User's Manual

USB2.0 Function Controller

User's Manual

Multimedia Processor for Mobile Applications
EMMA Mobile ^{TM} EV2

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Technology Corp. website (http://www.renesas.com).

Renesas Electronics
www.renesas.com

Rev.4.00 Mar 2012

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.

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 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 part number, confirm that the change will not lead to problems.

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

How to Use This Manual

1. Purpose and Target Readers

This manual is designed to provide the user with an understanding of the hardware functions and electrical characteristics of the MCU. It is intended for users designing application systems incorporating the MCU. A basic knowledge of electric circuits, logical circuits, and MCUs is necessary in order to use this manual.

The manual comprises an overview of the product; descriptions of the CPU, system control functions, peripheral functions, and electrical characteristics; and usage notes.

Particular attention should be paid to the precautionary notes when using the manual. These notes occur within the body of the text, at the end of each section, and in the Usage Notes section.

The revision history summarizes the locations of revisions and additions. It does not list all revisions. Refer to the text of the manual for details.

The following documents apply to the EMMA Mobile EV2. Make sure to refer to the latest versions of these documents. The newest versions of the documents listed may be obtained from the Renesas Electronics Web site.

Document Type	Description	Document Title	Document No.
Data Sheet	Hardware overview and electrical characteristics	EMMA Mobile EV2 Datasheet	R19DS0010EJxxxx
User’s manual (1chip)	Hardware whole specifications (pin assignments, memory maps, peripheral function specifications, electrical characteristics, timing charts) and operation description	EMMA Mobile EV2 User’s manual 1chip	R19UH0036EJxxxx
User’s manual (module)	Hardware each macro specifications and operation description.	EMMA Mobile EV2 User’s manual each module	See below

Document Name	Document No.	Document Name	Document No.
1chip	R19UH0036EJxxxx	DMA Controller	R19UH0043EJxxxx
System Management Unit	R19UH0037EJxxxx	LP-DDR/DDR2 Controller	R19UH0039EJxxxx
Timer	R19UH0054EJxxxx	SD Memory Card Interface	R19UH0061EJxxxx
System Timer	R19UH0055EJxxxx	SDIO Interface	R19UH0042EJxxxx
HD Video Decoder	R19UH0056EJxxxx	CF Card Interface	R19UH0062EJxxxx
Rotator	R19UH0057EJxxxx	Unified Serial Interface	R19UH0047EJxxxx
Resizer	R19UH0058EJxxxx	UART interface	R19UH0040EJxxxx
Image Composer	R19UH0038EJxxxx	USB 2.0 Host Controller	R19UH0045EJxxxx
LCD Interface	R19UH0044EJxxxx	USB 2.0 Function Controller (This manual)	R19UH0034EJxxxx
ITU-R BT.656 Interface	R19UH0059EJxxxx	IIC Interface	R19UH0052EJxxxx
Digital Terrestrial TV Interface	R19UH0048EJxxxx	General Purpose I/O Interface	R19UH0041EJxxxx
Camera Interface	R19UH0060EJxxxx	Pulse Width Modulation Interface	R19UH0063EJxxxx

4 digits of end shows the version.

2. Notation of Numbers and Symbols

The notation conventions for register names, bit names, numbers, and symbols used in this manual are described below.

(1) Register Names, Bit Names, and Pin Names

Registers, bits, and pins are referred to in the text by symbols. The symbol is accompanied by the word “register,” “bit,” or “pin” to distinguish the three categories.

Examples the PM03 bit in the PM0 register
P3_5 pin, VCC pin

(2) Notation of Numbers

The indication “b” is appended to numeric values given in binary format. However, nothing is appended to the values of single bits. The indication “h” is appended to numeric values given in hexadecimal format. Nothing is appended to numeric values given in decimal format.

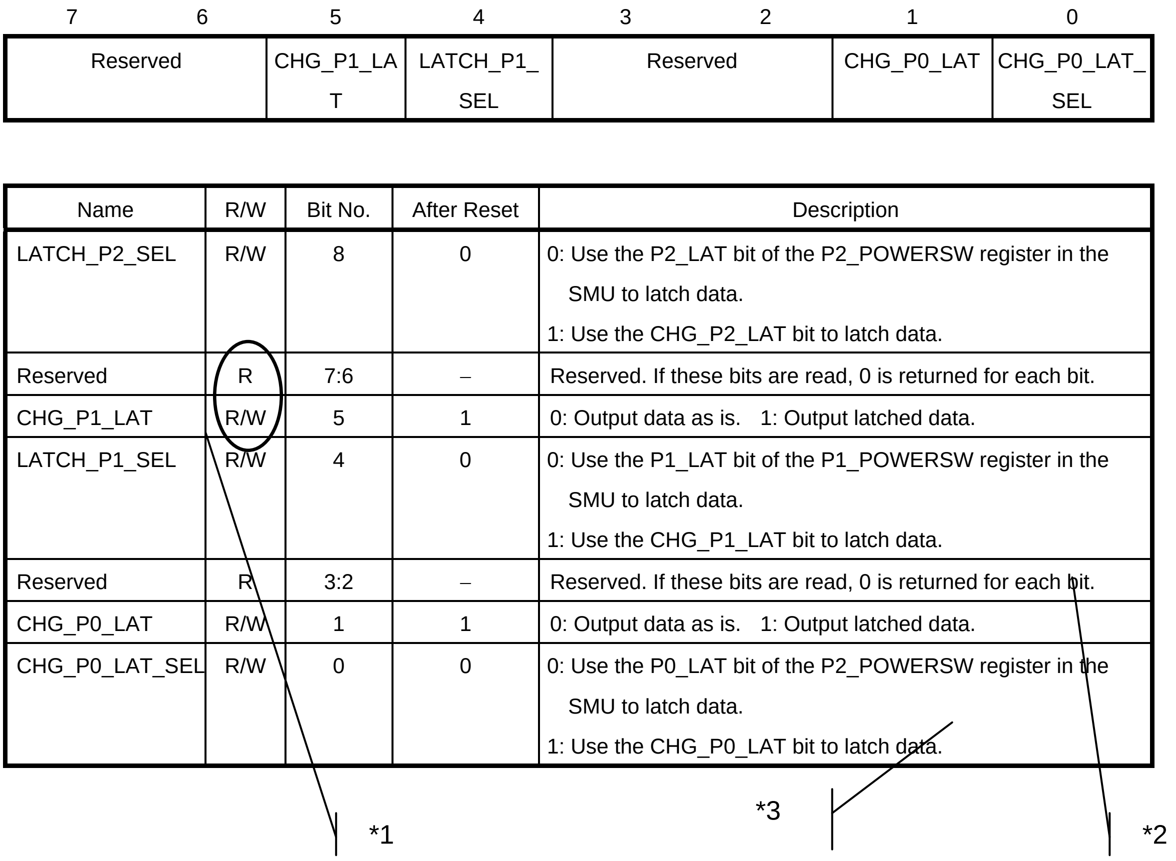
Examples Binary: 11b or 11
Hexadecimal: EFA0h
Decimal: 1234

3. Register Notation

The symbols and terms used in register diagrams are described below.

x.x.x XXX register

This register (XXXXXXX: xxxx_xxxxh) …..



7	6	5	4	3	2	1	0
Reserved		CHG_P1_LAT	LATCH_P1_SEL	Reserved		CHG_P0_LAT	CHG_P0_LAT_SEL

Name	R/W	Bit No.	After Reset	Description
LATCH_P2_SEL	R/W	8	0	0: Use the P2_LAT bit of the P2_POWERSW register in the SMU to latch data. 1: Use the CHG_P2_LAT bit to latch data.
Reserved	R	7:6	–	Reserved. If these bits are read, 0 is returned for each bit.
CHG_P1_LAT	R/W	5	1	0: Output data as is. 1: Output latched data.
LATCH_P1_SEL	R/W	4	0	0: Use the P1_LAT bit of the P1_POWERSW register in the SMU to latch data. 1: Use the CHG_P1_LAT bit to latch data.
Reserved	R	3:2	–	Reserved. If these bits are read, 0 is returned for each bit.
CHG_P0_LAT	R/W	1	1	0: Output data as is. 1: Output latched data.
CHG_P0_LAT_SEL	R/W	0	0	0: Use the P0_LAT bit of the P2_POWERSW register in the SMU to latch data. 1: Use the CHG_P0_LAT bit to latch data.

*1

R/W: Read and Write.

R: Read only.

W: Write only.

-: Nothing is assigned.

*2

Reserved bit.

Reserved bit. Set to specified value.

*3

- Nothing is assigned.

  Nothing is assigned to the bit. As the bit may be used for future functions, if necessary, set to 0.
- Do not set to a value.

  Operation is not guaranteed when a value is set.
- Function varies according to the operating mode.

  The function of the bit varies with the peripheral function mode. Refer to the register diagram for information on the individual modes.

4. List of Abbreviations and Acronyms

Abbreviation	Full Form
ACIA	Asynchronous Communication Interface Adapter
bps	bits per second
CRC	Cyclic Redundancy Check
DMA	Direct Memory Access
DMAC	Direct Memory Access Controller
GSM	Global System for Mobile Communications
Hi-Z	High Impedance
IEBus	Inter Equipment Bus
I/O	Input/Output
IrDA	Infrared Data Association
LSB	Least Significant Bit
MSB	Most Significant Bit
NC	Non-Connect
PLL	Phase Locked Loop
PWM	Pulse Width Modulation
SFR	Special Function Register
SIM	Subscriber Identity Module
UART	Universal Asynchronous Receiver/Transmitter
VCO	Voltage Controlled Oscillator

Table of Contents

USB2.0 Function Controller

EMMA Mobile EV2

R19UH0034EJ0400
Rev.4.00
Mar 30, 2012

1. Overview

. This document describes the features of the USB function controller used to control USB 2.0 functions (devices). A 32-bit AHB bus is assumed to be used for communication between USB 2.0 devices and the CPU, which facilitates incorporation of the USB function controller in an ARM CPU system.

1.1 Features

- Conforms to Universal Serial Bus Specification Revision 2.0.
- High-speed (480 Mbps) and full-speed (12 Mbps) transfer.
- USB 2.0 compliance test mode can be used.
- USB system clock: 24 MHz.
- The request address size: 8KB (Be sure to align in the 8KB unit.).

Table 1-1. Selectable Endpoint Configuration

※It doesn't correspond to Isochronous transfer.

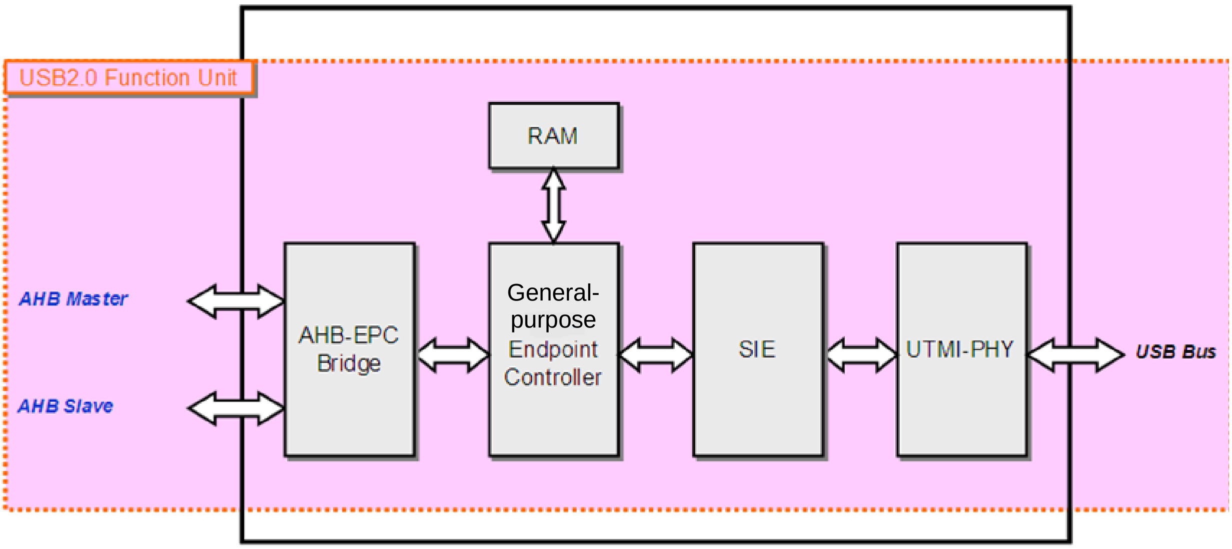
	Transfer Type	Buffer Type	Use of DMA	Transfer Direction	Max. Packet Size (Bytes)		Assumed RAM Capacity (Bytes)
					HS	FS	
EP0	Control	Single × 2	Used	Bidirectional	64	64	128
EP1	Bulk	Double	Used	Input or output	512	8/16/32/64	1,024
EP2	Bulk	Double	Used	Input or output	512	8/16/32/64	1,024
EP3	Interrupt	Single	Not used	Input	1 to 1,024	1 to 64	1,024
EP6	Bulk	Double	Used	Input or output	512	8/16/32/64	1,024
EP7	Bulk	Double	Used	Input or output	512	8/16/32/64	1,024
EP8	Interrupt	Single	Not used	Input	1 to 1,024	1 to 64	1,024
EP11	Bulk	Double	Used	Input or output	512	8/16/32/64	**Note**
EP12	Bulk	Double	Used	Input or output	512	8/16/32/64	**Note**
EP13	Interrupt	Single	Not used	Input	1 to 1,024	1 to 64	**Note**

Note These endpoints use RAM area assigned to other endpoints during use.

1.2 Block Overview

An overview of the USB function controller block and the included blocks is shown below.

Figure 1-1. Block Overview



(1) UTMI-PHY (UTMIPLUSPHY_TOP)

This macro includes a USB 2.0 transceiver and a USB 2.0 transceiver macrocell interface (UTMI+) and generates the clock to be supplied to the serial interface engine (SIE), endpoint controller, and AHB-EPC bridge at subsequent stages.

(2) SIE (NSU2UTMIFSIEV10)

This is a serial interface engine (SIE) module that is connected to the PHY.

(3) General-purpose endpoint controller EPC (NUSB2GEV22)

This module is connected to the SIE, and mainly controls the endpoint buffers of the USB devices and the USB transaction protocol.

(4) AHB-EPC bridge (USB2_AHB_BRIDGE)

This is a bridge circuit that converts PIO-to-endpoint-controller or DMA transfer cycles to AHB cycles. The CPU accesses the bridge registers and EPC registers by using the bridge's slave interface.

This circuit includes a DMA controller, which is used to perform DMA transfers between system memory area and endpoint buffers by using the bridge's master interface.

(5) RAM

This is a memory area used as the endpoint buffer.

1.3 Notes on Use

1.3.1 Specifications

(1) To make the USB function controller consistent with the rest of the system, some register settings associated with the implemented sub-blocks are restricted. The applicable settings are indicated in bold face in **CHAPTER 4 Registers.** See this chapter before setting the registers.

(2) The operation of the USB1_CLK and PMCLK clocks cannot be changed dynamically, except for stopping the clock.

(3) This module supports the direct power-down feature. For details about the direct power-down feature, see *10.3 Direct Power-Down Feature*.

(4) The CLKSEL pin status cannot be changed dynamically. Fix the level of this pin on the chip or externally.

(5) Registers in the USB function controller can only be accessed in 32-bit units.

1.3.2 Operations

(1) For details about the operating procedures, see the following sections:

- Initial setup and basic operation: **CHAPTER 12 Operating Procedures**
- Connection and disconnection: **9.3 VBUS Detection Overview**
- Mapping the 1-port RAM: **11.2 Specifyng the Base Address**
- Controlling power management: **CHAPTER 10 Power Management**

(2) When accessing an EPC register, make sure the clock is supplied to the USB function controller. For details, see *5.1 Notes on Accessing EPC Registers*.

1.3.3 Other notes

If the USB clock is not supplied to the USB devices, tasks might deadlock.

2. Pin Functions

2.1 System Interface

Table 2-1. System Interface Pins

Pin Name	I/O	Description	Synchronization Clock	Active Level	After Reset
USB1_CLK	Input	AHB clock	–	–	–
USB1_RSTZ	Input	AHB reset	Asynchronous	L	–
OSC1 clock	Input	USB reference clock	–	–	–

2.2 Interrupt Interface

Table 2-2. Interrupt Interface Pins

Pin Name	I/O	Description	Synchronization Clock	Active Level	After Reset
USB_INTF0	Output	Interrupt signal (BRIDGE) Only VBUS interrupts synchronize with PMCLK.	USB1_CLK PMCLK	H	0b
USB_INTF1	Output	Interrupt signal (EPC)	Internal 60 MHz clock	H	0b

2.3 USB Interface

Table 2-3. USB Interface Pins

Pin Name	I/O	Description	Synchronization Clock	Active Level	After Reset
USB_DP2	I/O	High-speed D+ signal for USB peripheral bus	Asynchronous	–	– (IN)
USB_DM2	I/O	High-speed D- signal for USB peripheral bus	Asynchronous	–	– (IN)
USB_VBUS	Input	Regulator pin	Asynchronous	–	–
USB_RREF2	–	Reference current generation	–	–	–

2.4 Power Interface

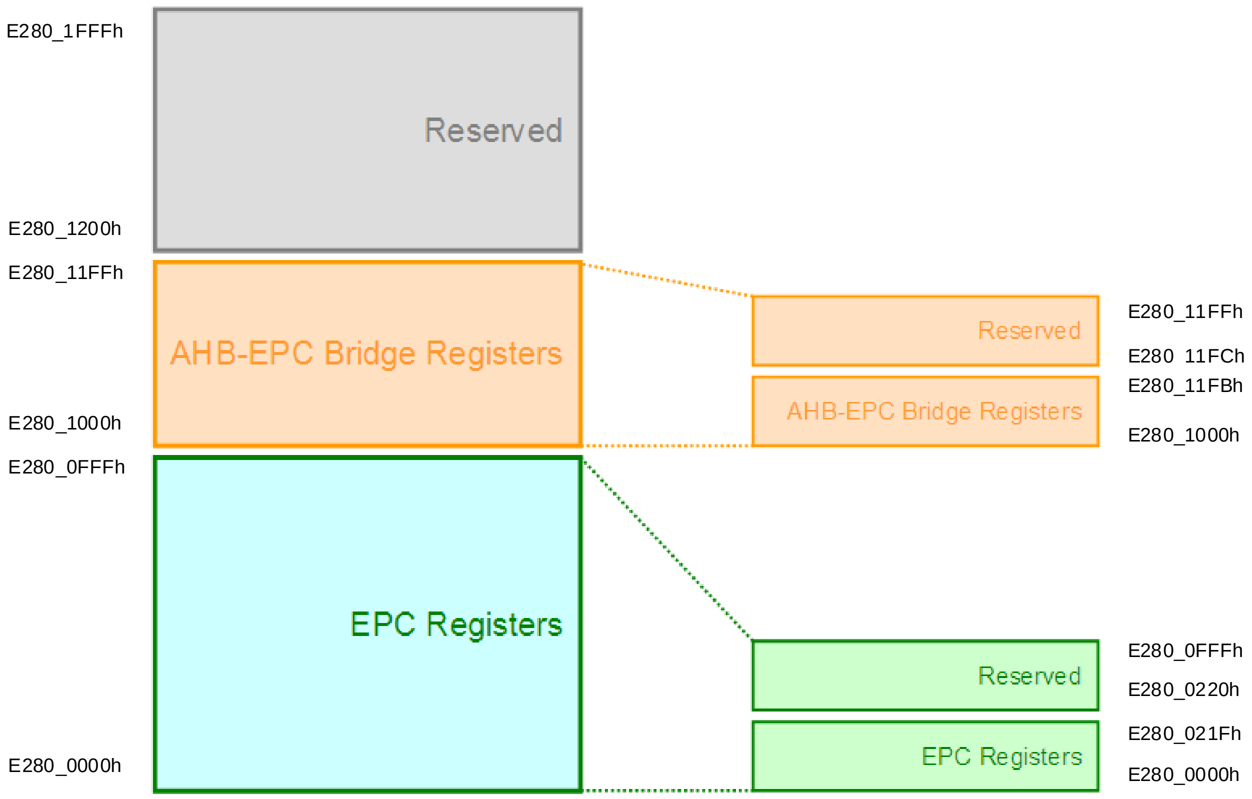
Table 2-4. Power Interface Pins

Pin Name	I/O	Description
USB_AVDD2	–	Regulator power supply pin
USB_AVSS2	–	Regulator ground pin
USB_PVSS2	–	PLL ground pin
USB_VD3312	–	IO power supply pin
USB_GND12	–	IO ground pin
USB_GND22	–	IO ground pin

3. Register Mapping

USB function controller registers are allocated to the EPC area and AHB-EPC bridge area. In both areas, address spaces that do not correspond to endpoint numbers implemented in the system are reserved, so do not access these spaces.

Figure 3-1. Register Mapping



Requested address size: 8 KB

Table 3-1. Register Mapping

Address Offset	Register Name	Symbol
0000h	USB control register	USB_CONTROL
0004h	USB status register	USB_STATUS
0008h	Frame number & USB address register	USB_ADDRESS
000Fh to 0014h	Reserved	–
0018h	Setup data 0 register	SETUP_DATA0
001Ch	Setup data 1 register	SETUP_DATA1
0020h	USB interrupt status register	USB_INT_STA
0024h	USB interrupt enable register	USB_INT_ENA
0028h	EP0 control register	EP0_CONTROL
002Ch	EP0 status register	EP0_STATUS
0030h	EP0 interrupt enable register	EP0_INT_ENA
0034h	EP0 length register	EP0_LENGTH
0038h	EP0 read register	EP0_READ
003Ch	EP0 write register	EP0_WRITE
0040h	EP1 control register	EP1_CONTROL
0044h	EP1 status register	EP1_STATUS
0048h	EP1 interrupt enable register	EP1_INT_ENA
004Ch	EP1 DMA control register	EP1_DMA_CTRL
0050h	EP1 max packet & base address register	EP1_PCKT_ADRS
0054h	EP1 length & DMA count register	EP1_LEN_DCNT
0058h	EP1 read register	EP1_READ
005Ch	EP1 write register	EP1_WRITE
0060h	EP2 control register	EP2_CONTROL
0064h	EP2 status register	EP2_STATUS
0068h	EP2 interrupt enable register	EP2_INT_ENA
006Ch	EP2 DMA control register	EP2_DMA_CTRL
0070h	EP2 max packet & base address register	EP2_PCKT_ADRS
0074h	EP2 length & DMA count register	EP2_LEN_DCNT
0078h	EP2 read register	EP2_READ
007Ch	EP2 write register	EP2_WRITE
0080h	EP3 control register	EP3_CONTROL
0084h	EP3 status register	EP3_STATUS
0088h	EP3 interrupt enable register	EP3_INT_ENA
008Ch	EP3 DMA control register	EP3_DMA_CTRL
0090h	EP3 max packet & base address register	EP3_PCKT_ADRS
0094h	EP3 length & DMA count register	EP3_LEN_DCNT
0098h	EP3 read register	EP3_READ
009Ch	EP3 write register	EP3_WRITE
00A0h	EP4 control register	EP4_CONTROL
00A4h	EP4 status register	EP4_STATUS
00A8h	EP4 interrupt enable register	EP4_INT_ENA
00ACh	EP4 DMA control register	EP4_DMA_CTRL

Address Offset	Register Name	Symbol
00B0h	EP4 max packet & base address register	EP4_PCKT_ADRS
00B4h	EP4 length & DMA count register	EP4_LEN_DCNT
00B8h	EP4 read register	EP4_READ
00BCh	EP4 write register	EP4_WRITE
00C0h	EP5 control register	EP5_CONTROL
00C4h	EP5 status register	EP5_STATUS
00C8h	EP5 interrupt enable register	EP5_INT_ENA
00CCh	EP5 DMA control register	EP5_DMA_CTRL
00D0h	EP5 max packet & base address register	EP5_PCKT_ADRS
00D4h	EP5 length & DMA count register	EP5_LEN_DCNT
00D8h	EP5 read register	EP5_READ
00DCh	EP5 write register	EP5_WRITE
00E0h	EP6 control register	EP6_CONTROL
00E4h	EP6 status register	EP6_STATUS
00E8h	EP6 interrupt enable register	EP6_INT_ENA
00ECh	EP6 DMA control register	EP6_DMA_CTRL
00F0h	EP6 max packet & base address register	EP6_PCKT_ADRS
00F4h	EP6 length & DMA count register	EP6_LEN_DCNT
00F8h	EP6 read register	EP6_READ
00FCh	EP6 write register	EP6_WRITE
0100h	EP7 control register	EP7_CONTROL
0104h	EP7 status register	EP7_STATUS
0108h	EP7 interrupt enable register	EP7_INT_ENA
010Ch	EP7 DMA control register	EP7_DMA_CTRL
0110h	EP7 max packet & base address register	EP7_PCKT_ADRS
0114h	EP7 length & DMA count register	EP7_LEN_DCNT
0118h	EP7 read register	EP7_READ
011Ch	EP7 write register	EP7_WRITE
0120h	EP8 control register	EP8_CONTROL
0124h	EP8 status register	EP8_STATUS
0128h	EP8 interrupt enable register	EP8_INT_ENA
012Ch	EP8 DMA control register	EP8_DMA_CTRL
0130h	EP8 max packet & base address register	EP8_PCKT_ADRS
0134h	EP8 length & DMA count register	EP8_LEN_DCNT
0138h	EP8 read register	EP8_READ
013Ch	EP8 write register	EP8_WRITE
0140h	EP9 control register	EP9_CONTROL
0144h	EP9 status register	EP9_STATUS
0148h	EP9 interrupt enable register	EP9_INT_ENA
014Ch	EP9 DMA control register	EP9_DMA_CTRL
0150h	EP9 max packet & base address register	EP9_PCKT_ADRS
0154h	EP9 length & DMA count register	EP9_LEN_DCNT

Address Offset	Register Name	Symbol
0158h	EP9 read register	EP9_READ
015Ch	EP9 write register	EP9_WRITE
0160h	EP10 control register	EP10_CONTROL
0164h	EP10 status register	EP10_STATUS
0168h	EP10 interrupt enable register	EP10_INT_ENA
016Ch	EP10 DMA control register	EP10_DMA_CTRL
0170h	EP10 max packet & base address register	EP10_PCKT_ADRS
0174h	EP10 length & DMA count register	EP10_LEN_DCNT
0178h	EP10 read register	EP10_READ
017Ch	EP10 write register	EP10_WRITE
0180h	EP11 control register	EP11_CONTROL
0184h	EP11 status register	EP11_STATUS
0188h	EP11 interrupt enable register	EP11_INT_ENA
018Ch	EP11 DMA control register	EP11_DMA_CTRL
0190h	EP11 max packet & base address register	EP11_PCKT_ADRS
0194h	EP11 length & DMA count register	EP11_LEN_DCNT
0198h	EP11 read register	EP11_READ
019Ch	EP11 write register	EP11_WRITE
01A0h	EP12 control register	EP12_CONTROL
01A4h	EP12 status register	EP12_STATUS
01A8h	EP12 interrupt enable register	EP12_INT_ENA
01ACh	EP12 DMA control register	EP12_DMA_CTRL
01B0h	EP12 max packet & base address register	EP12_PCKT_ADRS
01B4h	EP12 length & DMA count register	EP12_LEN_DCNT
01B8h	EP12 read register	EP12_READ
01BCh	EP12 write register	EP12_WRITE
01C0h	EP13 control register	EP13_CONTROL
01C4h	EP13 status register	EP13_STATUS
01C8h	EP13 interrupt enable register	EP13_INT_ENA
01CCh	EP13 DMA control register	EP13_DMA_CTRL
01D0h	EP13 max packet & base address register	EP13_PCKT_ADRS
01D4h	EP13 length & DMA count register	EP13_LEN_DCNT
01D8h	EP13 read register	EP13_READ
01DCh	EP13 write register	EP13_WRITE
01E0h to 0220h	Reserved	–
1000h	AHBSCTR	AHBSCTR
1004h	AHBMCTR	AHBMCTR
1008h	AHBBINT	AHBBINT
100Ch	AHBBINTEN	AHBBINTEN
1010h	EPCTR	EPCTR
1014h	EPTEST	EPTEST
1018h to 101Fh	Reserved	–

Address Offset	Register Name	Symbol
1020h	USBSSVER	USBSSVER
1024h	USBSSCONF	USBSSCONF
110Fh to 1028h	Reserved	–
1110h	EP1DCR1	EP1DCR1
1114h	EP1DCR2	EP1DCR2
1118h	EP1TADR	EP1TADR
111Fh to 111Ch	Reserved	–
1120h	EP2DCR1	EP2DCR1
1124h	EP2DCR2	EP2DCR2
1128h	EP2TADR	EP2TADR
112Fh to 112Ch	Reserved	–
1130h	EP3DCR1	EP3DCR1
1134h	EP3DCR2	EP3DCR2
1138h	EP3TADR	EP3TADR
113Fh to 113Ch	Reserved	–
1140h	EP4DCR1	EP4DCR1
1144h	EP4DCR2	EP4DCR2
1148h	EP4TADR	EP4TADR
114Fh to 114Ch	Reserved	–
1150h	EP5DCR1	EP5DCR1
1154h	EP5DCR2	EP5DCR2
1158h	EP5TADR	EP5TADR
115Fh to 115Ch	Reserved	–
1160h	EP6DCR1	EP6DCR1
1164h	EP6DCR2	EP6DCR2
1168h	EP6TADR	EP6TADR
116Fh to 116Ch	Reserved	–
1170h	EP7DCR1	EP7DCR1
1174h	EP7DCR2	EP7DCR2
1178h	EP7TADR	EP7TADR
117Fh to 117Ch	Reserved	–
1180h	EP8DCR1	EP8DCR1
1184h	EP8DCR2	EP8DCR2
1188h	EP8TADR	EP8TADR
118Fh to 118Ch	Reserved	–
1190h	EP9DCR1	EP9DCR1
1194h	EP9DCR2	EP9DCR2
1198h	EP9TADR	EP9TADR
119Fh to 119Ch	Reserved	–
11A0h	EP10DCR1	EP10DCR1
11A4h	EP10DCR2	EP10DCR2
11A8h	EP10TADR	EP10TADR

Address Offset	Register Name	Symbol
11AFh to 11ACh	Reserved	–
11B0h	EP11DCR1	EP11DCR1
11B4h	EP11DCR2	EP11DCR2
11B8h	EP11TADR	EP11TADR
11BFh to 11BCh	Reserved	–
11C0h	EP12DCR1	EP12DCR1
11C4h	EP12DCR2	EP12DCR2
11C8h	EP12TADR	EP12TADR
11CFh to 11CCh	Reserved	–
11D0h	EP13DCR1	EP13DCR1
11D4h	EP13DCR2	EP13DCR2
11D8h	EP13TADR	EP13TADR
11DFh to 11FCh	Reserved	–

4. Registers

This chapter describes the register features in detail. The meaning of the bit assignment table is shown below.

Add	100h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W																														*R*	*R* *W*	*R* *W*
Symbol																														SYMBOL_C	SYMBOL_B	SYMBOL_A
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	0

Add: Indicates the address offset.

Bit: Indicates the position of the bit in the 32-bit space.

RW: R indicates that the bit can be read, and W indicates that the bit can be written.

Symbol: Indicates the name of the bit.

R: Indicates the initial value after reset.

B: Indicates the value immediately after a USB bus reset signal is received (available in the EPC area only).

Remark The bits whose **Symbol** column is blank are reserved.

4.1 EPC Registers

4.1.1 USB control register (USB_CONTROL)

This register is used to control the basic EPC features and the USB device status.

Add	0000h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W														R W	R W	R W						R W		R W	R W	R W	R W	R W	R W	R W		
Symbol														USBTESTMODE2	USBTESTMODE1	USBTESTMODE0						INT_SEL		SOF_RCV	RSUM_IN	SUSPEND	CONF	DEFAULT	CONNECTB	PUE2		
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	0	1	–	–	–	–

Bit	Symbol	Description
[31:19]	–	Reserved. (Be sure to write 0b to this field.)
[18:16]	USBTESTMODE[2:0]	This bit is set after a status stage normal termination of SET_FEATURE_TEST_MODE request. When this bit is set besides the 000b, Function will be a HS mode immediately, and enters a test mode. 000b : Normal 001b : Test_J 010b : Test_K 011b : Test_SE0_NAK 100b : Test_Packet 101b : Test_Force_Enable
10	INT_SEL	Select the level of the USB_INTF1 interrupt output signal. The level cannot be specified for individual interrupt signals. Set this bit to 1b in the initial setup and do not change the value. 1: Level output If there are multiple interrupt sources, the interrupt output signals remain asserted until all the sources are cleared. 0: Pulse output If there are multiple interrupt sources and the sources are cleared, the interrupt output signal is deasserted. If a source remains, the interrupt output signal is asserted again 15 clock cycles (at 60 MHz) later.
9	–	Reserved. (The setting can't be changed from defaults.)
8	SOF_RCV	Select whether to enable automatic recovery when an SOF reception error occurs. Set this bit to 1b in the initial setup and do not change the value. 1: Enable 0: Disable
7	RSUM_IN	Specify whether to send the resume signal when using the remote wakeup feature. 1: Send the resume signal.

Bit	Symbol	Description
		0: Do not send the resume signal. **Caution While the system is in the Suspend state and the clock supply to the EPC, SIE, and UTMI-PHY blocks is stopped, the clock supply must be resumed first before setting this bit. For how to resume the clock supply, see *5.1 Notes on Accessing EPC Registers*.**
6	SUSPEND	If this bit is set to 1b while the USB device is in the Suspend state as defined in the USB specification, the clock supply to the EPC, SIE, and UTMI-PHY blocks is stopped to reduce power consumption. This bit is cleared to 0b when the resume signal is detected. For how to stop the clock, see *10.1 Power Down*. 1: Stop the clock supply. 0: Do not stop the clock supply.
5	CONF	Set this bit to enable endpoints other than endpoint 0. This bit is cleared to 0b when a bus reset signal is received. If this bit is cleared to 0b while the USB device is in a state other than the Suspend state, the USB device enters the Default or Address state. If this bit is set to 1b while the USB device is in a state other than the Suspend state, the USB device enters the Configured state. 1: Enable endpoints other than endpoint 0. 0: Disable endpoints other than endpoint 0. (No response is returned for tokens.)
4	DEFAULT	Set this bit to enable endpoint 0. This bit is set to 1b when a bus reset signal is received. If this bit is cleared to 0b, the USB device enters the Attached or Powered state as defined in the USB specification. If this bit is set to 1b, the state to be entered depends on the value of bit 5 (CONF). 1: Enable. 0: Disable. (No response is returned for tokens.)
3	CONNECTB	Set this bit to 1b when disconnecting a USB device to prevent the occurrence of a pseudo bus reset or suspend signal due to an unstable D+/D– signal operation on the USB bus. 1: Disable the USB signal sent to the UTMI-PHY block and then to the SIE block. 0: Enable the USB signal sent to the UTMI-PHY block and then to the SIE block.
2	PUE2	Specify whether to pull up the D+ signal. 1: Pull up the D+ signal. 0: Do not pull up the D+ signal.
[1:0]	–	Reserved. (Be sure to write 0b to this field.)

4.1.2 USB status register (USB_STATUS)

This is a read-only register that indicates the USB status and PHY core status.

Add	0004h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W																										R	R	R	R	R	R	
Symbol																										SPEED_MODE	CONF	DEFAULT	USB_RST	SPND_OUT	RSUM_OUT	
R	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	0	0	–	–	–	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	0	1	–	–	–	–

Bit	Symbol	Description
[31:7]	–	Reserved. (Be sure to write 0b to this field.)
6	SPEED_MODE	Indicates the USB bus speed. 1: High speed 0: Full speed
5	CONF	Indicates whether endpoints other than endpoint 0 are enabled. This bit reflects the value of bit 5 (CONF) of the USB control register. 1: Enabled 0: Disabled (No response is returned for tokens.)
4	DEFAULT	Indicates whether endpoint 0 is enabled. This bit reflects the value of bit 4 (DEFAULT) of the USB control register. 1: Enabled 0: Disabled (No response is returned for tokens.)
3	USB_RST	Indicates the bus reset signal status. 1: The bus reset signal is active. 0: The bus reset signal is inactive.
2	SPND_OUT	Indicates the PHY status. 1: PHY is in the Suspend state. 0: PHY is not in the Suspend state.
1	RSUM_OUT	Indicates the resume signal reception status on the PHY. 1: The resume signal is being received. 0: The resume is not being received.
0	–	Reserved. (Be sure to write 0b to this field.)

4.1.3 Frame number & USB address register (USB_ADDRESS)

This register is used to store the device address sent by the Set Address request.

Add	0008h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R W									R W	R W	R W	R W	R W	R W	R W	R	R	R	R		R	R	R	R	R	R	R	R	R	R	R
Symbol	SOF_DELAY_MODE									USB_ADDR[6:0]							SOF_STATUS	UFRAME[2:0]				FRAME[10:0]										
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	0	0	0	0	0	0	0	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
31	SOF_DELAY_MODE	Specify whether to enable a period in which an SOF is not accepted. If this bit is set to 1b, an SOF that is received within 64 cycles of the internal 60 MHz clock after an SOF is received normally or after an SOF recovery occurs is regarded as an illegal SOF token and ignored. 1: Enable a period in which an SOF is not accepted. 0: An SOF is accepted regardless of when it is received.
[30:23]	–	Reserved. (Be sure to write 0b to this field.)
[22:16]	USB_ADDR[6:0]	The value written to this field is determined as the device address after the status stage of the Set Address request completes normally. Write a value to this field before the status stage of the Set Address request completes. *This field is cleared to 0b when the bus reset signal is received*.
15	SOF_STATUS	Indicates the SOF (μSOF) reception status. This bit is set to 1b when an SOF (μSOF) reception error occurs. This bit is updated each time an SOF (μSOF) is received. 1: An SOF (μSOF) reception error occurred. 0: The SOF (μSOF) was received normally.
[14:12]	UFRAME[2:0]	Indicates the number of times the μSOF packet is received within a frame. If bit 8 (SOF_RCV) of the USB control register is set to 1b, this field is updated even if an automatic μSOF recovery occurs. This field is valid only when bit 5 (CONF) of the USB control register is 1b.
11	–	Reserved. (Be sure to write 0b to this field.)
[10:0]	FRAME[10:0]	Indicates the frame number of the SOF. If bit 8 (SOF_RCV) of the USB control register is set to 1b, this field is updated even if an automatic μSOF recovery occurs. This field is valid only when bit 5 (CONF) of the USB control register is 1b.

4.1.4 Setup data 0 register (SETUP_DATA0)

This register is used to store the first 4 bytes of 8-byte data received in a setup transaction.

Add	0018h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Symbol	SETUP4[7:0]								SETUP3[7:0]								SETUP2[7:0]								SETUP1[7:0]							
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:24]	SETUP4[7:0]	Stores the 4th byte of the received setup data. This field is updated each time setup data is received.
[23:16]	SETUP3[7:0]	Stores the 3rd byte of the received setup data. This field is updated each time setup data is received.
[15:8]	SETUP2[7:0]	Stores the 2nd byte of the received setup data. This field is updated each time setup data is received.
[7:0]	SETUP1[7:0]	Stores the 1st byte of the received setup data. This field is updated each time setup data is received.

4.1.5 Setup data 1 register (SETUP_DATA1)

This register is used to store the last 4 bytes of 8-byte data received in a setup transaction.

Add	001Ch																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Symbol	SETUP8[7:0]								SETUP7[7:0]								SETUP6[7:0]								SETUP5 [7:0]							
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:24]	SETUP8[7:0]	Stores the 8th byte of the received setup data. This field is updated each time setup data is received.
[23:16]	SETUP7[7:0]	Stores the 7th byte of the received setup data. This field is updated each time setup data is received.
[15:8]	SETUP6[7:0]	Stores the 6th byte of the received setup data. This field is updated each time setup data is received.
[7:0]	SETUP5[7:0]	Stores the 5th byte of the received setup data. This field is updated each time setup data is received.

4.1.6 USB interrupt status register (USB_INT_STA)

This register indicates the source of an interrupt output from the USB_INTF1 pin. The values of this register change when a state such as Bus Reset, Suspend, or Resume is detected, or according to the SOF reception status or the status of endpoint *n*.

Add	0020h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W											R	R	R	R	R	R	R	R	R	R	R	R	R	R		R W	R W	R W	R W	R W	R W	
Symbol											EP13_INT	EP12_INT	EP11_INT	EP10_INT	EP9_INT	EP8_INT	EP7_INT	EP6_INT	EP5_INT	EP4_INT	EP3_INT	EP2_INT	EP1_INT	EP0_INT		SPEED_MODE_INT	SOF_ERROR_INT	SOF_INT	USB_RST_INT	SPND_INT	RSUM_INT	
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:22]	–	Reserved. (Write 1b to this field.) *However, writing 0b causes no problem.*
[21:8]	EPn_INT/EP0_INT	Indicates whether an interrupt related to endpoint 0 or *n* has occurred. Read the EP0/EP*n* status register to determine the source. This field is cleared when 0b is written to the bit corresponding to the source in the EP0/EPn status register. Writing to this field does not clear the bit. (n: 1 to 13) This field is valid only for bits corresponding to endpoint numbers implemented in the system. The bits that do not correspond to an endpoint number are reserved. 1: An interrupt related to endpoint 0 or *n* has occurred. 0: No interrupt related to endpoint 0 or *n* has occurred.
7	–	Reserved. (Write 1b to this field.) *However, writing 0b causes no problem.*
6	SPEED_MODE_INT	Indicates a change in the speed mode. This bit is cleared by writing 0b. 1: The speed mode has changed from FS to HS. 0: The speed mode has not changed from FS to HS.
5	SOF_ERROR_INT	Indicates whether an SOF or μSOF reception error has occurred. This bit is set to 1b if an SOF or μSOF is not received during the period prescribed in the USB specification. In HS mode: 125 μs + 0.0625 μs In FS mode: 1 ms + 0.0005 ms This bit is cleared by writing 0b. This bit is invalid when bit 5 (CONF) of the USB control register is 0b. 1: An SOF or μSOF reception error has occurred. 0: No SOF or μSOF reception error has occurred.
4	SOF_INT	Indicates whether an SOF or μSOF has been received. This bit is cleared by writing 0b. This bit is invalid when bit 5 (CONF) of the USB control register is 0b. 1: An SOF or μSOF has been received. 0: No SOF or μSOF has been received.

Bit	Symbol	Description
3	USB_RST_INT	Indicates whether a bus reset has been issued. This bit is cleared by writing 0b. 1: A bus reset has been issued. 0: A bus reset has not been issued.
2	SPND_INT	Indicates whether the USB device is in the Suspend state. This bit is cleared by writing 0b. 1: The USB device is in the Suspend state. 0: The USB device is not in the Suspend state.
1	RSUM_INT	Indicates whether the resume signal sent from the USB host has been received. This bit is cleared by writing 0b. 1: The resume signal has been received. 0: The resume signal has not been received.
0	–	Reserved. (Write 1b to this field.) *However, writing 0b causes no problem.*

4.1.7 USB interrupt enable register (USB_INT_ENA)

This register is used to specify whether to enable or disable the interrupt sources assigned to the USB interrupt status register. If an interrupt source is disabled, the corresponding interrupt signal is not asserted even if the source occurs and the corresponding bit of the USB interrupt status register is set.

Add	0024h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W											R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W		R W	R W	R W	R W	R W	R W	
Symbol											EP13_EN	EP12_EN	EP11_EN	EP10_EN	EP9_EN	EP8_EN	EP7_EN	EP6_EN	EP5_EN	EP4_EN	EP3_EN	EP2_EN	EP1_EN	EP0_EN		SPEED_MODE_EN	SOF_ERROR_EN	SOF_EN	USB_RST_EN	SPND_EN	RSUM_EN	
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:24]	–	Reserved. (Be sure to write 0b to this field.)
[23:8]	EPn_EN/EP0_EN	Specify whether to enable bits [23:8] (EPn_INT/EP0_INT) of the USB interrupt status register. (n: 1 to 13) 1: Enable 0: Disable
7	–	Reserved. (Be sure to write 0b to this field.)
6	SPEED_MODE_EN	Specify whether to enable bit 6 (SPEED_MODE_INT) of the USB interrupt status register. 1: Enable 0: Disable
5	SOF_ERROR_EN	Specify whether to enable bit 5 (SOF_ERROR_INT) of the USB interrupt status register. 1: Enable 0: Disable
4	SOF_EN	Specify whether to enable bit 4 (SOF_INT) of the USB interrupt status register. 1: Enable 0: Disable
3	USB_RST_EN	Specify whether to enable bit 3 (USB_RST_INT) of the USB interrupt status register. 1: Enable 0: Disable
2	SPND_EN	Specify whether to enable bit 2 (SPND_INT) of the USB interrupt status register. 1: Enable 0: Disable
1	RSUM_EN	Specify whether to enable bit 1 (RSUM_INT) of the USB interrupt status register. 1: Enable 0: Disable
0	–	Reserved. (Be sure to write 0b to this field.)

4.1.8 EP0 control register (EP0_CONTROL)

This register is used to control endpoint 0.

Add	0028h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W														R W	R W	R W							W	W	W	W	W	W	W	R W	R W	R W
Symbol														EP0_STGSEL	EP0_OVERSEL	EP0_AUTO							EP0_PIDCLR	EP0_BCLR	EP0_DEND	EP0_DW1	EP0_DW0	EP0_INAK_EN	EP0_PERR_NAK_CLR	EP0_STL	EP0_INAK	EP0_ONAK
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:19]	–	Reserved. (Be sure to write 0b to this field.)
18	EP0_STGSEL	Select the operation when data other than null data is received at a status stage. Generally, do not change the default value (0b). If the value of this bit is changed during a transfer, the operation is not guaranteed. 1: Return a STALL. 0: Perform reception normally. Data other than null data is discarded and the packet is handled as null data. If an overrun occurs, however, no response is returned and the subsequent operation depends on the setting specified for bit 17 (EP0_OVERSEL).
17	EP0_OVERSEL	Select the operation when an overrun occurs during an OUT transfer and an OUT token is subsequently received. (No response is returned for the overrun packet.) Generally, do not change the default value (0b). If the value of this bit is changed during a transfer, the operation is not guaranteed. 1: Handle the next packet as a retry. 0: Return a STALL for the next packet.
16	EP0_AUTO	Select whether to automatically send a packet when a packet of the maximum packet size (64 bytes) is written to the EP0 write register (transmission buffer). Note that this feature might need to be disabled when sending a packet of the maximum packet size – 1, 2, or 3 bytes, as shown in the example below. If this feature is not disabled, 64 bytes of data become valid when writing the packet finishes. To prevent invalid data being included in the final written data, clear this bit to 0b and use bits 6 and 5 (EP0_DW[1:0]) instead. For example, when sending 61 to 63 bytes of data, it is handled as 64 bytes. **Caution Change the value of this bit while the buffer is empty.** 1: Set bit 7 (EP0_DEND). 0: Do not set bit 7 (EP0_DEND)
[15:10]	–	Reserved. (Be sure to write 0b to this field.)

Bit	Symbol	Description
9	EP0_PIDCLR	Write 1b to this bit to initialize the data PID for endpoint 0. As a result of setting this bit, both the transmission and reception PIDs are initialized to DATA1. If this bit is written during a USB transaction for endpoint 0, the write is held pending until the transaction finishes. The data PID is initialized when a bus reset is received. It is also initialized each time a SETUP token is received. Subsequent data PIDs are toggled by hardware. Therefore, setting this bit is generally not necessary. This bit is write-only. If it is read, 0b is always returned.
8	EP0_BCLR	Write 1b to this bit to clear the EP0 write and EP0 read registers (transmission and reception buffers). If this bit is written during a USB transaction for endpoint 0, the write is held pending until the transaction finishes. After this bit is set and before writing the next data, make sure bit 10 (EP0_IN_DATA) of the EP0 status register is 0b and bit 8 (EP0_IN_EMPTY) is 1b. **Caution The EP0 write and EP0 read registers are not cleared even if a USB bus reset is received.**
7	EP0_DEND	Write 1b to this bit to enable transmission of the data written to the EP0 write register (transmission buffer). When sending null data, make sure bit 8 (EP0_IN_EMPTY) of the EP0 status register is 1b, and then write 1b to this bit. This bit is write-only. If it is read, 0b is always returned.
[6:5]	EP0_DW[1:0]	Specify the number of valid bytes that were written last to the EP0 write register (transmission buffer). Set this field at the same time as setting bit 7 (EP0_DEND) to 1b. This field is write-only. If it is read, 0b is always returned. 00: 4 01: 1 10: 2 11: 3
4	EP0_INAK_EN	Write 1b to this bit to enable writing to bit 1 (EP0_INAK). This bit is write-only. If it is read, 0b is always returned. **Remark** This bit is used to prevent bit 1 (EP0_INAK) from being cleared unexpectedly when the EP0_INAK bit is set by hardware at the same time as a register is accessed by the firmware.
3	EP0_PERR_NAK_CLR	Write 1b to this bit to cancel the NAK returned when a token that indicates an unrecognized request configuration error is received. (For details about request configuration errors, see bit 17 (EP0_PERR_NAK) of the EP0 status register.) Generally, when a NAK is forcibly returned, bit 2 (EP0_STL) is set to return a STALL. Therefore, setting this bit is generally not necessary. This bit is write-only. If it is read, 0b is always returned.

Bit	Symbol	Description
2	EP0_STL	Use this bit to control the STALL returned in response to an IN, OUT, or PING token for endpoint 0. If this bit is set to 1b, a STALL is returned in response to all IN, OUT, and PING tokens for the data and status stages. Even if this bit is cleared to 0b, returning a STALL cannot be canceled. This bit is initialized each time a SETUP token is received. The setting specified for this bit takes precedence over the forced NAK response setting specified by bit 0 (EP0_ONAK) and bit 1 (EP0_INAK). This bit is set to 1b when: • data other than null data is received at a status stage while bit 18 (EP0_STGSEL) is 1b, or • an overrun occurs while bit 17 (EP0_OVERSEL) is 0b. 1: Return a STALL. 0: Do not return a STALL.
1	EP0_INAK	Use this bit to control the NAK returned in response to an IN token for endpoint 0. Generally ACK and NAK are controlled according to the status of the EP0 read register (reception buffer). Write 1b to this bit to send a NAK. This bit is set to 1b when a setup transaction completes normally. 1: Return a NAK even if data exists in the transmission buffer. 0: Transmit data if data exists in the transmission buffer.
0	EP0_ONAK	Use this bit to control the NAK returned in response to an OUT/PING token for endpoint 0. Generally ACK, NAK, and NYET are controlled according to the status of the EP0 read register (reception buffer). Write 1b to this bit to send a NAK. This bit is set to 1b when a setup transaction competes normally. 1: Return a NAK even if there is available space in the reception buffer. 0: Receive data if there is available space in the reception buffer.

4.1.9 EP0 status register (EP0_STATUS)

This register indicates the source of an interrupt related to endpoint 0, among the sources of interrupts output from the USB_INTF1 pin. If bit 8 (EP0_INT) of the USB interrupt status register is set, read the EP0_STATUS register to determine the interrupt source. This register also indicates the statuses of the SETUP tokens and data transmitted and received at endpoint 0, that the system has transitioned to a status stage, and the statuses of the buffers for endpoint 0.

The bits whose symbol names are prefixed by _INT indicate the interrupt sources. Even if another bit is set, USB_INTF1 is not asserted.

Add	002Ch																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W														R	R	R W	R W	R	R	R	R W	R	R	R	R W	R W	R W	R W	R W	R W	R W	R W
Symbol														EP0_PID	EP0_PERR_NAK	EP0_PERR_NAK_INT	EP0_OUT_NAK_INT	EP0_OUT_NULL	EP0_OUT_FULL	EP0_OUT_EMPTY	EP0_IN_NAK_INT	EP0_IN_DATA	EP0_IN_FULL	EP0_IN_EMPTY	EP0_OUT_NULL_INT	EP0_OUT_OR_INT	EP0_OUT_INT	EP0_IN_INT	EP0_STALL_INT	STG_END_INT	STG_START_INT	SETUP_INT
R	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:19]	–	Reserved. (Write 1b to this field.) *However, writing 0b causes no problem.*
18	EP0_PID	Indicates the value of the data PID to be sent next. 1: DATA1 0: DATA0
17	EP0_PERR_NAK	This bit is set to 1b when a NAK is forcibly returned due to reception of a token that indicates a request configuration error for endpoint 0. A NAK is returned in any of the following cases: • An IN or OUT token is received before a SETUP token is received (no setup stage) • An OUT token is received at the Control Read data stage • An IN token or an invalid OUT token is received at the Control Read status stage • An IN token or an invalid OUT token is received at the Control Write data stage • An OUT or PING token is received at the Control Write status stage • An OUT token is received at the No Data Control status stage While this bit is 1b, a NAK is sent in response to an IN, OUT, or PING token for endpoint 0. If this bit is set, set bit 2 (EP0_STL) of the EP0 control register to 1b to stall processing at endpoint 0. This bit is cleared to 0b each time a SETUP token is received. It is also cleared to 0b when 1b is written to bit 3 (EP0_PERR_NAK_CLR) of the EP0 control register. 1: A NAK was sent due to reception of a request configuration error token. 0: A NAK has not been sent even though a request configuration error token was received.
16	EP0_PERR_NAK_INT	This bit is set to 1b when a NAK is sent in response to a request configuration error

Bit	Symbol	Description
		token for endpoint 0. If this bit is set, set bit 2 (EP0_STL) of the EP0 control register to 1b to stall processing at endpoint 0. This bit is cleared by writing 0b. 1: A NAK was sent in response to a request configuration error token. 0: No request configuration error token has been received.
15	EP0_OUT_NAK_INT	This bit is set to 1b when a NAK is sent in response to an OUT or PING token for endpoint 0. This bit is cleared by writing 0b. 1: A NAK was sent in response to an OUT or PING token. 0: A NAK has not been sent in response to an OUT or PING token.
14	EP0_OUT_NULL	This bit is set to 1b when null data for endpoint 0 is received. This bit is updated each time received OUT data is stored in the EP0 read register (reception buffer). 1: Null data was received. 0: Null data has not been received
13	EP0_OUT_FULL	This bit is set to 1b if data of the maximum packet size (64 bytes) exists in the EP0 read register (reception buffer). This bit is updated when the buffer status changes. 1: The reception buffer is full. 0: The reception buffer is not full.
12	EP0_OUT_EMPTY	This bit is set to 1b if the EP0 read register (reception buffer) is empty. This bit is updated when the buffer status changes. 1: The reception buffer is empty. 0: The reception buffer is not empty.
11	EP0_IN_NAK_INT	This bit is set to 1b when a NAK is sent in response to an IN token for endpoint 0. This bit is cleared by writing 0b. 1: A NAK was sent in response to an IN token. 0: A NAK has not been sent in response to an IN token.
10	EP0_IN_DATA	This bit is set to 1b if there is data to be sent in the EP0 write register (transmission buffer) while bit 7 (EP0_DEND) of the EP0 control register is set. This bit is updated when the buffer status changes. 1: Data to be sent exists in the transmission buffer. 0: Data to be sent does not exist in the transmission buffer.
9	EP0_IN_FULL	This bit is set to 1b if the EP0 write register (transmission buffer) is full. This bit is updated when the buffer status changes. 1: The transmission buffer is full. 0: The transmission buffer is not full.
8	EP0_IN_EMPTY	This bit is set to 1b if the EP0 write register (transmission buffer) is empty. This bit is updated when the buffer status changes. 1: The transmission buffer is empty. 0: The transmission buffer is not empty.
7	EP0_OUT_NULL_INT	This bit is set to 1b when received null data is stored in the EP0 read register (reception buffer). This bit is cleared by writing 0b. 1: Null data was received. 0: Null data has not been received.
6	EP0_OUT_OR_INT	This bit is set to 1b if an overrun occurs while data is being received at endpoint 0. This bit is cleared by writing 0b. 1: An overrun occurred. 0: No overrun has occurred.
5	EP0_OUT_INT	This bit is set to 1b when storing valid data in the EP0 read register (reception buffer)

Bit	Symbol	Description
		has finished and the data can be read. This bit is cleared by writing 0b. 1: Data can be read from the reception buffer. 0: Data cannot be read from the reception buffer.
4	EP0_IN_INT	This bit is set to 1b when data in the EP0 write register (transmission buffer) has been sent normally and the next data can be written. This bit is cleared by writing 0b. 1: Data can be written to the transmission buffer. 0: Data cannot be written to the transmission buffer.
3	EP0_STALL_INT	This bit is set to 1b when processing at endpoint 0 is stalled. While bit 17 (EP0_OVERSEL) of the EP0 control register is 0, this bit is set to 1b when an overrun occurs. This bit is cleared by writing 0b. 1: Processing at endpoint 0 is stalled. 0: Processing at endpoint 0 is not stalled.
2	STG_END_INT	This bit is set to 1b when the status stage of a control transfer completes normally. This bit is cleared by writing 0b. This bit is cleared when the next SETUP token is received. 1: The status stage completed normally. 0: The status stage has not completed.
1	STG_START_INT	This bit is set to 1b when the status stage of a control transfer starts. This bit is cleared by writing 0b. This bit is cleared when the next SETUP token is received. 1: The status stage has started. 0: The status stage has not started.
0	SETUP_INT	This bit is set to 1b when valid setup data is received. At this time, to prevent the new setup data from being sent while the previously received request is being processed, clear this bit before processing of the request starts. This bit is cleared by writing 0b. 1: Valid setup data was received. 0: Valid setup data has not been received.

4.1.10 EP0 interrupt enable register (EP0_INT_ENA)

This register is used to specify whether to enable or disable the interrupt sources assigned to the EP0 status register. If an interrupt source is disabled, the corresponding interrupt signal is not asserted even if the source occurs and the corresponding bit of the EP0 status register is set.

Add	0030h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W																R W	R W				R W				R W	R W	R W	R W	R W	R W	R W	R W
Symbol																EP0_PERR_NAK_EN	EP0_OUT_NAK_EN				EP0_IN_NAK_EN				EP0_OUT_NULL_EN	EP0_OUT_OR_EN	EP0_OUT_EN	EP0_IN_EN	EP0_STALL_EN	STG_END_EN	STG_START_EN	SETUP_EN
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:17]	–	Reserved. (Be sure to write 0b to this field.)
16	EP0_PERR_NAK_EN	Specify whether to enable bit 16 (EP0_PERR_NAK_INT) of the EP0 status register. 1: Enable 0: Disable
15	EP0_OUT_NAK_EN	Specify whether to enable bit 15 (EP0_OUT_NAK_INT) of the EP0 status register. 1: Enable 0: Disable
[14:12]	–	Reserved. (Be sure to write 0b to this field.)
11	EP0_IN_NAK_EN	Specify whether to enable bit 11 (EP0_IN_NAK_EN) of the EP0 status register. 1: Enable 0: Disable
[10:8]	–	Reserved. (Be sure to write 0b to this field.)
7	EP0_OUT_NULL_EN	Specify whether to enable bit 7 (EP0_OUT_NULL_INT) of the EP0 status register. 1: Enable 0: Disable
6	EP0_OUT_OR_EN	Specify whether to enable bit 6 (EP0_OUT_OR_INT) of the EP0 status register. 1: Enable 0: Disable
5	EP0_OUT_EN	Specify whether to enable bit 5 (EP0_OUT_INT) of the EP0 status register. 1: Enable 0: Disable
4	EP0_IN_EN	Specify whether to enable bit 4 (EP0_IN_INT) of the EP0 status register. 1: Enable 0: Disable
3	EP0_STALL_EN	Specify whether to enable bit 3 (EP0_STALL_INT) of the EP0 status register. 1: Enable 0: Disable
2	STG_END_EN	Specify whether to enable bit 2 (STG_END_INT) of the EP0 status register. 1: Enable 0: Disable
1	STG_START_EN	Specify whether to enable bit 1 (STG_START_INT) of the EP0 status register. 1: Enable 0: Disable
0	SETUP_EN	Specify whether to enable bit 0 (SETUP_INT) of the EP0 status register.

Bit	Symbol	Description
		1: Enable 0: Disable

4.1.11 EP0 length register (EP0_LENGTH)

This register indicates the number of bytes in the OUT data received at endpoint 0.

Add	0034h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W																										R	R	R	R	R	R	R
Symbol																										EP0_LDATA[6:0]						
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:7]	–	Reserved. (Be sure to write 0b to this field.)
[6:0]	EP0_LDATA[6:0]	If OUT data is received normally in the EP0 read register (reception buffer), this field indicates the number of received bytes. The value is decremented each time data in the EP0 read register is read to indicate the number of bytes currently remaining in the EP0 read register.

4.1.12 EP0 read register (EP0_READ)

This register is used as a 64-byte reception buffer for endpoint 0.

Add	0038h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Symbol	EP0_RDATA4[7:0]								EP0_RDATA3[7:0]								EP0_RDATA2[7:0]								EP0_RDATA1[7:0]							
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:24]	EP0_RDATA4[7:0]	Data received at endpoint 0
[23:16]	EP0_RDATA3[7:0]	Data received at endpoint 0
[15:8]	EP0_RDATA2[7:0]	Data received at endpoint 0
[7:0]	EP0_RDATA1[7:0]	Data received at endpoint 0

4.1.13 EP0 write register (EP0_WRITE)

This register is used as a 64-byte transmission buffer for endpoint 0.

Add	003Ch																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
Symbol	EP0_WDATA4[7:0]								EP0_WDATA3[7:0]								EP0_WDATA2[7:0]								EP0_WDATA1[7:0]							
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:24]	EP0_WDATA4[7:0]	Data transmitted from endpoint 0
[23:16]	EP0_WDATA3[7:0]	Data transmitted from endpoint 0
[15:8]	EP0_WDATA2[7:0]	Data transmitted from endpoint 0
[7:0]	EP0_WDATA1[7:0]	Data transmitted from endpoint 0

4.1.14 EPn control register (EPn_CONTROL)

This register is used to control endpoint n (n: 1 to 13).

Add	0XX0h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R W	R				R W	R	R							R W	R W					W	W	W	W	W	W	W	W	R W	R W		R W
Symbol	EPn_EN	EPn_BUF_TYPE				EPn_DIR0	EPn_MODE[1:0]								EPn_OVERSEL	EPn_AUTO					EPn_IPIDCLR	EPn_OPIDCLR	EPn_BCLR	EPn_CBCLR	EPn_DEND	EPn_DW[1:0]		EPn_OSTL_EN	EPn_ISTL	EPn_OSTL		EPn_ONAK
R	0	–	0	0	0	0	–	–	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	0	0	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
31	EPn_EN	Specify whether to enable or disable endpoint n. If this bit is set while bit 5 (CONF) of the USB control register is 1b, the corresponding endpoint can respond to USB transactions. When this bit is cleared to 0b, the EPn write/read register (transmission/reception buffer) is cleared. If this bit is written during a USB transaction for endpoint *n*, the write is held pending until the transaction completes. 1: Enable endpoint *n*. 0: Disable endpoint *n*. **Cautions 1. By default, the value of this bit is assumed to be changed only when a bus reset is received, or when a USB request (such as Set Interface) is being processed.** **2. The data PID is not initialized even if this bit is cleared.**
30	EPn_BUF_TYPE	Indicates the type of EPn buffering. This bit is read-only. The buffering type cannot be changed by writing to this bit. 1: Double buffering 0: Single buffering **Caution Double-buffering endpoints operate as single-buffering endpoints when a hardware loopback occurs on the UDL side (when bit 0 of the TEST control register is 1b). Even in this case, however, this bit shows 1 (double buffering).**
[29:27]	–	Reserved. (Be sure to write 0b to this field.)
26	EPn_DIR0	Specify the direction of transfer at endpoint *n*. 1: Output 0: Input **Caution The value of this bit is assumed to be changed only during initial setup. If the value of this bit is changed during a transfer, the operation is not guaranteed.**

Bit	Symbol	Description
[25:24]	EPn_MODE[1:0]	Indicates the type of transfer performed at endpoint *n*. This bit is read-only. The endpoint transfer type cannot be changed by writing to this bit. 00b: Bulk 01b: Interrupt 10b: Reserved 11b: Reserved
[23:18]	–	Reserved. (Be sure to write 0b to this field.)
17	EPn_OVERSEL	Select the operation when an overrun occurs during an OUT transfer and an OUT is subsequently received. If the value of this bit is changed during a transfer, the operation is not guaranteed. Generally, do not change the default value (0b). 1: Handle the next packet as a retry. 0: Return a STALL for the next packet.
16	EPn_AUTO	Select whether to automatically send a packet when a packet of the maximum packet size (64 bytes) is written to the EPn write register (transmission buffer). Note that this feature might need to be disabled when sending a packet of the maximum packet size – 1, 2, or 3 bytes, as shown in the example below. If this feature is not disabled, 64 bytes of data become valid when writing the packet finishes. To prevent invalid data being included in the final written data, clear this bit to 0b and use bits 6 and 5 (EPn_DW[1:0]) instead. Example 1: If the CPU sends 509 to 511 bytes in 32-bit units while the maximum packet size is 512, 512 bytes are handled as valid data. Example 2: If the CPU sends 509 and 510 bytes in 32-bit units while the maximum packet size is 511, 511 bytes are handled as valid data. 1: Set bit 7 (EPn_DEND) automatically. 0: Do not set bit 7 (EPn_DEND) automatically. **Caution Changing the value of this bit is prohibited while bit 4 (EPn_DMA_EN) of the EPn DMA control register is 1b. In addition, do not enable this feature if the maximum packet size is 000h.**
[15:12]	–	Reserved. (Be sure to write 0b to this field.)
11	EPn_IPIDCLR	Write 1b to this bit to initialize the transmission data PID for endpoint *n*. If this bit is written during a USB transaction for endpoint *n*, the write is held pending until the transaction completes. This bit is write-only. If it is read, 0b is always returned. **Caution The data PID is initialized when a bus reset is received.**
10	EPn_OPIDCLR	Write 1b to this bit to initialize the reception data PID for endpoint *n*. If this bit is written during a USB transaction for endpoint *n*, the write is held pending until the transaction completes. This bit is write-only. If it is read, 0b is always returned. **Caution The data PID is initialized when a bus reset is received.**

Bit	Symbol	Description
9	EPn_BCLR	Write 1b to this bit to clear the EPn write and EPn read registers (transmission and reception buffers) on both the USB and CPU sides. If this bit is written during a USB transaction for endpoint *n*, the write is held pending until the transaction completes. If bit 26 (EPn_DIR0) is 0b (IN direction), after setting this bit, make sure bit 2 (EPn_IN_DATA) of the EPn status register is 0b and bit 8 (EPn_IN_EMPTY) of the EPn status register is 1b. This bit is write-only. If it is read, 0b is always returned. **Cautions 1. Setting this bit and bit 8 (EPn_CBCLR) at the same time is prohibited. In addition, setting this bit is prohibited if bit 4 (EPn_DMA_EN) of the EPn DMA control register is 1b.** **2. The EPn write and EPn read registers are not cleared even if a USB bus reset is received.**
8	EPn_CBCLR	Write 1b to this bit to clear the EPn write and EPn read registers (transmission and reception buffers) on the CPU side. This bit is invalid for single buffering endpoint. This bit is write-only. If it is read, 0b is always returned. **Caution Setting this bit and bit 9 (EPn_BCLR) at the same time is prohibited. In addition, setting this bit is prohibited if bit 4 (EPn_DMA_EN) of the EPn DMA control register is 1b.**
7	EPn_DEND	Write 1b to this bit to enable transmission of the data written to the EPn write register (transmission buffer). When sending null data, make sure bit 0 (EPn_IN_EMPTY) of the EPn status register is 1b, and then write 1b to this bit. **Caution Setting this bit is prohibited if bit 4 (EPn_DMA_EN) of the EPn DMA control register is 1b.**
[6:5]	EPn_DW[1:0]	Specify the number of valid bytes that were written last to the EPn write register (transmission buffer). Set this field at the same time as setting bit 7 (EPn_DEND) to 1b. This field is write-only. If it is read, 0b is always returned. 00b: 4 01b: 1 10b: 2 11b: 3
4	EPn_OSTL_EN	Write 1b to this bit to enable writing to bit 2 (EPn_OSTL). This bit is write-only. If it is read, 0b is always returned.
3	EPn_ISTL	Specify whether to return a STALL in response to an IN token for endpoint *n*. 1: Return a STALL in response to an IN token. 0: Do not return a STALL in response to an IN token.
2	EPn_OSTL	Specify whether to return a STALL in response to OUT or PING tokens for endpoint *n*. When setting this bit, be sure to set bit 4 (EPn_OSTL_EN) to 1b at the same time. This bit is set to 1b when an overrun occurs while bit 17 (EPn_OVERSEL) is 0b. 1: Return a STALL in response to an OUT or PING token. 0: Do not return a STALL in response to an OUT or PING token.
1	–	Reserved. (Be sure to write 0b to this field.)
0	EPn_ONAK	Use this bit to control the NAK returned in response to an OUT or PING token for endpoint *n*. Generally ACK, NAK, and NYET are controlled according to the status of the EP0 read register (reception buffer). Write 1b to this bit to send a NAK. 1: Return a NAK even if there is available space in the reception buffer. 0: Receive data if there is available space in the reception buffer.

4.1.15 EPn status register (EPn_STATUS)

This register indicates the source of an interrupt related to endpoint *n*, among the sources of interrupts output from the USB_INTF1 pin (n: 1 to 13). If bits 23 to 9 (EP[13:1]_INT) of the USB interrupt status register are set, read the EPn_STATUS register to determine the interrupt source. This register also indicates the statuses of data transmitted and received at endpoint n, errors, and the buffers for endpoint *n*.

The bits whose symbol names are prefixed by _INT indicate the interrupt sources. Even if another bit is set, USB_INTF1 is not asserted.

Add	0XX4h																														
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W				R					R W	R W	R W	R W	R W	R W	R	R						R			R W		R W	R W	R W	R	R	R
Symbol				EPn_OPID					EPn_OUT_END_INT	EPn_OUT_OR_INT	EPn_OUT_NAK_ERR_INT	EPn_OUT_STALL_INT	EPn_OUT_INT	EPn_OUT_NULL_INT	EPn_OUT_FULL	EPn_OUT_EMPTY						EPn_IPID			EPn_IN_END_INT		EPn_IN_NAK_ERR_INT	EPn_IN_STALL_INT	EPn_IN_INT	EPn_IN_DATA	EPn_IN_FULL	EPn_IN_EMPTY
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1
B	–	–	–	0	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	0	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:30]	–	Reserved. (Write 1b to this field.) *However, writing 0b causes no problem.*
29	–	Reserved
28	EPn_OPID	Indicates the value of the normal data PID to be received next. 1: DATA1 0: DATA0
27	–	Reserved
26	–	Reserved
25	–	Reserved. (Write 1b to this field.) *However, writing 0b causes no problem.*
24	–	Reserved
23	EPn_OUT_END_INT	This bit is set to 1b when an OUT-direction DMA transfer to read the buffer at endpoint *n* completes. When this bit is set, bit 4 (EPn_DMA_EN) of the EPn DMA control register is cleared to 0b. If bit 11 (EPn_STOP_MODE) and bit 8 (EPn_STOP_SET) of the EPn DMA control register are 1b, this bit is set to 1b when a short packet is received while the EPn_DMA_EN bit is 1b. This bit is cleared by writing 0b. 1: A DMA transfer to read the buffer has completed. 0: A DMA transfer to read the buffer is in progress.
22	EPn_OUT_OR_INT	This bit is set to 1b if an overrun occurs while data is being received at endpoint *n*. This bit is cleared by writing 0b. 1: An overrun occurred. 0: No overrun has occurred.

Bit	Symbol	Description
21	EPn_OUT_NAK_ERR_INT	This bit is set to 1b when a NAK is sent from endpoint *n* in response to an OUT or PING token during an interrupt or bulk transfer. • During interrupt or bulk transfer 1: A NAK was sent in response to an OUT or PING token. 0: A NAK has not been sent in response to an OUT or PING token.
20	EPn_OUT_STALL_INT	This bit is set to 1b when processing at endpoint *n* is stalled while bit 26 (EPn_DIR0) of the USB control register is 1b (OUT direction). This bit is cleared by writing 0b. 1: Processing at endpoint *n* is stalled. 0: Processing at endpoint *n* is not stalled.
19	EPn_OUT_INT	This bit is set to 1b when data other than normally received null data is stored in the EPn read register (reception buffer) and the data can be read from the buffer on the CPU side. If null data is received, bit 18 (EPn_OUT_NULL_INT) is set to 1b. This bit is cleared by writing 0b. 1: Data can be read from the reception buffer. 0: Data cannot be read from the reception buffer.
18	EPn_OUT_NULL_INT	This bit is set to 1b when null data is received and stored in the EPn read register (reception buffer) normally. For double-buffering, this bit is set to 1b when null data packets are copied to the CPU side. Null data packets are cleared when this bit is set. The reception buffer is then ready for the next packet reception. This bit is cleared by writing 0b. 1: Null data was received. 0: Null data has not been received.
17	EPn_OUT_FULL	This bit is set to 1b if the EPn read register (reception buffer) is full. For double-buffering, this bit is set to 1b if the buffer on the CPU side is full. This bit is updated when the reception buffer status changes. 1: The reception buffer is full. 0: The reception buffer is not full.
16	EPn_OUT_EMPTY	This bit is set to 1b if the EPn read register (reception buffer) is empty. For double-buffering, this bit is set to 1b if the buffer on the CPU side is empty. This bit is updated when the reception buffer status changes. 1: The reception buffer is empty. 0: The reception buffer is not empty.
[15:11]	–	Reserved. (Write 1b to this field.) *However, writing 0b causes no problem.*
10	EPn_IPID	Indicates the value of the data PID to be sent next. 1: DATA1 0: DATA0
9	–	Reserved
8	–	Reserved
7	EPn_IN_END_INT	This bit is set to 1b when an IN-direction DMA transfer to write to the buffer at endpoint *n* completes. If this bit is set, bit 4 (EPn_DMA_EN) of the EPn DMA control register is cleared to 0b. This bit is cleared by writing 0b. 1: A DMA transfer to write to the buffer has completed. 0: A DMA transfer to write to the buffer is in progress.
6	–	Reserved. (Write 1b to this field.) *However, writing 0b causes no problem.*

Bit	Symbol	Description
5	EPn_IN_NAK_ERR_INT	This bit is set to 1b when a NAK is sent from endpoint *n* in response to an IN token during an interrupt or bulk transfer. • During interrupt or bulk transfer 1: A NAK was sent in response to an IN token. 0: A NAK has not been sent in response to an IN token.
4	EPn_IN_STALL_INT	This bit is set to 1b when processing at endpoint *n* is stalled while bit 26 (EPn_DIR0) of the USB control register is 0b (IN direction). This bit is cleared by writing 0b. 1: Processing at endpoint *n* is stalled. 0: Processing at endpoint *n* is not stalled.
3	EPn_IN_INT	This bit is set to 1b when data in the EPn write register (transmission buffer) has been sent normally and the next data can be written to the buffer on the CPU side. • For single-buffering, this bit is set to 1b when data is received normally during an IN transaction. • For double-buffering, this bit is set to 1b when the data packet in the transmission-enabled buffer on the CPU side is copied to the USB device side. This bit is cleared by writing 0b. 1: Data can be written to the transmission buffer. 0: Data cannot be written to the transmission buffer.
2	EPn_IN_DATA	This bit is set to 1b if there is data to be sent in the EPn write register (transmission buffer). This bit is updated when the buffer status changes. 1: Data to be sent exists in the transmission buffer. 0: Data to be sent does not exist in the transmission buffer.
1	EPn_IN_FULL	This bit is set to 1b when the EPn write register (transmission buffer) on the CPU side becomes full. This bit is cleared to 0b when bit 7 (EPn_DEND) of the EPn control register is set to 1b to enable data transmission. This bit is updated when the EPn write register (transmission buffer) status changes. 1: The transmission buffer is full. 0: The transmission buffer is not full.
0	EPn_IN_EMPTY	This bit is set to 1b when the EPn write register (transmission buffer) on the CPU side becomes empty. This bit is updated when the EPn write register (transmission buffer) status changes. 1: The transmission buffer is empty. 0: The transmission buffer is not empty.

4.1.16 EPn interrupt enable register (EPn_INT_ENA)

This register is used to specify whether to enable or disable the interrupt sources assigned to the EP*n* status register (n: 1 to 13). If an interrupt source is disabled, the corresponding interrupt signal is not asserted even if the source occurs and the corresponding bit of the EP*n* interrupt status register is set.

Add	0XX8h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W									R W	R W	R W	R W	R W	R W											R W		R W	R W	R W			
Symbol									EPn_OUT_END_EN	EPn_OUT_OR_EN	EPn_OUT_NAK_ERR_EN	EPn_OUT_STALL_EN	EPn_OUT_EN	EPn_OUT_NULL_EN											EPn_IN_END_EN		EPn_IN_NAK_ERR_EN	EPn_IN_STALL_EN	EPn_IN_EN			
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:24]	–	Reserved. (Be sure to write 0b to this field.)
23	EPn_OUT_END_EN	Specify whether to enable bit 23 (EPn_OUT_END_INT) of the EPn status register. 1: Enable 0: Disable
22	EPn_OUT_OR_EN	Specify whether to enable bit 22 (EPn_OUT_OR_INT) of the EPn status register. 1: Enable 0: Disable
21	EPn_OUT_NAK_ERR_EN	Specify whether to enable bit 21 (EPn_OUT_NAK_ERR_INT) of the EPn status register. 1: Enable 0: Disable
20	EPn_OUT_STALL_EN	Specify whether to enable bit 20 (EPn_OUT_STALL_INT) of the EPn status register. 1: Enable 0: Disable
19	EPn_OUT_EN	Specify whether to enable bit 19 (EPn_OUT_INT) of the EPn status register. 1: Enable 0: Disable
18	EPn_OUT_NULL_EN	Specify whether to enable bit 18 (EPn_OUT_NULL_INT) of the EPn status register. 1: Enable 0: Disable
[17:8]	–	Reserved. (Be sure to write 0b to this field.)
7	EPn_IN_END_EN	Specify whether to enable bit 7 (EPn_IN_END_INT) of the EPn status register. 1: Enable 0: Disable
6	–	Reserved. (Be sure to write 0b to this field.)
5	EPn_IN_NAK_ERR_EN	Specify whether to enable bit 5 (EPn_IN_NAK_ERR_INT) of the EPn status register. 1: Enable 0: Disable
4	EPn_IN_STALL_EN	Specify whether to enable bit 4 (EPn_IN_STALL_INT) of the EPn status register. 1: Enable 0: Disable

Bit	Symbol	Description
3	EPn_IN_EN	Specify whether to enable bit 3 (EPn_IN_INT) of the EPn status register. 1: Enable 0: Disable
[2:0]	–	Reserved. (Be sure to write 0b to this field.)

4.1.17 EPn DMA control register (EPn_DMA_CTRL)

This register is used to set up DMA at endpoint *n* (n: 1 to 13).

Add	0XXCh																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W																					R W	R W	R W	R W				R W				R W
Sym bol																					EPn_STOP_MODE	EPn_DEND_SET	EPn_BURST_SET	EPn_STOP_SET				EPn_DMA_EN				EPn_DMAMODE0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:12]	–	Reserved. (Be sure to write 0b to this field.)
11	EPn_STOP_MODE	Specify the DMA stop conditions when bit 8 (EPn_STOP_SET) is 1b. 1: DMA transfer stops when a short packet is received and it is ready to be read out. (DMA transfer does stop if bit 4 (EPn_DMA_EN) is set to 1b when a previously received short packet is ready to be read out.) 0: DMA transfer stops when a short packet is received, and that packet is the last data.
10	EPn_DEND_SET	Specify whether to enable setting bit 7 (EPn_DEND) of the EPn control register to 1b when a DMA completion signal sent from the AHB-EPC bridge is received during an IN transaction (when bit 26 (EPn_DIR0) of the USB control register is 0). This bit is invalid for OUT transactions (EPn_DIR0 is 1b). Note that data less than the bus width cannot be transferred, regardless of the setting of this bit. 1: Enable 0: Disable

Bit	Symbol	Description
9	EPn_BURST_SET	Specify whether to enable clearing bit 4 (EPn_DMA_EN) to 0b each time one packet is transferred by using DMA. 1: Enable 0: Disable **Notes on writing by using DMA** (1) Set this bit and bit 16 (EPn_AUTO) of the EPn control register to 1b when successively transferring data of the maximum packet size by using DMA. (2) Clear this bit to 0b when transferring a short packet. If the last packet contains less than the maximum amount of data, write the data by using a PIO transfer. **Notes on reading by using DMA** (1) If a short packet that includes null data is received while bit 8 (EPn_STOP_SET) is 1b, bit 4 (EPn_DMA_EN) is cleared to 0b even if this bit is 1b. (2) Set this bit to 1b when successively transferring data of the maximum packet size by using DMA. (3) Clear this bit to 0b when transferring a short packet. If the last packet contains less than the maximum amount of data, read the data by using a PIO transfer.
8	EPn_STOP_SET	Select whether to clear bit 4 (EPn_DMA_EN) 0b and send the DMA completion signal to the AHB-EPC bridge to stop a DMA transfer if a short packet that includes null data is received while bit 26 (EPn_DIR0) of the USB control register is 1b (OUT direction). Use bit 11 (EPn_STOP_MODE) to specify how to stop a DMA transfer. Set this bit to 1b. 1: Clear bit 4 (EPn_DMA_EN) and send the DMA completion signal to the AHB-EPC bridge. 0: Do not clear bit 4 (EPn_DMA_EN) and do not send the DMA completion signal to the AHB-EPC bridge.
[7:5]	–	Reserved. (Be sure to write 0b to this field.)
4	EPn_DMA_EN	Specify whether to use DMA at endpoint *n*. 1: Use DMA. 0: Do not use DMA. This bit is cleared to 0b under the following conditions: (1) The DMA transfer specified by using the EPnDCR1 has completed. (2) When one packet has been transferred by using DMA while bits 24 to 16 (EPn_DMACNT) of the EPn length & DMA count register is 1b. (3) When one packet has been transferred by using DMA while bit 9 (EPn_BURST_SET) is 0b. (If the last packet contains less than the maximum amount of data, bits other than the last packet are cleared to 0b.) (4) If a short packet that includes null data is received while bit 8 (EPn_STOP_SET) is 1b and the data has been transferred by using a DMA transfer. (If bit 11 (EPn_STOP_MODE) is 1b, the DMA transfer stops when reading a short packet that includes null data becomes possible.) This bit is not cleared even if processing at endpoint *n* is stalled due to an overrun. If this bit is 1b, the EPn write and EPn read registers cannot be accessed by using PIO. Similarly, setting bit 7 (EPn_DEND) of the EPn control register is prohibited. Be sure to clear this bit to 0b before writing or reading data by using a PIO access after a DMA transfer completes or when setting the EPn_DEND bit. Before clearing this bit, stop the DMA master device in advance so that no transfer is performed. If this bit is cleared during a DMA transfer, the transferred data is not guaranteed.
[3:1]	–	Reserved. (Be sure to write 0b to this field.)

Bit	Symbol	Description
0	EPn_DMAMODE0	Select the DMA mode. Set this bit to 1b in the initial setup and do not change the value. 1: Demand mode 0: Single mode

4.1.18 EPn max packet & base address register (EPn_PCKT_ADRS)

This register is used to specify the maximum packet size to be transferred at each endpoint and the address to which each endpoint is mapped on the RAM (n: 1 to 13).

Add	0XX0h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Symbol				EPn_BASEAD[12:0]																		EPn_MPKT[10:0]										
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:29]	–	Reserved. (Be sure to write 0b to this field.)
[28:16]	EPn_BASEAD[12:0]	Specify the address to which the buffers of endpoint *n* are mapped to the RAM. For how to specify the address, see *11.2 Specifying the Base Address*. Specify values for this bit while bit 31 (EPn_EN) of the EPn control register is 0b.
[15:11]	–	Reserved. (Be sure to write 0b to this field.)
[10:0]	EPn_MPKT[10:0]	Specify the maximum packet size to be transferred at endpoint *n*. Specify values for this bit while bit 31 (EPn_EN) of the EPn control register is 0b.

4.1.19 EPn length & DMA count register (EPn_LEN_DCNT)

This register indicates the number of bytes received in the EPn read register (reception buffer) and is used to specify the number of packets to be sent by DMA (n: 1 to 13).

Add	0XX4h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W								RW	RW	RW	RW	RW	RW	RW	RW	RW						R	R	R	R	R	R	R	R	R	R	R
Symbol								EPn_DMACNT[8:0]														EPn_LDATA[10:0]										
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:25]	–	Reserved. (Be sure to write 0b to this field.)
[24:16]	EPn_DMACNT[8:0]	Specify the number of packets to be transferred successively by using DMA (burst DMA transfer). The value is decremented each time one packet is transferred by using DMA. When this field is 001h, bit 4 (EPn_DMA_EN) of the EPn DMA control register is cleared to 0b when one packet has been transferred by using DMA. Specify values for this field only when the EPn_DMA_EN is 0b. To enable burst DMA transfer, set bit 9 (EPn_BURST_SET) of the EPn DMA control register to 1b. Burst DMA transfer is disabled when this field is set to 000h. **Cautions 1. For the USB function controller, the maximum settable value is 100h (= 256 packets), according to the maximum settable value of the AHB-EPC bridge.** **2. Do not specify the number of transferred bytes.**
[15:11]	–	Reserved. (Be sure to write 0b to this field.)
[10:0]	EPn_LDATA[10:0]	Indicates the number of bytes stored in the EPn read register (reception buffer) on the CPU side that are ready to be read. The value is decremented each time data in the EPn read register is read.

4.1.20 EPn read register (EPn_READ)

This register is used as a reception buffer for endpoint *n* (n: 1 to 13).

Add	0XX8h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Symbol	EPn_RDATA4[7:0]								EPn_RDATA3[7:0]								EPn_RDATA2[7:0]								EPn_RDATA1[7:0]							
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:24]	EPn_RDATA4[7:0]	Data received at endpoint *n*
[23:16]	EPn_RDATA3[7:0]	Data received at endpoint *n*
[15:8]	EPn_RDATA2[7:0]	Data received at endpoint *n*
[7:0]	EPn_RDATA1[7:0]	Data received at endpoint *n*

4.1.21 EPn write register (EPn_WRITE)

This register is used as a transmission buffer for endpoint *n* (n: 1 to 13).

Add	0XXCh																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
Symbol	EPn_WDATA4[7:0]								EPn_WDATA3[7:0]								EPn_WDATA2[7:0]								EPn_WDATA1[7:0]							
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Bit	Symbol	Description
[31:24]	EPn_WDATA4[7:0]	Data transmitted from endpoint *n*
[23:16]	EPn_WDATA3[7:0]	Data transmitted from endpoint *n*
[15:8]	EPn_WDATA2[7:0]	Data transmitted from endpoint *n*
[7:0]	EPn_WDATA1[7:0]	Data transmitted from endpoint *n*

4.2 AHB-EPC Bridge Registers

4.2.1 AHBSCTR register

This register is used to set up the AHB slave features.

Add	1000h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W																																R W
Symbol																																WAIT_MODE
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	Symbol	Description
[31:1]	–	Reserved. (Be sure to write 0b to this field.)
0	WAIT_MODE	Set this bit to 1b in the initial setup and do not change the value.

4.2.2 AHBMCTR register

This register is used to set up the AHB master features.

Add	1004h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W																														R W		
Symbol																														WBURST_TYPE		
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

Bit	Symbol	Description
[31:6]	–	Reserved. (Be sure to write 0b to this field.)
[5:4]	–	Reserved. (Be sure to write *1b, not 0b*, to this field.)
3	–	Reserved. (Be sure to write 0b to this field.)
2	WBURST_TYPE	Set this bit to 1b in the initial setup and do not change the value.
[1:0]	–	Reserved. (Be sure to write 0b to this field.)

4.2.3 AHBBINT register

This register indicates the source of interrupts to the AHB-EPC bridge.

Add	1008h																														
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W	R W											R W		R W	R	R	R	R
Symbol	DMA_ENDINT[13:1]																									MBUS_ERRINT		SBUS_ERRINT0	ERR_MASTER[3:0]			
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Description
[31:17]	DMA_ENDINT[13:1]	When a DMA transfer completes, the bit corresponding to the relevant endpoint number is set to 1b. This field is cleared by writing 1b. Bit 29 corresponds to DMA_ENDINT13. Other bits correspond in descending order, with bit 17 corresponding to DMA_ENDINT1. *This field is valid only for bits corresponding to endpoint numbers implemented in the system. The bits that do not correspond to an endpoint number are reserved. Write 0b to the reserved bits.* 1: A DMA transfer at an endpoint has completed. 0: A DMA transfer at an endpoint has not completed.
[16:7]	–	Reserved. (Be sure to write 0b to this field.)
6	MBUS_ERRINT	Indicates whether an error response was received during AHB master operation. This field is cleared by writing 1b. 1: An error response was received. 0: No error response has been received.
5	–	Reserved. (Be sure to write 0b to this field.)
4	SBUS_ERRINT0	Indicates whether an error was returned in response to an AHB slave access in 32-bit or larger units. This field is cleared by writing 1b. 1: An error was returned in response to an access in 32-bit or larger units. 0: No error has been returned.
[3:0]	ERR_MASTER[3:0]	Use this field to store the number of the master that returned an error when bit 4 (SBUS_ERRINT0) is 1b. This field retains its value even if another interrupt source occurs until the SUBS_ERRINT0 bit is cleared to 0b.

4.2.4 AHBBINTEN register

This register is used to specify whether to enable or disable the interrupt sources assigned to the AHBBINT register. If an interrupt source is disabled, the corresponding interrupt signal is not asserted even if the source occurs and the corresponding bit of the AHBBINT register is set.

Add	100Ch																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW											RW		RW				
Symbol	DMA_ENDINTEN[13:1]																									MBUS_ERRINTEN		SBUS_ERRINT0EN				
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Description
[31:17]	DMA_ENDINTEN[13:1]	Specify whether to enable bits 31 to 17 (DMA_ENDINT) of the AHBBINT register. Bit 29 corresponds to DMA_ENDINTEN13. Other bits correspond in descending order, with bit 17 corresponding to DMA_ENDINTEN1. *This field is valid only for bits corresponding to endpoint numbers implemented in the system. The bits that do not correspond to an endpoint number are reserved. Write 0b to the reserved bits.* 1: Enable 0: Disable
[16:7]	–	Reserved. (Be sure to write 0b to this field.)
6	MBUS_ERRINTEN	Specify whether to enable bit 6 (MBUS_ERRINT) of the AHBBINT register. 1: Enable 0: Disable
5	–	Reserved. (Be sure to write 0b to this field.)
4	SBUS_ERRINT0EN	Specify whether to enable bit 4 (SBUS_ERRINT0) of the AHBBINT register. 1: Enable 0: Disable
[3:0]	–	Reserved. (Be sure to write 0b to this field.)

4.2.5 EPCTR register

This register is used to specify various items for controlling the USB function controller.

Add	1010h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W																				R W							R W	R				R W
Symbol																				DIRPD							PLL_RESUME	PLL_LOCK				EPC_RST
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	–	0	0	0	–	0	**Note**	**Note**	**Note**

Note Value after the USB1_RSTZ signal is asserted. These bits are cleared to 0b after the reset sequence completes.

Bit	Symbol	Description
[31:13]	–	Reserved. (Be sure to write 0b to this field.)
12	DIRPD	Set this bit to 1b to set the direct power-down mode. Clear this bit to 0b to exit from the direct power-down mode. For details about setting and exiting the direct power-down mode, see *10.3 Direct Power-Down Feature*. 1: Direct power-down mode 0: Normal operation
[11:6]	–	Reserved. (Be sure to write 0b to this field.)
5	PLL_RESUME	When the clock supply to the EPC and SIE blocks has been stopped while the PHY is in the Suspend state, set this bit to 1b to resume clock supply. After the clock supply is resumed, be sure to clear this bit to 0b. 1: Resume clock supply 0: Normal operation
4	PLL_LOCK	Indicates whether the PLL circuit has been locked up. 1: The PLL has been locked up. 0: The PLL has not been locked up.
[3:1]	–	Reserved. (Be sure to write 0b to this field.)
0	EPC_RST	Control the reset signal to be issued to the EPC and SIE blocks, and PHY interface. 1: Issue the EPC reset signal. 0: Deassert the EPC reset signal.

4.2.5 EPTEST register

This register is pin status detection of DP,DM

Add	1014h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W																							R	R								
Symbol																							F_LINESTATE[1]	F_LINESTATE[0]								
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Description
[31:10]	–	Reserved. (Be sure to write 0b to this field.)
[9]	F_LINESTATE[1]	Pin status detection of a DP.
[8]	F_LINESTATE[0]	Pin status detection of a DM.
[7:0]	–	Reserved. (Be sure to write 0b to this field.)

This register uses for 0xE2800000=0 after setting.

In case of VBUS detection, input of DP, DM is masked for input floating prevention.

A mask is released by 0xE2800000 bit[3]=0, and the state of the DP, DM can be confirmed.

4.2.6 USBSSVER register

This register indicates the versions of the implemented macros and USB function controller.

Add	1020h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W									R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Symbol									AHBB_VER[7:0]								EPC_VER[7:0]								SS_VER[7:0]							
R	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0

Bit	Symbol	Description
[31:24]	–	Reserved. (Be sure to write 0b to this field.)
[23:16]	AHBB_VER[7:0]	Indicates the AHB bridge version.
[15:8]	EPC_VER[7:0]	Indicates the endpoint controller version.
[7:0]	SS_VER[7:0]	Indicates the USB function controller version.

4.2.7 USBSSCONF register

This register shows the configuration of each endpoint.

Add	1024h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Symbol	EP_AVAILABLE[15:0]																DMA_AVAILABLE[15:0]															
R	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	0

Bit	Symbol	Description
[31:16]	EP_AVAILABLE[15:0]	Indicates whether an endpoint is available. Each bit in this field corresponds to an endpoint number. Because endpoint 0 is always available, bit 0 always indicates 1b. The value read from this field depends on the configuration of the incorporated endpoint controller. 1: Available 0: Not available
[15:0]	DMA_AVAILABLE [15:0]	Indicates whether an endpoint can be used for DMA transfers. Each bit in this field corresponds to an endpoint number. Because endpoint 0 cannot be used for DMA transfers, bit 0 always indicates 0b. The value read from this field depends on the configuration of the incorporated endpoint controller. 1: Usable 0: Not usable

4.2.8 EPnDCR1 register

This register is used to specify the parameters related to DMA transfers at endpoint *n* (n: 1 to 13).

Add	11X0h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W									R W	R W	R W	R W	R W	R W	R W	R W															R W	R W
Symbol									EPn_DMACNT[7:0]																						EPn_DIR0	EPn_REQEN
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Description
[31:24]	–	Reserved. (Be sure to write 0b to this field.)
[23:16]	EPn_DMACNT[7:0]	Specify the number of packets (not bytes) to be sent by using a DMA transfer. This field uses the same value as that of the EPn_DMACNT[8:0] bits of the EPn length & DMA counter register, except for processing that handles data packets that cannot be divided by 32 bits. However, if the EPn_DMACNT[8:0] bits are set to 100h, set this field to 00h. The value of this field is decremented each time one packet has been transferred by using a DMA transfer. **Caution Changing the value of this field is prohibited if bit 1 (EPn_REQEN) of the EPnDCR1 register is 1b.**
[15:2]	–	Reserved. (Be sure to write 0b to this field.)
1	EPn_DIR0	Specify the DMA transfer direction. Specify the same value as that of the EPn_DIR0 bit of the EPn control register. **Caution Changing the value of this field is prohibited if bit 1 (EPn_REQEN) of the EPnDCR1 register is 1b.** 1: OUT direction (from EPC to AHB) 0: IN direction (from AHB to EPC)
0	EPn_REQEN	Specify whether to enable DMA transfer requests from the endpoint controller. This bit is cleared to 0b when the number of packets specified by the EPn_DMACNT bits have been transferred or when the EPC receives a short packet and it ends the DMA transfer. 1: Enabled (DMA transfer is enabled) 0: Disabled (DMA transfer is disabled)

4.2.9 EPnDCR2 register

This register is used to specify the parameters related to DMA transfers at endpoint *n* (n: 1 to 13).

Add	11X4h																														
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W						R W	R W	R W	R W	R W	R W	R W	R W	R W	R	R						R W	R W	R W	R W	R W	R W	R W	R W	R W	R	R
Symbol						EPn_LMPKT[10:0]																EPn_MPKT[10:0]										
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Description
[31:27]	–	Reserved. (Be sure to write 0b to this field.)
[26:16]	EPn_LMPKT[10:0]	Indicates or used to specify the number of bytes in the last packet transferred by using DMA. The effect of this field depends on the DMA transfer direction. • IN transfer (bit 2 (EPn_DIR0) of EPnDCR1 register = 0b) Specify the number of bytes in the last packet to be transferred by using DMA. The DMA transfer completes when the specified maximum packet size has been transferred. Because data is transferred in 32-bit units, the lower 2 bits of this field are ignored. If the last packet contains less than 3 bytes, write the data by using a PIO transfer. Example: - If the maximum packet size is 512 bytes, set this field to 200h (512 bytes). - If the maximum packet size is 511 bytes (short packet), set this field to 1FCh (508 bytes). **Caution Changing the value of this field is prohibited if bit 1 (EPn_REQEN) of the EPnDCR1 register is 1b.** • OUT transfer (bit 2 (EPn_DIR0) of EPnDCR1 register = 1b) Indicates the number of bytes in the last packet transferred by using DMA. Because data is transferred in 32-bit units, the lower 2 bits of this field are ignored. Writing to this field is invalid during OUT transfers.
[15:11]	–	Reserved. (Be sure to write 0b to this field.)
[10:0]	EPn_MPKT[10:0]	Specify the maximum packet size to be transferred at endpoint *n*. Specify the same value as that of the EPn_MPKT[10:0] bits of the EPn max packet & base address register. **Caution Changing the value of this field is prohibited if bit 1 (EPn_REQEN) of the EPnDCR1 register is 1b.**

4.2.10 EPnTADR register

This register is used to specify the transfer start address on the AHB side for DMA transfers at endpoint *n*. (n: 1 to 13)

Add	11X8h																															
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	R	R
Symbol	EPn_TADR[31:0]																															
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	Symbol	Description
[31:0]	EPn_TADR[31:0]	Specify the transfer start address on the AHB side for DMA transfers. Because data is transferred in 32-bit units, the lower 2 bits of this field are ignored. **Caution Changing the value of this field is prohibited if bit 1 (EPn_REQEN) of the EPnDCR1 register is 1b.**

5. Accessing Registers

5.1 Notes on Accessing EPC Registers

When accessing a register in the EPC area, make sure the clock is supplied to the USB function controller. If not, the operation might deadlock on the AHB bus. Read bit 4 (PLL_LOCK) of the EPCTR register to check the clock supply status.

Be sure to check the clock supply status in the following cases:

- When accessing an EPC register for the first time after the PLL reset signal is deasserted
- When the resume signal is deasserted by using the remote wakeup feature. (See *10.2.2 Remote wakeup* for details.)

5.2 Notes on Accessing a Reserved Area

Address spaces that do not correspond to endpoint numbers implemented in the system are reserved, so do not access these spaces.

6. Clock System

6.1 Externally Supplied Clocks

The USB function controller requires the following clocks to be supplied externally:

Table 6-1. Externally Supplied Clocks

Clock Signal	Description	Restriction on Frequency
USB1_CLK	AHB clock	During communication: 0 < USB1_CLK ≤ 133 MHz When communication is not performed: 0 ≤ USB1_CLK ≤ 133 MHz
OSC1 clock	USB reference clock	24 MHz

The clocks are described below.

- USB1_CLK

  The USB1_CLK signal is supplied to the AHB-EPC bridge. Connect the AHB clock to the USB1_CLK clock. This clock can be stopped temporarily to reduce the power consumed by the AHB-EPC bridge while the USB device is disconnected from the main system.

  The USB1_CLK signal status cannot be changed dynamically, except for stopping the clock.

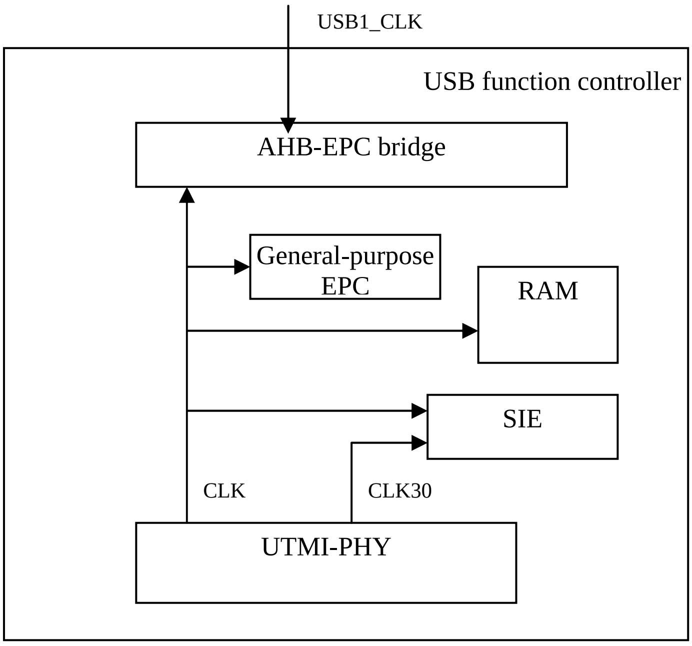
- OSC1 clock

  The OSC1 clock is used as a reference clock for generating a 480 MHz clock, which is used for USB transfers. The frequency is 24 MHz.

6.2 Clock System Diagram

Figure 6-1 shows the clock system of the USB function controller.

Figure 6-1. Clock System Diagram



7. Reset System

7.1 Reset Signals

The USB function controller is reset by using the USB1_RSTZ signal. When the USB1_RSTZ signal is asserted, all the circuits in the USB function controller are reset. The USB1_RSTZ signal functions as a power-on reset, and is assumed to be connected to the USB1_RSTZ signal line on the AHB. The USB1_RSTZ signal output from the USB function controller is an asynchronous reset signal, which is directly connected to the F/F reset pin.

Table 7-1 lists the reset signals for the USB function controller.

Table 7-1. Reset Signals

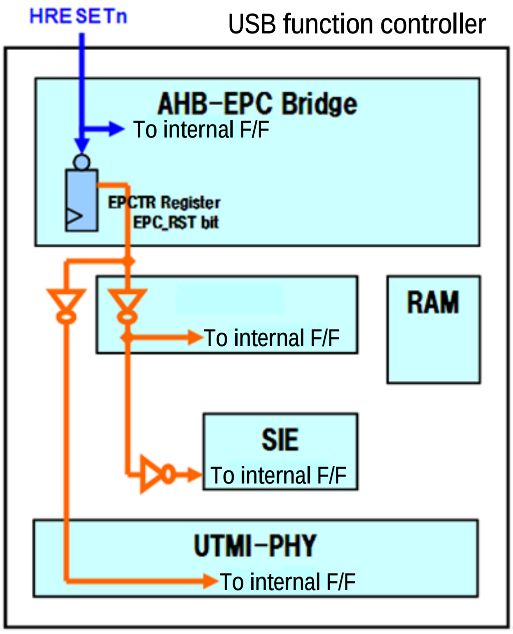
Reset Signal	Supplied From	Description
USB1_RSTZ	External	Power-on reset signal for the USB function controller This signal resets the USB function controller.
EPC_RST	Internal	Reset signal for the USB devices (EPC/PHY)

When the USB1_RSTZ signal is asserted, the EPC_RST signal is also asserted. The EPC_RST signal can be controlled by using the AHB-EPC bridge registers. After the USB1_RSTZ signal is asserted, manipulate these registers to deassert the reset signal. For details about the reset sequence, see *12.1 Reset Sequence*.

7.2 Reset System Diagram

Figure 7-1 shows the reset system in the USB function controller.

Figure 7-1. Reset System Diagram



8. Interrupts

8.1 Interrupt Signals

Table 8-1 lists the interrupt signals that can be handled by using the USB function controller. Each signal is detected at its active level. Using these interrupt signals independently is recommended because the AHB-EPC bridge registers cannot be used to check the USB_INTF1 status.

Table 8-1. Interrupt Signals

Interrupt Signals	Synchronization Clock	Active Level	Description
USB_INTF0	USB1_CLK	High	An interrupt signal generated by the AHB-EPC bridge macro.
USB_INTF1	Internal 60 MHz clock	High	An interrupt signal generated by the EPC.

8.2 Interrupt Control Registers

8.2.1 USB_INTF0 control register

This register shows the interrupts generated by the AHB-EPC bridge. Use the AHB-EPC bridge registers to check the status of, clear, and enable the interrupts.

Table 8-2. USB_INTF0 Control Register

Control	Target Registers
Checking the status of and clearing the interrupts	AHBBINT register
Enabling the interrupts	AHBBINTEN register

8.2.2 USB_INTF1 control register

This register shows the interrupts generated by the EPC. Use the EPC registers to check the status of, clear, and enable the interrupts.

Table 8-3. USB_INTF1 Control Register

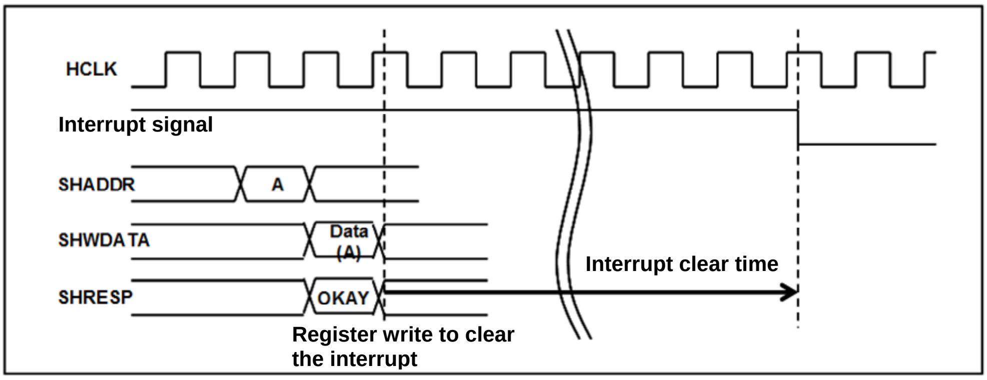
Control	Target Registers
Checking the status of and clearing the interrupts	USB interrupt status register EP0/EPn status registers
Enabling the interrupts	USB interrupt enable register EP0/EPn interrupt enable registers

8.3 Time Required for Clearing an Interrupt

After an interrupt is generated by a USB device and a register that is used to clear the USB_INTF1 interrupt is accessed, a time lag might occur until the interrupt is actually cleared, depending on the internal bus access status. As a result, an interrupt with the same status might be acknowledged multiple times (see Figure 8-1). Take appropriate measures to prevent this, such as accessing the register to clear USB_INTF1, and then accessing the relevant USB function register immediately. Accessing the USB function register is suspended until USB_INTF1 is cleared; that is, the interrupt must have already been cleared by the time the USB function register is accessed.

When the USB1_CLK frequency is 100 MHz, it generally takes about 150 ns to clear an interrupt. When a DMA transfer is being performed on the internal bus between the AHB-EPC bridge and EPC, it takes about 600 ns in the worst case (34 cycles of an internal 60 MHz clock + 3 USB1_CLK cycles).

Figure 8-1. Time Required for Clearing an Interrupt



8.4 Notes on Clearing Interrupt Status Registers

How to clear interrupt sources depends on the interrupt status register to be manipulated.

(1) AHBBINT register
Write 1b to the active interrupt status bits (to which 1b is written).
Write 0b to the other bits.

(2) USB interrupt status register, EP0 status register, and EPn status register
Write 0b to the active interrupt status bits (to which 1b is written).
Write 1b to the other bits.

9. VBUS Detection

9.1 External Circuit for Detecting the VBUS Level

An external circuit is required to monitor the VBUS level on the USB bus.
This circuit converts 5 V to 3 V, prevents chattering by using capacitors or other measures, and then inputs the signal as the VBUS signal to the USB function controller.

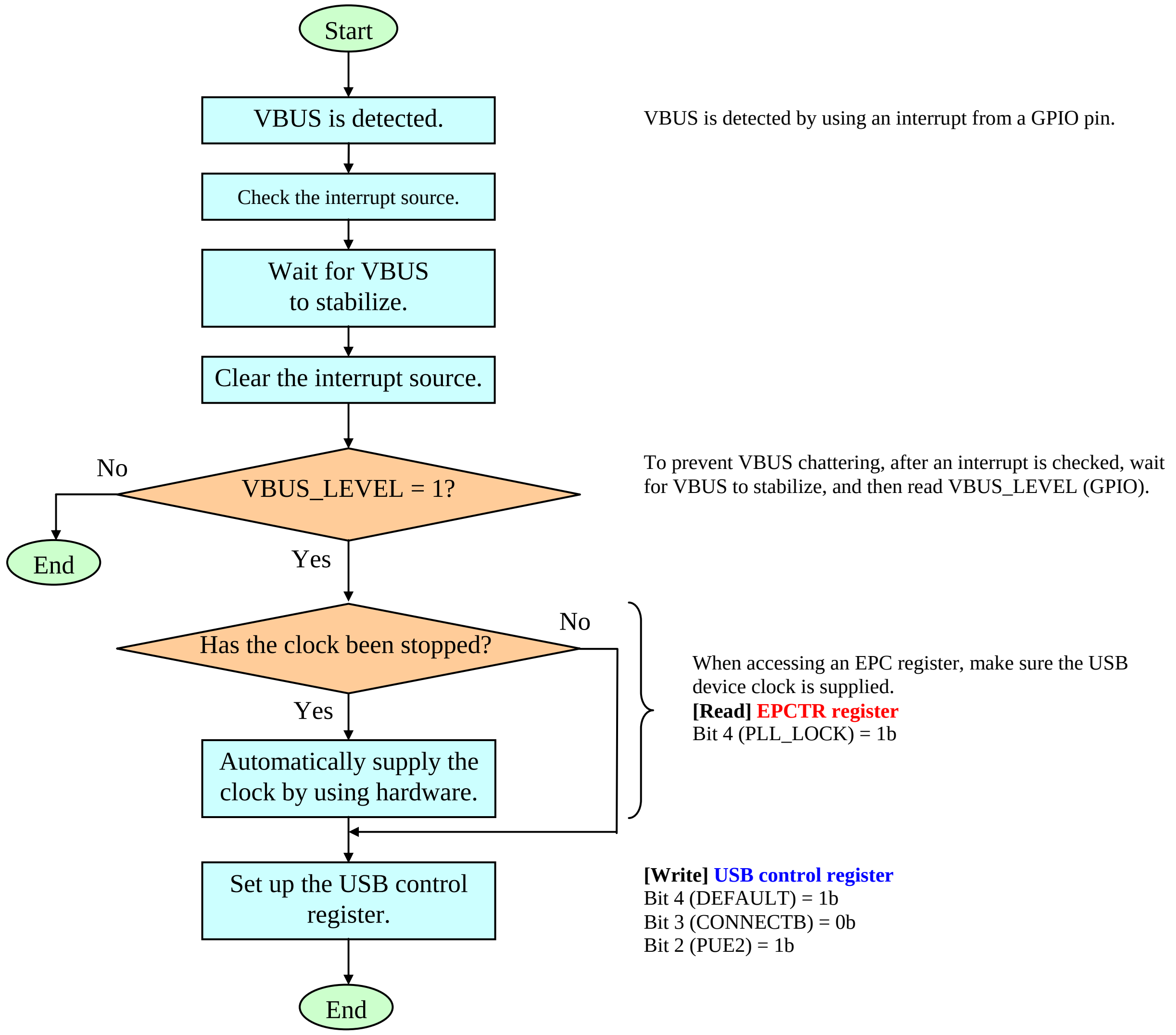
9.2 Detecting VBUS

Use general-purpose I/O ports (GPIO) to detect VBUS.
The detected VBUS level is sent to EPC registers by the CPU and is used to control the SIE and UTMI-PHY blocks.

9.3 VBUS Detection Overview

9.3.1 When a USB device is connected

Figure 9-1. VBUS Detection Overview (When a USB Device Is Connected)


Start
VBUS is detected.
Check the interrupt source.
Wait for VBUS to stabilize.
Clear the interrupt source.
VBUS_LEVEL = 1?
No
End
Yes
Has the clock been stopped?
No
Yes
Automatically supply the clock by using hardware.
Set up the USB control register.
End

VBUS is detected by using an interrupt from a GPIO pin.

To prevent VBUS chattering, after an interrupt is checked, wait for VBUS to stabilize, and then read VBUS_LEVEL (GPIO).

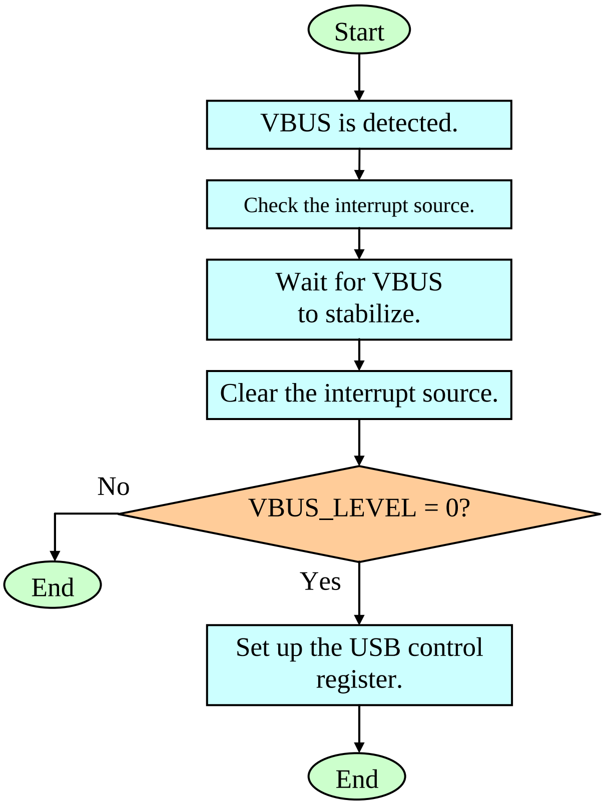
When accessing an EPC register, make sure the USB device clock is supplied.
[Read] **EPCTR register**
Bit 4 (PLL_LOCK) = 1b

[Write] **USB control register**
Bit 4 (DEFAULT) = 1b
Bit 3 (CONNECTB) = 0b
Bit 2 (PUE2) = 1b

Caution **If no external circuit is connected, the chattering time might be up to 100 ms. Even if an external circuit is connected, the VBUS stabilization time depends on the user's system. It is therefore recommended to evaluate the time in the user's system.**

9.3.2 When a USB device is disconnected

Figure 9-2. VBUS Detection Overview (When a USB Device Is Disconnected)


Start
VBUS is detected.
Check the interrupt source.
Wait for VBUS to stabilize.
Clear the interrupt source.
VBUS_LEVEL = 0?
No
End
Yes
Set up the USB control register.
End

VBUS is detected by using an interrupt from a GPIO pin.

To prevent VBUS chattering, after an interrupt is checked, wait for VBUS to stabilize, and then read VBUS_LEVEL (GPIO).

[Write] USB control register
Bit 4 (DEFAULT) = 0b
Bit 3 (CONNECTB) = 1b
Bit 2 (PUE2) = 0b

Remark VBUS_LEVEL is assumed to be input when it is active (high) (VBUS_LEVEL = 1b: VBUS on). If a USB device is disconnected while it is in the Suspend state and the clock supply to the USB function controller has been stopped, reset the EPC block, SIE block, and PHY interface by using bit 0 (EPC_RST) of the EPCTR register.

Caution If no external circuit is connected, the chattering time might be up to 100 ms. Even if an external circuit is connected, the VBUS stabilization time depends on the user's system. It is therefore recommended to evaluate the time in the user's system.

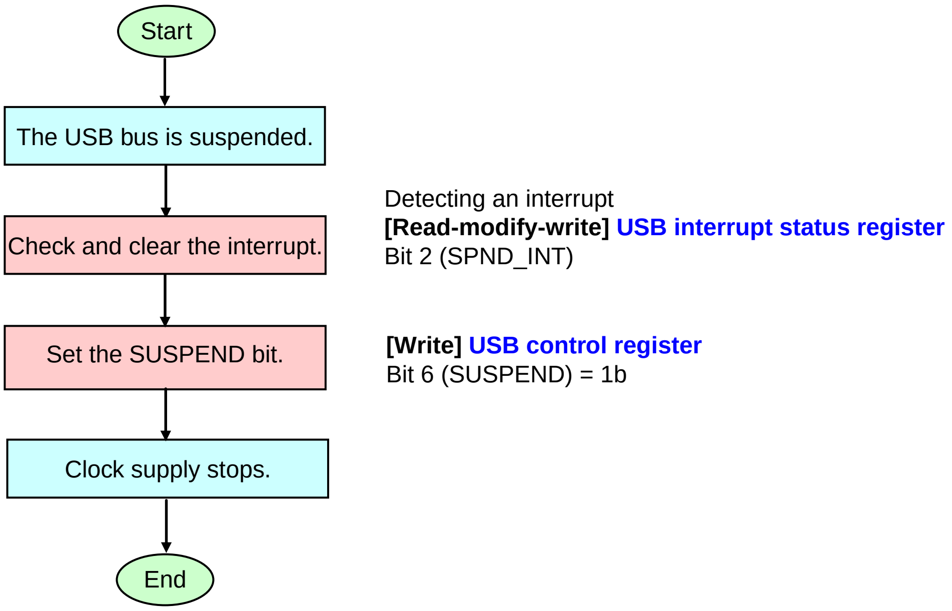
10. Power Management

This chapter describes the power management features.

10.1 Power Down

The following shows an overview of powering down.

Figure 10-1. Power Down Overview

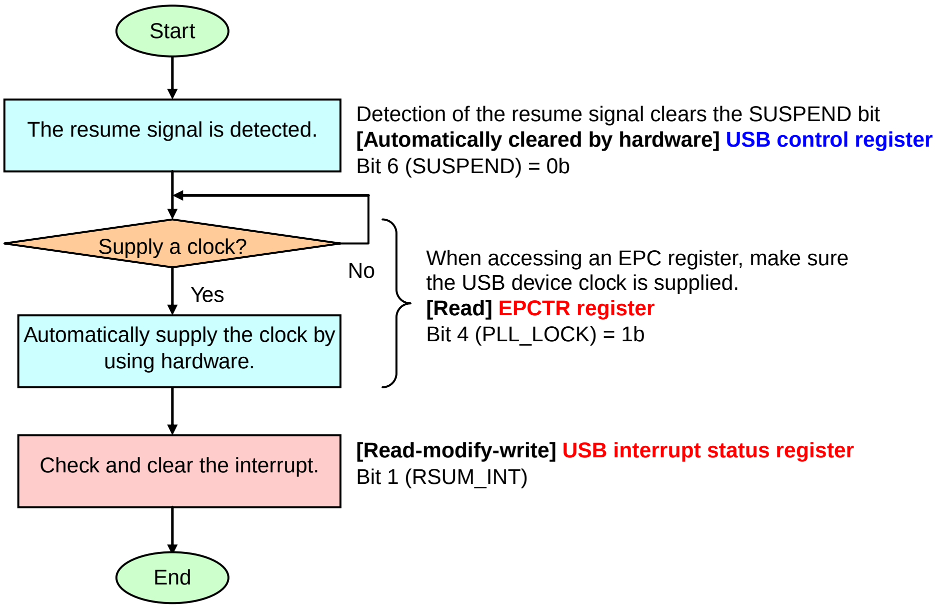

Start
The USB bus is suspended.
Check and clear the interrupt.
Detecting an interrupt
[Read-modify-write] USB interrupt status register
Bit 2 (SPND_INT)
Set the SUSPEND bit.
[Write] USB control register
Bit 6 (SUSPEND) = 1b
Clock supply stops.
End

10.2 Power On

The power can be turned on either by using the resume signal from the USB host or by using the remote wakeup feature.

10.2.1 Resume

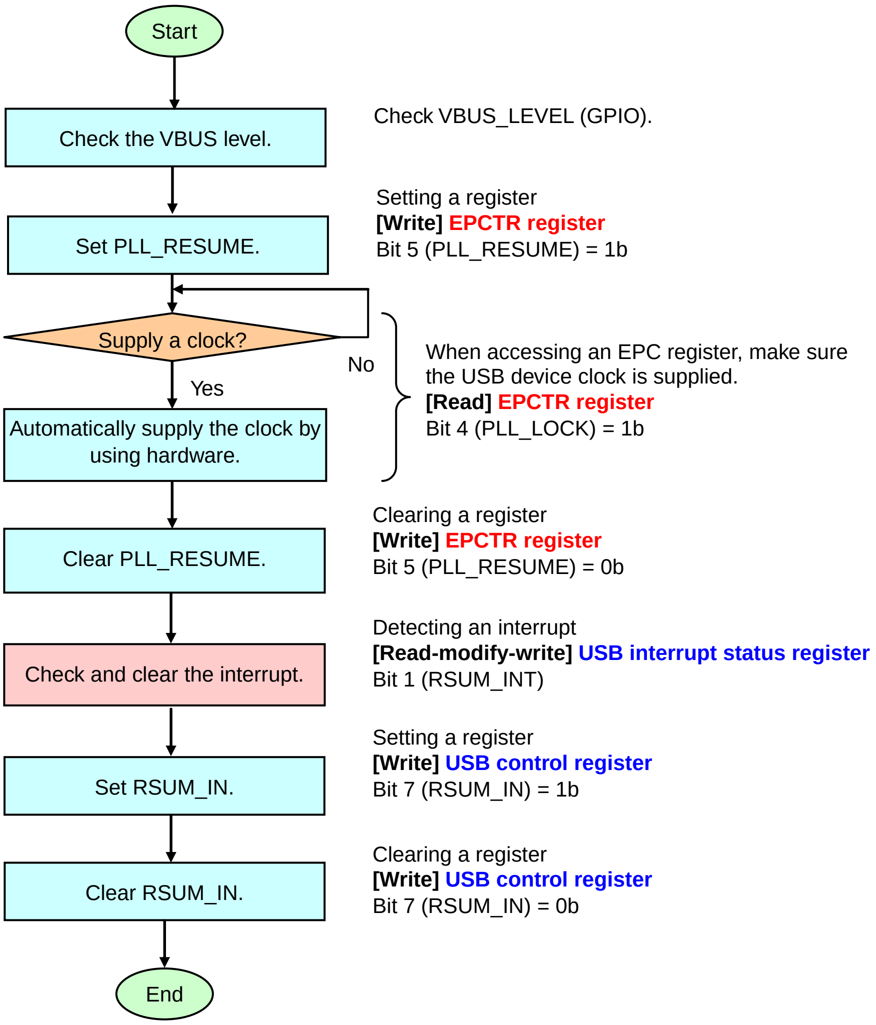
Figure 10-2. Power On By Detecting the Resume Signal


Start
The resume signal is detected.
Detection of the resume signal clears the SUSPEND bit
[Automatically cleared by hardware] USB control register
Bit 6 (SUSPEND) = 0b
Supply a clock?
No
Yes
Automatically supply the clock by using hardware.
When accessing an EPC register, make sure the USB device clock is supplied.
[Read] EPCTR register
Bit 4 (PLL_LOCK) = 1b
Check and clear the interrupt.
[Read-modify-write] USB interrupt status register
Bit 1 (RSUM_INT)
End

When the resume signal is detected, the SUSPEND bit of the USB control register is cleared and the RSUM_INT interrupt occurs. Confirm that this interrupt has occurred and clear it.

10.2.2 Remote wakeup

Figure 10-3. Power On by Using the Remote Wakeup Feature



The USB cable might be disconnected while the USB device is in the Suspend state. Therefore, when using the remote wakeup feature, make sure VBUS_LEVEL is 1, and then set the PLL_RESUME bit of the EPCTR register. In the same way as when using the resume signal, the SUSPEND bit of the USB control register is cleared and the clock is supplied. Read the PLL_LOCK bit of the EPCTR register to check whether the clock is being supplied. After that, set the RSUM_IN bit of the USB control register to power on remotely. Then clear the RSUM_IN bit.

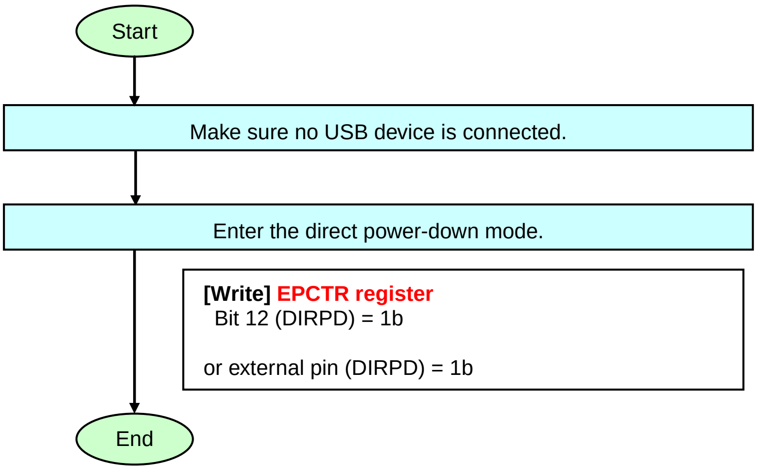
10.3 Direct Power-Down Feature

When the USB function controller is not used in the chip, the power consumption can be reduced by using the direct power-down feature.

10.3.1 Entering the direct power-down mode

Figure 10-4 shows how to enter the direct power-down mode.

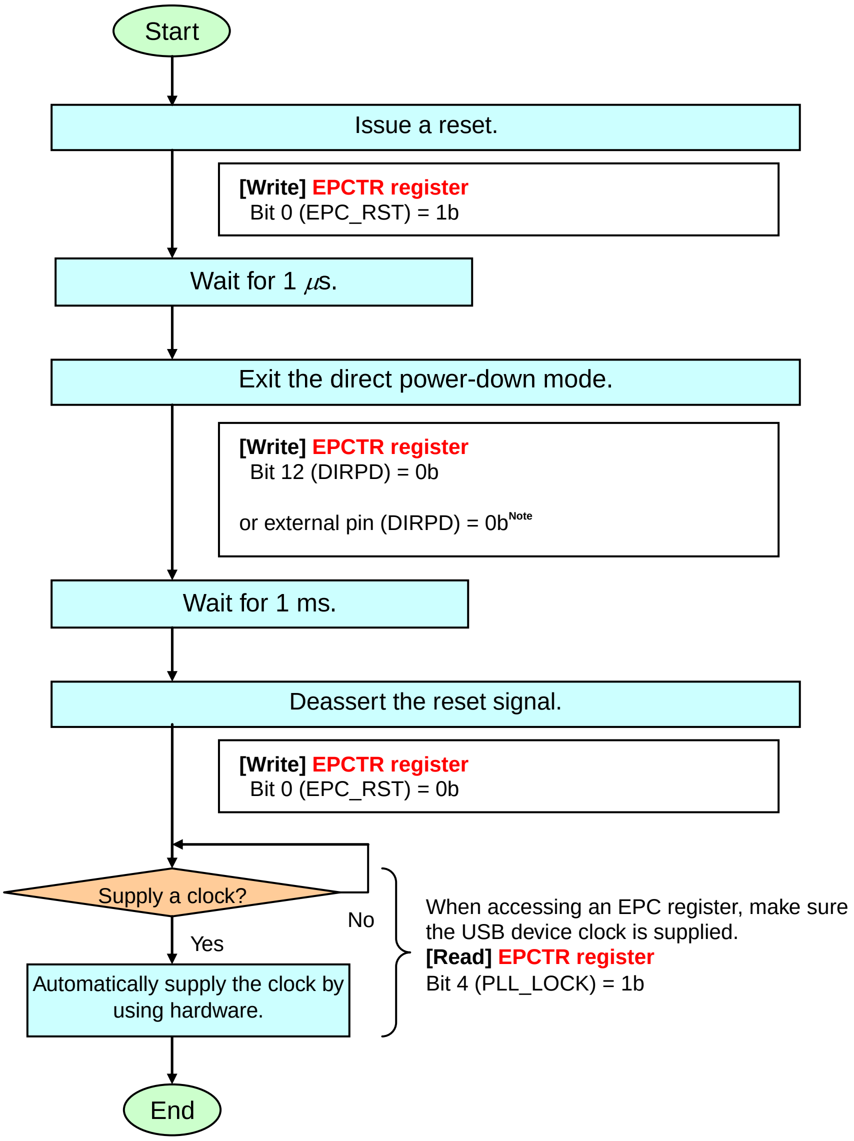
Figure 10-4. Entering the Direct Power-Down Mode


Start
Make sure no USB device is connected.
Enter the direct power-down mode.
[Write] EPCTR register
Bit 12 (DIRPD) = 1b
or external pin (DIRPD) = 1b
End

10.3.2 Exiting the direct power-down mode

Figure 10-5 shows how to exit the direct power-down mode.
When doing so, the DIRPD bit must be cleared while bit 0 (EPC_RST) of the EPCTR register is 1b.

Figure 10-5. Exiting the Direct Power-Down Mode



Note If the external pin DIRPD is used to enter the direct power-down mode, exit the mode by using the DIRPD pin. If the EPCTR register is used, exit the mode by using the EPCTR register.

11. Internal RAM

The USB function controller uses one 14.464-byte RAM module for the buffers of the general-purpose endpoint controller. This chapter describes how to specify the RAM base addresses by using software.

In the USB function controller, processes executed at each endpoint share the internal RAM. Therefore, which endpoint uses which RAM area must be specified by using the EPn_BASEAD[12:0] bits of the EPn max packet & base address register.

Table 11-1 shows an endpoint configuration example.

Table 11-1. Endpoint Configuration Example

	Transfer Type	Buffer Type	Use of DMA	Transfer Direction	Maximum Packet Size	
					In HS Mode	In FS Mode
EP0 (fixed)	Control	Single × 2 (RD/WR)	Not used	I/O	64 bytes	64 bytes
EP1	Bulk	Double	Used	Output	512 bytes	64 bytes
EP2	Bulk	Double	Used	Input	512 bytes	64 bytes
EP3	Interrupt	Single	Not used	Output	8 bytes	8 bytes
EP4	Bulk	Double	Used	Output	512 bytes	64 bytes
EP5	Bulk	Double	Used	Input	512 bytes	64 bytes

11.1 Calculating the Required RAM Size

To calculate the required RAM size, calculate the total size of the buffers used by the endpoints.
Table 11-2 shows the number of words that must be allocated in RAM per endpoint for each configuration.
In the case of Table 11-1, the required RAM size is 1,058 words (sum of 32, 256, 256, 2, 256, and 256).

Table 11-2. RAM Size Required by Each Endpoint

Endpoint Configuration			Required Size	Description
EP0			32 words (= 64 bytes × 2)	A 64-byte buffer for transmission and reception
EPn	Bulk	Single	128 words (= 512 bytes)	An area equivalent to the size used in HS mode is required.
		Double	256 words (= 512 bytes × 2)	An area equivalent to twice the size used in HS mode is required.
	Interrupt	Single	Maximum packet size in words	The required buffer size is allocated. If the maximum packet size is divided by words and less than a word remains, round the required size up to the nearest word.

11.2 Specifying the Base Address

To specify which endpoint buffer is allocated to which RAM area, specify the RAM base address by using the EPn_BASEAD[12:0] bits of the EPn max packet & base address register. This setting is required for each endpoint. For the case shown in Table 11-1, the settings are specified for each endpoint as follows:

- The EP0 buffers are always allocated starting from 000h in the RAM, so no register setting is required.
  Because EP0 requires 32 words, the EP0 buffers are always allocated from 000h to 01Fh.
- The EP1 buffers are allocated after the EP0 buffer area, to the area starting from 020h.
  Because EP1 requires 256 words, the EP1 buffers are allocated from 020h to 11Fh.
  Specify the EP1 max packet & base address register as follows:
  EP1_MPKT[10:0] = 200h (512 bytes) (in HS mode) or 040h (64 bytes) (in FS mode)
  EP1_BASEAD[12:0] = 0020h
- The EP2 buffers are allocated after the EP1 buffer area (0020h to 011Fh), to the area starting from 0120h.
  Specify the EP2_MPKT[10:0] bits in the same way as for EP1.
- Specify the EP3 and subsequent buffer areas in the same way.

Table 11-3 lists the EPn max packet & base address register settings for the case shown in Table 11-1.

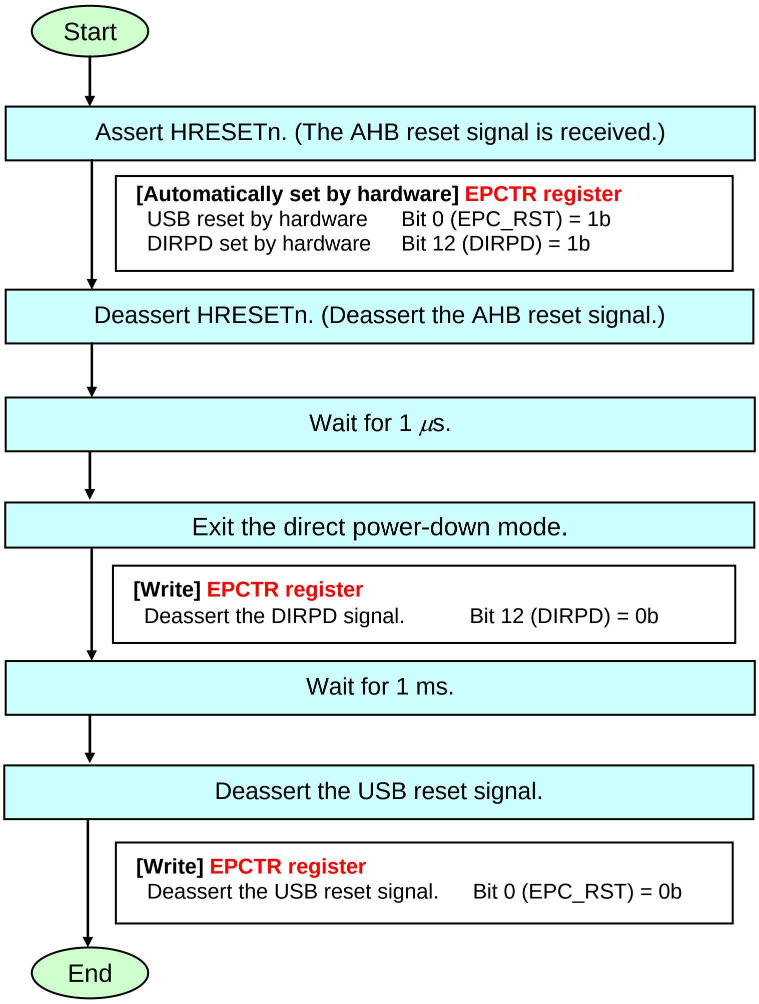
Table 11-3. Settings for the Case Shown in Table 11-1

	EPn_BASEAD[12:0]	EPn_MPKT[10:0]	Value Specified for EPn Max Packet & Base Address Register
EP0 (fixed)	–	–	–
EP1	0020h	200h(HS)/040h(FS)	0020_0200h (HS) 0020_0040h (FS)
EP2	0120h	200h(HS)/040h(FS)	0120_0200h (HS) 0120_0040h (FS)
EP3	0220h	008h	0220_0008h
EP4	0222h	200h(HS)/040h(FS)	0222_0200h (HS) 0222_0040h (FS)
EP5	0322h	200h(HS)/040h(FS)	0322_0200h (HS) 0322_0040h (FS)

12. Operating Procedures

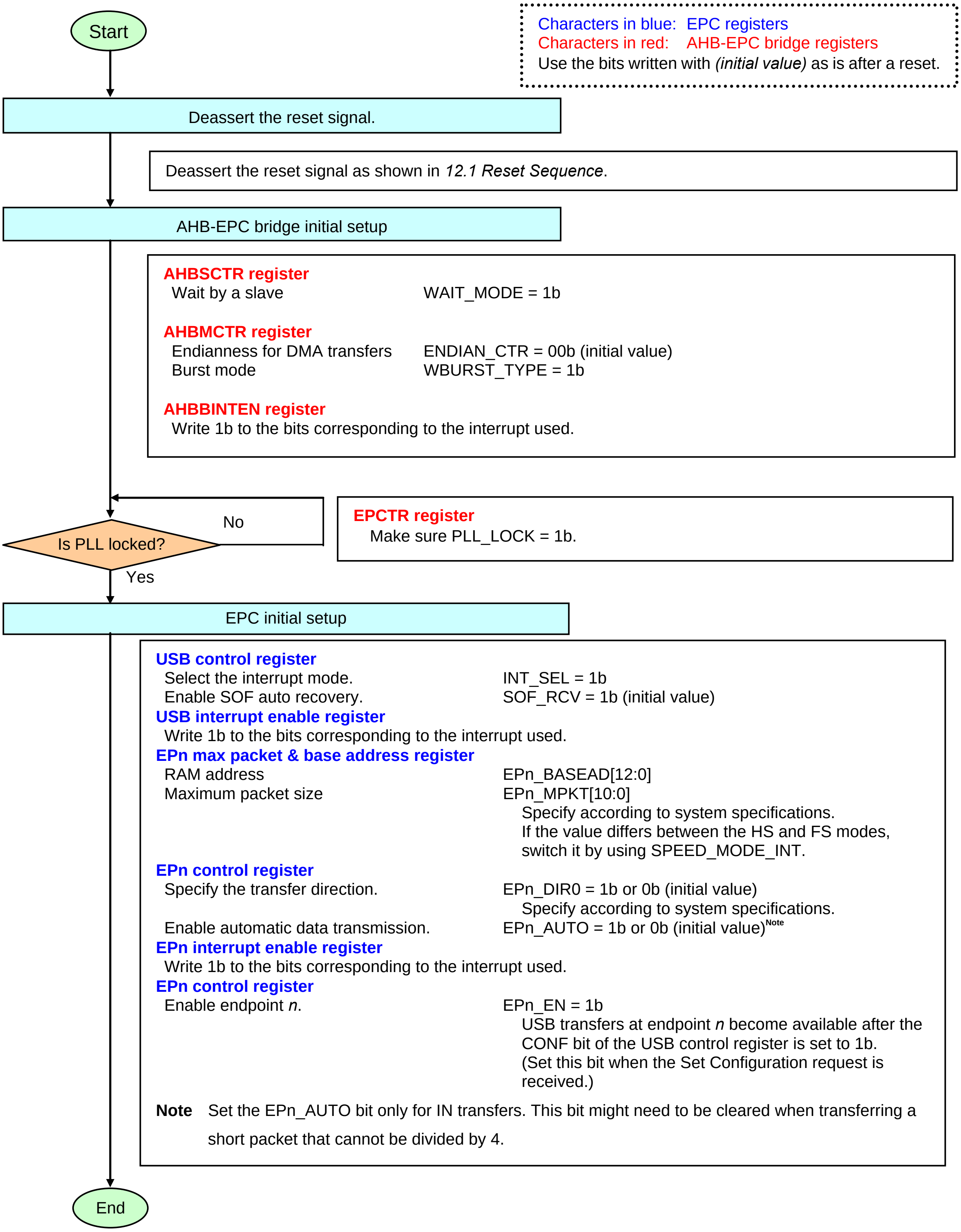
12.1 Reset Sequence

Figure 12-1. Reset Sequence


Start
Assert HRESETn. (The AHB reset signal is received.)
[Automatically set by hardware] EPCTR register
USB reset by hardware Bit 0 (EPC_RST) = 1b
DIRPD set by hardware Bit 12 (DIRPD) = 1b
Deassert HRESETn. (Deassert the AHB reset signal.)
Wait for 1 μs.
Exit the direct power-down mode.
[Write] EPCTR register
Deassert the DIRPD signal. Bit 12 (DIRPD) = 0b
Wait for 1 ms.
Deassert the USB reset signal.
[Write] EPCTR register
Deassert the USB reset signal. Bit 0 (EPC_RST) = 0b
End

12.2 Initial Setup Sequence

Figure 12-2. USB Device Initial Setup Sequence



Characters in blue: EPC registers
Characters in red: AHB-EPC bridge registers
Use the bits written with *(initial value)* as is after a reset.

Deassert the reset signal.

Deassert the reset signal as shown in *12.1 Reset Sequence*.

AHB-EPC bridge initial setup

AHBSCTR register
Wait by a slave WAIT_MODE = 1b

AHBMCTR register
Endianness for DMA transfers ENDIAN_CTR = 00b (initial value)
Burst mode WBURST_TYPE = 1b

AHBBINTEN register
Write 1b to the bits corresponding to the interrupt used.

EPCTR register
Make sure PLL_LOCK = 1b.

EPC initial setup

USB control register
Select the interrupt mode. INT_SEL = 1b
Enable SOF auto recovery. SOF_RCV = 1b (initial value)

USB interrupt enable register
Write 1b to the bits corresponding to the interrupt used.

EPn max packet & base address register
RAM address EPn_BASEAD[12:0]
Maximum packet size EPn_MPKT[10:0]
Specify according to system specifications.
If the value differs between the HS and FS modes, switch it by using SPEED_MODE_INT.

EPn control register
Specify the transfer direction. EPn_DIR0 = 1b or 0b (initial value)
Specify according to system specifications.
Enable automatic data transmission. EPn_AUTO = 1b or 0b (initial value)[Note]

EPn interrupt enable register
Write 1b to the bits corresponding to the interrupt used.

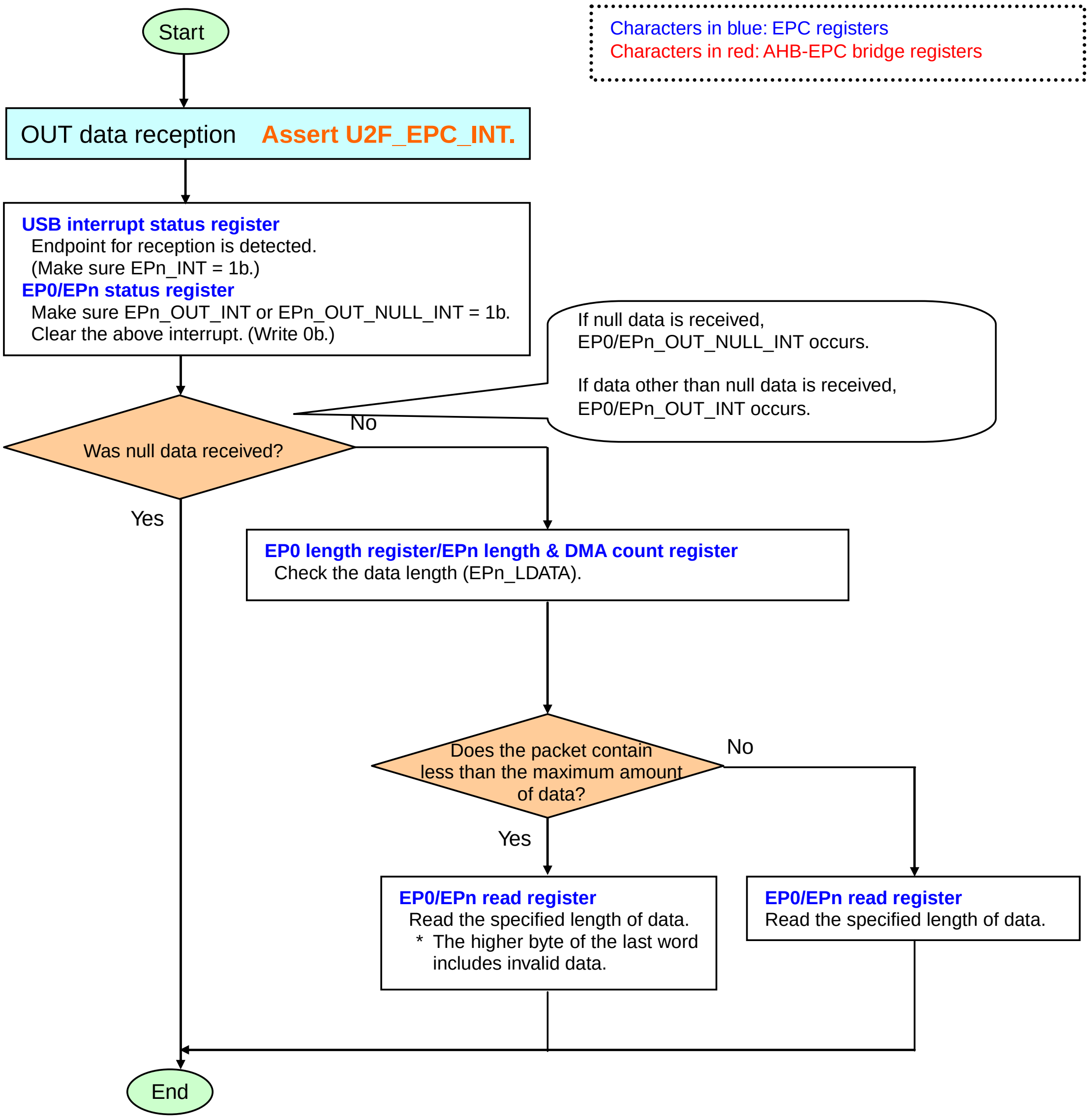
EPn control register
Enable endpoint *n*. EPn_EN = 1b
USB transfers at endpoint *n* become available after the CONF bit of the USB control register is set to 1b.
(Set this bit when the Set Configuration request is received.)

Note Set the EPn_AUTO bit only for IN transfers. This bit might need to be cleared when transferring a short packet that cannot be divided by 4.

12.3 Transfer Overview

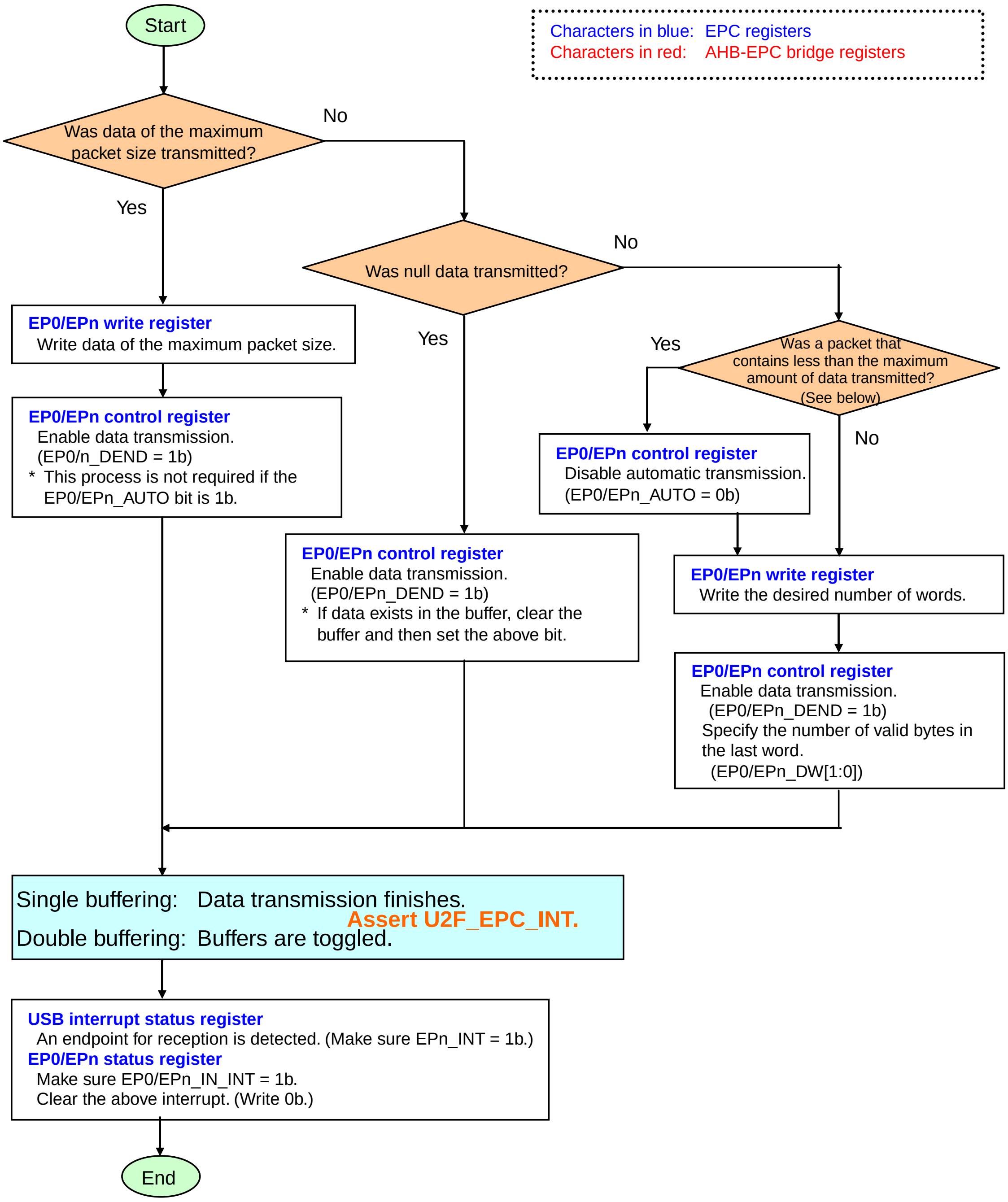
12.3.1 PIO OUT transfer

Figure 12-3. PIO OUT Transfer Overview


Characters in blue: EPC registers
Characters in red: AHB-EPC bridge registers
Start
OUT data reception Assert U2F_EPC_INT.
USB interrupt status register
Endpoint for reception is detected.
(Make sure EPn_INT = 1b.)
EP0/EPn status register
Make sure EPn_OUT_INT or EPn_OUT_NULL_INT = 1b.
Clear the above interrupt. (Write 0b.)
If null data is received,
EP0/EPn_OUT_NULL_INT occurs.
If data other than null data is received,
EP0/EPn_OUT_INT occurs.
Was null data received?
No
Yes
EP0 length register/EPn length & DMA count register
Check the data length (EPn_LDATA).
Does the packet contain less than the maximum amount of data?
No
Yes
EP0/EPn read register
Read the specified length of data.
* The higher byte of the last word includes invalid data.
EP0/EPn read register
Read the specified length of data.
End

12.3.2 PIO IN transfer

Figure 12-4. PIO IN Transfer Overview


Start
Characters in blue: EPC registers
Characters in red: AHB-EPC bridge registers
Was data of the maximum packet size transmitted?
No
Yes
Was null data transmitted?
No
Yes
Was a packet that contains less than the maximum amount of data transmitted? (See below)
Yes
No
EP0/EPn write register
Write data of the maximum packet size.
EP0/EPn control register
Enable data transmission.
(EP0/n_DEND = 1b)
* This process is not required if the EP0/EPn_AUTO bit is 1b.
EP0/EPn control register
Enable data transmission.
(EP0/EPn_DEND = 1b)
* If data exists in the buffer, clear the buffer and then set the above bit.
EP0/EPn control register
Disable automatic transmission.
(EP0/EPn_AUTO = 0b)
EP0/EPn write register
Write the desired number of words.
EP0/EPn control register
Enable data transmission.
(EP0/EPn_DEND = 1b)
Specify the number of valid bytes in the last word.
(EP0/EPn_DW[1:0])
Single buffering: Data transmission finishes.
Assert U2F_EPC_INT.
Double buffering: Buffers are toggled.
USB interrupt status register
An endpoint for reception is detected. (Make sure EPn_INT = 1b.)
EP0/EPn status register
Make sure EP0/EPn_IN_INT = 1b.
Clear the above interrupt. (Write 0b.)
End

A packet that contains less than the maximum amount of data does not mean a short packet, but rather a packet whose data size reaches the maximum packet size when the transmitted data is written in words.

Example 1: When the maximum packet size is set to 64 bytes and 63, 62, or 61 bytes are sent

Example 2: When the maximum packet size is set to 15 bytes and 14 or 13 bytes are sent

When transmitting such packet, be sure to clear the EP0/EPn_AUTO bit. Otherwise, a packet of the maximum packet size will be transmitted.

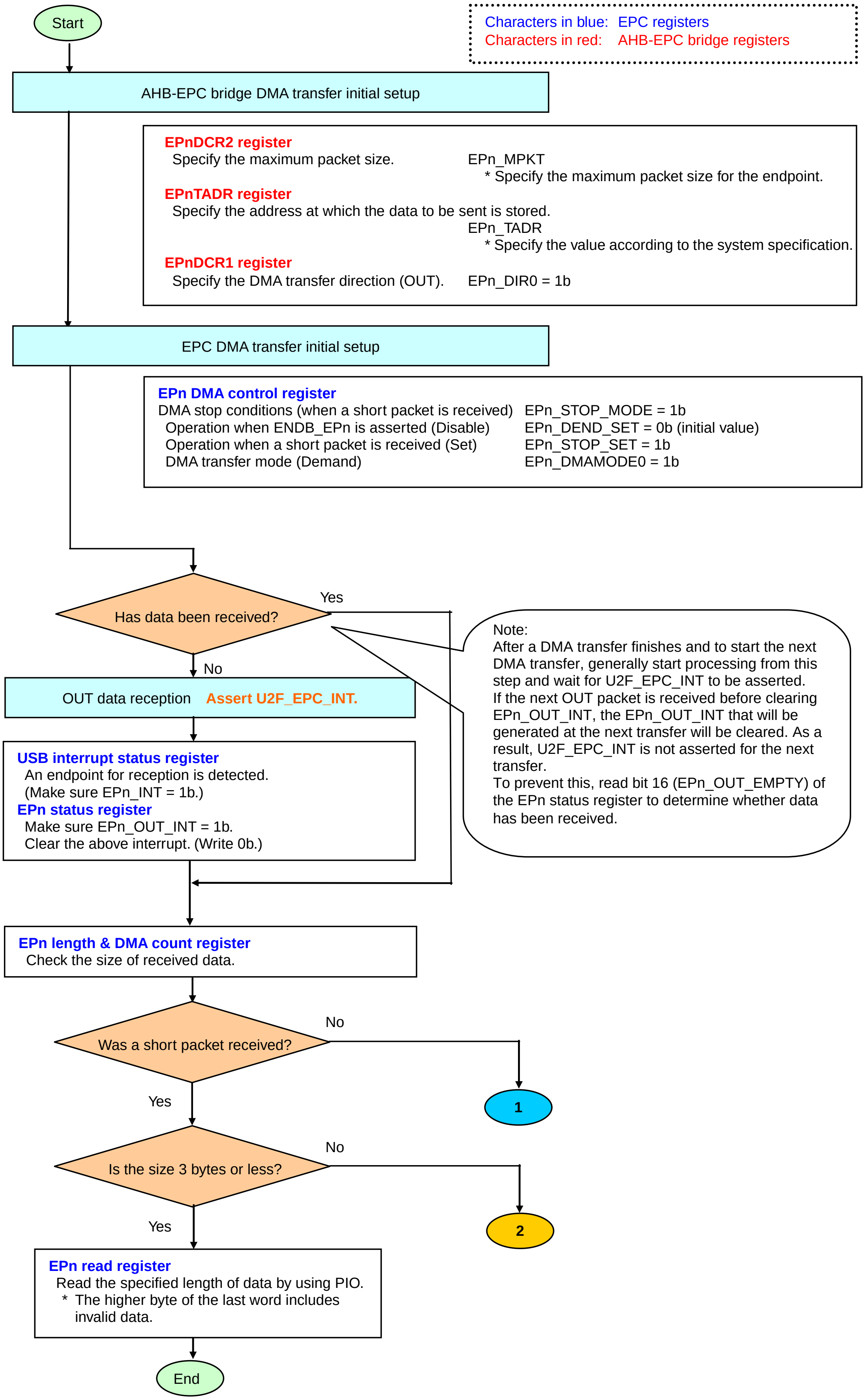
12.3.3 DMA OUT transfer

To perform a DMA transfer, specify the number of transferred packets by using the EPn_DMACNT bit of the EPnDCR1 register and EPn length & DMA count register.
When DMA transfer is performed the number of times specified by the EPn_DMACNT bit of the EPn_LEN_DCNT register or when a short packet that contains null data is received, the USB function controller stops the DMA transfer and generates the USB_INTF0 or USB_INTF1 interrupt.

A DMA OUT transfer assumes the following conditions:

- The number of packets sent from the USB host is unknown. (The value specified for the EPn_DMACNT bit must be determined according to the user's system.)
- A short packet might be received during a transfer.

Figure 12-5. DMA OUT Transfer Overview



Characters in blue: EPC registers
Characters in red: AHB-EPC bridge registers

AHB-EPC bridge DMA transfer initial setup

EPnDCR2 register
Specify the maximum packet size. EPn_MPKT
* Specify the maximum packet size for the endpoint.

EPnTADR register
Specify the address at which the data to be sent is stored.
EPn_TADR
* Specify the value according to the system specification.

EPnDCR1 register
Specify the DMA transfer direction (OUT). EPn_DIR0 = 1b

EPC DMA transfer initial setup

EPn DMA control register

DMA stop conditions (when a short packet is received)	EPn_STOP_MODE = 1b
Operation when ENDB_EPn is asserted (Disable)	EPn_DEND_SET = 0b (initial value)
Operation when a short packet is received (Set)	EPn_STOP_SET = 1b
DMA transfer mode (Demand)	EPn_DMAMODE0 = 1b

Has data been received?

Note:
After a DMA transfer finishes and to start the next DMA transfer, generally start processing from this step and wait for U2F_EPC_INT to be asserted.
If the next OUT packet is received before clearing EPn_OUT_INT, the EPn_OUT_INT that will be generated at the next transfer will be cleared. As a result, U2F_EPC_INT is not asserted for the next transfer.
To prevent this, read bit 16 (EPn_OUT_EMPTY) of the EPn status register to determine whether data has been received.

OUT data reception Assert U2F_EPC_INT.

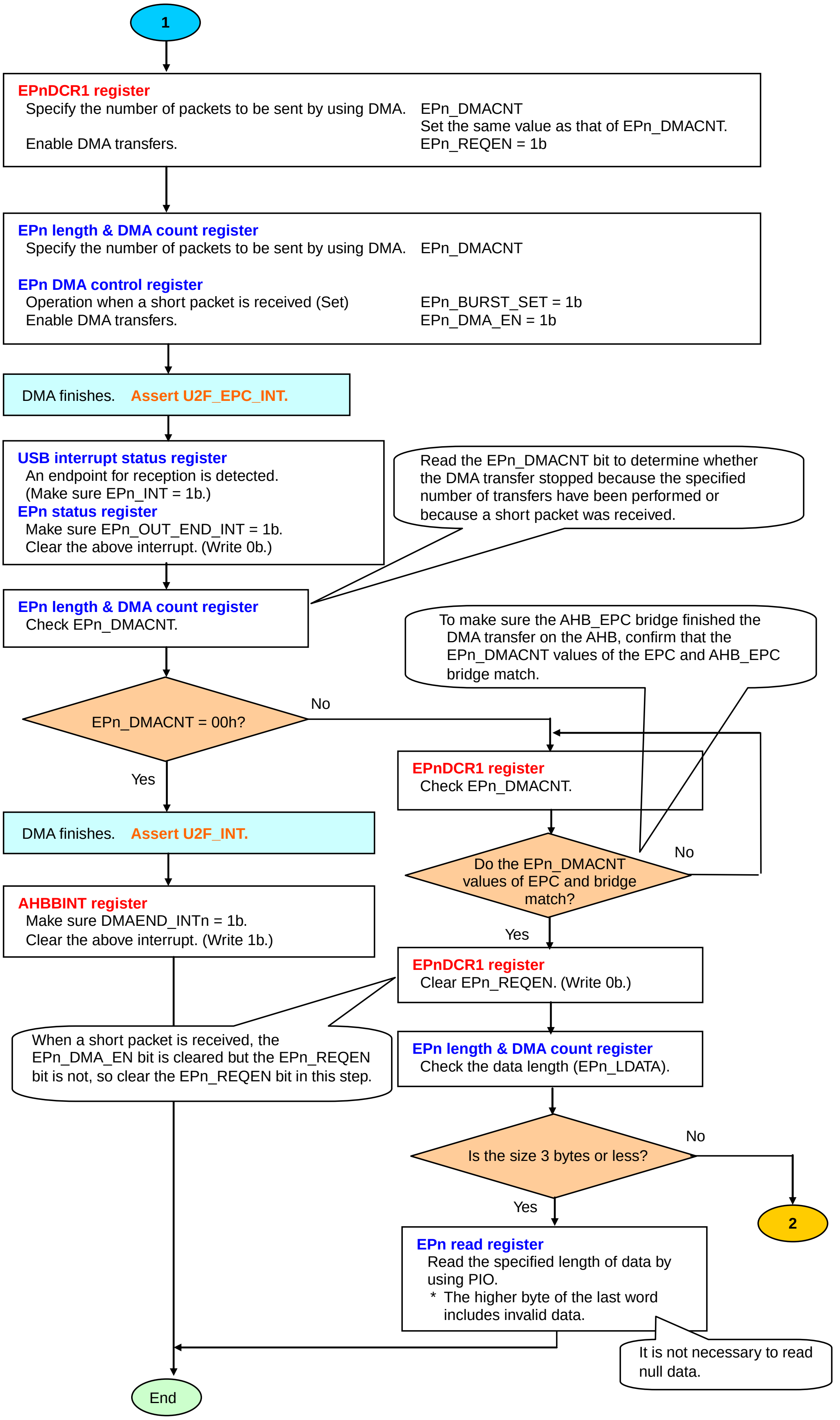
USB interrupt status register
An endpoint for reception is detected.
(Make sure EPn_INT = 1b.)
EPn status register
Make sure EPn_OUT_INT = 1b.
Clear the above interrupt. (Write 0b.)

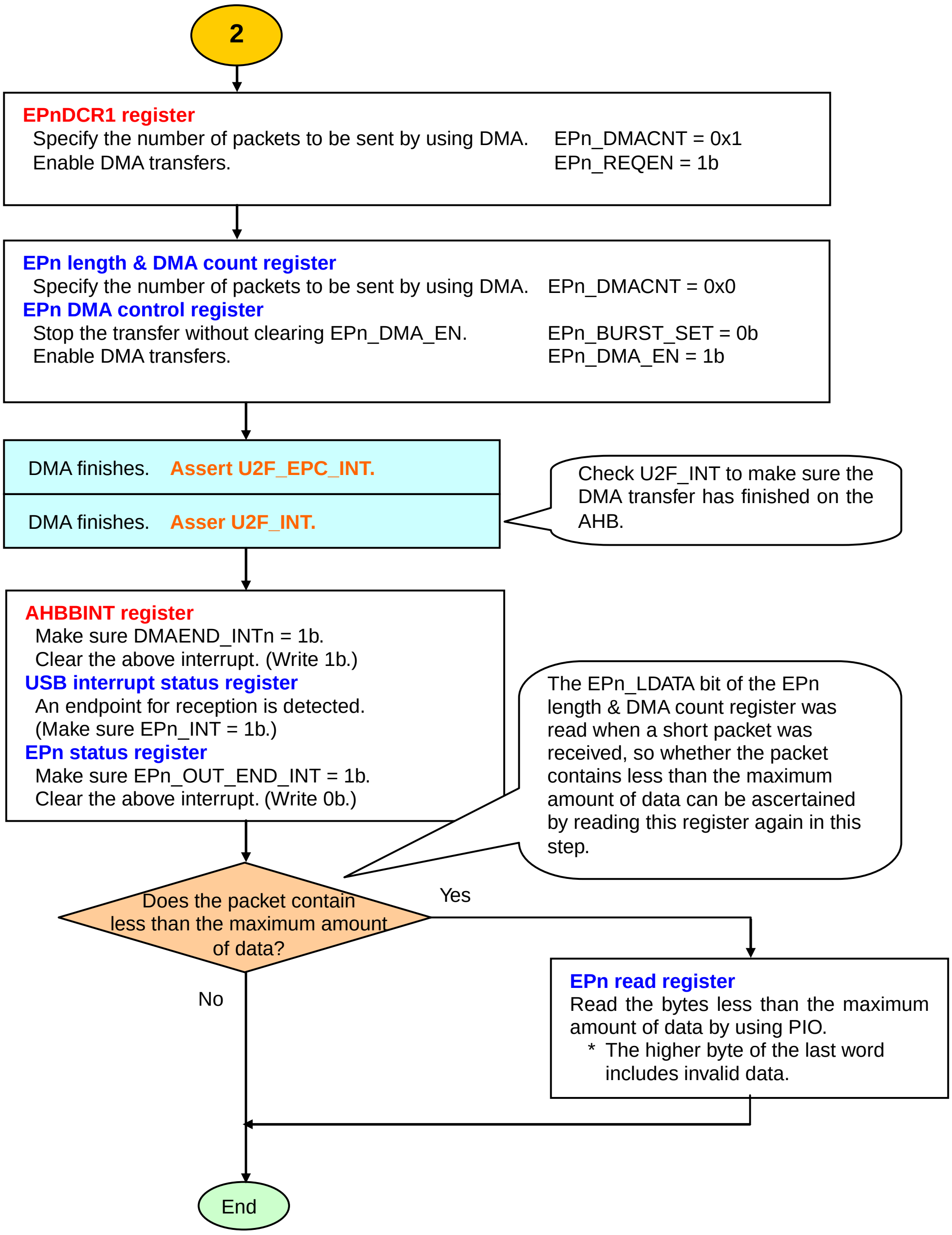
EPn length & DMA count register
Check the size of received data.

Was a short packet received?

Is the size 3 bytes or less?

EPn read register
Read the specified length of data by using PIO.
* The higher byte of the last word includes invalid data.


1
EPnDCR1 register
Specify the number of packets to be sent by using DMA.
EPn_DMACNT
Set the same value as that of EPn_DMACNT.
Enable DMA transfers.
EPn_REQEN = 1b
EPn length & DMA count register
Specify the number of packets to be sent by using DMA.
EPn_DMACNT
EPn DMA control register
Operation when a short packet is received (Set)
EPn_BURST_SET = 1b
Enable DMA transfers.
EPn_DMA_EN = 1b
DMA finishes. Assert U2F_EPC_INT.
USB interrupt status register
An endpoint for reception is detected.
(Make sure EPn_INT = 1b.)
EPn status register
Make sure EPn_OUT_END_INT = 1b.
Clear the above interrupt. (Write 0b.)
Read the EPn_DMACNT bit to determine whether the DMA transfer stopped because the specified number of transfers have been performed or because a short packet was received.
EPn length & DMA count register
Check EPn_DMACNT.
To make sure the AHB_EPC bridge finished the DMA transfer on the AHB, confirm that the EPn_DMACNT values of the EPC and AHB_EPC bridge match.
EPn_DMACNT = 00h?
No
Yes
EPnDCR1 register
Check EPn_DMACNT.
DMA finishes. Assert U2F_INT.
Do the EPn_DMACNT values of EPC and bridge match?
No
Yes
AHBBINT register
Make sure DMAEND_INTn = 1b.
Clear the above interrupt. (Write 1b.)
EPnDCR1 register
Clear EPn_REQEN. (Write 0b.)
When a short packet is received, the EPn_DMA_EN bit is cleared but the EPn_REQEN bit is not, so clear the EPn_REQEN bit in this step.
EPn length & DMA count register
Check the data length (EPn_LDATA).
Is the size 3 bytes or less?
No
Yes
2
EPn read register
Read the specified length of data by using PIO.
* The higher byte of the last word includes invalid data.
It is not necessary to read null data.
End



EPnDCR1 register

Specify the number of packets to be sent by using DMA. EPn_DMACNT = 0x1

Enable DMA transfers. EPn_REQEN = 1b

EPn length & DMA count register

Specify the number of packets to be sent by using DMA. EPn_DMACNT = 0x0

EPn DMA control register

Stop the transfer without clearing EPn_DMA_EN. EPn_BURST_SET = 0b

Enable DMA transfers. EPn_DMA_EN = 1b

DMA finishes. **Assert U2F_EPC_INT.**

DMA finishes. **Asser U2F_INT.**

Check U2F_INT to make sure the DMA transfer has finished on the AHB.

AHBBINT register

Make sure DMAEND_INTn = 1b.

Clear the above interrupt. (Write 1b.)

USB interrupt status register

An endpoint for reception is detected.

(Make sure EPn_INT = 1b.)

EPn status register

Make sure EPn_OUT_END_INT = 1b.

Clear the above interrupt. (Write 0b.)

The EPn_LDATA bit of the EPn length & DMA count register was read when a short packet was received, so whether the packet contains less than the maximum amount of data can be ascertained by reading this register again in this step.

Does the packet contain less than the maximum amount of data?

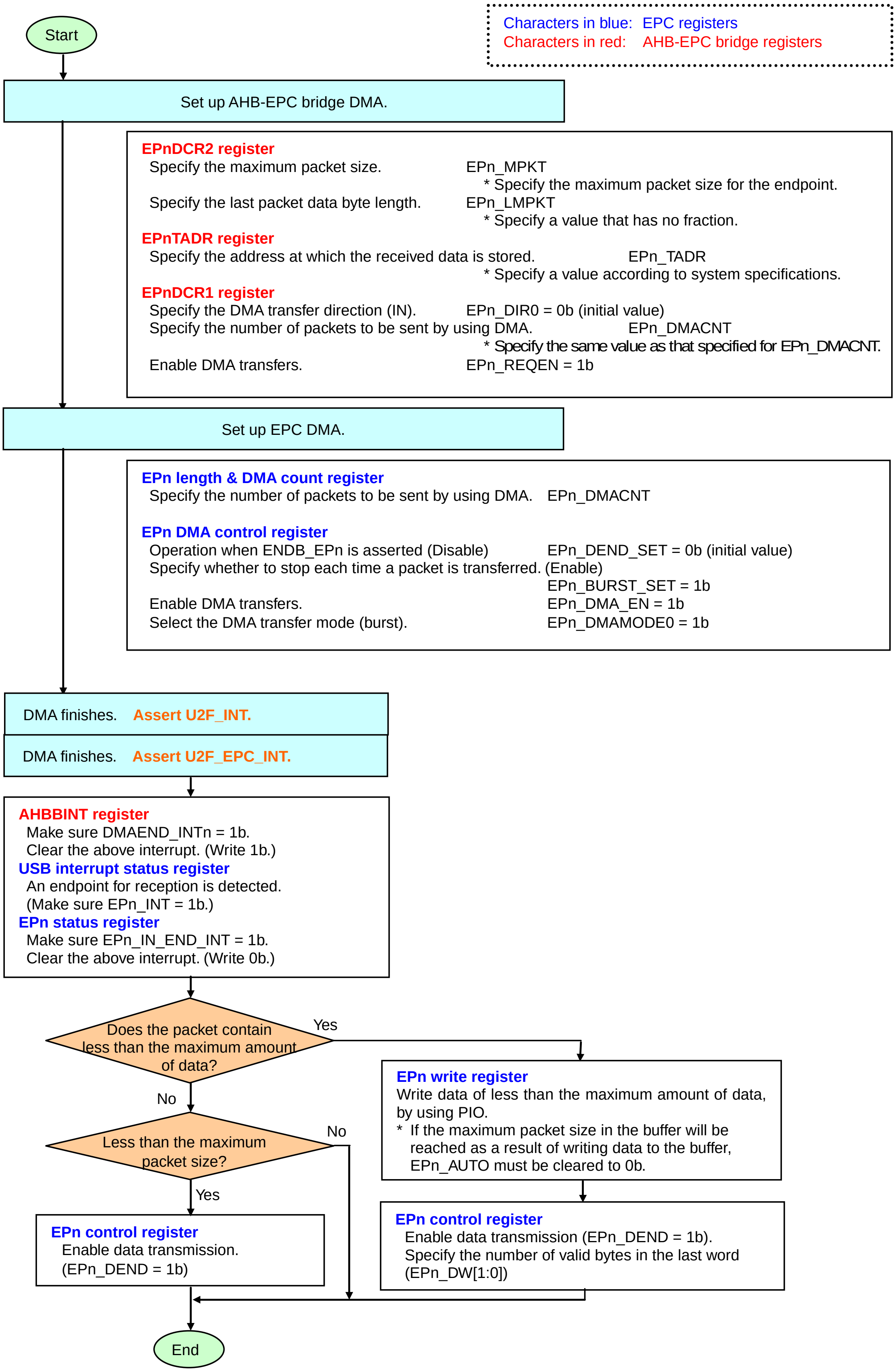
EPn read register

Read the bytes less than the maximum amount of data by using PIO.

* The higher byte of the last word includes invalid data.

12.3.4 DMA IN transfer

Figure 12-6. DMA IN Transfer Overview



Characters in blue: EPC registers
Characters in red: AHB-EPC bridge registers

Set up AHB-EPC bridge DMA.

EPnDCR2 register
Specify the maximum packet size. EPn_MPKT
* Specify the maximum packet size for the endpoint.
Specify the last packet data byte length. EPn_LMPKT
* Specify a value that has no fraction.

EPnTADR register
Specify the address at which the received data is stored. EPn_TADR
* Specify a value according to system specifications.

EPnDCR1 register
Specify the DMA transfer direction (IN). EPn_DIR0 = 0b (initial value)
Specify the number of packets to be sent by using DMA. EPn_DMACNT
* Specify the same value as that specified for EPn_DMACNT.
Enable DMA transfers. EPn_REQEN = 1b

Set up EPC DMA.

EPn length & DMA count register
Specify the number of packets to be sent by using DMA. EPn_DMACNT

EPn DMA control register
Operation when ENDB_EPn is asserted (Disable) EPn_DEND_SET = 0b (initial value)
Specify whether to stop each time a packet is transferred. (Enable)
EPn_BURST_SET = 1b
Enable DMA transfers. EPn_DMA_EN = 1b
Select the DMA transfer mode (burst). EPn_DMAMODE0 = 1b

DMA finishes. **Assert U2F_INT.**

DMA finishes. **Assert U2F_EPC_INT.**

AHBBINT register
Make sure DMAEND_INTn = 1b.
Clear the above interrupt. (Write 1b.)

USB interrupt status register
An endpoint for reception is detected.
(Make sure EPn_INT = 1b.)

EPn status register
Make sure EPn_IN_END_INT = 1b.
Clear the above interrupt. (Write 0b.)

Does the packet contain less than the maximum amount of data?

EPn write register
Write data of less than the maximum amount of data, by using PIO.
* If the maximum packet size in the buffer will be reached as a result of writing data to the buffer, EPn_AUTO must be cleared to 0b.

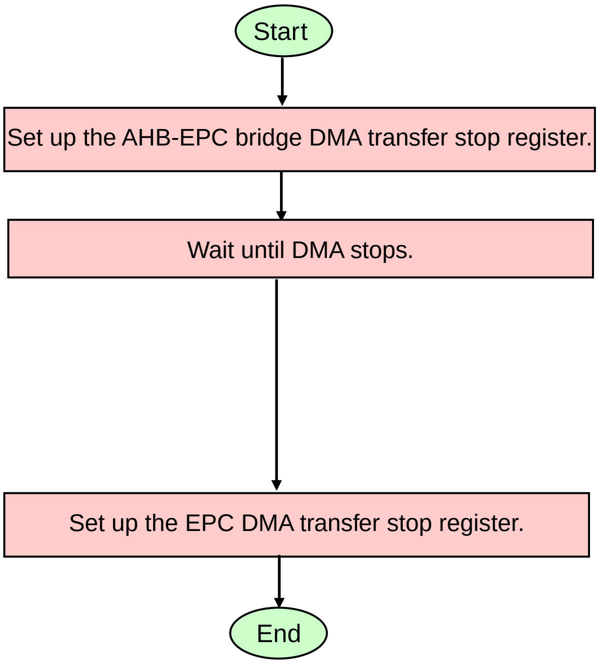
EPn control register
Enable data transmission (EPn_DEND = 1b).
Specify the number of valid bytes in the last word
(EPn_DW[1:0])

Less than the maximum packet size?

EPn control register
Enable data transmission.
(EPn_DEND = 1b)

12.3.5 Stopping a DMA transfer

Figure 12-7. Stopping a DMA Transfer


Start
Set up the AHB-EPC bridge DMA transfer stop register.
Wait until DMA stops.
Set up the EPC DMA transfer stop register.
End

[Write] EPnDCR1 register

Bit 0 (EPn_REQEN) = 0b

If the EPn_REQEN bit is cleared to 0b during a DMA transfer, the AHB-EPC bridge stops the DMA transfer when one packet is transferred. Due to this, the system must wait until the DMA transfer stops completely. The wait time might depend on the AHB bus occupation status, so the wait time must be determined according to the user's system.

[Write] EPn DMA control register

Bit 4 (EPn_DMA_EN) = 0b

12.3.6 Notes on using DMA transfers

Note the following when executing DMA transfers:

(1) After setting the EPn_REQEN bit of the EPnDCR1 register and the EPn_DMA_EN bit of the EPn DMA control register, an interrupt corresponding to the EPn_IN_INT or EPn_OUT_INT bit of the EPn status register occurs each time a packet is received or transmitted, even if a DMA transfer is being executed.

Before setting the above bits, clear the EPn_IN_EN and EPn_OUT_EN bits of the EPn interrupt enable register, which are used to enable interrupts corresponding to the above bits.

(2) Due to (1), after a DMA transfer finishes, the EPn_IN_INT or EPn_OUT_INT bit of the EPn status register is set. However, this indicates the status of the interrupt generated for already transmitted or received packets, not the current status of whether reading or writing to EPn is enabled or disabled.

(3) Due to (2), wait for USB_INTF1 to be asserted before reading data by using PIO after a DMA transfer finishes.

Note, however, that if a new OUT packet is received before EPn_OUT_INT is cleared, EPn_OUT_INT is cleared and therefore the USB_INTF1 signal for the new packet is not asserted.

To avoid this, read the following bits to check whether data has been received:

- EPn status register
  - EPn_OUT_FULL or EPn_OUT_EMPTY bits
- EPn length & DMA count register
  - EPn_LDATA[10:0] bits

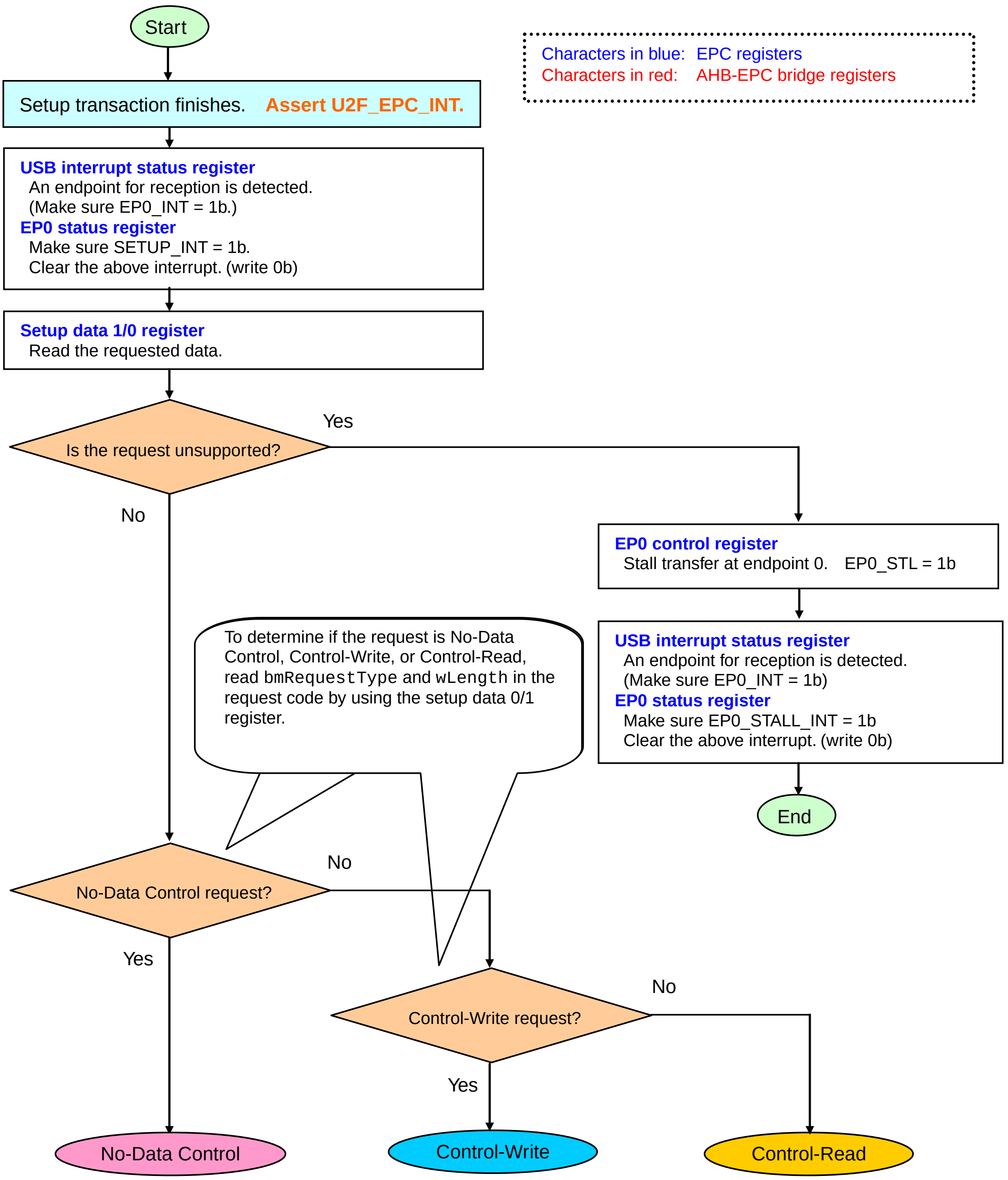
(4) Due to (2), read the following bits to check whether data can be written to the transmission buffer before writing data by using PIO after a DMA transfer finishes:

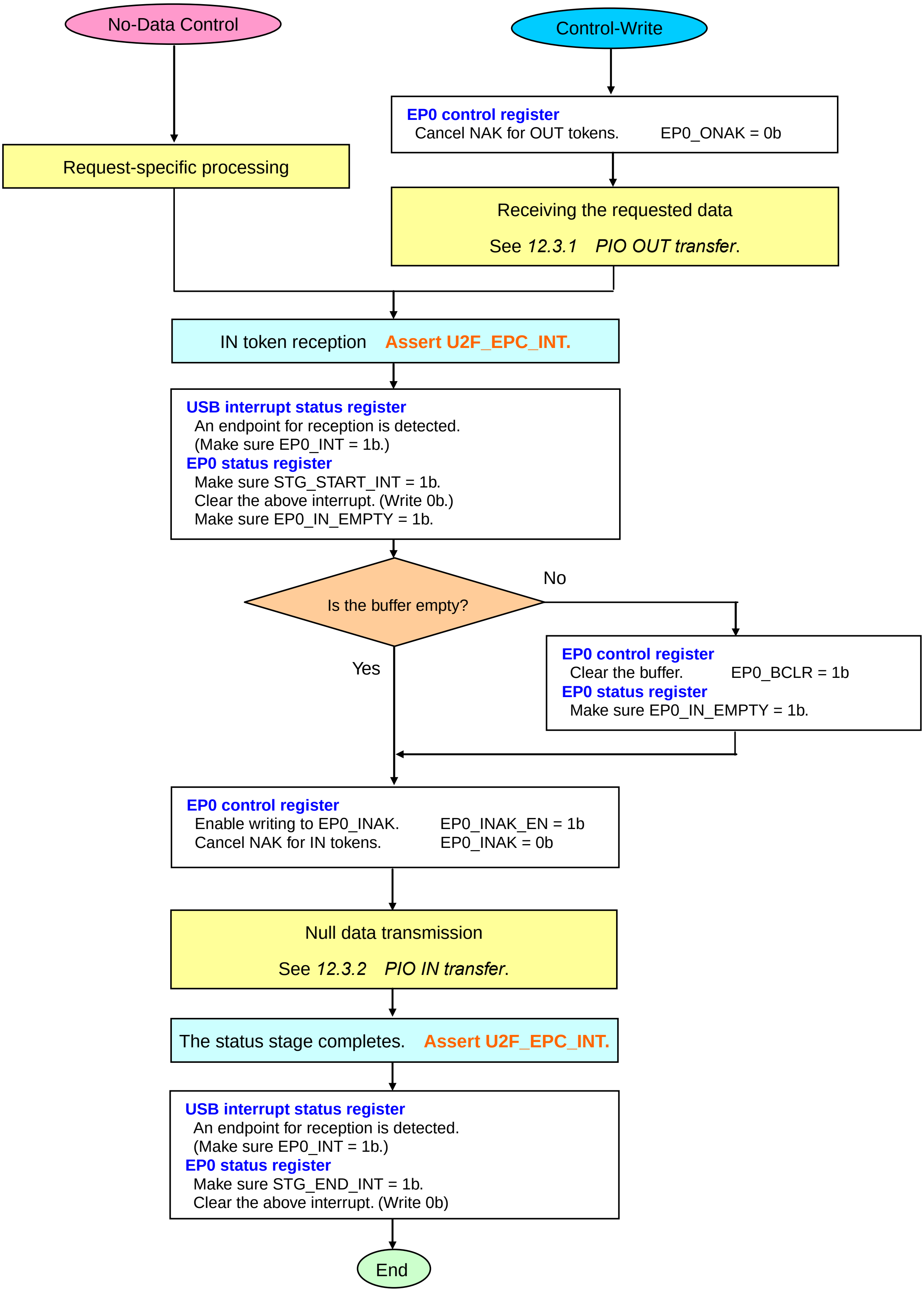
- EPn status register
  - EPn_IN_FULL/EPn_IN_EMPTY

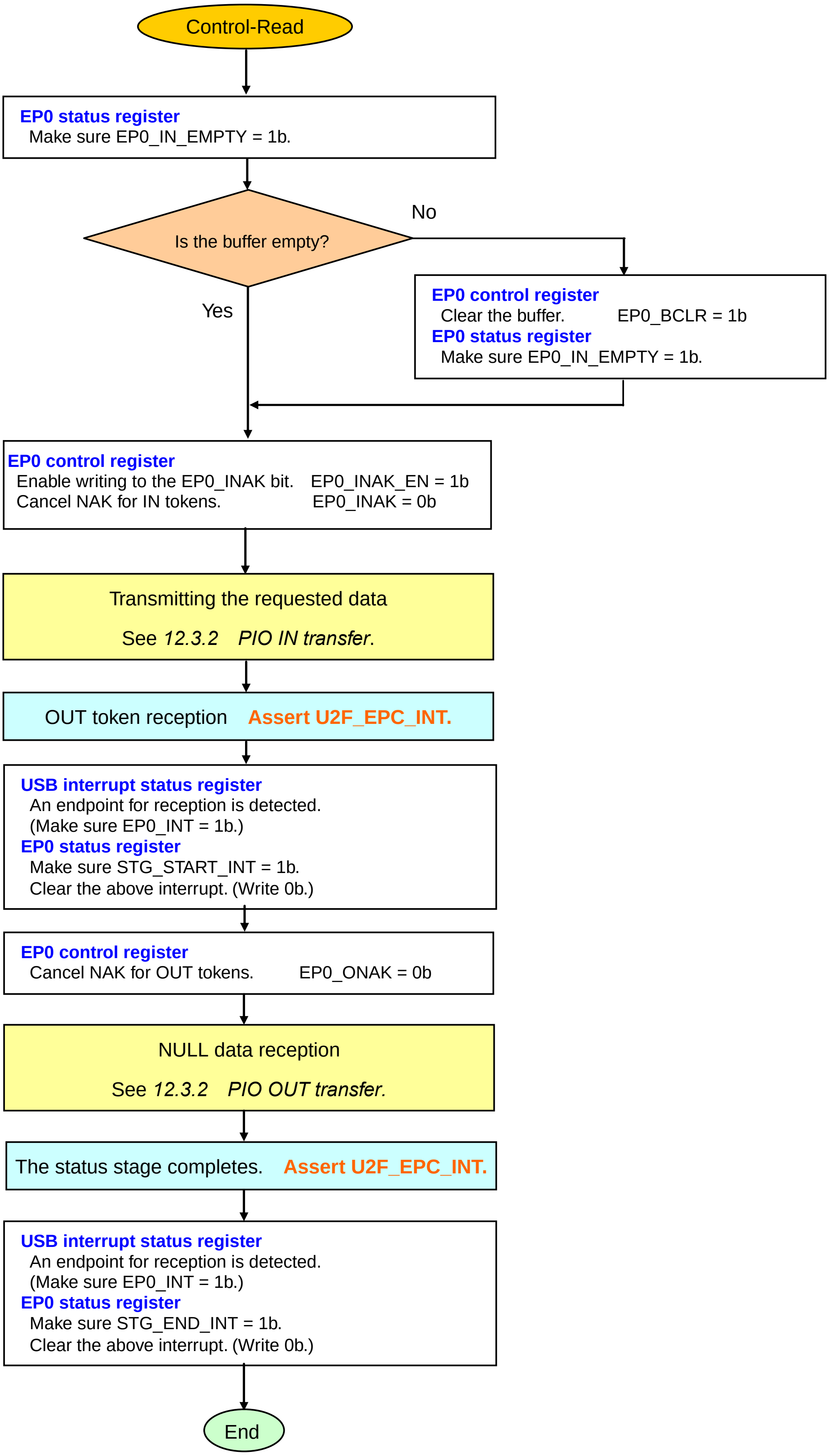
(5) There is no restriction on performing DMA transfer after a PIO transfer is finished.

12.3.7 Control transfer

Figure 12-8. Control Transfer Overview


Start
Characters in blue: EPC registers
Characters in red: AHB-EPC bridge registers
Setup transaction finishes. Assert U2F_EPC_INT.
USB interrupt status register
An endpoint for reception is detected.
(Make sure EP0_INT = 1b.)
EP0 status register
Make sure SETUP_INT = 1b.
Clear the above interrupt. (write 0b)
Setup data 1/0 register
Read the requested data.
Is the request unsupported?
Yes
No
EP0 control register
Stall transfer at endpoint 0. EP0_STL = 1b
USB interrupt status register
An endpoint for reception is detected.
(Make sure EP0_INT = 1b)
EP0 status register
Make sure EP0_STALL_INT = 1b
Clear the above interrupt. (write 0b)
End
To determine if the request is No-Data Control, Control-Write, or Control-Read, read bmRequestType and wLength in the request code by using the setup data 0/1 register.
No-Data Control request?
No
Yes
Control-Write request?
No
Yes
No-Data Control
Control-Write
Control-Read


No-Data Control
Control-Write
EP0 control register
Cancel NAK for OUT tokens. EP0_ONAK = 0b
Request-specific processing
Receiving the requested data
See *12.3.1 PIO OUT transfer*.
IN token reception Assert U2F_EPC_INT.
USB interrupt status register
An endpoint for reception is detected.
(Make sure EP0_INT = 1b.)
EP0 status register
Make sure STG_START_INT = 1b.
Clear the above interrupt. (Write 0b.)
Make sure EP0_IN_EMPTY = 1b.
Is the buffer empty?
No
Yes
EP0 control register
Clear the buffer. EP0_BCLR = 1b
EP0 status register
Make sure EP0_IN_EMPTY = 1b.
EP0 control register
Enable writing to EP0_INAK. EP0_INAK_EN = 1b
Cancel NAK for IN tokens. EP0_INAK = 0b
Null data transmission
See *12.3.2 PIO IN transfer*.
The status stage completes. Assert U2F_EPC_INT.
USB interrupt status register
An endpoint for reception is detected.
(Make sure EP0_INT = 1b.)
EP0 status register
Make sure STG_END_INT = 1b.
Clear the above interrupt. (Write 0b)
End


Control-Read
EP0 status register
Make sure EP0_IN_EMPTY = 1b.
Is the buffer empty?
No
Yes
EP0 control register
Clear the buffer. EP0_BCLR = 1b
EP0 status register
Make sure EP0_IN_EMPTY = 1b.
EP0 control register
Enable writing to the EP0_INAK bit. EP0_INAK_EN = 1b
Cancel NAK for IN tokens. EP0_INAK = 0b
Transmitting the requested data
See 12.3.2 PIO IN transfer.
OUT token reception Assert U2F_EPC_INT.
USB interrupt status register
An endpoint for reception is detected.
(Make sure EP0_INT = 1b.)
EP0 status register
Make sure STG_START_INT = 1b.
Clear the above interrupt. (Write 0b.)
EP0 control register
Cancel NAK for OUT tokens. EP0_ONAK = 0b
NULL data reception
See 12.3.2 PIO OUT transfer.
The status stage completes. Assert U2F_EPC_INT.
USB interrupt status register
An endpoint for reception is detected.
(Make sure EP0_INT = 1b.)
EP0 status register
Make sure STG_END_INT = 1b.
Clear the above interrupt. (Write 0b.)
End

12.3.8 Protocol error NAK processing

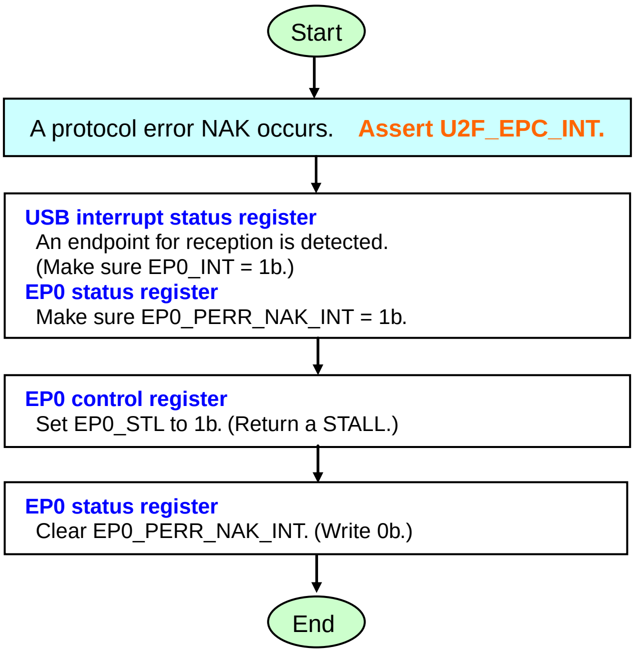
A protocol error NAK is a response returned when an invalid token whose explicit device response is not prescribed in the USB specification is received during a control transfer.

A protocol error NAK is generated in the following cases:

- An IN or OUT token is received before a SETUP token is received. (No setup stage is established.)
- An OUT token is received at a Control Read data stage.
- An IN token is received at a Control Read status stage or an OUT token is received for data PID0.
- An IN token is received at a Control Write status stage or an OUT token is received for data PID0 at the beginning of a data stage.
- An OUT token or PING token is received at the Control Write status stage.
- An OUT token is received at a No Data Control status stage.

If a protocol error NAK is returned, bit 16 of the EP0 status register is set and an EP0_PERR_NAK_INT interrupt occurs. If this interrupt is detected, halt the operation at EP0 and send a STALL for the subsequent tokens.

Figure 12-9. Protocol Error NAK Processing Overview


Start
A protocol error NAK occurs. Assert U2F_EPC_INT.
USB interrupt status register
An endpoint for reception is detected.
(Make sure EP0_INT = 1b.)
EP0 status register
Make sure EP0_PERR_NAK_INT = 1b.
EP0 control register
Set EP0_STL to 1b. (Return a STALL.)
EP0 status register
Clear EP0_PERR_NAK_INT. (Write 0b.)
End

12.3.9 Processing specific to each request

This section describes the standard requests for USB devices, states of the device, and the processes that must be performed by the device.

In the tables that show each request below, the **Request Code Field** column shows the code included in the standard request. The **Action** column shows the actions that must be taken by the device in each Default, Address, or Configured state stage. *STALL response* indicates that the device must send a STALL because a request error has occurred. To enable a STALL to be sent, set bit 2 (EP0_STL) of the EP0 control register (0028h) to 1b.

The device states are defined in the following table:

Table 12-1. Device States

Default	This is the state in which the USB device address is reset to 00h after a USB bus reset signal is received.
Address	This is the state in which the USB device address is set to a value other than 00h.
Configured	This is the state in which the configuration value is set to a value other than 00h, and accessing the relevant endpoint is enabled.

(1) Clear Feature

Table 12-2. Clear Feature Request

Request Code Field					Action		
bmRequest-Type	bRequest	wValue	wIndex	wLength	Default	Address	Configured
00h (device)	01h	0001h (remote wakeup)	0000h	0000h	Response (1) **Note**	Response (1) **Note**	Response (1) **Note**
01h (interface)	01h	0000h	0000h	0000h	STALL response	STALL response	STALL response
02h (endpoint)	01h	0000h (endpoint halt)	Endpoint number	0000h	Response (2)	Response (2)	Response (3)

Note The device must send a STALL if it does not support the remote wakeup feature.

○ Transfer type: No Data Control (SETUP-IN)

○ Processing details

Table 12-3. Clear Feature Request Processing

Response (1)	• Clear the remote wakeup flag. (The USB function controller does not need to perform special processing. However, if the remote wakeup flag is cleared, the USB function controller can no longer use the remote wakeup feature.)
Response (2)	• If `wIndex` is 0000h, clear the buffer for endpoint 0. (After setting bit 8 (EP0_BCLR) of the EP0 control register to 1b, make sure bit 8 (EP0_IN_EMPTY) of the EP0 status register is 1b.) • If `wIndex` is not 0000h, send a STALL.
Response (3)	• If `wIndex` is 0000h, clear the buffer for endpoint 0. (Perform the same processing as that of *Response (2)*). • If `wIndex` indicates the supported endpoint, clear the transmission/reception data PID, clear the halt status, and clear the buffers for the endpoint. (Clear the EPn_OPIDCLR, EPn_IPIDCLR, EPn_OSTL, EPn_ISTL, and EPn_BCLR bits of the EPn control register to 0b.) • If `wIndex` does not indicate a supported endpoint, send a STALL.

(2) Get Configuration

Table 12-4. Get Configuration Request

Request Code Field					Action		
bmRequest-Type	bRequest	wValue	wIndex	wLength	Default	Address	Configured
80h	08h	00h	0000h	0001h	Response (1)	Response (1)	Response (2)

○ Transfer type: Control Read (SETUP-IN-OUT)

○ Processing details

Table 12-5. Get Configuration Request Processing

Response (1)	• Send 00h (1 byte).
Response (2)	• Send the current configuration number (1 byte).

(3) Get Descriptor

Table 12-6. Get Descriptor Request

Request Code Field					Action		
bmRequest-Type	bRequest	wValue	wIndex	wLength	Default	Address	Configured
80h	06h	0100h (device)	0000h	Descriptor size	Response (1)	Response (1)	Response (1)
80h	06h	020Xh (config)	0000h	Descriptor size	Response (2)	Response (2)	Response (2)
80h	06h	030Xh (string)	0000h or language ID	Descriptor size	Response (3)	Response (3)	Response (3)
80h	06h	0600h (device qualifier)	0000h	Descriptor size	Response (4)	Response (4)	Response (4)
80h	06h	070Xh (other speed config)	0000h	Descriptor size	Response (5)	Response (5)	Response (5)

○ Transfer type: Control Read (SETUP-IN-OUT)

○ Processing details

Send the type specified by `wValue` and the index descriptor for the size specified by `wLength`. The following assumes that the value sent using `bMaxPacketSize(0)` of the device descriptor is 64.

- If the descriptor size is greater than `wLength`, send the descriptor for the size specified by `wLength`, from the beginning of the descriptor.
- If the descriptor size is less than `wLength`, send a short packet (64 bytes or less of data) at the end of descriptor transmission. At this time, if the descriptor size is a multiple of 64 bytes, send null data at the end.

Table 12-7. Get Descriptor Request Processing

Response (1)	Send a device descriptor.
Response (2)	Send the configuration descriptor indexed by the lower bits of `wValue`. If the configuration descriptor size is less than `wLength`, send all the interface and endpoint descriptors contained in the configuration successively.
Response (3)	Send the string descriptor indexed by the lower bits of `wValue`.
Response (4)	Send the device qualifier descriptor.
Response (5)	Send the other speed configuration descriptor indexed by the lower bits of `wValue`. If the configuration descriptor size is less than `wLength`, send all the interface and endpoint descriptors contained in the configuration successively.

(4) Get Interface

Table 12-8. Get Interface Request

Request Code Field					Action		
`bmRequest-Type`	`bRequest`	`wValue`	`wIndex`	`wLength`	Default	Address	Configured
81h	0Ah	00h	Interface number	0001h	STALL response	STALL response	Response (1)

○ Transfer type: Control Read (SETUP-IN-OUT)

○ Processing details

Table 12-9. Get Interface Request Processing

Response (1)	• Send the current alternate setting number for the interface specified by `wIndex`. • If no alternate setting is supported, send a STALL.

(5) Get Status

Table 12-10. Get Status Request

Request Code Field					Action		
`bmRequest-Type`	`bRequest`	`wValue`	`wIndex`	`wLength`	Default	Address	Configured
80h (device)	00h	00h	0000h	0002h	Response (1)	Response (1)	Response (1)
81h (interface)	00h	00h	0000h	0002h	Response (2)	Response (2)	Response (2)
82h (endpoint)	00h	00h	Endpoint number	0002h	Response (3)	Response (3)	Response (4)

○ Transfer type: Control Read (SETUP-IN-OUT)

○ Processing details

Table 12-11. Get Status Request Processing

Response (1)	• Send a return value in which whether self-powered devices are supported is specified for D0 and whether the remote wakeup feature is supported is specified for D1. (For example, if self-powered devices and the remote wakeup feature are not supported, send 0001h.)
Response (2)	• Send 0000h or a STALL.
Response (3)	• If `wIndex` is 0000h, send 0000h. • If `wIndex` is not 0000h, send a STALL.
Response (4)	• If `wIndex` is 0000h, send 0000h. • If `wIndex` indicates a supported endpoint, send a return value in which the halt (STALL) state is specified for D0. (Send 0001h if bit 3 or 2 (EPn_ISTL or EPn_OSTL) of the EPn control register is 1b. If 0b, send 0000b.) • If `wIndex` does not indicate a supported endpoint, send a STALL.

(6) Set Address

Table 12-12. Set Address Request

Request Code Field					Action		
`bmRequest -Type`	`bRequest`	`wValue`	`wIndex`	`wLength`	Default	Address	Configured
00h	05h	Device address	0000h	0000h	Response (1)	Response (1)	Response (1)

○ Transfer type: No Data Control (SETUP-IN)

○ Processing details

Table 12-13. Set Address Request Processing

Response (1)	• If `wValue` is less than or equal to 127, set the `wVaule` value to the frame number & USB address register. • If `wValue` is greater than or equal to 128, send a STALL.

(7) Set Configuration

Table 12-14. Set Configuration Request

Request Code Field					Action		
`bmRequest -Type`	`bRequest`	`wValue`	`wIndex`	`wLength`	Default	Address	Configured
00h	09h	Config value	0000h	0000h	Response (1)	Response (2)	Response (3)

○ Transfer type: No Data Control (SETUP-IN)

○ Processing details

Table 12-15. Set Configuration Request Processing

Response (1)	• No special processing is required. Send null data at the status stage.
Response (2)	• No special processing is required if `wValue` is 0. Send null data at the status stage. • If the `wValue` values matches the value of a supported configuration, the transaction transitions to the Configured state. (Set bit 5 (CONF) of the USB control register to 1b.) • If the `wValue` value does not match the value of a supported configuration, send a STALL.
Response (3)	• If `wValue` is 0, the transaction returns to the Address state. (Clear bit 5 (CONF) of the USB control register to 0b.) • No special processing is required if the `wValue` value matches the current `bConfigurationValue` value. Send null data at the status stage. • If the `wValue` value matches the value of a supported configuration, change the configuration. • If the `wValue` value does not match the value of the supported configuration, send a STALL.

(8) Set Descriptor

Table 12-16. Set Descriptor Request

Request Code Field					Action		
`bmRequest-Type`	`bRequest`	`wValue`	`wIndex`	`wLength`	Default	Address	Configured
00h	07h	Descriptor type	0000h or language ID	Descriptor size	Response (1)	Response (1)	Response (1)

○ Transfer type: Control Write (SETUP-OUT-IN)

○ Processing details

Table 12-17. Set Descriptor Request Processing

Response (1)	• If rewriting a descriptor is not enabled, send a STALL. • If rewriting a descriptor is enabled, read the data and write it to the descriptor again by using software. (The USB function controller does not need to perform special processing.)

(9) Set Feature

Table 12-18. Set Feature Request

Request Code Field					Action		
`bmRequest-Type`	`bRequest`	`wValue`	`wIndex`	`wLength`	Default	Address	Configured
00h (device)	03h	0001h (remote wakeup)	0000h	0000h	When remote wakeup is supported		
					Response (1)	Response (1)	Response (1)
					When remote wakeup is not supported		
					STALL response	STALL response	STALL response
01h (interface)	03h	0000h	0000h	0000h	STALL response	STALL response	STALL response
02h (endpoint)	03h	0000h (endpoint halt)	Endpoint number	0000h	Response (2)	Response (2)	Response (3)

○ Transfer type: No Data Control (SETUP-IN)

○ Processing details

Table 12-19. Set Feature Request Processing

Response (1)	• Set the remote wakeup flag. (The USB function controller does not need to perform special processing.)
Response (2)	• No special processing is required if `wIndex` is 0000h. Send null data at the status stage. • If `wIndex` is not 0000h, send a STALL.
Response (3)	• If `wIndex` is 0000h, clear the buffer for endpoint 0. (Perform the same processing as that of *Response (2)*). • If `wIndex` indicates a supported endpoint, specify the halt status for the endpoint. (Set bits 2 and 3 (EPn_OSTL and EPn_ISTL) of the EPn control register to 1b.) • If `wIndex` does not indicate a supported endpoint, send a STALL.

(10) Set Interface

Table 12-20. Set Interface Request

Request Code Field					Action		
`bmRequest-Type`	`bRequest`	`wValue`	`wIndex`	`wLength`	Default	Address	Configured
01h	0Bh	Alternate setting	Interface number	0000h	STALL response	STALL response	Response (1)

○ Transfer type: No Data Control (SETUP-IN)

○ Processing details

Table 12-21. Set Interface Request Processing

Response (1)	• If the alternate setting number specified by `wValue` differs from the current alternate setting number, change the values of the EPn max packet & base address register. • If the alternate setting number specified by `wValue` is the same as the current alternate setting number, no special processing is required. Send null data at the status stage.

(11) Sync Frame

Table 12-22. Sync Frame Request

Request Code Field					Action		
`bmRequest-Type`	`bRequest`	`wValue`	`wIndex`	`wLength`	Default	Address	Configured
82h	0Ch	0000h	EP number	0000h	STALL response	STALL response	STALL response

○ Transfer type: Control Read (SETUP-IN-OUT)

12.3.10 Descriptors

The types of standard descriptors and setting examples are described below.

(1) Device descriptor

A device descriptor is used to send basic information about the device.

Table 12-23. Device Descriptor

Field	Offset	Size	Value (Example)	Description
`bLength`	0	Byte	12h	Specify the descriptor size (18 bytes).
`bDescriptorType`	1	Byte	01h	For device descriptors, always set this field to 01h.
`bcdUSB`	2	Word	0200h	Indicates the USB standard version the device conforms to. 0200h indicates that the device conforms to USB 2.0.
`bDeviceClass`	4	Byte	*xx*h	Specify the class the device belongs to.
`bDeviceSubClass`	5	Byte	*xx*h	Specify the subclass the device belongs to.
`bDeviceProtocol`	6	Byte	00h	Specify the protocol defined by the class or subclass the device belongs to.
`bMaxPacktetSize0`	7	Byte	40h	Specify the maximum packet size (64 bytes) to be transferred at endpoint 0.
`idVendor`	8	Word	*xxxx*h	Specify the vendor ID.
`idProduct`	10	Word	*xxxx*h	Specify the product ID.
`bcdDevice`	12	Word	0100h	Indicates the device version.
`iManufacturer`	14	Byte	01h	Specify the index for the string descriptor that describes the manufacturer of the product.
`iProduct`	15	Byte	02h	Specify the index for the string descriptor that describes the product.
`iSerialNumber`	16	Byte	03h	Specify the index for the string descriptor that describes the serial number of the product.
`bNumConfigurations`	17	Byte	01h	Specify the number of supported configurations.

(2) Device qualifier descriptor

A device qualifier descriptor is used to send information about the device descriptor fields whose values change when HS and FS are switched.

Table 12-24. Device Qualifier Descriptor

Field	Offset	Size	Value (Example)	Description
`bLength`	0	Byte	0Ah	Specify the descriptor size (10 bytes).
`bDescriptorType`	1	Byte	06h	For device qualifier descriptors, always set this field to 06h.
`bcdUSB`	2	Word	0200h	Indicates the USB standard version the device conforms to. 0200h indicates that the device conforms to USB 2.0.
`bDeviceClass`	4	Byte	*xx*h	Specify the class the device belongs to.
`bDeviceSubClass`	5	Byte	*xx*h	Specify the subclass the device belongs to.
`bDeviceProtocol`	6	Byte	00h	Specify the protocol defined by the class or subclass the device belongs to.
`bMaxPacktetSize0`	7	Byte	40h	Specify the maximum packet size (64 bytes) to be transferred at endpoint 0.
`bNumConfigurations`	8	Byte	01h	Specify the number of supported configurations.
Reserved	9	Byte	00h	

(3) Configuration descriptor and other speed configuration descriptor

A configuration descriptor and other speed configuration descriptor are used to send information about the configuration of the device. When a device that supports both the fast speed and high speed is operating at either speed, the other speed configuration descriptor shows the configuration of the device if it operates at the other speed.

These descriptors are generally sent by using a single Get Descriptor Configuration request.

Table 12-25. Configuration Descriptor and Other Speed Configuration Descriptor

Field	Offset	Size	Value (Example)	Description
`bLength`	0	Byte	09h	Specify the descriptor size (9 bytes).
`bDescriptorType`	1	Byte	02h 07h	• For configuration descriptors, always set this field to 02h. • For other speed configuration descriptors, always set this field to 07h.
`wTotalLength`	2	Word	9 + (9 × *M*) + (7 × *N*)	Specify the size of the descriptor to be transferred. Specify a total of the following: Configuration descriptor size (9 bytes) Interface descriptor size (9 bytes) × number of descriptors *M* Endpoint descriptor size (7 bytes) × number of descriptor *N*
`bNumInterface`	4	Byte	01h	Specify the number of interfaces supported by this configuration.
`bConfigurationValue`	5	Byte	01h	Specify the configuration number.
`iConfiguration`	6	Byte	00h	Specify the index for the string descriptor that describes this configuration.
`bmAttributes`	7	Byte	C0h	• Bit 7: Always set to 1b. • Bit 6: Set to 1b if the device is self-powered. • Bit 5: Set to 1b if the device supports the remote wakeup feature. • Bits 4 to 0: Always set to 0b.
`MaxPower`	8	Byte	00h	Specify the current value required by the device. Value = current consumption value/2 (mA).

(4) Interface descriptor

An interface descriptor is used to send information about the device interface.

Table 12-26. Interface Descriptor

Field	Offset	Size	Value (Example)	Description
`bLength`	0	Byte	09h	Specify the descriptor size (9 bytes).
`bDescriptorType`	1	Byte	04h	For interface descriptors, always set this field to 04h.
`bInterfaceNumber`	2	Byte	00h	Specify the interface number.
`bAlternateSetting`	3	Byte	01h	• When using an alternate setting, specify the number for this field. • If there is not alternate setting, set this field to 00h.
`bNumEndpoints`	4	Byte	*xx*h	Specify the number of endpoints other than endpoint 0 supported by this interface.
`bInterfaceClass`	5	Byte	*xx*h	Specify the class the interface belongs to. This is equivalent to `bDeviceClass` in device descriptors.
`bInterfaceSubClass`	6	Byte	00h	Specify the subclass the interface belongs to. This is equivalent to `bDeviceSubClass` in device descriptors.
`bInterfaceProtocol`	7	Byte	*xx*h	Specify the protocol defined by the class or subclass the interface belongs to. This is equivalent to `bDeviceProtocol` in device descriptors.
`iInterface`	8	Byte	*xx*h	Specify the index for the string descriptor that describes this interface.

(5) Endpoint descriptor

An endpoint descriptor is used to send information about the endpoint used.

Table 12-27. Endpoint Descriptor

Field	Offset	Size	Value (Example)	Description
`bLength`	0	Byte	07h	Specify the descriptor size (7 bytes).
`bDescriptorType`	1	Byte	05h	For endpoint descriptors, always set this field to 05h.
`bEndpointAddress`	2	Byte	*xx*h	Specify the endpoint number and transfer direction. • Bit 7: Transfer direction (0b: OUT, 1b: IN) • Bits 6 to 4: Always set to 000b. • Bits 3 to 0: Endpoint number
`bmAttributes`	3	Byte	*xx*h	Specify the transfer type at the endpoint. • Bits 7 and 6: Always set to 00b. • Bits 5 and 4: Usage type (00b: Data endpoint, 01b: Feedback endpoint, 10b: Implicit feedback date endpoint, 11b: Reserved) • Bits 3 and 2: Synchronization type (00b: No Synchronization, 01b: Asynchronous, 10b: Adaptive, 11b: Synchronous) • Bits 1 and 0: Transfer type (00b: Control, 01b: Reserved, 10b: Bulk, 11b: Interrupt)
`wMaxPacketSize`	4	Word	0200h	Specify the maximum packet size to be sent at the endpoint. • Bits 15 to 13: Always set to 000b. • Bits 12 and 11: Specify the number of additional transfers per microframe. 00: None (Once per μFrame) 01: 1 (Twice per μFrame) 10: 2 (Three times per μFrame) 11: Reserved • Bits 10 to 0: Specify the maximum packet size.
`bInterval`	6	Byte	00h	For interrupt transfers, specify the maximum wait time. For bulk and control transfers, specify the maximum NAK ratio.

(6) String descriptor

A string descriptor is used to send the text for description.

Table 12-28. String Descriptor

Field	Offset	Size	Value (Example)	Description
bLength	0	Byte	*xx*h	Specify the descriptor size. The size depends on the text to be sent.
bDescriptorType	1	Byte	03h	For string descriptors, always set this field to 03h.
wLANGID bString	2	Byte	0409h	• When wIndex is 0, specify the language of the string for wLANGID. • When wIndex is not 0, specify a character string indexed by the lower bits of the wValue value for bString, by using the language specified by wIndex.

REVISION HISTORY	EMMA Mobile EV2 User’s Manual: USB 2.0 Function Controller

Rev.	Date	Description	
		Page	Summary
1.00	Sep 30, 2010	—	1st revision release
2.00	May 31, 2011	—	Incremental update from comments to the 1.0.
3.00	Sep 30, 2011	47	Chapter 4.2.5 added.
4.00	Mar 30, 2012	—	Isochronous transfer related description deleted.

EMMA Mobile EV2 User’s Manual:
USB 2.0 Function Controller

Publication Date:	Rev.1.00	Sep 30, 2010
	Rev.4.00	Mar 30, 2012

Published by: Renesas Electronics Corporation

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: +44-1628-585-100, Fax: +44-1628-585-900

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.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

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

Colophon 1.1

USB2.0 Function Controller

EMMA Mobile EV2

Renesas Electronics Corporation