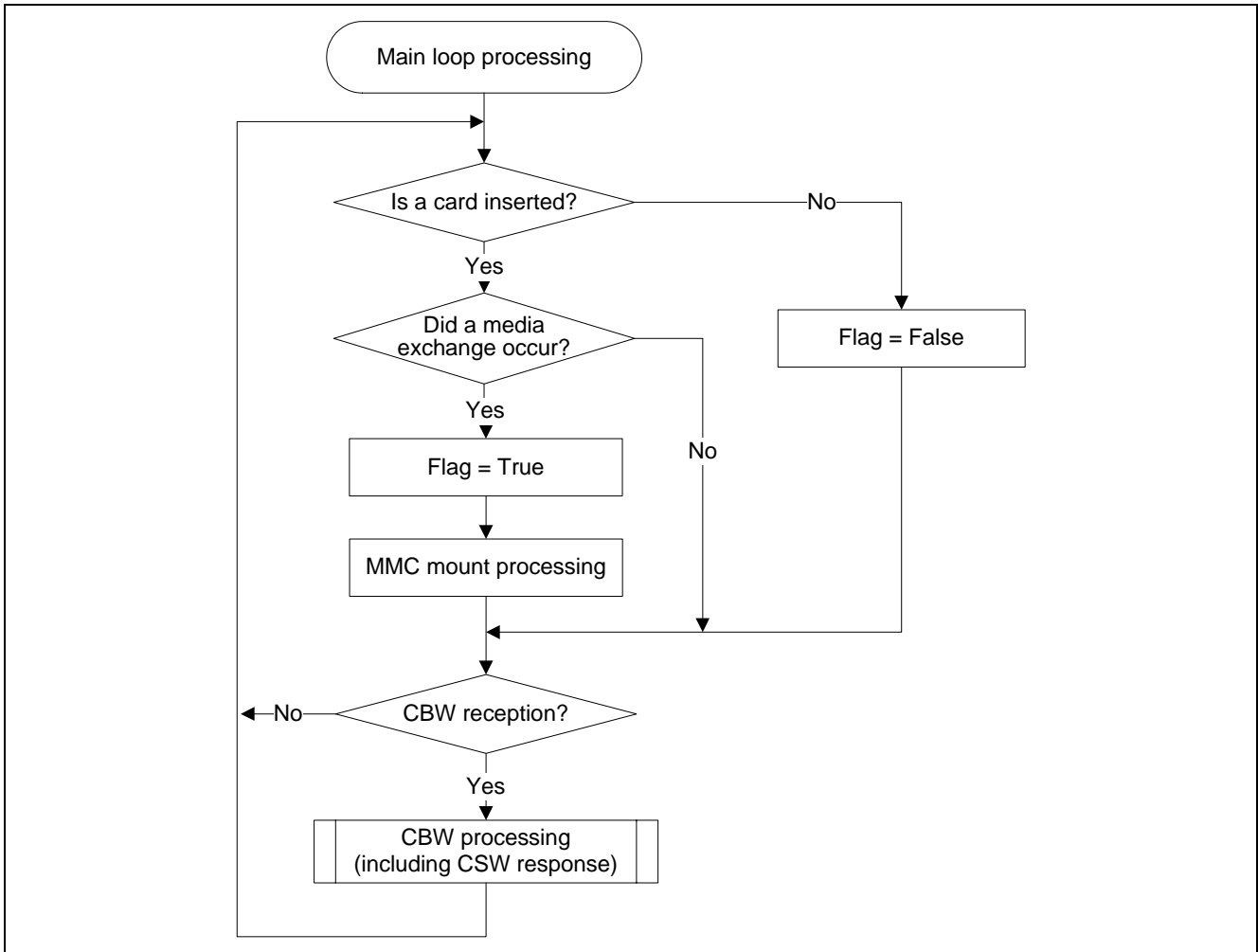


Figure 3.2 Main Loop Processing

4. Descriptors

This section describes the descriptors used for USB communication in this V850E2/ML4 sample program.

4.1 Device Descriptors

Table 4.1 Device Descriptors

Field	Size (byte)	Setting Value	Description
bLength	1	0x12	Descriptor size: 18 bytes
bDescriptorType	1	0x01	Descriptor type: device
bcdUSB	2	0x0200	USB specifications release number: USB 2.0
bDeviceClass	1	0x00	Class code: none
bDeviceSubClass	1	0x00	Subclass code: none
bDeviceProtocol	1	0x00	Protocol code: The specified protocol is unused.
bMaxPacketSize0	1	0x40	Maximum endpoint packet size: 64
idVendor	2	0x045B	Vender ID: Renesas Electronics Corporation
idProduct	2	0x0228	Vendor ID: V850E2/ML4
bcdDevice	2	0x0001	Device release number: Version 1
iManufacturer	1	0x00	Index to the string descriptor that indicates the manufacturer: 0
iProduct	1	0x00	Index to the string descriptor that indicates the product: 0
iSerialNumber	1	0x01	Index to the string descriptor that indicates the device manufacture number: 1
bNumConfigurations	1	0x01	Configuration count: 1

Note: The vendor ID and product ID should be set for the user's application system.

4.2 Configuration Descriptors

Table 4.2 Configuration Descriptors

Field	Size (byte)	Setting Value	Description
bLength	1	0x09	Descriptor size: 9 bytes
bDescriptorType	1	0x02	Descriptor type: configuration
wTotalLength	2	0x0020	Total number of bytes for the configuration, interface, and endpoint descriptors: 32 bytes
bNumInterfaces	1	0x01	Number of interfaces that have this configuration: 1
bConfigurationValue	1	0x01	Identifier for this descriptor: 1
iConfiguration	1	0x00	Index to the string descriptor the codes this descriptor: 0
bmAttributes	1	0xC0	Features of this configuration: self powered, no remote wakeup
bMaxPower	1	0x1B	Maximum current consumption for this configuration: 54 mA

4.3 Interface Descriptors

Table 4.2 Interface Descriptors

Field	Size (byte)	Setting Value	Description
bLength	1	0x09	Descriptor size: 9 bytes
bDescriptorType	1	0x04	Descriptor type: interface
bInterfaceNumber	1	0x00	Identifier for this interface: 0
bAlternateSetting	1	0x00	Alternate settings for this interface: none
bNumEndpoints	1	0x02	Number of endpoints provided by this interface: 2
bInterfaceClass	1	0x08	Class code: Mass Storage Class
bInterfaceSubClass	1	0x06	Subclass code: SCSI transparent command set
bInterfaceProtocol	1	0x50	Protocol code: Bulk Only Transfer
iInterface	1	0x00	Index to the string descriptor that codes this interface: 0

4.4 Endpoint Descriptors

The endpoint address can be switched by enabling or disabling the following define in the header file (usb850.h); the default is enabled.

```
#define USE_EP_BK11
#define USE_EP_BK01
```

Table 4.4 Endpoint Combinations

	Bulk IN	Bulk OUT
define enabled	EP1	EP2
define disabled	EP3	EP4

Table 4.5 EP1 (Bulk IN) Endpoint Descriptor

Field	Size (byte)	Setting Value	Description
bLength	1	0x07	Descriptor size: 7 bytes
bDescriptorType	1	0x05	Descriptor type: endpoint
bEndpointAddress	1	0x81	Transfer direction for this endpoint: IN Address of this endpoint: 1
bmAttributes	1	0x02	Transfer type of this endpoint: bulk
wMaxPacketSize	2	0x0040	Maximum packet size for this endpoint: 64 bytes
bInterval	1	0x00	Polling interval for this endpoint: 0 ms

Table 4.6 EP2 (Bulk OUT) Endpoint Descriptor

Field	Size (byte)	Setting Value	Description
bLength	1	0x07	Descriptor size: 7 bytes
bDescriptorType	1	0x05	Descriptor type: endpoint
bEndpointAddress	1	0x02	Transfer direction for this endpoint: OUT Address of this endpoint: 2
bmAttributes	1	0x02	Transfer type of this endpoint: bulk
wMaxPacketSize	2	0x0040	Maximum packet size for this endpoint: 64 bytes
bInterval	1	0x00	Polling interval for this endpoint: 0 ms

4.5 String Descriptors

Table 4.7 String Descriptor (String 0)

Field	Size (byte)	Setting Value	Description
bLength	1	0x04	Descriptor size: 4 bytes
bDescriptorType	1	0x03	Descriptor type: string
bString	2	0x09,0x04	Language code: (U.S.)

Table 4.8 String Descriptor (String 1)

Field	Size (byte)	Setting Value	Description
bLength	1	0x1A	Descriptor size: 4 bytes
bDescriptorType	1	0x03	Descriptor type: string
bString	24	—	Serial number: 022708065010

5. Function Specifications

The table below lists the API functions provided by the MultiMediaCard driver.

Table 5.1 MultiMediaCard Driver API Functions

Function	Description
R_mmc_Init_Driver	MultiMediaCard driver initialization
R_mmc_Init_Slot	MultiMediaCard initialization
R_mmc_Detach	Stops the MultiMediaCard driver
R_mmc_Read_Data	Reads data from a MultiMediaCard
R_mmc_Write_Data	Writes data to a MultiMediaCard
R_mmc_Chk_Detect	Checks the media insertion state
R_mmc_Get_MmcInfo	Acquires media information (such as capacity)

The following USB MMC driver functions are modified so that MultiMediaCards can be accessed. Although the target of the access is changed from a virtual disk (internal memory) to the MultiMediaCard, the function names and arguments are not changed. See the USB MSC driver application note for the specifications of these functions.

Table 5.2 Functions that Use MultiMediaCard Driver API Functions

Function	Description
ata_read6	Uses R_mmc_Read_Data() to read data from the media.
ata_read10	Uses R_mmc_Read_Data() to read data from the media.
ata_write6	Uses R_mmc_Write_Data() to write data to the media.
ata_write10	Uses R_mmc_Write_Data() to write data to the media.
ata_verify	Uses R_mmc_Read_Data() to read data from the media.
ata_write_verify	Uses R_mmc_Write_Data() to write data to the media.
ata_read_format_capacities	Uses R_mmc_Get_MmcInfo() to get the media capacity.
ata_read_capacity	Uses R_mmc_Get_MmcInfo() to get the media capacity.
ata_test_unit_ready	Uses R_mmc_Chk_Detect() to get the media inserted/not inserted state.
Common to all ata_xxxx functions (however, there are exceptions)	Uses R_mmc_Chk_Detect() and returns an error if the card not inserted state is detected.

6. CUSTOMIZATION

6.1 Descriptor Content

The data (see section 4., Descriptors) that the USB driver registers for the USB function controller at initialization is defined in the file `usb850_desc.h`. Information, such as the target device attributes can be set through the sample program by modifying the values in this file to match the actual application system.

Furthermore, arbitrary information can be registered in the string descriptors. Since a serial number is defined in the sample program, this value should be modified appropriately.

```

:
/* 0 : Language Code*/
DSTR(LangString, 2, (0x09,0x04));
/* 1 : Serial Number*/
USTR(SerialString, 12, ('0','2','2','7','0','8','0','6','5','0','1','0'));
:

```

Figure 6.1 String Descriptor Setting Section in `usb850_desc.h`

6.2 Vendor and Product Name Settings

The names displayed as the vendor name and product name for the disk drive can be modified by editing the INQUIRY command response value defined in the file `scsi_cmd.c`.

The figure below shows the INQUIRY TABLE coded in the file `scsi_cmd.c`.

```

1  UINT8  INQUIRY_TABLE[INQUIRY_LENGTH]={
2      0x00,                                /*Qualifier, device type code*/
3      0x80,                                /*RMB, device type modification child*/
4      0x02,                                /*ISO Version, ECMA Version, ANSI Version*/
5      0x02,                                /*AENC, TrmIOP, response data form*/
6      0x1F,                                /*addition data length*/
7      0x00,0x00,0x00,                    /*reserved*/
8      'R','e','n','e','s','a','s',' ', /*vender ID*/                               <1>
9      'S','t','o','r','a','g','e','F','n','c','D','r','i','v','e','r',
                                         /*product ID*/                               <2>
10     '0','.','0','1'                      /*Product Revision*/
11 };

```

Figure 6.2 INQUIRY_TABLE in the File `scsi_cmd.c`

The vendor name is defined at <1> on line 8 and the product name is defined at <2> on line 9. The vendor name is 8 bytes (8 "half-width" characters from the Japanese character set) and the product name is 16 bytes (16 "half-width" Japanese characters).

During data transmission, each character is transmitted as the ASCII code value. This means that code values that cannot be interpreted as ASCII codes will not be displayed correctly.

The vendor name and product name specified in INQUIRY_TABLE are displayed as the disk drive name by the device manager.

7. CubeSuite+ Environment Settings

7.1 Installation Procedure

1. File Copying

The files supplied in the sample program can be copied to any directory as long as the folder structure is not modified.

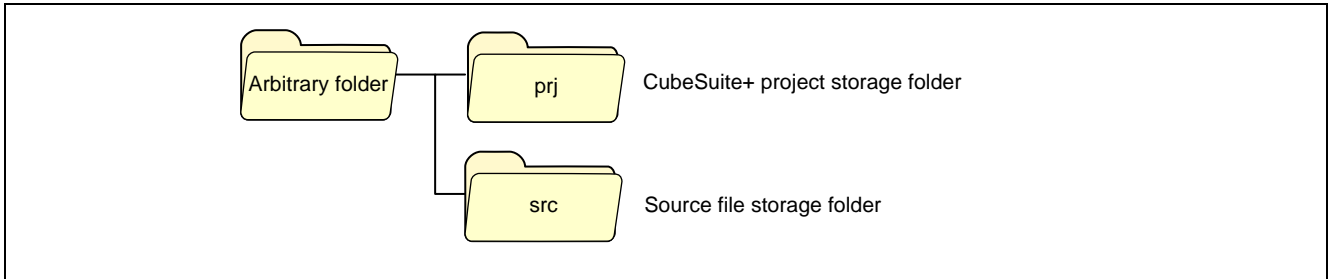


Figure 7.1 Folder Structure

2. Workspace Settings

- Start CubeSuite+ and select Open from the File menu.
- The File Open dialog will open. Specify the CubeSuite+ project file in the prj directory that holds the sample program.

3. Build Tool Settings

- Select CX (Build Tool) from the Project Tree in CubeSuite+ and display the properties.
- Select the Version Selection item and set the compiler package version used in the Used Compiler Package Version item.
- Select V850E2/ML4 E1 (Debugging Tool) from the Project Tree, select Used Debugging Tool from the right-click menu and select "V850E2 E1".

7.2 CubeSuite+ Environment Debugging

7.2.1 Load module generation

To write a program to the target device, the C language and assembler files must be converted using C compilers and assembler to generate a load module.

A load module can be generated by selecting Build Project from the Build menu in CubeSuite+.

7.2.2 Loading and execution

The generated load module can be written to (loaded into) the target and executed.

1. Loading the Load Module

- Start the debugger by selecting Download to debugging tool from the Debug menu.
- This starts downloading of the load module through the debugging tool.

2. Program Execution

Click the  button in CubeSuite+, or, alternatively, select Run from the Debug menu.

8. Verifying Program Operation

8.1 Jumper Settings

Verify the jumper settings before supplying the power to the evaluation board.

1. USB Interface Block

Set the jumpers on the V850E2/ML4 CPU evaluation board (R0K0F4022C000BR) as listed in the table below.

Jumper	Setting	Remarks
JP1	1-2	Enables the VBUS.
JP4	1-2	Enables D+ control by P2_4.
JP9	2-3	Connects P2_10/CSI1F_CS4 to JA1-23.

2. MultiMediaCard Interface Block

Set this jumper on the V850E2/ML4 CPU evaluation board (R0K0F4022C000BR) as listed in the table below. Since software control of the card power supply is not implemented in this sample program, pins 1 and 2 on this jumper must be shorted together.

Jumper	Setting	Remarks
JP1	1-2	Uses the CD signal to control the card power supply.

9. Reference Documents

Hardware Manual

- [1] V850E2/ML4 User's Manual: Hardware (R01UH0262EJ)
(The latest version can be downloaded from the Renesas Electronics website.)
- [2] V850E2/ML4 CPU Board User's Manual (R20UT0778EJ)
- [3] Memory Car Slot Expansion Board User's Manual (R20UT2111EJ)

Software Manual

- V850E2M User's Manual: Architecture (R01US0001EJ)
(The latest version can be downloaded from the Renesas Electronics website.)

Website and Support

Renesas Electronics Website

<http://www.renesas.com/>

Inquiries

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

All trademarks and registered trademarks are the property of their respective owners.

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Nov.08.12	—	First edition issued

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 manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins

Handle unused pins in accord 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 one with a different type number, confirm that the change will not lead to problems.

- The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

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.



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.

2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited

1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited

Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH

Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.

7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.

Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited

Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852-2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.

13F, No. 363, Fu Shing North Road, Taipei, 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 906, 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 Korea Co., Ltd.

11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141