User's Manual

32

RZ/N1D Group, RZ/N1S Group, RZ/N1L Group

User's Manual: System Control and Peripheral

RZ Family
RZ/N Series

Renesas Electronics
www.renesas.com

Rev.1.20 Dec, 2021

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating the safety of the final products or systems 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. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on
   The state of the product is undefined at the time 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 time 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 time 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 time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins
   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the 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.
5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is 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. Additionally, 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.
6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

How to Use This Manual

1. Objective and Target Users

This manual was written to explain the hardware functions and electrical characteristics of this LSI to the target users, i.e. those who will be using this LSI in the design of application systems. Target users are expected to understand the fundamentals of electrical circuits, logic circuits, and microcomputers.

This manual is organized in the following items: an overview of the product, descriptions of the CPU, system control functions, and peripheral functions, electrical characteristics of the device, and usage notes.

When designing an application system that includes this LSI, take all points to note into account.
Points to note are given in their contexts and at the final part of each section, and in the section giving usage notes.

The list of revisions is a summary of major points of revision or addition for earlier versions. It does not cover all revised items. For details on the revised points, see the actual locations in the manual.

The following documents have been prepared for reference.

■ Documents related to RZ/N1

Document Name	Document Number
RZ/N1D Group, RZ/N1S Group, RZ/N1L Group DATASHEET	R01DS0323EJ****
RZ/N1D Group, RZ/N1S Group, RZ/N1L Group User's Manual: System Introduction, Multiplexing, Electrical and Mechanical Information	R01UH0750EJ****
RZ/N1D Group, RZ/N1S Group, RZ/N1L Group User's Manual: System Control and Peripheral	R01UH0751EJ**** (this manual)
RZ/N1D Group, RZ/N1S Group, RZ/N1L Group User's Manual: Peripherals	R01UH0752EJ****
RZ/N1D Group, RZ/N1S Group, RZ/N1L Group User's Manual: R-IN Engine and Ethernet Peripherals	R01UH0753EJ****
RZ/N1D Group, RZ/N1S Group, RZ/N1L Group User's Manual: PWMTimer	R01UH0913EJ****

2. Description of Registers

Each register description includes a bit chart, illustrating the arrangement of bits, and a table of bits, describing the meanings of the bit settings. The standard format and notation for bit charts and tables are described below.

X.X.X [Register Name]

(2)

Address: XXXX XXXXh

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	[Bit Field]					—	—	[Bit Field]		—	[Bit]	[Bit]	[Bit]
Value after reset	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0

Table X.X [Register Name] Register Contents

Bit Position	Bit Name	Function	R/W
b12 to b8	[Bit Field]	[Description]	R/W
b5 to b4	[Bit Field]	[Description] 2'b00: Hi-Z 2'b01: L Output Others: Prohibited (3)	R/W
b2	[Bit]	[Description] 1'b0: H-Z 1'b1: Output (defult)	R/W
b1	[Bit]	[Description] 1'b0: H-Z 1'b1: Output (defult)	R/W
b0	[Bit]	[Description] 1'b0: H-Z 1'b1: Output	R/W (1)

(1) R/W: The bit or field is readable and writable.

R/(W): The bit or field is readable and writable. However, writing to this bit or field has some limitations. For details on the limitations, see the description or notes of respective registers.

R: The bit or field is readable. Writing to this bit or field has no effect.

W: The bit or field is writable. Reading to this bit or field is not guaranteed.

(2) Reserved. Make sure to use the specified value when writing to this bit or field; otherwise, the correct operation is not guaranteed.

(3) Setting prohibited. The correct operation is not guaranteed if such a setting is performed.

3. List of Abbreviations and Acronyms

Abbreviation	Full Form
AHB	Arm Advanced High-performance Bus
APB	Arm Advanced Peripheral Bus
AXI	Arm Advanced eXtensible Interface
bps	bits per second
CA7	Arm Cortex-A7 module
CM3	Arm Cortex-M3 module
CRC	Cyclic Redundancy Check
DMA	Direct Memory Access
DMAC	Direct Memory Access Controller
Hi-Z	High Impedance
HSR	High-availability Seamless Redundancy
HW-RTOS	Hard Ware Real Time OS
I/O	Input/Output
INTC	Interrupt Controller
LSB	Least Significant Bit
MSB	Most Significant Bit
NC	Non-Connect
NoC	Network-on-Chip
PLL	Phase Locked Loop
PWM	Pulse Width Modulation
UART	Universal Asynchronous Receiver/Transmitter
OTP	One Time Programmable
PTP	Precision Time Protocol
PRP	Parallel Redundancy Protocol
SoC	System On Chip

4. Description of the Access Size

Access size:

8 bits = Byte

16 bits = Halfword

32 bits = Word

Table of Contents

Section 1 CPU

1.1 Arm Cortex-A7

1.1.1 Overview

Cortex®-A7 processors in RZ/N1 coupled with subsystem composed of FPU, ETM, debug function, L1 cache, L2 cache, and GIC are integrated inside “Cortex-A7 Core Subsystem”.

More information of Cortex-A7 can be obtained from:

https://developer.arm.com/products/processors/cortex-a/cortex-a7

The figures below show the interfaces of the Arm® Cortex-A7 core and its connections to other blocks.

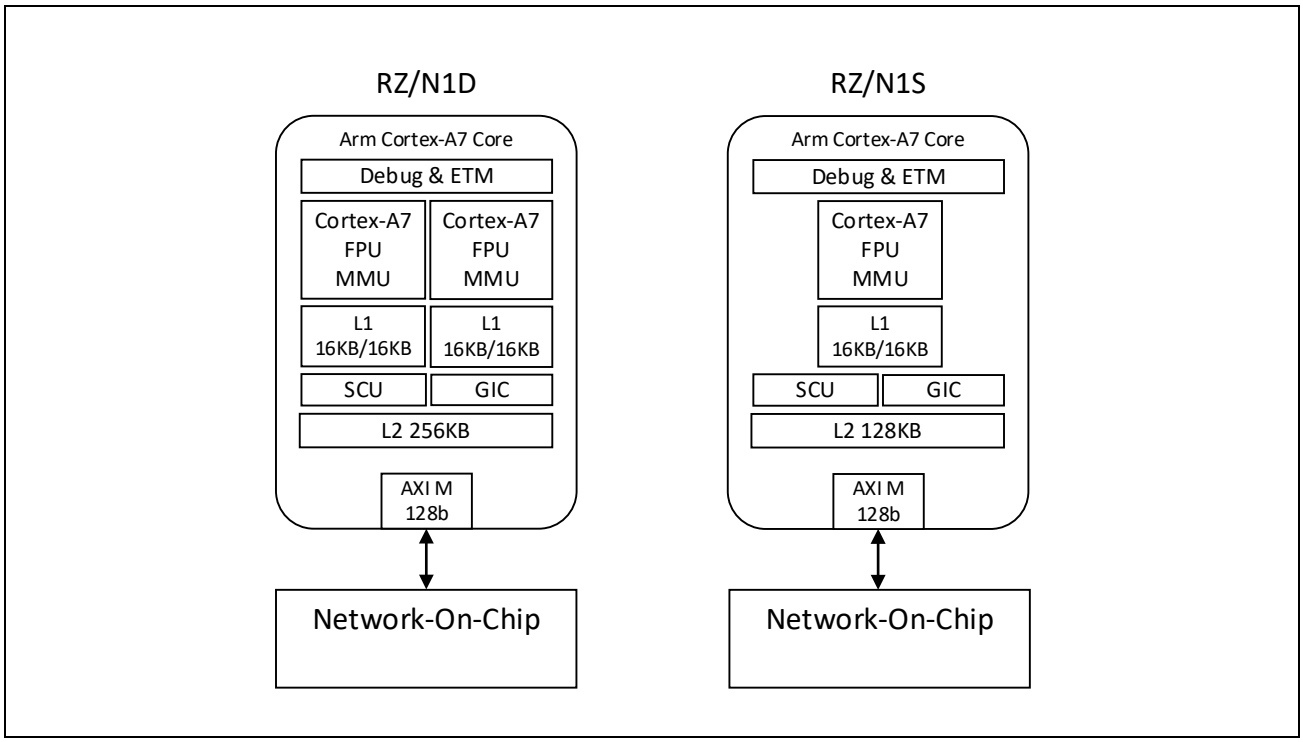


Figure 1.1 Arm Cortex-A7 Core Interfaces and Connections

The Cortex-A7 processors are configured with following features:

- Arm Cortex-A7 revision: r0p5
- L1 cache 16 KB (instruction)/16 KB (data)
- L2 cache (RZ/N1D: 256 KB, RZ/N1S: 128 KB)
- FPU revision VFPv4-D16
- MMU
- Integrated GIC
- Hardware coherent caches
  - The cache coherency is not going down to the NoC.
- Frequency: up to 500 MHz

- Target Frequency depends on NoC frequency
  - Can be dynamically changed by PWRCTRL_CA7DIV register
  - Clkin variation with NoC (x1): Up to 125 MHz
  - Clkin variation with NoC (x2): Up to 250 MHz
  - Clkin variation with NoC (x4): Up to 500 MHz

  See more detail in *Section 3.5, Clock Frequency Scaling* in *the RZ/N1D Group, RZ/N1S Group, RZ/N1L Group User's Manual: System Introduction, Multiplexing, Electrical and Mechanical Information*.
- Generic timer: 6.25 MHz
- Configuration Base Address Register (CBAR) value: 4410_0000h
- Not support for TrustZone

  Only secure world (default) is available.

1.1.2 Usage Notes

Cortex-A7 frequency can be 1 or 2 or 4 times of NoC frequency. The PWRCTRL_CA7DIV register controls the divider setting value and it can be written by Cortex-A7.

Ex.) Changing the divider setting value from 2 (default) to 4 (= NoC frequency)

Step 1: Wait if PWRCTRL_CA7DIV.BUSY = 1
Step 2: Write 8000_0004h to PWRCTRL_CA7DIV
Step 3: Wait if PWRCTRL_CA7DIV.BUSY = 1

1.2 Arm Cortex-M3

1.2.1 Overview

The Cortex-M3 is a low-power processor that features low interrupt latency, and low-cost debug. It is optimal for deeply embedded applications that require optimal interrupt response features and supports Armv7-M Thumb® instruction set.

More information of Cortex-M3 can be obtained from:

https://developer.arm.com/products/processors/cortex-m/cortex-m3

The Cortex-M3 processors is configured with following features:

- Arm Cortex-M3 revision: r2p1
- ITM, TPIU, FPB, DWT, ETM debug function supported
- Target Frequency depends on NoC frequency

  Clkin variation: From 15.625 MHz (Divider: 16) to 125 MHz (Divider: 2)
- Memory Protection Unit (MPU) supported

1.2.2 Signal Interface

Signal Name	Input Output	Description
Clock		
CM3_FCLK	Input	Cortex-M3 core clock
CM3_HCLK	Input	Cortex-M3 core and AHB clock
CM3_STCLK	Input	Systick clock. Fixed 6.25MHz, no clock gating
Related Clock		
RINBUS_HCLK	—	R-IN Engine clock. Since Cortex-M3 belongs to R-IN Engine, this clock should be supplied.
External Signal		
NMI_CORTEXM3	input	Non-maskable Interrupt, active high

1.2.3 Usage Notes

In case of accessing registers, please set the “Device” or “Strongly Ordered” memory attribute. Otherwise, a register which has a read-clear bit may result in being cleared unintentionally by prefetch operation.

Following function area can be accessed from Cortex-M3 only.

- HW-RTOS resource

1.2.3.1 Restriction

Since NoC (Network-on-Chip) does not support Lock access, the “bit-banding” feature of Cortex-M3 via NoC does not result in real atomic accesses. Software is responsible for the consequences.

For details of the restrictions on Cortex-M3 core, please refer to above Arm website.

Section 2 Network-On-Chip

2.1 Overview

On-chip connectivity for SoC function blocks implementing combinations of internal bus protocols and Clock gating management.

2.2 Features

Network-On-Chip has the following features:

- Clock gating is built around NoC and Clock Controller allowing the following when it is possible:
  - Disconnect & Connect services
  - Bus error management on disconnect port.
  - Clock gating
- Can be dynamically changed by dedicated integer divider
  - Clkin variation: From 15.625 MHz (Divider: 16) to 125 MHz (Divider: 2)
- Secure/Not Secure Access management
- Priority/Round Robin Arbitration Combination scheme to ensure high memory bandwidth utilization

CAUTION

- The cache coherency is not going down to the NoC. This feature is only managed between L1 & L2 cache of Cortex-A7
- No LOCKED access supported:
  Therefore the “bit-banding” feature of Cortex-M3 via NoC does not result in real atomic accesses. Software is responsible for the consequences.

Table 2.1 RZ/N1D Bus Connection Map

Bus Slave \ Bus Master	Cortex-A7	Cortex-M3	NAND Flash	SDIO1 SDIO2	USBh USBf	DMAC1	DMAC2	GMAC1 GMAC2	MSEBI Slave	LCDC	Core Sight AHB	Core Sight ETR
DDR2/3 (mem) port0*1	✓				✓						✓	✓
DDR2/3 (mem) port1*1								✓	✓			
DDR2/3 (mem) port2*1			✓	✓		✓						
DDR2/3 (mem) port3*1		✓					✓			✓		
DDR2/3 (reg)	✓	✓									✓	
2MB SRAM (mem)*2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
QSPI1 (mem)	✓	✓					✓				✓	
QSPI1 (reg)	✓	✓									✓	
NAND Flash	✓	✓									✓	
SDIO1/SDIO2	✓	✓									✓	
USBh/USBf	✓	✓									✓	
DMAC1/DMAC2	✓	✓									✓	
R-IN Engine Accessory Register Ethernet Accessory Register	✓	✓									✓	
GMAC1/GMAC2	✓	✓									✓	
A5PSW	✓	✓									✓	
HSR	✓	✓									✓	
ETHERCAT/SERCOSIII	✓	✓				✓	✓				✓	
MSEBIM/MSEBIS	✓	✓					✓				✓	
Peripheral Group1/2/3/4	✓	✓				✓	✓		✓		✓	
Mailbox	✓	✓									✓	
Peripheral Group0	✓	✓					✓				✓	
System Control Watchdog for CA7/CM3	✓	✓									✓	
RTC	✓	✓									✓	
ROM	✓											
CoreSight	✓	✓									✓	

Note 1. Each port can access the same address area (Up to 2 GB address space).

Note 2. There is a port for instruction and data respectively.

Table 2.2 RZ/N1S Bus Connection Map

Bus Master / Bus Slave	Cortex-A7	Cortex-M3	NAND Flash	SDIO1 SDIO2	USBh USBf	DMAC1	DMAC2	GMAC1 GMAC2	MSEBI Slave	LCDC	Core Sight AHB	Core Sight ETR
4MB SRAM (mem)*1	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
4MB SRAM_CTRL (reg)	✓	✓									✓	
2MB SRAM (mem)*2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
QSPI1/QSPI2 (mem)	✓	✓					✓				✓	
QSPI1/QSPI2 (reg)	✓	✓									✓	
NAND Flash	✓	✓									✓	
SDIO1/SDIO2	✓	✓									✓	
USBh/USBf	✓	✓									✓	
DMAC1/DMAC2	✓	✓									✓	
R-IN Engine Accessory Register Ethernet Accessory Register	✓	✓									✓	
GMAC1/GMAC2	✓	✓									✓	
A5PSW	✓	✓									✓	
ETHERCAT/SERCOSIII	✓	✓				✓	✓				✓	
MSEBIM/MSEBIS	✓	✓					✓				✓	
Peripheral Group1/2/3/4	✓	✓				✓	✓		✓		✓	
Mailbox	✓	✓									✓	
Peripheral Group0	✓	✓					✓				✓	
System Control Watchdog for CA7/CM3	✓	✓									✓	
RTC	✓	✓									✓	
ROM	✓											
CoreSight	✓	✓									✓	

Note 1. There is a memory bank with independent port every 1 MB.

Note 2. There is a port for instruction and data respectively.

Table 2.3 RZ/N1L Bus Connection Map

Bus Master / Bus Slave	Cortex-M3	NAND Flash	SDIO1	USBh USBf	DMAC1	DMAC2	GMAC1	MSEBI Slave	Core Sight AHB	Core Sight ETR
4MB SRAM (mem)*1	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
4MB SRAM_CTRL (reg)	✓								✓	
2MB SRAM (mem)*2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
QSPI1 (mem)	✓					✓			✓	
QSPI1 (reg)	✓								✓	
NAND Flash	✓								✓	
SDIO1/SDIO2	✓								✓	
USBh/USBf	✓								✓	
DMAC1/DMAC2	✓								✓	
R-IN Engine Accessory Register Ethernet Accessory Register	✓								✓	
GMAC1 GMAC2	✓								✓	
A5PSW	✓								✓	
ETHERCAT/SERCOSIII	✓				✓	✓			✓	
MSEBIS	✓					✓			✓	
Peripheral Group1/2/3/4	✓				✓	✓		✓	✓	
Peripheral Group0	✓					✓			✓	
System Control Watchdog for CM3	✓								✓	
CoreSight	✓								✓	

Note 1. There is a memory bank with independent port every 1 MB.

Note 2. There is a port for instruction and data respectively.

Section 3 2MB SRAM

All RZ/N1 have 2MB SRAM in R-IN Engine block. It consists of 2 of 1MB SRAMs with ECC function and can be accessed from CPU and some circuits.

3.1 Overview

Table 3.1 2MB SRAM Summary

Item	Description
RAM capacity	2 Mbytes (Instruction RAM: 1 Mbyte, Data RAM: 1 Mbyte)
RAM address	Instruction RAM 0400 0000h to 040F FFFFh Data RAM 2000 0000h to 200F FFFFh
Error checking	Correction of single error and detection of double error Self-Testing of the ECC Circuit Interrupt management

3.2 Signal Interfaces

Signal Name	Input Output	Description
Clock		
SRAM2MB_HCLK	Input	Internal bus clock (AHB), no clock gating
RINBUS_HCLK	Input	R-IN Engine local bus clock, commonly used for Cortex-M3, HW-RTOS, R-IN Engine Accessory register
Interrupt		
ECC_2MB_Int	Output	Pulse sensitive interrupt, Active High

3.3 Register Map

2MB SRAM registers can be accessed by 32-bit units.

For accessing this module registers, RINBUS_HCLK must be enabled and RIN BUS sub system must not have been reset. The detail is described on Register Description of PWRCTRL_RINCTRL.

NOTE

RAMEDC and RAMEEC registers are write protected by RAMPCMD.

Table 3.2 2MB SRAM Control Register List

Address	Register Symbol	Register Name
400F 3000h	RAMPCMD	RAM_SYS Protect Command Register
400F 3100h	RAMEDC	RAM_SYS ECC Decoder Config Register
400F 3104h	RAMEEC	RAM_SYS ECC Encoder Config Register
400F 3108h	RAMDBEST	RAM_SYS Double Bit ECC Error Status Register
400F 310Ch	RAMDBEAD	RAM_SYS Double Bit ECC Error Address Register
400F 3110h	RAMDBECNT	RAM_SYS Double Bit ECC Error Counter Register

3.4 Register Description

3.4.1 RAMPCMD — RAM_SYS Protect Command Register

Please refer to **Section 3.6, Usage Notes**, for more detail.

Address: 400F 3000h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	PROTREL
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3.3 RAMPCMD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b1	Reserved		R
b0	PROTREL	Write access to protection registers are permitted. 1: Write access to protection registers are permitted. 0: Write access to protection registers are not permitted.	R/W

3.4.2 RAMEDC — RAM_SYS ECC Decoder Config Register

RAMECC register is write protected by RAMPCMD.

Please refer to **Section 3.6, Usage Notes**, for more detail.

Address: 400F 3100h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	ECC_ENABLE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3.4 RAMEDC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b1	Reserved		R
b0	ECC_ENABLE	ECC decoder enable. 0: Disable 1: Enable	R/W

3.4.3 RAMEEC — RAM_SYS ECC Encoder Config Register

RAMEEC register is write protected by RAMPCMD.

Please refer to **Section 3.6, Usage Notes**, for more detail.

Address: 400F 3104h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DBE_DIST15	DBE_DIST14	DBE_DIST13	DBE_DIST12	DBE_DIST11	DBE_DIST10	DBE_DIST9	DBE_DIST8	DBE_DIST7	DBE_DIST6	DBE_DIST5	DBE_DIST4	DBE_DIST3	DBE_DIST2	DBE_DIST1	DBE_DIST0
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3.5 RAMEEC Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b20	Reserved		R
b19 to b16	Reserved	Keep the initial value.	R/W
b15	DBE_DIST15	ECC error injection enable of Data RAM BANK1 WAY3. 0: Disable 1: Enable	R/W
b14	DBE_DIST14	ECC error injection enable of Data RAM BANK1 WAY2. Same bit function as DBE_DIST15	R/W
b13	DBE_DIST13	ECC error injection enable of Data RAM BANK1 WAY1. Same bit function as DBE_DIST15	R/W
b12	DBE_DIST12	ECC error injection enable of Data RAM BANK1 WAY0. Same bit function as DBE_DIST15	R/W
b11	DBE_DIST11	ECC error injection enable of Data RAM BANK0 WAY3. Same bit function as DBE_DIST15	R/W
b10	DBE_DIST10	ECC error injection enable of Data RAM BANK0 WAY2. Same bit function as DBE_DIST15	R/W
b9	DBE_DIST9	ECC error injection enable of Data RAM BANK0 WAY1. Same bit function as DBE_DIST15	R/W
b8	DBE_DIST8	ECC error injection enable of Data RAM BANK0 WAY0. Same bit function as DBE_DIST15	R/W
b7	DBE_DIST7	ECC error injection enable of Instruction RAM BANK1 WAY3. Same bit function as DBE_DIST15	R/W
b6	DBE_DIST6	ECC error injection enable of Instruction RAM BANK1 WAY2. Same bit function as DBE_DIST15	R/W
b5	DBE_DIST5	ECC error injection enable of Instruction RAM BANK1 WAY1. Same bit function as DBE_DIST15	R/W
b4	DBE_DIST4	ECC error injection enable of Instruction RAM BANK1 WAY0. Same bit function as DBE_DIST15	R/W
b3	DBE_DIST3	ECC error injection enable of Instruction RAM BANK0 WAY3. Same bit function as DBE_DIST15	R/W
b2	DBE_DIST2	ECC error injection enable of Instruction RAM BANK0 WAY2. Same bit function as DBE_DIST15	R/W

Table 3.5 RAMEEC Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b1	DBE_DIST1	ECC error injection enable of Instruction RAM BANK0 WAY1. Same bit function as DBE_DIST15	R/W
b0	DBE_DIST0	ECC error injection enable of Instruction RAM BANK0 WAY0. Same bit function as DBE_DIST15	R/W

3.4.4 RAMDBEST — RAM_SYS Double Bit ECC Error Status Register

Please refer to **Section 3.6, Usage Notes**, for more detail.

Address: 400F 3108h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DBE_RAM15	DBE_RAM14	DBE_RAM13	DBE_RAM12	DBE_RAM11	DBE_RAM10	DBE_RAM9	DBE_RAM8	DBE_RAM7	DBE_RAM6	DBE_RAM5	DBE_RAM4	DBE_RAM3	DBE_RAM2	DBE_RAM1	DBE_RAM0
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3.6 RAMDBEST Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved		R
b15	DBE_RAM15	ECC double bit error detection flag of Data RAM BANK1 WAY3. 0: Error not occurred 1: Error occurred	R
b14	DBE_RAM14	ECC double bit error detection flag of Data RAM BANK1 WAY2. Same bit function as DBE_RAM15	R
b13	DBE_RAM13	ECC double bit error detection flag of Data RAM BANK1 WAY1. Same bit function as DBE_RAM15	R
b12	DBE_RAM12	ECC double bit error detection flag of Data RAM BANK1 WAY0. Same bit function as DBE_RAM15	R
b11	DBE_RAM11	ECC double bit error detection flag of Data RAM BANK0 WAY3. Same bit function as DBE_RAM15	R
b10	DBE_RAM10	ECC double bit error detection flag of Data RAM BANK0 WAY2. Same bit function as DBE_RAM15	R
b9	DBE_RAM9	ECC double bit error detection flag of Data RAM BANK0 WAY1. Same bit function as DBE_RAM15	R
b8	DBE_RAM8	ECC double bit error detection flag of Data RAM BANK0 WAY0. Same bit function as DBE_RAM15	R
b7	DBE_RAM7	ECC double bit error detection flag of Instruction RAM BANK1 WAY3. Same bit function as DBE_RAM15	R
b6	DBE_RAM6	ECC double bit error detection flag of Instruction RAM BANK1 WAY2. Same bit function as DBE_RAM15	R
b5	DBE_RAM5	ECC double bit error detection flag of Instruction RAM BANK1 WAY1. Same bit function as DBE_RAM15	R
b4	DBE_RAM4	ECC double bit error detection flag of Instruction RAM BANK1 WAY0. Same bit function as DBE_RAM15	R
b3	DBE_RAM3	ECC double bit error detection flag of Instruction RAM BANK0 WAY3. Same bit function as DBE_RAM15	R
b2	DBE_RAM2	ECC double bit error detection flag of Instruction RAM BANK0 WAY2. Same bit function as DBE_RAM15	R

Table 3.6 RAMDBEST Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b1	DBE_RAM1	ECC double bit error detection flag of Instruction RAM BANK0 WAY1. Same bit function as DBE_RAM15	R
b0	DBE_RAM0	ECC double bit error detection flag of Instruction RAM BANK0 WAY0. Same bit function as DBE_RAM15	R

3.4.5 RAMDBEAD — RAM_SYS Double Bit ECC Error Address Register

Please refer to **Section 3.6, Usage Notes**, for more detail.

Address: 400F 310Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	BANK		ADDRESS		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ADDRESS														—	LOCK
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3.7 RAMDBEAD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b21	Reserved		R
b20, b19	BANK	ECC double bit Error BANK These bits indicate a bank number where an ECC double bit error was encountered. 00b: Instruction RAM BANK0 01b: Instruction RAM BANK1 10b: Data RAM BANK0 11b: Data RAM BANK1	R
b18 to b2	ADDRESS	ECC double bit Error Address The double bit ECC error address is retained.	R
b1	Reserved		R
b0	LOCK	Lock Enable 0: Register unlocked (Double bit ECC error address can be captured.) 1: Register locked (Double bit ECC error address cannot be captured.) Read this register to unlock the registers.	R

3.4.6 RAMDBECNT — RAM_SYS Double Bit ECC Error Counter Register

Please refer to **Section 3.6, Usage Notes**, for more detail.

Address: 400F 3110h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	ERRCOUNT			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3.8 RAMDBECNT Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	Reserved		R
b3 to b0	ERRCOUNT	Double bit ECC error counter.	R

3.5 Operation

3.5.1 Configuration of Memory Map

The instruction RAM and data RAM consist of 512-Kbyte RAMs, each having a two-bank four-way configuration.

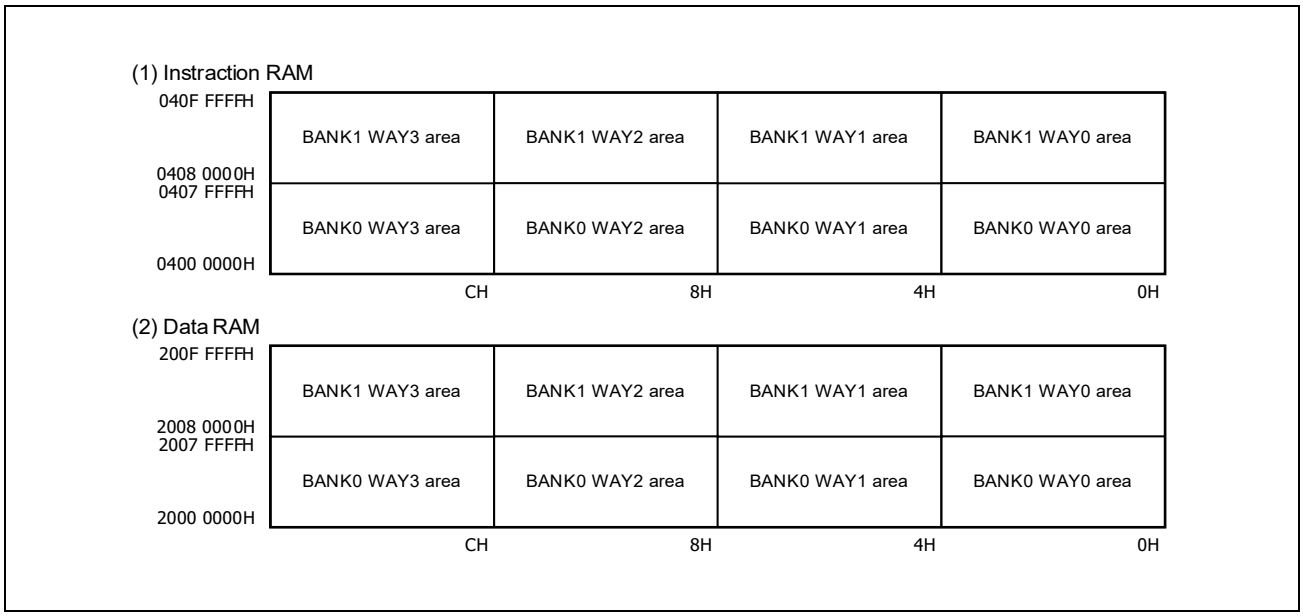


Figure 3.1 Configuration of Memory

3.5.2 Initializing

For normal operation of the ECC circuit, it is necessary to initialize the instruction RAM and data RAM.
(When the boot of RZ/N1, CPU (Cortex-A7 or Cortex-M3) uses the instruction RAM and data RAM. At the time, ECC error protection is not started.) For accessing this module registers, RINBUS_HCLK must be enabled and RIN BUS sub system must not have been reset. The detail is described on Register Description of PWRCTRL_RINCTRL.

RAM initialize:

ECC error correction cannot be enabled before initializing the instruction RAM and data RAM. After the initializing sequence below, set RAMEDC.ECC_ENABLE register to 1 to enable ECC error correction.

Initialize from 0x04000000 to 0x040FFFFF and 0x20000000 to 0x200FFFFF.

1. Disable cache in case of Cortex-A7, disable NMI in case of Cortex-M3
2. Mask the ECC interrupt
3. Read the data from the initialize area.
4. Write the Read data to the same address.

3.5.3 ECC Error Correction Function

RAMEDC register enables or disables ECC error correction in the 2-Mbyte space. ECCs can be used to correct single error and detect double error. Cortex-A7 and Cortex-M3 detects the sources of double errors, and RAMDBEST register is used to check the WAY in which an error was found. Moreover, RAMDBEAD register can be used to identify the address where a double error was found, and RAMDBECNT indicates the number of double errors that have been encountered.

3.5.4 Self-Testing of the ECC Circuit

Self-testing of the ECC circuit proceeds in way units. RAMEEC register sets the target area for each way. The following shows an example of the procedure for self-testing of the ECC circuit.

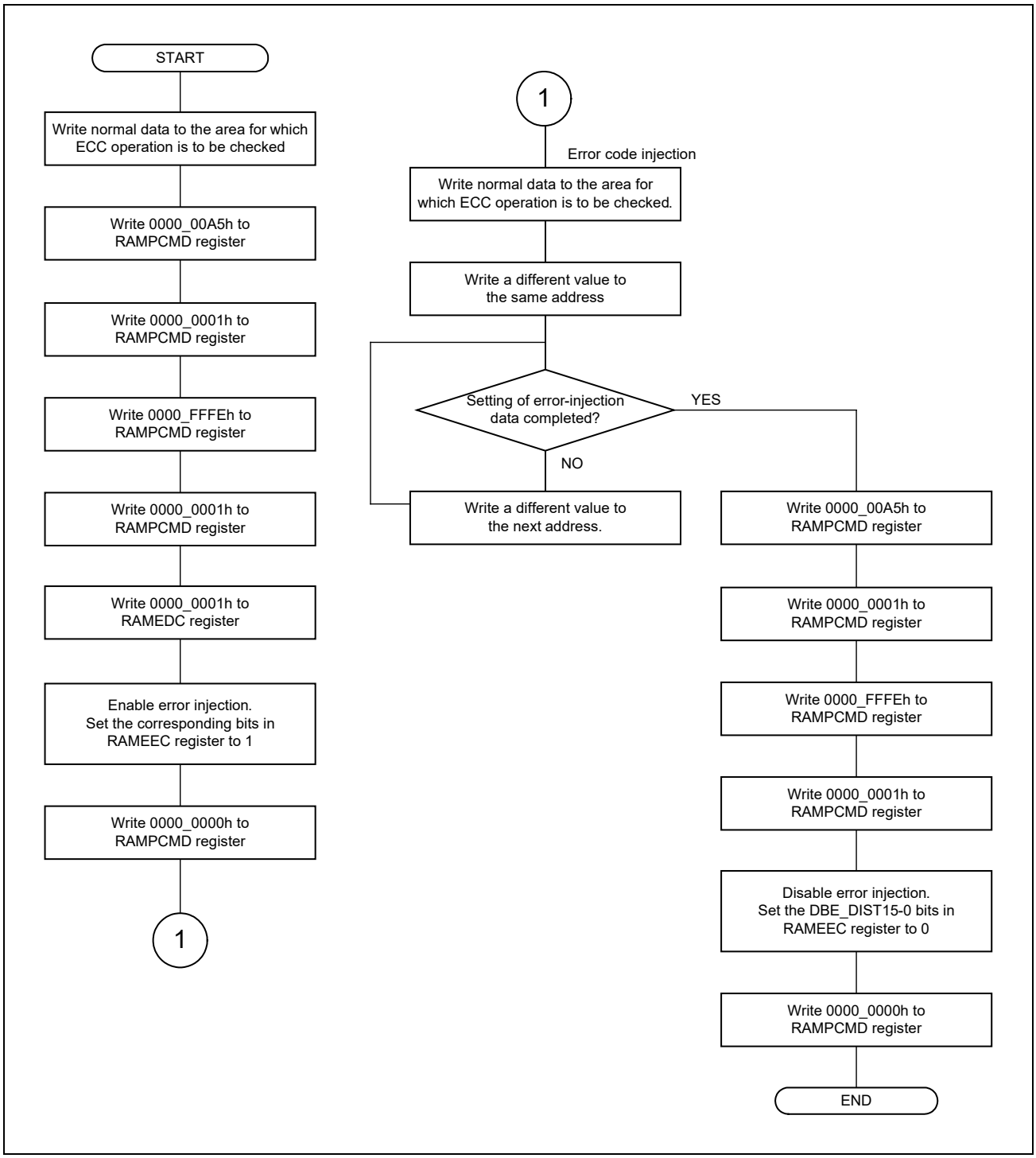


Figure 3.2 Example of ECC Error-Injection Setting Procedure

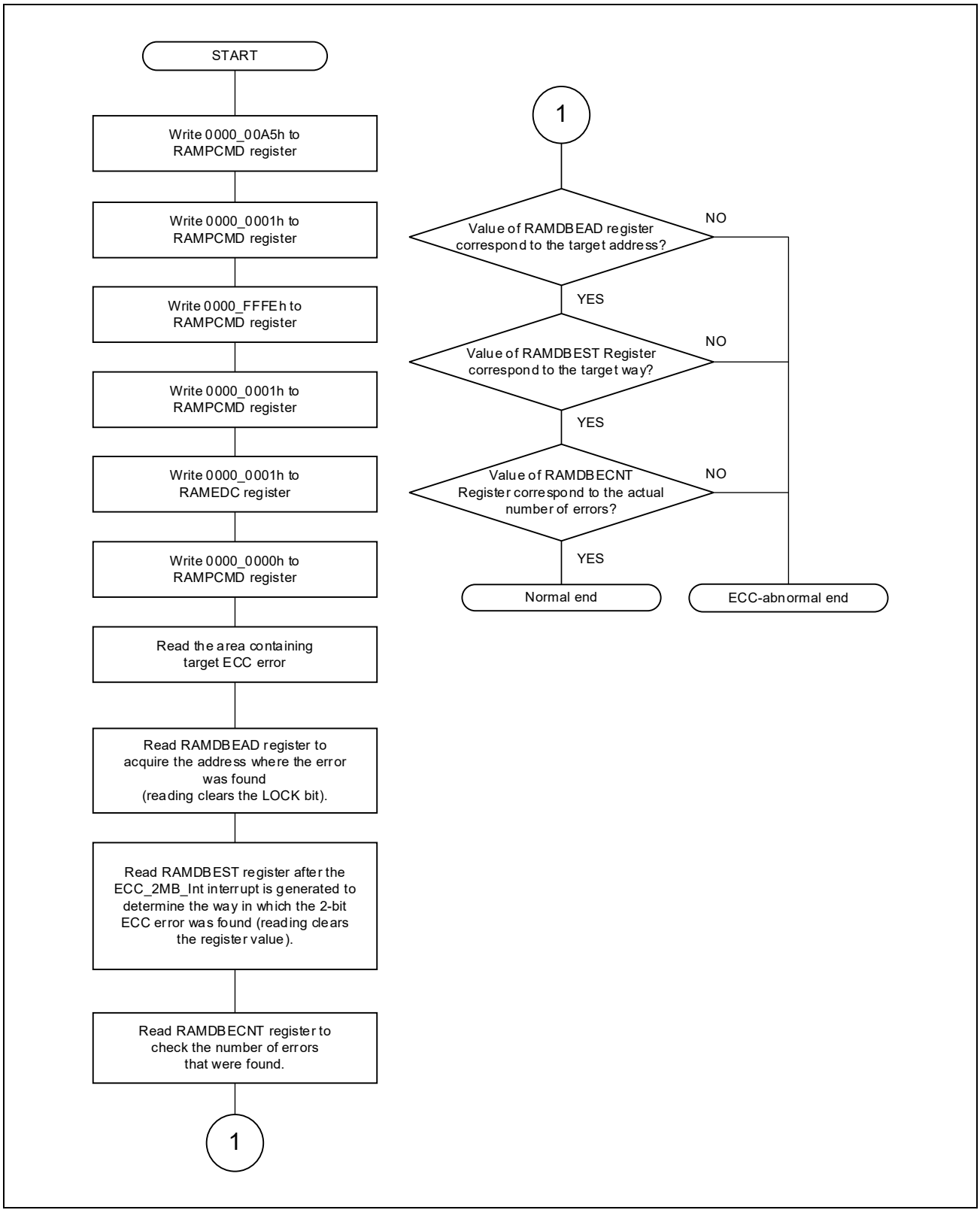


Figure 3.3 Procedure for Checking ECC Operation

3.6 Usage Notes

3.6.1 Protect Command Register (RAMPCMD)

RAMPCMD register is used to provide write protection for registers that might seriously affect the system in order to prevent the application system from inadvertently stopping due to, for example, runaway of a program. Writing to the protected registers is disabled unless the PROTREL bit is set to 1.

To set the PROTREL bit of RAMPCMD register to 1, the sequence shown below must be used. No special sequence is required to clear this bit to 0 or to read the register.

(1) Write 0000_00A5h to RAMPCMD register as a specific value

(2) Write 0000_0001h to RAMPCMD register

(3) Write 0000_FFFEh to RAMPCMD register

(4) Write 0000_0001h to RAMPCMD register

CAUTIONS

1. In steps 1, 2, and 3, nothing is written to the register.
2. Be sure to clear the PROTREL bit to 0 after writing to target registers is completed.

RAMPCMD bit clear procedure:

(1) Write 0b to PROTREL of RAMPCMD register. (Write protect enabled.)

3.6.2 ECC Decoder Config Register (RAMEDC)

RAMEDC register controls the ECC decoders for instruction RAM and data RAM.

When RAMEDC.ECC_ENABLE bit is set to 1, ECC decoder logic for instruction RAM and data RAM is enabled, and the following functions are enabled, and this is conveyed to interrupt controller (Cortex-A7, Cortex-M3).

- For single ECC error: Correct the read data.
- For double ECC error: Generate a double bit ECC error interrupt from the instruction RAM and data RAM, and convey the interrupt signal to interrupt controller (Cortex-A7, Cortex-M3).

When RAMEDC.ECC_ENABLE bit is set to 0 (the “disabled” setting), even if an ECC error occurs, read data is not corrected and the error signal is not conveyed to interrupt controller. Therefore, retention of the status in RAMDBEST register, capturing an error address in RAMDBEAD register, and error count in RAMDBECNT register are disabled.

CAUTIONS

1. Switch this register while no masters are accessing the RAM.
2. Writing to this register is disabled unless the write-protection is canceled by RAMPCMD register.

3.6.3 ECC Encoder Config Register (RAMEEC)

RAMEEC register controls the self-test for the ECC circuit of the instruction RAM and data RAM. If RAMEEC.DBE_DIST[n] (n = 0..15) bit is set to 1, the Syndrome value (ECC redundancy bit data) is latched when the RAM corresponding to each bit is accessed. Then, when the RAM is accessed the next time, the latched Syndrome value is written to the RAM to inject an ECC error. If RAMEEC.DBE_DIST[n] (n = 0..15) bit is set to 0, the normal Syndrome value is always written to the RAM corresponding to each bit.

CAUTIONS

1. Switch this register while no masters are accessing the RAM.
2. Writing to this register is disabled unless the write-protection is canceled by RAMPCMD register.

3.6.4 Double Bit ECC Error Status Register (RAMDBEST)

RAMDBEST register indicates the double bit ECC error status for the instruction RAM and data RAM. After a double bit ECC error interrupt is generated from instruction RAM and data RAM, read this register to identify the BANK and WAY in which a double bit ECC error occurred.

CAUTION

This register is read-clear.

3.6.5 Double Bit ECC Error Address Register (RAMDBEAD)

RAMDBEAD register is a read-only register that holds the address where a double bit ECC error was found. When a double bit ECC error is detected, the ECC error address is captured with the detection signal as a trigger, and then is stored in the ADDRESS[16:0] bits. The register in which an ECC error address has been captured cannot retain the next ECC error address unless the register is read and the LOCK bit is cleared. Therefore, if you want to capture a new ECC error address, you must first read this register.

CAUTION

If double bit ECC errors occur at the same time in different WAYs, the priority of captured addresses is as follows:

Instruction RAM BANK0 WAY0 > Instruction RAM BANK0 WAY1 >
Instruction RAM BANK0 WAY2 > Instruction RAM BANK0 WAY3 >
Instruction RAM BANK1 WAY0 > Instruction RAM BANK1 WAY1 >
Instruction RAM BANK1 WAY2 > Instruction RAM BANK1 WAY3 >
Data RAM BANK0 WAY0 > Data RAM BANK0 WAY1 >
Data RAM BANK0 WAY2 > Data RAM BANK0 WAY3 >
Data RAM BANK1 WAY0 > Data RAM BANK1 WAY1 >
Data RAM BANK1 WAY2 > Data RAM BANK1 WAY3

3.6.6 Double Bit ECC Error Counter Register (RAMDBECNT)

RAMDBECNT register is a read-only register that retains the double bit ECC error count. If a double bit ECC error is detected, the error counter is incremented with the detection signal as a trigger. If the counter value exceeds the maximum (Fh), it is cleared to 0h.

Section 4 4MB SRAM

RZ/N1S and RZ/N1L have on-chip 4MB RAM with ECC function. There are 4 NoC access ports for each 1MB area.

4.1 Overview

Table 4.1 4MB SRAM Summary

Item	Description
RAM capacity	4 Mbytes
RAM address	Area0: 8000 0000h to 800F FFFFh Area1: 8010 0000h to 801F FFFFh Area2: 8020 0000h to 802F FFFFh Area3: 8030 0000h to 803F FFFFh
Error checking	Correction of single error and detection of double error Self-Testing of the ECC Circuit Interrupt management

4.2 Signal Interfaces

Signal Name	Input Output	Description
Clock		
SRAM4MB_HCLK	Input	Internal bus clock (AHB), no clock gating
Interrupt		
ECC_4MB_Int	Output	Pulse sensitive interrupt, Active High

4.3 Register Map

4MB SRAM registers can be accessed by 32-bit units.

NOTE

SR4EDC and SR4EEC registers are write protected by SR4PCMD.

Table 4.2 4MB SRAM Control Register List

Address	Register Symbol	Register Name
8040 0000h	SR4PCMD	SRAM 4MB Protect Command Register
8040 0100h	SR4EDC	SRAM 4MB ECC Decoder Config Register
8040 0104h	SR4EEC	SRAM 4MB ECC Encoder Config Register
8040 0108h	SR4DBEST	SRAM 4MB Double Bit ECC Error Status Register
8040 010Ch	SR4DBEAD	SRAM 4MB Double Bit ECC Error Address Register
8040 0110h	SR4DBECNT	SRAM 4MB Double Bit ECC Error Counter Register

4.4 Register Description

4.4.1 SR4PCMD — SRAM 4MB Protect Command Register

Please refer to **Section 4.6, Usage Notes**, for more detail.

Address: 8040 0000h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	PROTREL
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4.3 SR4PCMD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b1	Reserved		R
b0	PROTREL	Write access to protection registers are permitted. 1: Write access to protection registers are permitted. 0: Write access to protection registers are not permitted.	R/W

4.4.2 SR4EDC — SRAM 4MB ECC Decoder Config Register

SR4EDC register is write protected by SR4PCMD.

Please refer to **Section 4.6, Usage Notes**, for more detail.

Address: 8040 0100h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	ECC_ENABLE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4.4 SR4EDC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b1	Reserved		R
b0	ECC_ENABLE	ECC decoder enable. 0: Disable 1: Enable	R/W

4.4.3 SR4EEC — SRAM 4MB ECC Encoder Config Register

SR4EEC register is write protected by SR4PCMD.

Please refer to **Section 4.6, Usage Notes**, for more detail.

Address: 8040 0104h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	DBE_DIST3	DBE_DIST2	DBE_DIST1	DBE_DIST0
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4.5 SR4EEC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	Reserved		R
b3	DBE_DIST3	ECC error injection enable of 4MB SRAM Area3 (8030 0000h to 803F FFFFh). 0: Disable 1: Enable	R/W
b2	DBE_DIST2	ECC error injection enable of 4MB SRAM Area2 (8020 0000h to 802F FFFFh). Same bit function as DBE_DIST3	R/W
b1	DBE_DIST1	ECC error injection enable of 4MB SRAM Area1 (8010 0000h to 801F FFFFh). Same bit function as DBE_DIST3	R/W
b0	DBE_DIST0	ECC error injection enable of 4MB SRAM Area0 (8000 0000h to 800F FFFFh). Same bit function as DBE_DIST3	R/W

4.4.4 SR4DBEST — SRAM 4MB Double Bit ECC Error Status Register

Please refer to **Section 4.6, Usage Notes**, for more detail.

Address: 8040 0108h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	DBE_RAM3	DBE_RAM2	DBE_RAM1	DBE_RAM0
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4.6 SR4DBEST Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	Reserved		R
b3	DBE_RAM3	ECC double bit error detection flag of 4MB SRAM Area3 (8030 0000h to 803F FFFFh). 0: Error not occurred 1: Error occurred	R
b2	DBE_RAM2	ECC double bit error detection flag of 4MB SRAM Area2 (8020 0000h to 802F FFFFh). Same bit function as DBE_RAM3	R
b1	DBE_RAM1	ECC double bit error detection flag of 4MB SRAM Area1 (8010 0000h to 801F FFFFh). Same bit function as DBE_RAM3	R
b0	DBE_RAM0	ECC double bit error detection flag of 4MB SRAM Area0 (8000 0000h to 800F FFFFh). Same bit function as DBE_RAM3	R

4.4.5 SR4DBEAD — SRAM 4MB Double Bit ECC Error Address Register

Please refer to **Section 4.6, Usage Notes**, for more detail.

Address: 8040 010Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	ADDRESS					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ADDRESS												—	—	—	LOCK
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4.7 SR4DBEAD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b22	Reserved		R
b21 to b4	ADDRESS	ECC double bit Error Address The double bit ECC error address is retained.	R
b3 to b1	Reserved		R
b0	LOCK	Lock Enable 0: Register unlocked (Double bit ECC error address can be captured.) 1: Register locked (Double bit ECC error address cannot be captured.) Read this register to unlock the registers.	R

4.4.6 SR4DBECNT — SRAM 4MB Double Bit ECC Error Counter Register

Please refer to **Section 4.6, Usage Notes**, for more detail.

Address: 8040 0110h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	ERRCOUNT			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4.8 SR4DBECNT Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	Reserved		R
b3 to b0	ERRCOUNT	Double bit ECC error counter.	R

4.5 Operation

4.5.1 Configuration of Memory Map

The 4MB SRAM have 4 separated 1-Mbyte RAMs.

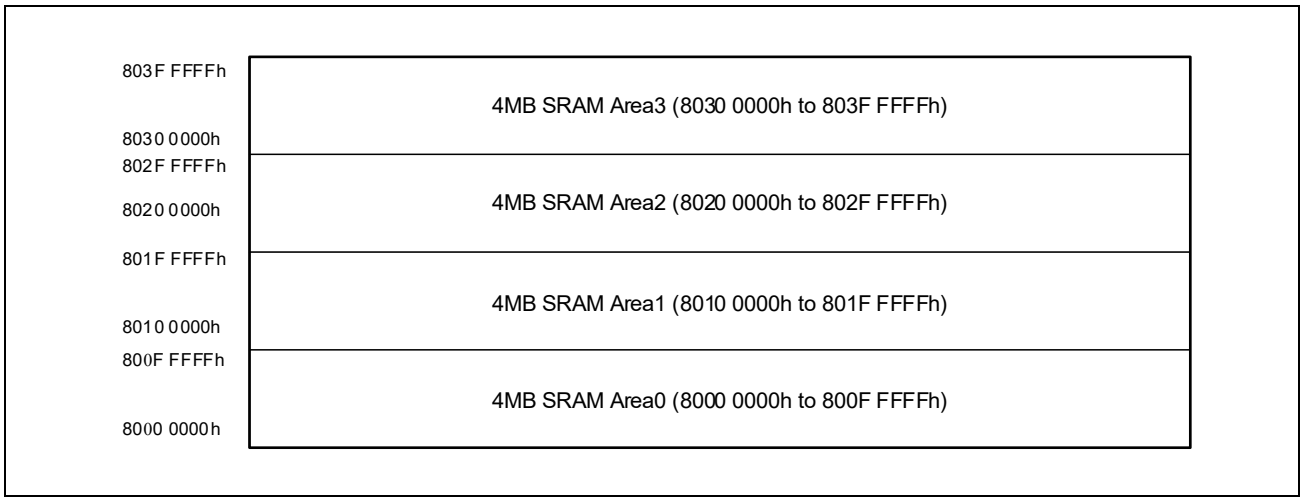


Figure 4.1 Configuration of Memory

4.5.2 Initializing

For normal operation of the ECC circuit, it is necessary to initialize the 4MB SRAM.

RAM initialize:

ECC error correction cannot be enabled before initializing the 4MB SRAM. After the initializing sequence below, set SR4EDC.ECC_ENABLE register to 1 to enable ECC error correction.

Initialize from 0x80000000 to 0x803FFFFF.

1. Disable cache in case of Cortex-A7, Disable NMI in case of Cortex-M3
2. Mask the ECC interrupt
3. Read the data from the initialize area.
4. Write the Read data to the same address.

4.5.3 ECC Error Correction Function

SR4EDC register enables or disables ECC error correction in the 4-Mbyte space. ECCs can be used to correct single error and detect double error. Cortex-A7 and Cortex-M3 detects the sources of double errors, and SR4DBEST register is used to check the area in which an error was found. Moreover, SR4DBEAD register can be used to identify the address where a double error was found, and SR4DBECNT indicates the number of double errors that have been encountered.

4.5.4 Self-Testing of the ECC Circuit

Self-testing of the ECC circuit proceeds in area units. Use only 32-bit accesses. SR4EEC register sets the target area. The following shows an example of the procedure for self-testing of the ECC circuit.

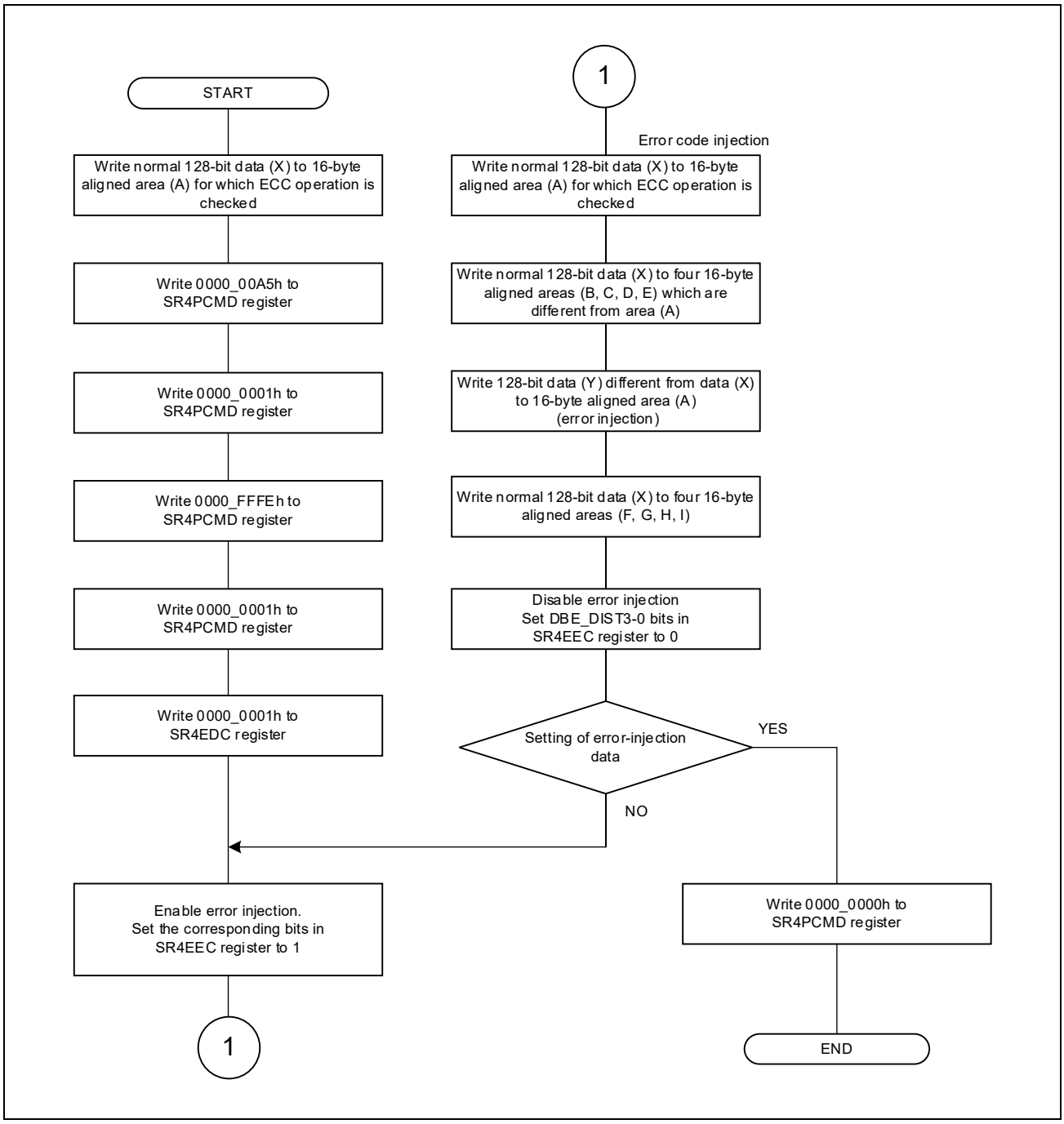


Figure 4.2 Example of ECC Error-Injection Setting Procedure

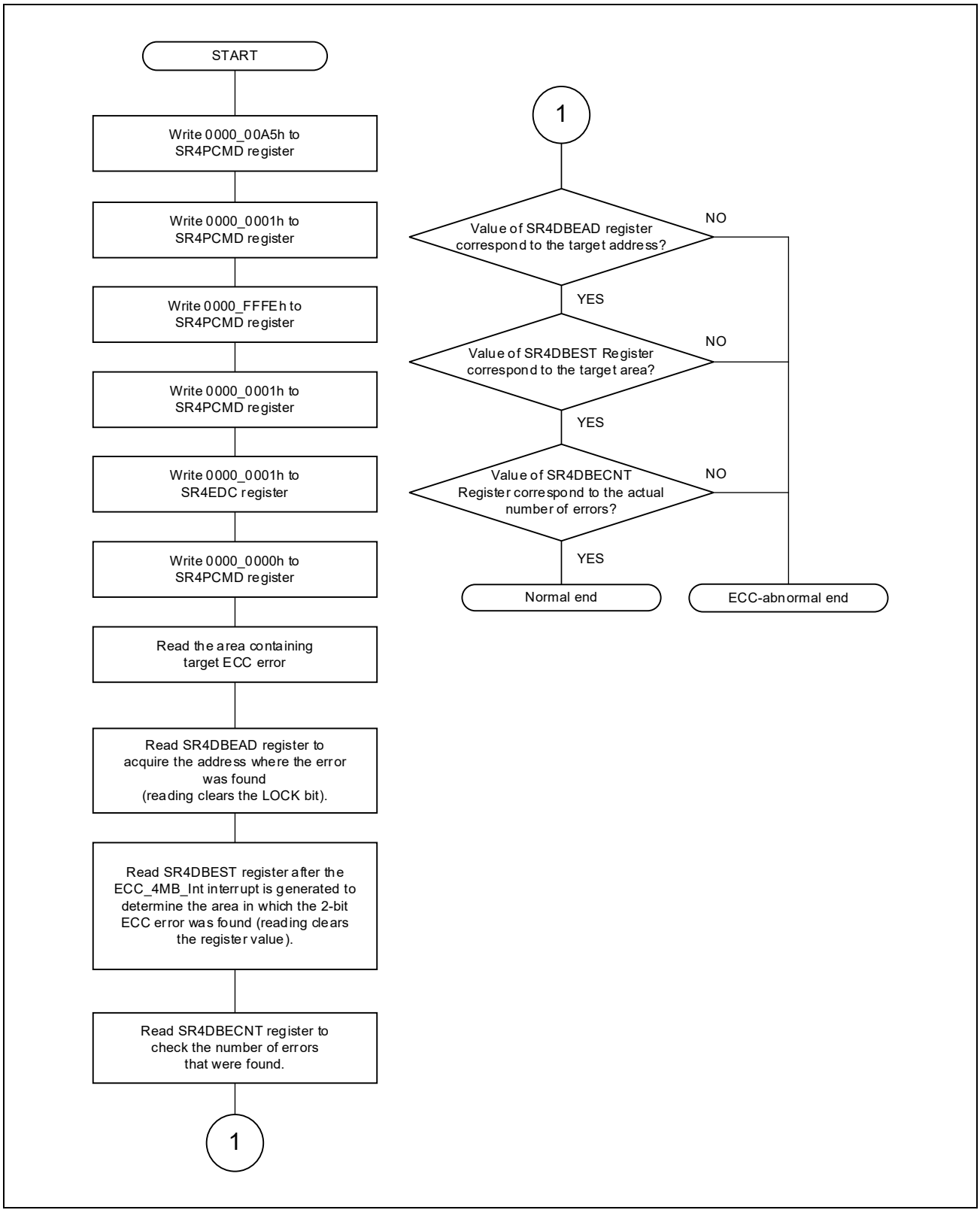


Figure 4.3 Procedure for Checking ECC Operation

4.6 Usage Notes

4.6.1 SRAM 4MB Protect Command Register (SR4PCMD)

SR4PCMD register is used to provide write protection for registers that might seriously affect the system in order to prevent the application system from inadvertently stopping due to, for example, runaway of a program. Writing to the protected registers is disabled unless PROTREL bit is set to 1.

To set PROTREL bit of SR4PCMD register to 1, the sequence shown below must be used. No special sequence is required to clear this bit to 0 or to read the register.

(1) Write 0000_00A5h to SR4PCMD register as a specific value

(2) Write 0000_0001h to SR4PCMD register

(3) Write 0000_FFFEh to SR4PCMD register

(4) Write 0000_0001h to SR4PCMD register

CAUTIONS

1. In steps 1, 2, and 3, nothing is written to the register.
2. Be sure to clear the PROTREL bit to 0 after writing to target registers is completed

SR4PCMD bit clear procedure:

(1) Write 0b to PROTREL of SR4PCMD register. (Write protect enabled.)

4.6.2 SRAM 4MB ECC Decoder Config Register (SR4EDC)

SR4EDC register controls the ECC decoders for 4MB SRAM.

When SR4EDC.ECC_ENABLE bit is set to 1, ECC decoder logic for 4MB SRAM is enabled, and the following functions are enabled, and this is conveyed to interrupt controller (Cortex-A7, Cortex-M3).

- For single ECC error: Correct the read data.
- For double ECC error: Generate a double bit ECC error interrupt from the 4MB SRAM and convey the interrupt signal to interrupt controller (Cortex-A7, Cortex-M3).

When SR4EDC.ECC_ENABLE bit is set to 0 (the “disabled” setting), even if an ECC error occurs, read data is not corrected and the error signal is not conveyed to interrupt controller. Therefore, retention of the status in SR4DBEST register, capturing an error address in SR4DBEAD register, and error count in SR4DBECNT register are disabled.

CAUTIONS

1. Switch this register while no masters are accessing the RAM.
2. Writing to this register is disabled unless the write-protection is canceled by SR4PCMD register.

4.6.3 SRAM 4MB ECC Encoder Config Register (SR4EEC)

SR4EEC register controls the self-test for the ECC circuit of the 4MB SRAM. If SR4EEC.DBE_DIST[n] (n = 0.. 3) bit is set to 1, the Syndrome value (ECC redundancy bit data) is latched when the RAM corresponding to each bit is accessed. Then, when the RAM is accessed the next time, the latched Syndrome value is written to the RAM to inject an ECC error. If SR4EEC.DBE_DIST[n] (n = 0..3) bit is set to 0, the normal Syndrome value is always written to the RAM corresponding to each bit.

CAUTIONS

1. Switch this register while no masters are accessing the RAM.
2. Writing to this register is disabled unless the write-protection is canceled by SR4PCMD register.

4.6.4 SRAM 4MB Double Bit ECC Error Status Register (SR4DBEST)

SR4DBEST register indicates the double bit ECC error status for the 4MB SRAM. After a double bit ECC error interrupt is generated from 4MB SRAM, read this register to identify the area in which a double bit ECC error occurred.

CAUTION

This register is read-clear.

4.6.5 SRAM 4MB Double Bit ECC Error Address Register (SR4DBEAD)

SR4DBEAD register is a read-only register that holds the address where a double bit ECC error was found. When a double bit ECC error is detected, the ECC error address is captured with the detection signal as a trigger, and then is stored in the ADDRESS[17:0] bits. The register in which an ECC error address has been captured cannot retain the next ECC error address unless the register is read and the LOCK bit is cleared. Therefore, if you want to capture a new ECC error address, you must first read this register.

CAUTION

If double bit ECC errors occur at the same time in different areas, the priority of captured addresses is as follows:

4MB SRAM Area0 (8000 0000h to 800F FFFFh) >
4MB SRAM Area1 (8010 0000h to 801F FFFFh) >
4MB SRAM Area2 (8020 0000h to 802F FFFFh) >
4MB SRAM Area3 (8030 0000h to 803F FFFFh)

4.6.6 SRAM 4MB Double Bit ECC Error Counter Register (SR4DBECNT)

SR4DBECNT register is a read-only register that retains the double bit ECC error count. If a double bit ECC error is detected, the error counter is incremented with the detection signal as a trigger. If the counter value exceeds the maximum (Fh), it is cleared to 0h.

Section 5 Debugging Interface

5.1 Overview

The On-chip Debug and Trace Unit shall include CoreSight™-compliant logic to provide the whole infrastructure for multi-core debug and trace to debug, monitor and optimize the performance of the Arm Cortex-A7 Core and Arm Cortex-M3 processors.

- 32 KB Embedded Trace Buffer (ETB)
  - On-chip storage of trace data
  - The ETB accepts trace data from either ETM components connected to the Arm Cortex-A7 Core or Arm Cortex-M3 Core or ITM
- Embedded Trace Router (ETR)
  - For large storage capability of the trace data across to the internal RAM (2 MB or 4 MB) and DDR2/3
- Embedded Trace Macrocell (ETM)
  - Real-time trace module providing instruction and data tracing of a processor
- SWJ-DP debug port
  - A combined JTAG-DP and SW-DP that enables to connect either a SWD or JTAG probe to a target

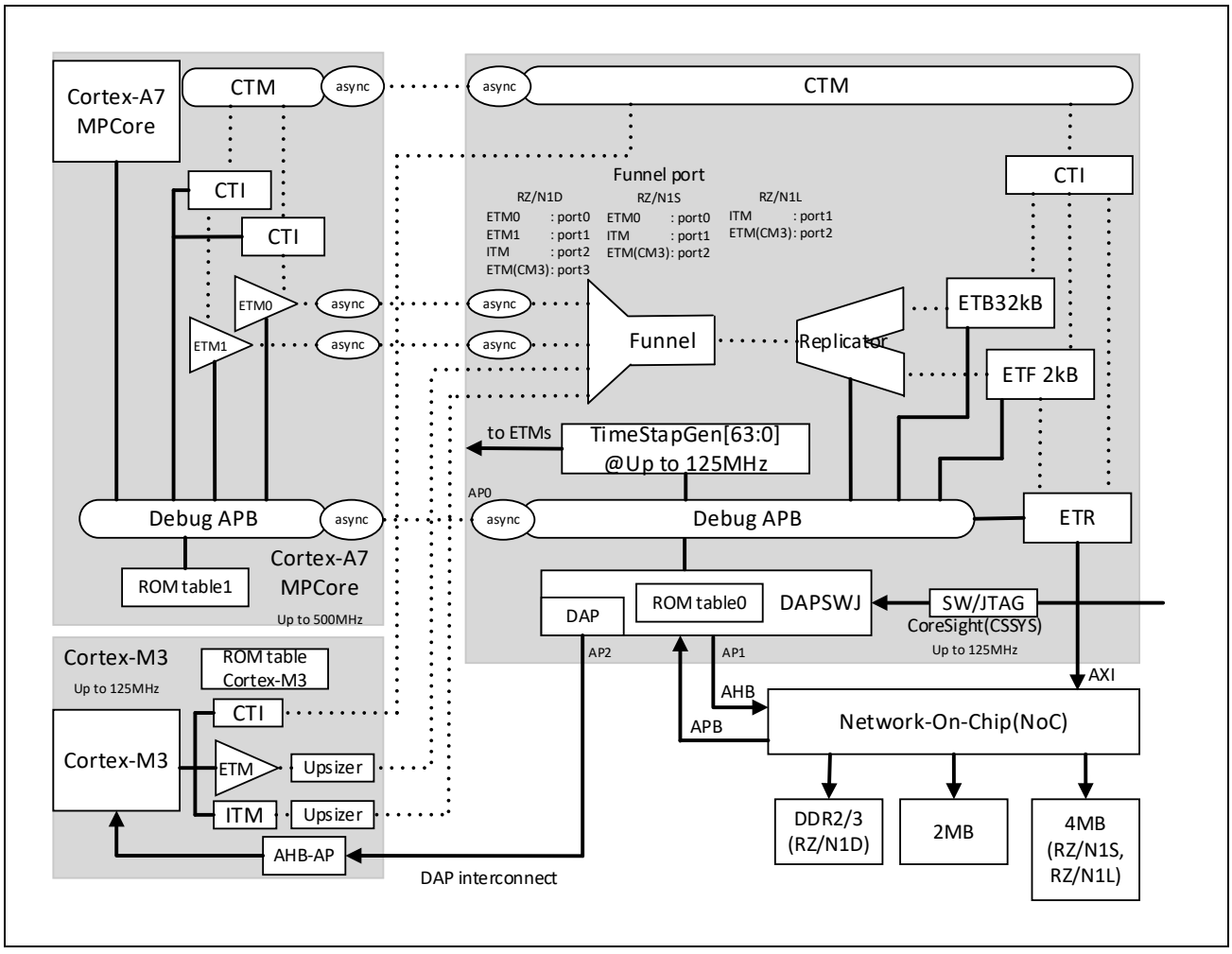


Figure 5.1 High-level Block Diagram of the On-chip Debug & Trace Unit

5.2 JTAG Interface

5.2.1 Circuit Recommendation of JTAG Interface

If the JTAG interface is unused, the JTAG_TRST_N input should be connected to digital GND via a 4.7kΩ resistor. The circuit recommendation for the interface looks as follows.

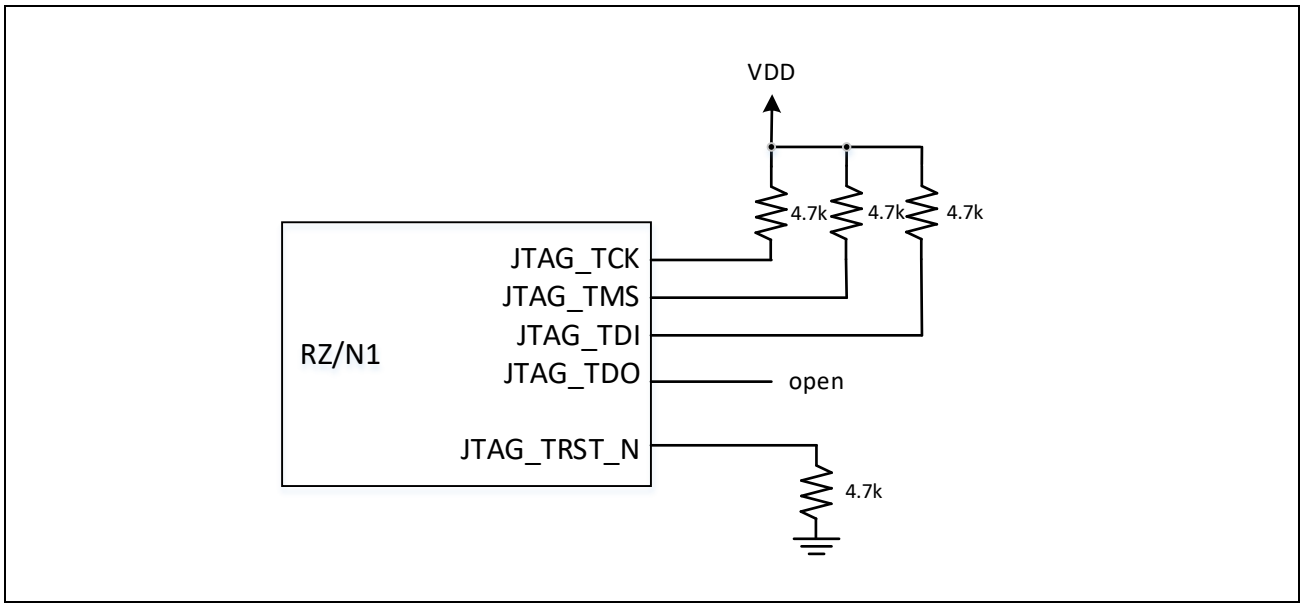


Figure 5.2 Unused JTAG Interface

If it uses the JTAG interface for boundary scan test, it is necessary to check whether the boundary scan tool has specific requirements with respect to the JTAG_TRST_N circuit in the target system. If the boundary scan tool has specific requirements, the circuit in the target system must be made configurable. The below figure shows the situation, that the boundary scan tool requires a pull-up at the JTAG_TRST_N input of the chip.

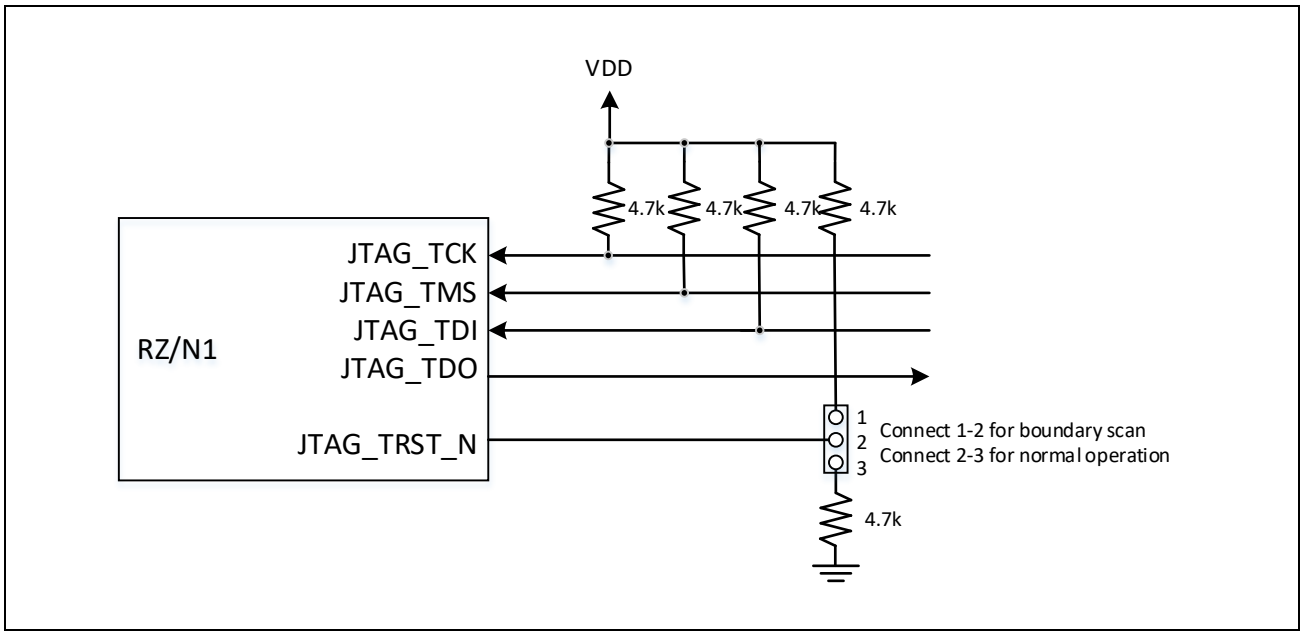


Figure 5.3 JTAG Interface Usable for Boundary Scan

5.2.2 Circuit Recommendation of JTAG-Debug Interface

The following figure shows example of JTAG-Debug connection scheme (according to standard JTAG IEEE 1149.1 mode).

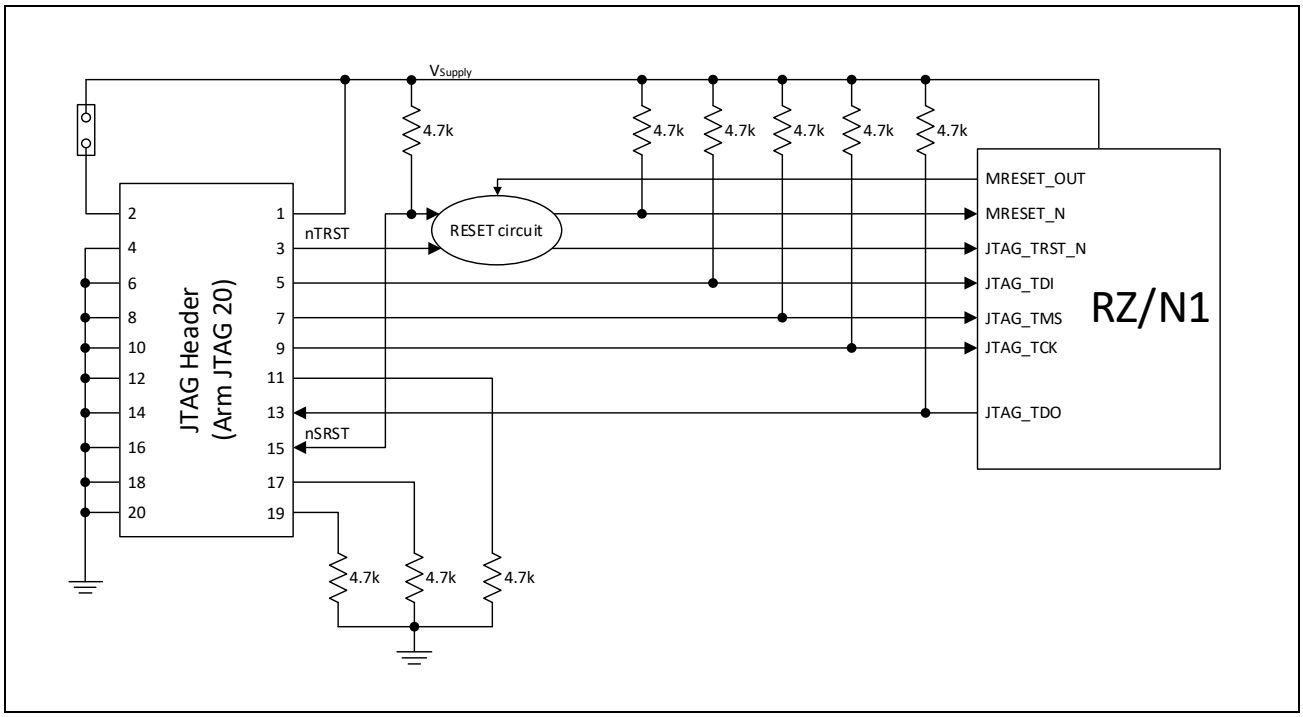


Figure 5.4 Example of PCB / JTAG Connection

The following items should be considered:

- Signals TDI, TMS, TCK, RTCK and TDO are typically pulled up on the target board to keep them stable when the debug equipment is not connected.
- DBGRQ and DBGACK are typically pulled down on the target board as RZ/N1 doesn't support these.

There is no RTCK signal provided on the RZ/N1, therefore it can either be pulled to a fixed logic level or connected to the TCK signal to provide a direct loop-back.

5.2.3 Circuit Recommendation of Serial Wire Debug Interface

The following figure shows example of Serial Wire Debug (SWD) connection scheme.

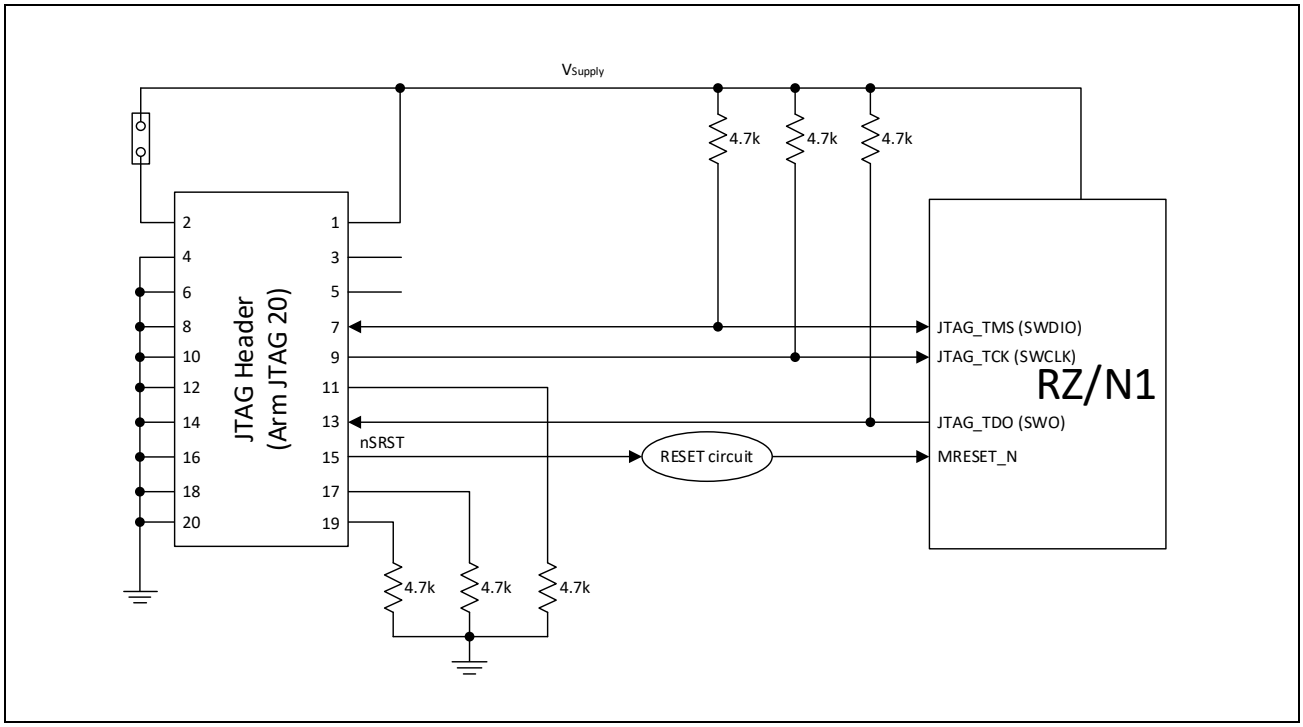


Figure 5.5 Example of SWD Connection

The following items should be considered:

- SWDIO, SWCLK and SWO signals are typically pulled up on the target board to keep them stable when the debug equipment is not connected.
- Other JTAG signals are not used in Serial Wire Debug (SWD) mode.

5.3 Reset Considerations

5.3.1 RZ/N1 Reset Signals

The RZ/N1 has a reset called MRESET_N (resets the entire CPU core, debug logic and peripherals). The RZ/N1 device includes the JTAG interface and has a second reset input called JTAG_TRST_N (TAP reset). This resets the Debug Access Port (DAP) (JTAG Debug Port (JTAG-DP) / Serial Wire Debug Port (SW-DP)) and the boundary scan cells.

It is strongly recommended that both signals are separately available on the JTAG connector. If the MRESET_N and JTAG_TRST_N signals are linked together, resetting the system also resets the TAP controller. This means that:

- It is not possible to debug a system from reset, because any breakpoints previously set are lost
- You might have to start the debug session from the beginning, because the debugger (e.g. DSTREAM) might not recover when the TAP controller state is changed.

5.3.2 Debugger Reset Signals

Most of the debug units have two reset signals connected to the RZ/N1:

- nTRST drives the JTAG_TRST_N signal on the Arm processor of RZ/N1. It is an output that is activated whenever the debug software has to re-initialize the debug interface in the target system.
- nSRST is a bidirectional signal that both drives and senses the system reset signals of RZ/N1 (MRESET_N, MRESET_OUT). By default, this output is driven LOW by the debugger to re-initialize the RZ/N1 system.

JTAG_TRST_N should be pulled-up to assure normal operation when the JTAG interface is disconnected. For further information, please refer to the corresponding debug tool vendor.

5.3.3 Example Reset Circuit

The circuit in the following figure shows example of reset circuit logic for the RZ/N1 reset signals and the debug unit reset signals.

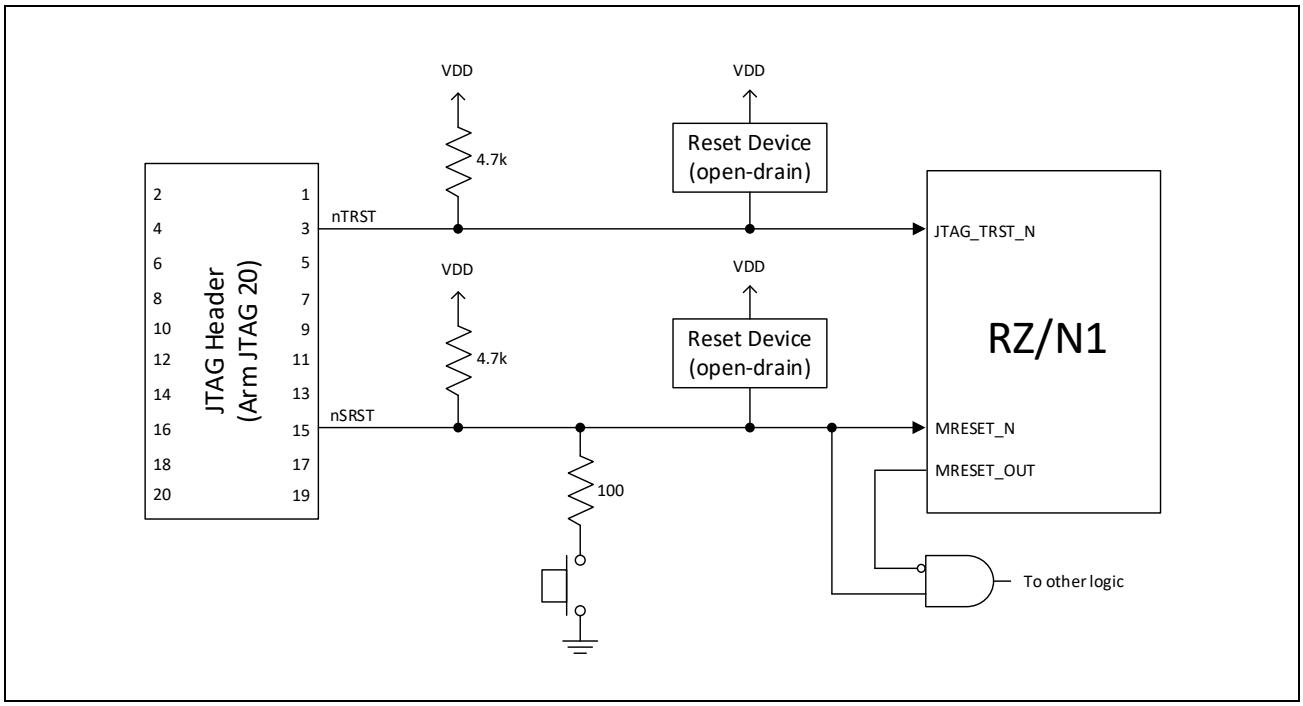


Figure 5.6 Example of Reset Circuit

It should be noted that the MRESET_OUT output signal reflects the MRESET_N input signal. Therefore, the MRESET_OUT must not be connected to nSRST (open drain system reset) together with MRESET_N input to avoid a “deadlock” connection.

The external debugger can cause a system hang-up by issuing a system reset when nSRST is connected to MRESET_N, due to the fact that the MRESET_N resets entire debug logic of RZ/N1. An internal system reset (e.g. issued by watchdog or software bit) will be seen by the debugger, if the MRESET_OUT is connected to nSRST, as an internal system reset will not reset the debug logic.

Section 6 16b DDR2/3 Controller

6.1 Overview

RZ/N1D integrates DDR2/3 Controller, and it is able to support DDR2 and DDR3 double data rate memory devices with fixed operating frequencies. The multi ports architecture ensures that memory is shared efficiently among different high-bandwidth client modules.

- DDR2 & 3 Memory controllers with fixed frequency, asynchronous to NoC

  DDR3-1000
  (1) DDR controller: 250 MHz
  (2) DDR PHY: 500 MHz

  DDR2-500
  (1) DDR controller: 125 MHz
  (2) DDR PHY: 250 MHz
- DQ/DQS Ratio 8/1
- 2 Chip Select
- 2 ODT
- 16-bit Address Bus A0..15
- Maximum 2 GB address space
- Programmable memory data path size of full memory 16-bit data width or half memory 8-bit data width
  - 16 bits, 8 bits, 8+ECC bits
- ECC function software configurable (enable/disable)
  - 2 bits ECC by byte access
  - ECC scheme: 7 check bits over 32 data bits (this will result in 2 bits adder for 8 data bits)
  - Single-bit Error correction (SEC) and double-bit Error Detection (DED) error reporting and automatic correction of single-bit error events.
  - Programmable reporting and correction
  - Programmable removal of ECC storage.
- Flexible priority scheme software configurable
- Configurable DQ/DQS output impedance/slope and on die termination.
- DDR2/DDR3 Low power control management (by software)
  - Active Power-Down
  - Pre-Charge Power-Down
  - Self-Refresh
  - Self-Refresh with Memory Clock Gating
  - Self-Refresh with Memory and Controller Clock Gating
  - Optimized command scheduling with Activate/Pre-charging support
  - Support for Refresh Per Chip Select, to avoid peak current when refreshing.

- Port Address Protection Check
  - Incoming addresses and instruction type for each port are verified according to register setting.
  - Up to 16 address protection regions per port

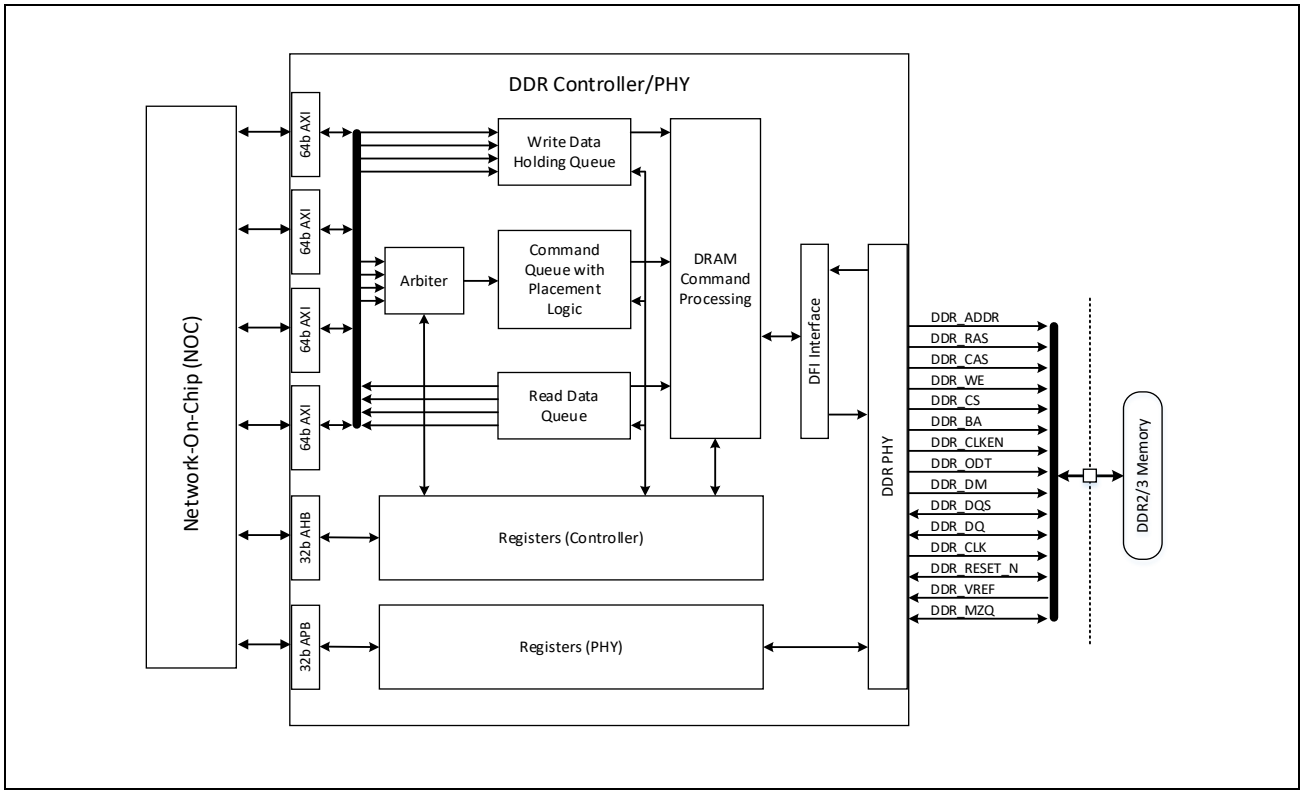


Figure 6.1 DDR Controller Interfaces and Connections

6.2 Signal Interfaces

Signal Name	Input Output	Description
Clock		
DDR_XCLK	Input	Internal bus clock (AXI)
DDR_HCLK	Input	Internal bus clock (AHB, APB), no clock gating
DDR_DFICLK	Input	Clock for DDR PHY interface
Interrupt		
DDRC_Int	Output	Level sensitive interrupt, Active High
External signal (dedicated pin only)		
DDR_CLKP DDR_CLKN	Output	Differential clock
DDR_CLKEN	Output	Clock enable, high active
DDR_RESET_N	Output	DDR Reset, low active
DDR_ADDR[15:0]	Output	Address bus
DDR_BA[2:0]	Output	Bank address
DDR_DQ[15:0]	I/O	Data bus
DDR_DM[1:0]	Output	Data mask
DDR_DQS1 DDR_DQS_N1	I/O	Differential bidirectional data strobe for upper byte lane
DDR_DQS0 DDR_DQS_N0	I/O	Differential bidirectional data strobe for lower byte lane
DDR_WE	Output	Write enable, low active
DDR_RAS	Output	Row Address Strobe, low active
DDR_CAS	Output	Column Address Strobe, low active
DDR_CS[1:0]	Output	Chip Select, low active
DDR_ODT[1:0]	Output	ODT control

6.3 Register Map

6.3.1 DDR Controller

Table 6.1 DDR Controller Register Map (1/4)

Address	Register Symbol	Register Name
4000 D000h	DDR_CTL_00	DDR-Controller Status & Control 00
4000 D004h	DDR_CTL_01	DDR-Controller Status & Control 01
4000 D008h	DDR_CTL_02	DDR-Controller Status & Control 02
4000 D00Ch	DDR_CTL_03	DDR-Controller Status & Control 03
4000 D010h	DDR_CTL_04	DDR-Controller Status & Control 04
4000 D014h	DDR_CTL_05	DDR-Controller Status & Control 05
4000 D018h	DDR_CTL_06	DDR-Controller Status & Control 06
4000 D01Ch	DDR_CTL_07	DDR-Controller Status & Control 07
4000 D020h	DDR_CTL_08	DDR-Controller Status & Control 08
4000 D024h	DDR_CTL_09	DDR-Controller Status & Control 09
4000 D028h	DDR_CTL_10	DDR-Controller Status & Control 10
4000 D02Ch	DDR_CTL_11	DDR-Controller Status & Control 11
4000 D030h	DDR_CTL_12	DDR-Controller Status & Control 12
4000 D034h	DDR_CTL_13	DDR-Controller Status & Control 13
4000 D038h	DDR_CTL_14	DDR-Controller Status & Control 14
4000 D03Ch	DDR_CTL_15	DDR-Controller Status & Control 15
4000 D040h	DDR_CTL_16	DDR-Controller Status & Control 16
4000 D044h	DDR_CTL_17	DDR-Controller Status & Control 17
4000 D048h	DDR_CTL_18	DDR-Controller Status & Control 18
4000 D04Ch	DDR_CTL_19	DDR-Controller Status & Control 19
4000 D050h	DDR_CTL_20	DDR-Controller Status & Control 20
4000 D054h	DDR_CTL_21	DDR-Controller Status & Control 21
4000 D058h	DDR_CTL_22	DDR-Controller Status & Control 22
4000 D05Ch	DDR_CTL_23	DDR-Controller Status & Control 23
4000 D060h	DDR_CTL_24	DDR-Controller Status & Control 24
4000 D064h	DDR_CTL_25	DDR-Controller Status & Control 25
4000 D068h	DDR_CTL_26	DDR-Controller Status & Control 26
4000 D06Ch	DDR_CTL_27	DDR-Controller Status & Control 27
4000 D070h	DDR_CTL_28	DDR-Controller Status & Control 28
4000 D074h	DDR_CTL_29	DDR-Controller Status & Control 29
4000 D078h	DDR_CTL_30	DDR-Controller Status & Control 30
4000 D07Ch	DDR_CTL_31	DDR-Controller Status & Control 31
4000 D080h	DDR_CTL_32	DDR-Controller Status & Control 32
4000 D084h	DDR_CTL_33	DDR-Controller Status & Control 33
4000 D088h	DDR_CTL_34	DDR-Controller Status & Control 34
4000 D08Ch	DDR_CTL_35	DDR-Controller Status & Control 35
4000 D090h	DDR_CTL_36	DDR-Controller Status & Control 36
4000 D094h	DDR_CTL_37	DDR-Controller Status & Control 37
4000 D098h	DDR_CTL_38	DDR-Controller Status & Control 38
4000 D09Ch	DDR_CTL_39	DDR-Controller Status & Control 39
4000 D0A0h	DDR_CTL_40	DDR-Controller Status & Control 40
4000 D0A4h	DDR_CTL_41	DDR-Controller Status & Control 41

Table 6.1 DDR Controller Register Map (2/4)

Address	Register Symbol	Register Name
4000 D0A8h	DDR_CTL_42	DDR-Controller Status & Control 42
4000 D0ACh	DDR_CTL_43	DDR-Controller Status & Control 43
4000 D0B0h	DDR_CTL_44	DDR-Controller Status & Control 44
4000 D0B4h	DDR_CTL_45	DDR-Controller Status & Control 45
4000 D0B8h	DDR_CTL_46	DDR-Controller Status & Control 46
4000 D0BCh	DDR_CTL_47	DDR-Controller Status & Control 47
4000 D0C0h	DDR_CTL_48	DDR-Controller Status & Control 48
4000 D0C4h	DDR_CTL_49	DDR-Controller Status & Control 49
4000 D0C8h	DDR_CTL_50	DDR-Controller Status & Control 50
4000 D0CCh	DDR_CTL_51	DDR-Controller Status & Control 51
4000 D0D0h	DDR_CTL_52	DDR-Controller Status & Control 52
4000 D0D4h	DDR_CTL_53	DDR-Controller Status & Control 53
4000 D0D8h	DDR_CTL_54	DDR-Controller Status & Control 54
4000 D0DCh	DDR_CTL_55	DDR-Controller Status & Control 55
4000 D0E0h	DDR_CTL_56	DDR-Controller Status & Control 56
4000 D0E4h	DDR_CTL_57	DDR-Controller Status & Control 57
4000 D0E8h	DDR_CTL_58	DDR-Controller Status & Control 58
4000 D0ECh	DDR_CTL_59	DDR-Controller Status & Control 59
4000 D0F0h	DDR_CTL_60	DDR-Controller Status & Control 60
4000 D0F4h	DDR_CTL_61	DDR-Controller Status & Control 61
4000 D0F8h	DDR_CTL_62	DDR-Controller Status & Control 62
4000 D0FCh	DDR_CTL_63	DDR-Controller Status & Control 63
4000 D100h	DDR_CTL_64	DDR-Controller Status & Control 64
4000 D104h	DDR_CTL_65	DDR-Controller Status & Control 65
4000 D108h	DDR_CTL_66	DDR-Controller Status & Control 66
4000 D10Ch	DDR_CTL_67	DDR-Controller Status & Control 67
4000 D110h	DDR_CTL_68	DDR-Controller Status & Control 68
4000 D114h	DDR_CTL_69	DDR-Controller Status & Control 69
4000 D118h	DDR_CTL_70	DDR-Controller Status & Control 70
4000 D11Ch	DDR_CTL_71	DDR-Controller Status & Control 71
4000 D120h	DDR_CTL_72	DDR-Controller Status & Control 72
4000 D124h	DDR_CTL_73	DDR-Controller Status & Control 73
4000 D128h	DDR_CTL_74	DDR-Controller Status & Control 74
4000 D12Ch	DDR_CTL_75	DDR-Controller Status & Control 75
4000 D130h	DDR_CTL_76	DDR-Controller Status & Control 76
4000 D134h	DDR_CTL_77	DDR-Controller Status & Control 77
4000 D138h	DDR_CTL_78	DDR-Controller Status & Control 78
4000 D13Ch	DDR_CTL_79	DDR-Controller Status & Control 79
4000 D140h	DDR_CTL_80	DDR-Controller Status & Control 80
4000 D144h	DDR_CTL_81	DDR-Controller Status & Control 81
4000 D148h	DDR_CTL_82	DDR-Controller Status & Control 82
4000 D14Ch	DDR_CTL_83	DDR-Controller Status & Control 83
4000 D150h	DDR_CTL_84	DDR-Controller Status & Control 84
4000 D154h	DDR_CTL_85	DDR-Controller Status & Control 85
4000 D158h	DDR_CTL_86	DDR-Controller Status & Control 86

Table 6.1 DDR Controller Register Map (3/4)

Address	Register Symbol	Register Name
4000 D15Ch	DDR_CTL_87	DDR-Controller Status & Control 87
4000 D160h	DDR_CTL_88	DDR-Controller Status & Control 88
4000 D164h	DDR_CTL_89	DDR-Controller Status & Control 89
4000 D168h	DDR_CTL_90	DDR-Controller Status & Control 90
4000 D16Ch + 8h × n	DDR_CTL_[k] (n = 0..15) (k = 91 + n × 2)	Port0 Range[n] Start Address Setting Register
4000 D170h + 8h × n	DDR_CTL_[k] (n = 0..15) (k = 92 + n × 2)	Port0 Range[n] End Address Setting Register
4000 D1ECh + 8h × n	DDR_CTL_[k] (n = 0..15) (k = 123 + n × 2)	Port1 Range[n] Start Address Setting Register
4000 D1F0h + 8h × n	DDR_CTL_[k] (n = 0..15) (k = 124 + n × 2)	Port1 Range[n] End Address Setting Register
4000 D26Ch + 8h × n	DDR_CTL_[k] (n = 0..15) (k = 155 + n × 2)	Port2 Range[n] Start Address Setting Register
4000 D270h + 8h × n	DDR_CTL_[k] (n = 0..15) (k = 156 + n × 2)	Port2 Range[n] End Address Setting Register
4000 D2ECh + 8h × n	DDR_CTL_[k] (n = 0..15) (k = 187 + n × 2)	Port3 Range[n] Start Address Setting Register
4000 D2F0h + 8h × n	DDR_CTL_[k] (n = 0..14) (k = 188 + n × 2)	Port3 Range[n] End Address Setting Register
4000 D368h	DDR_CTL_218	Port3 Range15 End Address Setting Register
4000 D36Ch + 8h × n	DDR_CTL_[k] (n = 0..15) (k = 219 + n × 2)	Port0 Range[n] Protect Setting Register1
4000 D370h + 8h × n	DDR_CTL_[k] (n = 0..14) (k = 220 + n × 2)	Port0 Range[n] Protect Setting Register2
4000 D3E8h	DDR_CTL_250	Port0 Range15 Protect Setting Register2
4000 D3ECh + 8h × n	DDR_CTL_[k] (n = 0..15) (k = 251 + n × 2)	Port1 Range[n] Protect Setting Register1
4000 D3F0h + 8h × n	DDR_CTL_[k] (n = 0..14) (k = 252 + n × 2)	Port1 Range[n] Protect Setting Register2
4000 D468h	DDR_CTL_282	Port1 Range15 Protect Setting Register2
4000 D46Ch + 8h × n	DDR_CTL_[k] (n = 0..15) (k = 283 + n × 2)	Port2 Range[n] Protect Setting Register1
4000 D470h + 8h × n	DDR_CTL_[k] (n = 0..14) (k = 284 + n × 2)	Port2 Range[n] Protect Setting Register2
4000 D4E8h	DDR_CTL_314	Port2 Range15 Protect Setting Register2
4000 D4ECh + 8h × n	DDR_CTL_[k] (n = 0..15) (k = 315 + n × 2)	Port3 Range[n] Protect Setting Register1
4000 D4F0h + 8h × n	DDR_CTL_[k] (n = 0..14) (k = 316 + n × 2)	Port3 Range[n] Protect Setting Register2
4000 D568h	DDR_CTL_346	Port3 Range15 Protect Setting Register2
4000 D56Ch	DDR_CTL_347	DDR-Controller Status & Control 347
4000 D570h	DDR_CTL_348	DDR-Controller Status & Control 348
4000 D574h	DDR_CTL_349	DDR-Controller Status & Control 349
4000 D578h	DDR_CTL_350	DDR-Controller Status & Control 350
4000 D57Ch	DDR_CTL_351	DDR-Controller Status & Control 351
4000 D580h	DDR_CTL_352	DDR-Controller Status & Control 352
4000 D584h	DDR_CTL_353	DDR-Controller Status & Control 353
4000 D588h	DDR_CTL_354	DDR-Controller Status & Control 354
4000 D58Ch	DDR_CTL_355	DDR-Controller Status & Control 355

Table 6.1 DDR Controller Register Map (4/4)

Address	Register Symbol	Register Name
4000 D590h	DDR_CTL_356	DDR-Controller Status & Control 356
4000 D594h	DDR_CTL_357	DDR-Controller Status & Control 357
4000 D598h	DDR_CTL_358	DDR-Controller Status & Control 358
4000 D59Ch	DDR_CTL_359	DDR-Controller Status & Control 359
4000 D5A0h	DDR_CTL_360	DDR-Controller Status & Control 360
4000 D5A4h	DDR_CTL_361	DDR-Controller Status & Control 361
4000 D5A8h	DDR_CTL_362	DDR-Controller Status & Control 362
4000 D5ACh	DDR_CTL_363	DDR-Controller Status & Control 363
4000 D5B0h	DDR_CTL_364	DDR-Controller Status & Control 364
4000 D5B4h	DDR_CTL_365	DDR-Controller Status & Control 365
4000 D5B8h	DDR_CTL_366	DDR-Controller Status & Control 366
4000 D5BCh	DDR_CTL_367	DDR-Controller Status & Control 367
4000 D5C0h	DDR_CTL_368	DDR-Controller Status & Control 368
4000 D5C4h	DDR_CTL_369	DDR-Controller Status & Control 369
4000 D5C8h	DDR_CTL_370	DDR-Controller Status & Control 370
4000 D5CCh	DDR_CTL_371	DDR-Controller Status & Control 371
4000 D5D0h	DDR_CTL_372	DDR-Controller Status & Control 372
4000 D5D4h	DDR_CTL_373	DDR-Controller Status & Control 373
4000 D5D8h	DDR_CTL_374	DDR-Controller Status & Control 374

6.3.2 DDR PHY

Table 6.2 DDR PHY Register Map

Address	Register Symbol	Register Name
4000 E000h	FUNCCTRL	Function Control Register
4000 E004h	DLLCTRL	MDLL Control Register
4000 E008h	ZQCALCTRL	ZQ Calibration Control Register
4000 E00Ch	ZQODTCTRL	ZQODT Control Register
4000 E010h	RDCTRL	Read Control Register
4000 E014h	RDTMG	READ Timing Control Register
4000 E018h	FIFOINIT	FIFO Initialization Register
4000 E01Ch	OUTCTRL	Output Control Register
4000 E040h	WLCTRL1	Write Leveling Control Register 1
4000 E0E8h	DQCALOFS1	DQS Offset Setting

6.4 Register Description

6.4.1 DDR Controller Register Description

6.4.1.1 DDR_CTL_00 — DDR-Controller Status & Control 00

Address: 4000 D000h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	VERSION															
Value after reset	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	DRAM_CLASS				—	—	—	—	—	—	—	START
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.3 DDR_CTL_00 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	VERSION	Holds the controller version number.	R
b15 to b12	Reserved	NA	R
b11 to b8	DRAM_CLASS	Defines the mode of operation of the controller. 3'b0100 = DDR2 3'b0110 = DDR3 All other settings are reserved and must not be set.	R/W
b7 to b1	Reserved	NA	R
b0	START	Initiate command processing in the controller. Set to 1 to initiate.	R/W

6.4.1.2 DDR_CTL_01 — DDR-Controller Status & Control 01

Address: 4000 D004h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	READ_DATA_FIFO_DEPTH								—	—	—	—	—	—	MAX_CS_REG	
Value after reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	MAX_COL_REG				—	—	—	MAX_ROW_REG				
Value after reset	0	0	0	0	1	0	1	1	0	0	0	1	0	0	0	0

Table 6.4 DDR_CTL_01 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	READ_DATA_FIFO_DEPTH	Reports the depth of the controller read data queue.	R
b23 to b18	Reserved	NA	R
b17, b16	MAX_CS_REG	Holds the maximum number of chip selects available.	R
b15 to b12	Reserved	NA	R
b11 to b8	MAX_COL_REG	Holds the maximum width of column address in DRAMs.	R
b7 to b5	Reserved	NA	R
b4 to b0	MAX_ROW_REG	Holds the maximum width of memory address bus.	R

6.4.1.3 DDR_CTL_02 — DDR-Controller Status & Control 02

Address: 4000 D008h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	ASYNC_CDC_STAGES								WRITE_DATA_FIFO_PTR_WIDTH							
Value after reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	WRITE_DATA_FIFO_DEPTH								READ_DATA_FIFO_PTR_WIDTH							
Value after reset	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0

Table 6.5 DDR_CTL_02 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	ASYNC_CDC_STAGES	Reports the number of stages used for synchronizer.	R
b23 to b16	WRITE_DATA_FIFO_PTR_WIDTH	Reports the width of the controller write data latency queue pointer.	R
b15 to b8	WRITE_DATA_FIFO_DEPTH	Reports the depth of the controller write data latency queue.	R
b7 to b0	READ_DATA_FIFO_PTR_WIDTH	Reports the width of the controller read data queue pointer.	R

6.4.1.4 DDR_CTL_03 — DDR-Controller Status & Control 03

Address: 4000 D00Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	AXI0_WRCMD_PROC_FIFO_LOG2_DEPTH								AXI0_WRFIFO_LOG2_DEPTH							
Value after reset	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI0_RDFIFO_LOG2_DEPTH								AXI0_CMDFIFO_LOG2_DEPTH							
Value after reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

Table 6.6 DDR_CTL_03 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	AXI0_WRCMD_PROC_FIFO_LOG2_DEPTH	Reports the depth of the AXI port 0 Write Command Processing FIFO. Value is the log2 value of the depth.	R
b23 to b16	AXI0_WRFIFO_LOG2_DEPTH	Reports the depth of the AXI port 0 Write Data FIFO. Value is the log2 value of the depth.	R
b15 to b8	AXI0_RDFIFO_LOG2_DEPTH	Reports the depth of the AXI port 0 Read Data FIFO. Value is the log2 value of the depth.	R
b7 to b0	AXI0_CMDFIFO_LOG2_DEPTH	Reports the depth of the AXI port 0 Command FIFO. Value is the log2 value of the depth.	R

6.4.1.5 DDR_CTL_04 — DDR-Controller Status & Control 04

Address: 4000 D010h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	AXI1_WRCMD_PROC_FIFO_LOG2_DEPTH								AXI1_WRFIFO_LOG2_DEPTH							
Value after reset	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI1_RDFIFO_LOG2_DEPTH								AXI1_CMDFIFO_LOG2_DEPTH							
Value after reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

Table 6.7 DDR_CTL_04 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	AXI1_WRCMD_PROC_FIFO_LOG2_DEPTH	Reports the depth of the AXI port 1 Write Command Processing FIFO. Value is the log2 value of the depth.	R
b23 to b16	AXI1_WRFIFO_LOG2_DEPTH	Reports the depth of the AXI port 1 Write Data FIFO. Value is the log2 value of the depth.	R
b15 to b8	AXI1_RDFIFO_LOG2_DEPTH	Reports the depth of the AXI port 1 Read Data FIFO. Value is the log2 value of the depth.	R
b7 to b0	AXI1_CMDFIFO_LOG2_DEPTH	Reports the depth of the AXI port 1 Command FIFO. Value is the log2 value of the depth.	R

6.4.1.6 DDR_CTL_05 — DDR-Controller Status & Control 05

Address: 4000 D014h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	AXI2_WRCMD_PROC_FIFO_LOG2_DEPTH								AXI2_WRFIFO_LOG2_DEPTH							
Value after reset	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI2_RDFIFO_LOG2_DEPTH								AXI2_CMDFIFO_LOG2_DEPTH							
Value after reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

Table 6.8 DDR_CTL_05 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	AXI2_WRCMD_PROC_FIFO_LOG2_DEPTH	Reports the depth of the AXI port 2 Write Command Processing FIFO. Value is the log2 value of the depth.	R
b23 to b16	AXI2_WRFIFO_LOG2_DEPTH	Reports the depth of the AXI port 2 Write Data FIFO. Value is the log2 value of the depth.	R
b15 to b8	AXI2_RDFIFO_LOG2_DEPTH	Reports the depth of the AXI port 2 Read Data FIFO. Value is the log2 value of the depth.	R
b7 to b0	AXI2_CMDFIFO_LOG2_DEPTH	Reports the depth of the AXI port 2 Command FIFO. Value is the log2 value of the depth.	R

6.4.1.7 DDR_CTL_06 — DDR-Controller Status & Control 06

Address: 4000 D018h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	AXI3_WRCMD_PROC_FIFO_LOG2_DEPTH								AXI3_WRFIFO_LOG2_DEPTH							
Value after reset	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI3_RDFIFO_LOG2_DEPTH								AXI3_CMDFIFO_LOG2_DEPTH							
Value after reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

Table 6.9 DDR_CTL_06 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	AXI3_WRCMD_PROC_FIFO_LOG2_DEPTH	Reports the depth of the AXI port 3 Write Command Processing FIFO. Value is the log2 value of the depth.	R
b23 to b16	AXI3_WRFIFO_LOG2_DEPTH	Reports the depth of the AXI port 3 Write Data FIFO. Value is the log2 value of the depth.	R
b15 to b8	AXI3_RDFIFO_LOG2_DEPTH	Reports the depth of the AXI port 3 Read Data FIFO. Value is the log2 value of the depth.	R
b7 to b0	AXI3_CMDFIFO_LOG2_DEPTH	Reports the depth of the AXI port 3 Command FIFO. Value is the log2 value of the depth.	R

6.4.1.8 DDR_CTL_07 — DDR-Controller Status & Control 07

Address: 4000 D01Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	TINIT							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TINIT															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.10 DDR_CTL_07 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b0	TINIT	DRAM TINIT value (memory clock cycles)	R/W

6.4.1.9 DDR_CTL_08 — DDR-Controller Status & Control 08

Address: 4000 D020h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TRST_PWRON															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TRST_PWRON															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.11 DDR_CTL_08 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	TRST_PWRON	Duration of memory reset during power-on initialization (memory clock cycles)	R/W

6.4.1.10 DDR_CTL_09 — DDR-Controller Status & Control 09

Address: 4000 D024h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	CKE_INACTIVE															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	CKE_INACTIVE															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.12 DDR_CTL_09 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	CKE_INACTIVE	Number of cycles after reset before CKE will be active (memory clock cycles)	R/W

6.4.1.11 DDR_CTL_10 — DDR-Controller Status & Control 10

Address: 4000 D028h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	TCPD							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TCPD								—	—	—	—	INITAREF			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.13 DDR_CTL_10 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b8	TCPD	DRAM TCPD value (memory clock cycles)	R/W
b7 to b4	Reserved	NA	R
b3 to b0	INITAREF	Number of auto-refresh commands to execute during DRAM initialization.	R/W

6.4.1.12 DDR_CTL_11 — DDR-Controller Status & Control 11

Address: 4000 D02Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	CASLAT_LIN						—	—	—	—	—	—	—	NO_CMD_INIT
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TDLL															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.14 DDR_CTL_11 Register Contents

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved	NA	R
b29 to b24	CASLAT_LIN	Sets latency from read command send to data receive from/to controller. The upper bits [5:1] define memory CAS latency for the controller. The bit [0] should be kept the initial value.	R/W
b23 to b17	Reserved	NA	R
b16	NO_CMD_INIT	Disable DRAM commands until the TDLL parameter has expired during initialization. Set to 1 to disable.	R/W
b15 to b0	TDLL	DRAM TDLL value (memory clock cycles)	R/W

6.4.1.13 DDR_CTL_12 — DDR-Controller Status & Control 12

Address: 4000 D030h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	TCCD					—	—	—	—	—	TBST_INT_INTERVAL		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	ADDITIVE_LAT					—	—	—	WRLAT				
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.15 DDR_CTL_12 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b29	Reserved	NA	R
b28 to b24	TCCD	DRAM CAS-to-CAS value (memory clock cycles)	R/W
b23 to b19	Reserved	NA	R
b18 to b16	TBST_INT_INTERVAL	DRAM burst interrupt interval value (memory clock cycles)	R/W
b15 to b13	Reserved	NA	R
b12 to b8	ADDITIVE_LAT	DRAM additive latency value (memory clock cycles)	R/W
b7 to b5	Reserved	NA	R
b4 to b0	WRLAT	DRAM WRLAT value (memory clock cycles)	R/W

6.4.1.14 DDR_CTL_13 — DDR-Controller Status & Control 13

Address: 4000 D034h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	TWTR						TRAS_MIN							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TRC								TRRD							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.16 DDR_CTL_13 Register Contents

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved	NA	R
b29 to b24	TWTR	DRAM TWTR value (memory clock cycles)	R/W
b23 to b16	TRAS_MIN	DRAM TRAS_MIN value (memory clock cycles)	R/W
b15 to b8	TRC	DRAM TRC value (memory clock cycles)	R/W
b7 to b0	TRRD	DRAM TRRD value (memory clock cycles)	R/W

6.4.1.15 DDR_CTL_14 — DDR-Controller Status & Control 14

Address: 4000 D038h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	TMRD					—	—	—	—	TRTP			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	TFAW						—	—	—	TRP				
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.17 DDR_CTL_14 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b29	Reserved	NA	R
b28 to b24	TMRD	DRAM TMRD value (memory clock cycles)	R/W
b23 to b20	Reserved	NA	R
b19 to b16	TRTP	DRAM TRTP value (memory clock cycles)	R/W
b15, b14	Reserved	NA	R
b13 to b8	TFAW	DRAM TFAW value (memory clock cycles)	R/W
b7 to b5	Reserved	NA	R
b4 to b0	TRP	DRAM TRP value (memory clock cycles)	R/W

6.4.1.16 DDR_CTL_15 — DDR-Controller Status & Control 15

Address: 4000 D03Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	TRAS_MAX								
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TRAS_MAX								TMOD							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.18 DDR_CTL_15 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24 to b8	TRAS_MAX	DRAM TRAS_MAX value (memory clock cycles)	R/W
b7 to b0	TMOD	Number of cycles after MRS command and before any other command (memory clock cycles)	R/W

6.4.1.17 DDR_CTL_16 — DDR-Controller Status & Control 16

Address: 4000 D040h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TRCD								—	—	—	—	—	—	—	WRITEINTERP
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TCKESR								—	—	—	—	—	TCKE		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.19 DDR_CTL_16 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	TRCD	DRAM TRCD value (memory clock cycles)	R/W
b23 to b17	Reserved	NA	R
b16	WRITEINTERP	Allow controller to interrupt a write burst to the DRAMs with a read command. Set to 1 to allow interruption.	R/W
b15 to b8	TCKESR	Minimum CKE low pulse width during a self-refresh (memory clock cycles)	R/W
b7 to b3	Reserved	NA	R
b2 to b0	TCKE	Minimum CKE pulse width (memory clock cycles)	R/W

6.4.1.18 DDR_CTL_17 — DDR-Controller Status & Control 17

Address: 4000 D044h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	TRAS_LOCKOUT	—	—	—	—	—	—	—	CONCURRENTAP
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	AP	—	—	TWR					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.20 DDR_CTL_17 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	TRAS_LOCKOUT	Allow the controller to execute auto pre-charge commands before the TRAS_MIN parameter expires. Set to 1 to enable.	R/W
b23 to b17	Reserved	NA	R
b16	CONCURRENTAP	Allow controller to issue commands to other banks while a bank is in auto pre-charge. Set to 1 to enable.	R/W
b15 to b9	Reserved	NA	R
b8	AP	Enable auto pre-charge mode of controller. Set to 1 to enable.	R/W
b7, b6	Reserved	NA	R
b5 to b0	TWR	DRAM TWR value (memory clock cycles)	R/W

6.4.1.19 DDR_CTL_18 — DDR-Controller Status & Control 18

Address: 4000 D048h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	REG_DIMM_ENABLE	—	—	—	TRP_AB				
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	BSTLEN			—	—	TDAL					
Value after reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Table 6.21 DDR_CTL_18 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	REG_DIMM_ENABLE	Enable registered DIMM operation of the controller. Set to 1 to enable.	R/W
b23 to b21	Reserved	NA	R
b20 to b16	TRP_AB	DRAM TRP all bank value (memory clock cycles)	R/W
b15 to b11	Reserved	NA	R
b10 to b8	BSTLEN	Encoded burst length sent to DRAMs during initialization. Set to 1 for BL2, set to 2 for BL4, or set to 3 for BL8.	R/W
b7, b6	Reserved	NA	R
b5 to b0	TDAL	DRAM TDAL value (memory clock cycles)	R/W

6.4.1.20 DDR_CTL_19 — DDR-Controller Status & Control 19

Address: 4000 D04Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	TREF_ENABLE	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	AREFRESH	—	—	—	—	—	—	ADDRESS_MIRRORING	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.22 DDR_CTL_19 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	TREF_ENABLE	Issue auto-refresh commands to the DRAMs at the interval defined in the TREF parameter. Set to 1 to enable.	R/W
b23 to b17	Reserved	NA	R
b16	Reserved	Reserved for future use. It should be kept the initial value.	R/W
b15 to b9	Reserved	NA	R
b8	AREFRESH	Initiate auto-refresh at the end of the current burst boundary. Set to 1 to trigger.	W
b7 to b2	Reserved	NA	R
b1, b0	ADDRESS_MIRRORING	Indicates which chip selects support address mirroring. Bit (0) controls cs0, bit (1) controls cs1, etc. Set each bit to 1 to enable.	R/W

6.4.1.21 DDR_CTL_20 — DDR-Controller Status & Control 20

Address: 4000 D050h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	TREF													
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	TRFC									
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.23 DDR_CTL_20 Register Contents

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved	NA	R
b29 to b16	TREF	DRAM TREF value (memory clock cycles)	R/W
b15 to b10	Reserved	NA	R
b9 to b0	TRFC	DRAM TRFC value (memory clock cycles)	R/W

6.4.1.22 DDR_CTL_21 — DDR-Controller Status & Control 21

Address: 4000 D054h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	TREF_INTERVAL													
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.24 DDR_CTL_21 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved	NA	R
b15, b14	Reserved	NA	R
b13 to b0	TREF_INTERVAL	Defines the cycles between refreshes to different chip selects (memory clock cycles)	R/W

6.4.1.23 DDR_CTL_22 — DDR-Controller Status & Control 22

Address: 4000 D058h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TXPDLL															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TPDEX															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.25 DDR_CTL_22 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	TXPDLL	DRAM TXPDLL value (memory clock cycles)	R/W
b15 to b0	TPDEX	DRAM TPDEX value (memory clock cycles)	R/W

6.4.1.24 DDR_CTL_23 — DDR-Controller Status & Control 23

Address: 4000 D05Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TXARDS															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TXARD															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.26 DDR_CTL_23 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	TXARDS	DRAM TXARDS value (memory clock cycles)	R/W
b15 to b0	TXARD	DRAM TXARD value (memory clock cycles)	R/W

6.4.1.25 DDR_CTL_24 — DDR-Controller Status & Control 24

Address: 4000 D060h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TXSNR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TXSR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.27 DDR_CTL_24 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	TXSNR	DRAM TXSNR value (memory clock cycles)	R/W
b15 to b0	TXSR	DRAM TXSR value (memory clock cycles)	R/W

6.4.1.26 DDR_CTL_25 — DDR-Controller Status & Control 25

Address: 4000 D064h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	CKE_DELAY			—	—	—	—	—	—	—	ENABLE_QUICK_SREFRESH
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	SREFRESH_EXIT_NO_REFRESH	—	—	—	—	—	—	—	PWRUP_SREFRESH_EXIT
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.28 DDR_CTL_25 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b27	Reserved	NA	R
b26 to b24	CKE_DELAY	Additional cycles to delay CKE for status reporting (DFICLK clock cycles)	R/W
b23 to b17	Reserved	NA	R
b16	ENABLE_QUICK_SREFRESH	Allow user to interrupt memory initialization to enter self-refresh mode. Set to 1 to allow interruption.	R/W
b15 to b9	Reserved	NA	R
b8	SREFRESH_EXIT_NO_REFRESH	Disables the automatic refresh request associated with self-refresh exit. Set to 1 to disable.	R/W
b7 to b1	Reserved	NA	R
b0	PWRUP_SREFRESH_EXIT	Allow power up via self-refresh instead of full memory initialization. Set to 1 to enable.	R/W

6.4.1.27 DDR_CTL_26 — DDR-Controller Status & Control 26

Address: 4000 D068h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	LP_CMD								CKSRX							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	CKSRE								—	—	—	—	—	—	LOWPOWER_REFRESH_ENABLE	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.29 DDR_CTL_26 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	LP_CMD	Defines the low power command requested through the software programmable interface. When this command is completed, the low power command complete interrupt (bit 9) in the INT_STATUS parameter will be set to 1. Re-programming attempts to this parameter until the interrupt is set will be ignored. The bits are defined as follows: Bit[7] = Lock 1'b0 = No action 1'b1 = Lock command Bit[6] = Controller clock gating 1'b0 = No action 1'b1 = Gate the controller clock Bit[5] = Memory clock gating 1'b0 = No action 1'b1 = Gate the memory clock Bits[4:2] = Low power state 3'b000 = Active power-down 3'b001 = Pre-charge power-down 3'b010 = Self-refresh All other settings are Reserved Bit[1] = Entry command 1'b0 = No action 1'b1 = Enter the specified state Bit[0] = Exit command 1'b0 = No action 1'b1 = Exit any low power	W
b23 to b16	CKSRX	Clock stable delay on self-refresh exit (memory clock cycles)	R/W
b15 to b8	CKSRE	Clock hold delay on self-refresh entry (memory clock cycles)	R/W
b7 to b2	Reserved	NA	R
b1, b0	LOWPOWER_REFRESH_ENABLE	NA. It should be kept the initial value.	R/W

6.4.1.28 DDR_CTL_27 — DDR-Controller Status & Control 27

Address: 4000 D06Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	LP_AUTO_EXIT_EN			—	—	—	—	—	LP_AUTO_ENTRY_EN		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	LP_ARB_STATE				—	—	LP_STATE					
Value after reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Table 6.30 DDR_CTL_27 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b27	Reserved	NA	R
b26 to b24	LP_AUTO_EXIT_EN	Enable auto exit from each of the low power states when a read or write command enters the command queue. Bit (0) controls power-down, bit (1) controls self-refresh, and bit (2) controls self-refresh with memory and controller clock gating. Set each bit to 1 to enable.	R/W
b23 to b19	Reserved	NA	R
b18 to b16	LP_AUTO_ENTRY_EN	Enable auto entry into each of the low power states when the associated idle timer expires. Bit (0) controls power-down, bit (1) controls self-refresh, and bit (2) controls self-refresh with memory and controller clock gating. Set each bit to 1 to enable.	R/W
b15 to b12	Reserved	NA	R
b11 to b8	LP_ARB_STATE	Reports on the state of the arbiter. Bits (2:0) indicate which interface has control of the low power control module and bit (3) indicates if the software programmable interface has an active lock on the arbiter. For bits (2:0), value of 0 indicates module is idle, value of 1 indicates software programmable interface is in control, value of 3 indicates automatic interface is in control, value of 4 indicates dynamic power control per chip select interface is in control, and value of 5 indicates that the controller is in control.	R
b7, b6	Reserved	NA	R
b5 to b0	LP_STATE	Holds the state of the DRAM memories: Bit[5] = Valid status. This bit will be cleared to 1'b0 when a command is accepted on the software programmable interfaces, and remain at 1'b0 until the command has completed. 1'b0 = Invalid, low power state currently in transition 1'b1 = Valid, stable low power state Bits [4:0] = Low power state 5'b00000 = Idle 5'b00001 = Active Power-Down 5'b00011 = Pre-Charge Power-Down 5'b00101 = Self-Refresh 5'b00110 = Self-Refresh with Memory Clock Gating 5'b00111 = Self-Refresh with Memory and Controller Clock Gating All other settings are Reserved, must not be set ***Note*)** If an active power-down was requested, this parameter will reflect the requested active power-down state even if pre-charge power-down is used.	R

6.4.1.29 DDR_CTL_28 — DDR-Controller Status & Control 28

Address: 4000 D070h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	LP_AUTO_SR_IDLE								—	—	—	—	LP_AUTO_PD_IDLE			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	LP_AUTO_PD_IDLE								—	—	—	—	—	—	LP_AUTO_MEM_GATE_EN	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.31 DDR_CTL_28 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	LP_AUTO_SR_IDLE	Number of long count sequences until the controller will place memory in self-refresh.	R/W
b23 to b20	Reserved	NA	R
b19 to b8	LP_AUTO_PD_IDLE	Defines the idle time until the controller will place memory in active power-down. (DFICLK clock cycles)	R/W
b7 to b2	Reserved	NA	R
b1, b0	LP_AUTO_MEM_GATE_EN	Enable memory clock gating when entering a low power state via the auto low power counters. Bit (0) controls power-down, and bit (1) controls self-refresh. Set each bit to 1 to enable.	R/W

6.4.1.30 DDR_CTL_29 — DDR-Controller Status & Control 29

Address: 4000 D074h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	LP_AUTO_SR_MC_GATE_IDLE							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.32 DDR_CTL_29 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b15	Reserved	NA	R
b14 to b8	Reserved	Reserved for future use. It should be kept the initial value.	R/W
b7 to b0	LP_AUTO_SR_MC_GATE_IDLE	Number of long count sequences until the controller will place memory in self-refresh with controller and memory clock gating.	R/W

6.4.1.31 DDR_CTL_30 — DDR-Controller Status & Control 30

Address: 4000 D078h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	WRITE_MODEREG									
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	WRITE_MODEREG															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.33 DDR_CTL_30 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b26	Reserved	NA	R
b25 to b0	WRITE_MODEREG	Write memory mode register data to the DRAMs. Bits (7:0) define the memory mode register number if bit (23) is set bits (15:8) define the chip select if bit (24) is clear bits (23:16) define which memory mode register/s to write bit (24) defines whether all chip selects will be written bit (25) triggers the write. Writeable but may also be changed by the internal logic.	R/W

6.4.1.32 DDR_CTL_31 — DDR-Controller Status & Control 31

Address: 4000 D07Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	MR0_DATA_0							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	MR0_DATA_0								MRW_STATUS							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.34 DDR_CTL_31 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b8	MR0_DATA_0	Data to program into memory mode register 0 for chip select 0.	R/W
b7 to b0	MRW_STATUS	Status of Write memory mode register. Bit (0) set indicates a WRITE_MODEREG parameter programming error.	R

6.4.1.33 DDR_CTL_32 — DDR-Controller Status & Control 32

Address: 4000 D080h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	MR2_DATA_0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	MR1_DATA_0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.35 DDR_CTL_32 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	MR2_DATA_0	Data to program into memory mode register 2 for chip select 0.	R/W
b15 to b0	MR1_DATA_0	Data to program into memory mode register 1 for chip select 0.	R/W

6.4.1.34 DDR_CTL_33 — DDR-Controller Status & Control 33

Address: 4000 D084h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	MR3_DATA_0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	MRSINGLE_DATA_0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.36 DDR_CTL_33 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	MR3_DATA_0	Data to program into memory mode register 3 for chip select 0.	R/W
b15 to b0	MRSINGLE_DATA_0	Data to program into memory mode register single write to chip select 0.	R/W

6.4.1.35 DDR_CTL_34 — DDR-Controller Status & Control 34

Address: 4000 D088h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	MR1_DATA_1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	MR0_DATA_1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.37 DDR_CTL_34 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	MR1_DATA_1	Data to program into memory mode register 1 for chip select 1.	R/W
b15 to b0	MR0_DATA_1	Data to program into memory mode register 0 for chip select 1.	R/W

6.4.1.36 DDR_CTL_35 — DDR-Controller Status & Control 35

Address: 4000 D08Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	MRSINGLE_DATA_1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	MR2_DATA_1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.38 DDR_CTL_35 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	MRSINGLE_DATA_1	Data to program into memory mode register single write to chip select 1.	R/W
b15 to b0	MR2_DATA_1	Data to program into memory mode register 2 for chip select 1.	R/W

6.4.1.37 DDR_CTL_36 — DDR-Controller Status & Control 36

Address: 4000 D090h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	FWC	—	—	—	—	—	—	—	ECC_EN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	MR3_DATA_1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.39 DDR_CTL_36 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	FWC	Force a write check. Xor the XOR_CHECK_BITS parameter with the ECC code and write to memory. Set to 1 to trigger.	W
b23 to b17	Reserved	NA	R
b16	ECC_EN	ECC error checking and correcting control. Set to 0 to disable ECC or set to 1 for ECC reporting and correcting. To enable the ECC function, set the “REDUC” to 1 to enable the half datapath feature.	R/W
b15 to b0	MR3_DATA_1	Data to program into memory mode register 3 for chip select 1.	R/W

6.4.1.38 DDR_CTL_37 — DDR-Controller Status & Control 37

Address: 4000 D094h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	ECC_DISABLE_W_UC_ERR
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	XOR_CHECK_BITS													
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.40 DDR_CTL_37 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b17	Reserved	NA	R
b16	ECC_DISABLE_W_UC_ERR	Controls auto-corruption of ECC when un-correctable errors occur in read/modify/write operations. Set to 1 to disable corruption.	R/W
b15, b14	Reserved	NA	R
b13 to b0	XOR_CHECK_BITS	Value to xor with generated ECC codes for forced write check.	R/W

6.4.1.39 DDR_CTL_38 — DDR-Controller Status & Control 38

Address: 4000 D098h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	ECC_U_ADDR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ECC_U_ADDR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.41 DDR_CTL_38 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	ECC_U_ADDR	Address of uncorrectable ECC event.	R

6.4.1.40 DDR_CTL_39 — DDR-Controller Status & Control 39

Address: 4000 D09Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	ECC_U_SYND						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.42 DDR_CTL_39 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved	NA	R
b7	Reserved	NA	R
b6 to b0	ECC_U_SYND	Syndrome for uncorrectable ECC event.	R

6.4.1.41 DDR_CTL_40 — DDR-Controller Status & Control 40

Address: 4000 D0A0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	ECC_U_DATA															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ECC_U_DATA															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.43 DDR_CTL_40 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	ECC_U_DATA	Data associated with uncorrectable ECC event.	R

6.4.1.42 DDR_CTL_41 — DDR-Controller Status & Control 41

Address: 4000 D0A4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	ECC_C_ADDR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ECC_C_ADDR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.44 DDR_CTL_41 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	ECC_C_ADDR	Address of correctable ECC event.	R

6.4.1.43 DDR_CTL_42 — DDR-Controller Status & Control 42

Address: 4000 D0A8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	ECC_C_SYND						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.45 DDR_CTL_42 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved	NA	R
b7	Reserved	NA	R
b6 to b0	ECC_C_SYND	Syndrome for correctable ECC event.	R

6.4.1.44 DDR_CTL_43 — DDR-Controller Status & Control 43

Address: 4000 D0ACh

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	ECC_C_DATA															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ECC_C_DATA															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.46 DDR_CTL_43 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	ECC_C_DATA	Data associated with correctable ECC event.	R

6.4.1.45 DDR_CTL_44 — DDR-Controller Status & Control 44

Address: 4000 D0B0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	LONG_COUNT_MASK				
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.47 DDR_CTL_44 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b21	Reserved	NA	R
b20 to b16	LONG_COUNT_MASK	Reduces the length of the long counter from 1024 cycles. The only supported values are 0x00 (1024 cycles), 0x10 (512 clocks), 0x18 (256 clocks), 0x1C (128 clocks), 0x1E (64 clocks) and 0x1F (32 clocks).	R/W
b15 to b0	Reserved	NA	R

6.4.1.46 DDR_CTL_45 — DDR-Controller Status & Control 45

Address: 4000 D0B4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	ZQCL											
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	ZQINIT											
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.48 DDR_CTL_45 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b28	Reserved	NA	R
b27 to b16	ZQCL	Number of cycles needed for a ZQCL command (memory clock cycles)	R/W
b15 to b12	Reserved	NA	R
b11 to b0	ZQINIT	Number of cycles needed for a ZQINIT command (memory clock cycles)	R/W

6.4.1.47 DDR_CTL_46 — DDR-Controller Status & Control 46

Address: 4000 D0B8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	ZQ_ON_SREF_EXIT		—	—	—	—	—	—	ZQ_REQ	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	ZQCS											
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.49 DDR_CTL_46 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b26	Reserved	NA	R
b25, b24	ZQ_ON_SREF_EXIT	Defines the type of ZQ calibration performed at self-refresh exit. Bit (0) selects ZQCS and bit (1) selects ZQCL. Set either bit to 1 to enable.	R/W
b23 to b18	Reserved	NA	R
b17, b16	ZQ_REQ	User request to initiate a ZQ calibration. Bit (0) controls ZQCS and bit (1) controls ZQCL. Set either bit to 1 to trigger.	W
b15 to b12	Reserved	NA	R
b11 to b0	ZQCS	Number of cycles needed for a ZQCS command (memory clock cycles)	R/W

6.4.1.48 DDR_CTL_47 — DDR-Controller Status & Control 47

Address: 4000 D0BCh

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	ZQ_INTERVAL															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ZQ_INTERVAL															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.50 DDR_CTL_47 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	ZQ_INTERVAL	Number of long count sequences allowed between automatic ZQCS commands.	R/W

6.4.1.49 DDR_CTL_48 — DDR-Controller Status & Control 48

Address: 4000 D0C0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	ROW_DIFF			—	—	—	—	—	—	BANK_DIFF	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	ZQCS_ROTATE	—	—	—	—	—	—	—	ZQ_IN_PROGRESS
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.51 DDR_CTL_48 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b27	Reserved	NA	R
b26 to b24	ROW_DIFF	Difference between number of address pins available (16) and number being used.	R/W
b23 to b18	Reserved	NA	R
b17, b16	BANK_DIFF	Encoded number of banks on the DRAM(s). 2'b00 = 8 banks 2'b01 = 4 banks 2'b10 = 2 banks 2'b11 = Reserved	R/W
b15 to b9	Reserved	NA	R
b8	ZQCS_ROTATE	Selects whether a ZQCS command will calibrate just one chip select or all chip selects. Set to 1 for rotating CS.	R/W
b7 to b1	Reserved	NA	R
b0	ZQ_IN_PROGRESS	Indicates that a ZQ command is currently in progress. Value of 1 indicates command in progress.	R

6.4.1.50 DDR_CTL_49 — DDR-Controller Status & Control 49

Address: 4000 D0C4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	COMMAND_AGE_COUNT								AGE_COUNT							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	APREBIT				—	—	—	—	COL_DIFF			
Value after reset	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0

Table 6.52 DDR_CTL_49 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	COMMAND_AGE_COUNT	Initial value of individual command aging counters for command aging (DFICLK clock cycles)	R/W
b23 to b16	AGE_COUNT	Initial value of master aging-rate counter for command aging (DFICLK clock cycles)	R/W
b15 to b12	Reserved	NA	R
b11 to b8	APREBIT	Location of the auto pre-charge bit in the DRAM address.	R/W
b7 to b4	Reserved	NA	R
b3 to b0	COL_DIFF	Difference between number of column pins available (11) and number being used.	R/W

6.4.1.51 DDR_CTL_50 — DDR-Controller Status & Control 50

Address: 4000 D0C8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	PLACEMENT_EN	—	—	—	—	—	—	—	BANK_SPLIT_EN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	ADDR_CMP_EN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.53 DDR_CTL_50 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	PLACEMENT_EN	Enable placement logic for command queue. Set to 1 to enable.	R/W
b23 to b17	Reserved	NA	R
b16	BANK_SPLIT_EN	Enable bank splitting as a rule for command queue placement. Set to 1 to enable.	R/W
b15 to b9	Reserved	NA	R
b8	Reserved	Reserved for future use. It should be set to 1.	R/W
b7 to b1	Reserved	NA	R
b0	ADDR_CMP_EN	Enable address collision detection as a rule for command queue placement. Set to 1 to enable.	R/W

6.4.1.52 DDR_CTL_51 — DDR-Controller Status & Control 51

Address: 4000 D0CCh

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	CS_SAME_EN	—	—	—	—	—	—	—	RW_SAME_PAGE_EN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	RW_SAME_EN	—	—	—	—	—	—	—	PRIORITY_EN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.54 DDR_CTL_51 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	CS_SAME_EN	Enable chip select grouping when read/write grouping as a rule for command queue placement. This is only valid when the RW_SAME_EN parameter is set. Set to 1 to enable.	R/W
b23 to b17	Reserved	NA	R
b16	RW_SAME_PAGE_EN	Enable page grouping when read/write grouping as a rule for command queue placement. This is only valid when the RW_SAME_EN parameter is set. Set to 1 to enable.	R/W
b15 to b9	Reserved	NA	R
b8	RW_SAME_EN	Enable read/write grouping as a rule for command queue placement. Set to 1 to enable.	R/W
b7 to b1	Reserved	NA	R
b0	PRIORITY_EN	Enable priority as a rule for command queue placement. Set to 1 to enable.	R/W

6.4.1.53 DDR_CTL_52 — DDR-Controller Status & Control 52

Address: 4000 D0D0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	SWAP_EN	—	—	—	—	—	NUM_Q_ENTRIES_ACT_DISABLE		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	DISABLE_RW_GROUP_W_BNK_CONFLICT		—	—	—	—	—	—	—	W2R_SPLIT_EN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.55 DDR_CTL_52 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	SWAP_EN	Enable command swapping logic in execution unit. Set to 1 to enable.	R/W
b23 to b19	Reserved	NA	R
b18 to b16	NUM_Q_ENTRIES_ACT_DISABLE	Number of queue entries in which ACT requests will be disabled. Setting to X will disable ACT requests from the X entries lowest in the command queue.	R/W
b15 to b10	Reserved	NA	R
b9, b8	DISABLE_RW_GROUP_W_BNK_CONFLICT	Disables placement to read/write group when grouping creates a bank collision. Bit (0) controls placement next to bank conflict command and bit (1) controls placement 2 away from bank conflict command. Set each bit to 1 to disable.	R/W
b7 to b1	Reserved	NA	R
b0	W2R_SPLIT_EN	Enable splitting of commands to the same chip select from a write to a read command as a rule for command queue placement. Set to 1 to enable.	R/W

6.4.1.54 DDR_CTL_53 — DDR-Controller Status & Control 53

Address: 4000 D0D4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	BURST_ON_FLY_BIT				—	—	—	—	—	—	CS_MAP	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	INHIBIT_DRAM_CMD	—	—	—	—	—	—	—	DISABLE_RD_INTERLEAVE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.56 DDR_CTL_53 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b28	Reserved	NA	R
b27 to b24	BURST_ON_FLY_BIT	Identifies the burst-on-fly bit in the memory mode registers.	R/W
b23 to b18	Reserved	NA	R
b17, b16	CS_MAP	Defines which chip selects are active.	R/W
b15 to b9	Reserved	NA	R
b8	INHIBIT_DRAM_CMD	Inhibit read/write command traffic and associated bank commands. Set to 1 to inhibit.	R/W
b7 to b1	Reserved	NA	R
b0	DISABLE_RD_INTERLEAVE	Disable read data interleaving for commands from the same port. 0: Allow read data interleaving 1: Disable read data interleaving	R/W

6.4.1.55 DDR_CTL_54 — DDR-Controller Status & Control 54

Address: 4000 D0D8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	CONTROLLER_BUSY	—	—	—	—	—	—	—	IN_ORDER_ACCEPT
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	Q_FULLNESS			—	—	—	—	—	—	—	REDUC
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.57 DDR_CTL_54 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	CONTROLLER_BUSY	Indicator that the controller is processing a command. Evaluates all ports for outstanding transactions. Value of 1 indicates controller busy.	R
b23 to b17	Reserved	NA	R
b16	IN_ORDER_ACCEPT	Forces the controller to accept commands in the order in which they are placed in the command queue. 0: Uses the selection logic to select the ideal command for execution 1: Disables the selection logic and executes commands in order	R/W
b15 to b11	Reserved	NA	R
b10 to b8	Q_FULLNESS	Quantity that determines command queue full.	R/W
b7 to b1	Reserved	NA	R
b0	REDUC	Enable the half datapath feature of the controller. Set to 1 to enable. Data is transmitted on the lower datapath.	R/W

6.4.1.56 DDR_CTL_55 — DDR-Controller Status & Control 55

Address: 4000 D0DCh

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	CTRLUPD_REQ_PER_AREF_EN	—	—	—	—	—	—	—	CTRLUPD_REQ
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.58 DDR_CTL_55 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved	NA	R
b15 to b9	Reserved	NA	R
b8	CTRLUPD_REQ_PER_AREF_EN	Enable an automatic controller-initiated update (dfi_ctrlupd_req) after every refresh. Set to 1 to enable.	R/W
b7 to b1	Reserved	NA	R
b0	CTRLUPD_REQ	Assert the DFI controller-initiated update request signal dfi_ctrlupd_req. Set to 1 to trigger.	W

6.4.1.57 DDR_CTL_56 — DDR-Controller Status & Control 56

Address: 4000 D0E0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	INT_STATUS						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	INT_STATUS															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.59 DDR_CTL_56 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23	Reserved	NA	R
b22 to b0	INT_STATUS	The INT_STATUS bits correspond to these interrupts: Bit [22] = Logical OR of all lower bits. Bit [21] = The user-initiated DLL resync has completed. Bit [20] = A state change has been detected on the dfi_init_complete signal after initialization. Bit [19] = The assertion of the INHIBIT_DRAM_CMD parameter has successfully inhibited the command queue. Bit [18] = The register interface-initiated mode register write has completed and another mode register write may be issued. Bit [17:16] = Reserved Bit [15] = A DFI update error has occurred. Error information can be found in the UPDATE_ERROR_STATUS parameter. Bit [14:12] = Reserved Bit [11] = The user has programmed an invalid setting associated with 64-bit-defined-word per burst. Bit [10] = Wrap cycle crossing a DRAM page has been detected. This is unsupported & may result in memory data corruption. Bit [9] = The low power operation has been completed. Bit [8] = The memory controller initialization has been completed. Bit [7] = An error occurred on the port command channel. Bit [6] = Multiple uncorrectable ECC events have been detected. Bit [5] = An uncorrectable ECC event has been detected. Bit [4] = Multiple correctable ECC events have been detected. Bit [3] = A correctable ECC event has been detected. Bit [2] = Multiple accesses outside the defined PHYSICAL memory space have occurred. Bit [1] = A memory access outside the defined PHYSICAL memory space has occurred. Bit [0] = The memory reset is valid on the DFI bus.	R

6.4.1.58 DDR_CTL_57 — DDR-Controller Status & Control 57

Address: 4000 D0E4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	INT_ACK					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	INT_ACK															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.60 DDR_CTL_57 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23, b22	Reserved	NA	R
b21 to b0	INT_ACK	Clear bit of the INT_STATUS parameter. 0: No effect 1: Clears the associated bit in the INT_STATUS	W

6.4.1.59 DDR_CTL_58 — DDR-Controller Status & Control 58

Address: 4000 D0E8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	INT_MASK						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	INT_MASK															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.61 DDR_CTL_58 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23	Reserved	NA	R
b22 to b0	INT_MASK	Mask for DDRC_Int signals from the INT_STATUS parameter. Bit [22] 0: Mask interrupts individually based on the settings of bits [21:0]. 1: Mask all interrupts. The settings in bits [21:0] are not relevant. Bit [21:0] 0: Do not mask interrupt 1: Mask interrupt	R/W

6.4.1.60 DDR_CTL_59 — DDR-Controller Status & Control 59

Address: 4000 D0ECh

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	OUT_OF_RANGE_ADDR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	OUT_OF_RANGE_ADDR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.62 DDR_CTL_59 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	OUT_OF_RANGE_ADDR	Address of command that caused an out-of-range interrupt.	R

6.4.1.61 DDR_CTL_60 — DDR-Controller Status & Control 60

Address: 4000 D0F0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	OUT_OF_RANGE_TYPE						—	OUT_OF_RANGE_LENGTH						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.63 DDR_CTL_60 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b14	Reserved	NA	R
b13 to b8	OUT_OF_RANGE_TYPE	Type of command that caused an out-of-range interrupt.	R
b7	Reserved	NA	R
b6 to b0	OUT_OF_RANGE_LENGTH	Length of command that caused an out-of-range interrupt.	R

6.4.1.62 DDR_CTL_61 — DDR-Controller Status & Control 61

Address: 4000 D0F4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	PORT_CMD_ERROR_ADDR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	PORT_CMD_ERROR_ADDR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.64 DDR_CTL_61 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	PORT_CMD_ERROR_ADDR	Address of command that caused the PORT command error.	R

6.4.1.63 DDR_CTL_62 — DDR-Controller Status & Control 62

Address: 4000 D0F8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	ODT_WR_MAP_CS0		—	—	—	—	—	—	ODT_RD_MAP_CS0	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	PORT_CMD_ERROR_TYPE			—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.65 DDR_CTL_62 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b26	Reserved	NA	R
b25, b24	ODT_WR_MAP_CS0	Determines which chip(s) will have termination when a write occurs on chip select 0. Set bit X to enable termination on csX when cs0 is performing a write.	R/W
b23 to b18	Reserved	NA	R
b17, b16	ODT_RD_MAP_CS0	Determines which chip(s) will have termination when a read occurs on chip select 0. Set bit X to enable termination on csX when cs0 is performing a read.	R/W
b15 to b11	Reserved	NA	R
b10 to b8	PORT_CMD_ERROR_TYPE	Type of error and access type that caused the PORT command error.	R
b7 to b0	Reserved	NA	R

6.4.1.64 DDR_CTL_63 — DDR-Controller Status & Control 63

Address: 4000 D0FCh

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	TODTH_WR				TODTL_2CMD							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	ODT_WR_MAP_CS1		—	—	—	—	—	—	ODT_RD_MAP_CS1	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.66 DDR_CTL_63 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b28	Reserved	NA	R
b27 to b24	TODTH_WR	Defines the DRAM minimum ODT high time after an ODT assertion for a write command (memory clock cycles)	R/W
b23 to b16	TODTL_2CMD	Defines the DRAM delay from an ODT de-assertion to the next non-write, non-read command (memory clock cycles)	R/W
b15 to b10	Reserved	NA	R
b9, b8	ODT_WR_MAP_CS1	Determines which chip(s) will have termination when a write occurs on chip select 1. Set bit X to enable termination on csX when cs1 is performing a write.	R/W
b7 to b2	Reserved	NA	R
b1, b0	ODT_RD_MAP_CS1	Determines which chip(s) will have termination when a read occurs on chip select 1. Set bit X to enable termination on csX when cs1 is performing a read.	R/W

6.4.1.65 DDR_CTL_64 — DDR-Controller Status & Control 64

Address: 4000 D100h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	RD_TO_ODTH							—	WR_TO_ODTH						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	ODT_EN	—	—	—	—	TODTH_RD			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.67 DDR_CTL_64 Register Contents

Bit Position	Bit Name	Function	R/W
b31	Reserved	NA	R
b30 to b24	RD_TO_ODTH	Defines the delay from a read command to ODT assertion (memory clock cycles)	R/W
b23	Reserved	NA	R
b22 to b16	WR_TO_ODTH	Defines the delay from a write command to ODT assertion (memory clock cycles)	R/W
b15 to b9	Reserved	NA	R
b8	ODT_EN	Enable support of DRAM ODT. When enabled, controller will assert and de-assert ODT output to DRAM as needed. Set to 1 to enable.	R/W
b7 to b4	Reserved	NA	R
b3 to b0	TODTH_RD	Defines the DRAM minimum ODT high time after an ODT assertion for a read command (memory clock cycles)	R/W

6.4.1.66 DDR_CTL_65 — DDR-Controller Status & Control 65

Address: 4000 D104h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	OBSOLETE0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	OBSOLETE0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.68 DDR_CTL_65 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	OBSOLETE0	NA	R

6.4.1.67 DDR_CTL_66 — DDR-Controller Status & Control 66

Address: 4000 D108h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	W2W_DIFFCS_DLY				—	—	—	—	—	W2R_DIFFCS_DLY		
Value after reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	R2W_DIFFCS_DLY			—	—	—	—	—	R2R_DIFFCS_DLY		
Value after reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

Table 6.69 DDR_CTL_66 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b28	Reserved	NA	R
b27 to b24	W2W_DIFFCS_DLY	Additional delay to insert between writes to different chip selects. Program to a non-zero value. (memory clock cycles)	R/W
b23 to b19	Reserved	NA	R
b18 to b16	W2R_DIFFCS_DLY	Additional delay to insert between writes and reads to different chip selects. Allowed programming dependent on memory system. (memory clock cycles)	R/W
b15 to b11	Reserved	NA	R
b10 to b8	R2W_DIFFCS_DLY	Additional delay to insert between reads and writes to different chip selects. Program to a non-zero value. (memory clock cycles)	R/W
b7 to b3	Reserved	NA	R
b2 to b0	R2R_DIFFCS_DLY	Additional delay to insert between reads to different chip selects. Program to a non-zero value. (memory clock cycles)	R/W

6.4.1.68 DDR_CTL_67 — DDR-Controller Status & Control 67

Address: 4000 D10Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	W2W_SAMECS_DLY			—	—	—	—	—	W2R_SAMECS_DLY		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	R2W_SAMECS_DLY			—	—	—	—	—	R2R_SAMECS_DLY		
Value after reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Table 6.70 DDR_CTL_67 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b27	Reserved	NA	R
b26 to b24	W2W_SAMECS_DLY	Additional delay to insert between two writes to the same chip select. Any value including 0x0 supported. (memory clock cycles)	R/W
b23 to b19	Reserved	NA	R
b18 to b16	W2R_SAMECS_DLY	Additional delay to insert between writes and reads to the same chip select (memory clock cycles)	R/W
b15 to b11	Reserved	NA	R
b10 to b8	R2W_SAMECS_DLY	Additional delay to insert between reads and writes to the same chip select. Program to a non-zero value. (memory clock cycles)	R/W
b7 to b3	Reserved	NA	R
b2 to b0	R2R_SAMECS_DLY	Additional delay to insert between two reads to the same chip select. Any value including 0x0 supported. (memory clock cycles)	R/W

6.4.1.69 DDR_CTL_68 — DDR-Controller Status & Control 68

Address: 4000 D110h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	SWLVL_LOAD	—	—	—	—	—	—	SW_LEVELING_MODE	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	OCD_ADJUST_PUP_CS_0					—	—	—	OCD_ADJUST_PDN_CS_0				
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.71 DDR_CTL_68 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	SWLVL_LOAD	NA	W
b23 to b18	Reserved	NA	R
b17, b16	SW_LEVELING_MODE	NA. The bit should be kept the initial value.	R/W
b15 to b13	Reserved	NA	R
b12 to b8	OCD_ADJUST_PUP_CS_0	NA. The bit should be kept the initial value.	R/W
b7 to b5	Reserved	NA	R
b4 to b0	OCD_ADJUST_PDN_CS_0	NA. The bit should be kept the initial value.	R/W

6.4.1.70 DDR_CTL_69 — DDR-Controller Status & Control 69

Address: 4000 D114h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	LVL_STATUS				—	—	—	—	—	—	—	SWLVL_OP_DONE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	SWLVL_EXIT	—	—	—	—	—	—	—	SWLVL_START
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.72 DDR_CTL_69 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b28	Reserved	NA	R
b27 to b24	LVL_STATUS	NA	R
b23 to b17	Reserved	NA	R
b16	SWLVL_OP_DONE	NA	R
b15 to b9	Reserved	NA	R
b8	SWLVL_EXIT	NA	W
b7 to b1	Reserved	NA	R
b0	SWLVL_START	NA	W

6.4.1.71 DDR_CTL_70 — DDR-Controller Status & Control 70

Address: 4000 D118h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	WRLVL_REQ	SWLVL_RESP_2							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SWLVL_RESP_1								SWLVL_RESP_0							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.73 DDR_CTL_70 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	WRLVL_REQ	NA	W
b23 to b16	SWLVL_RESP_2	NA	R
b15 to b8	SWLVL_RESP_1	NA	R
b7 to b0	SWLVL_RESP_0	NA	R

6.4.1.72 DDR_CTL_71 — DDR-Controller Status & Control 71

Address: 4000 D11Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	WRLVL_EN	—	—	WLMRD					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	WLDQSEN						—	—	—	—	—	—	—	WRLVL_CS
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.74 DDR_CTL_71 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	WRLVL_EN	NA. The bit should be kept the initial value.	R/W
b23, b22	Reserved	NA	R
b21 to b16	WLMRD	NA. The bit should be kept the initial value.	R/W
b15, b14	Reserved	NA	R
b13 to b8	WLDQSEN	NA. The bit should be kept the initial value.	R/W
b7 to b1	Reserved	NA	R
b0	WRLVL_CS	NA. The bit should be kept the initial value.	R/W

6.4.1.73 DDR_CTL_72 — DDR-Controller Status & Control 72

Address: 4000 D120h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	WRLVL_ERROR_STATUS								—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	WRLVL_INTERVAL															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.75 DDR_CTL_72 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	WRLVL_ERROR_STATUS	NA	R
b23 to b19	Reserved	NA	R
b18 to b16	Reserved	Reserved for future use. It should be kept the initial value.	R/W
b15 to b0	WRLVL_INTERVAL	NA. The bit should be kept the initial value.	R/W

6.4.1.74 DDR_CTL_73 — DDR-Controller Status & Control 73

Address: 4000 D124h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	WRLVL_DELAY_0							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	WRLVL_DELAY_0								—	—	—	—	—	—	—	WRLVL_REG_EN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.76 DDR_CTL_73 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b8	WRLVL_DELAY_0	NA. The bit should be kept the initial value.	R/W
b7 to b1	Reserved	NA	R
b0	WRLVL_REG_EN	NA. The bit should be kept the initial value.	R/W

6.4.1.75 DDR_CTL_74 — DDR-Controller Status & Control 74

Address: 4000 D128h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	WRLVL_DELAY_2															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	WRLVL_DELAY_1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.77 DDR_CTL_74 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	WRLVL_DELAY_2	NA. The bit should be kept the initial value.	R/W
b15 to b0	WRLVL_DELAY_1	NA. The bit should be kept the initial value.	R/W

6.4.1.76 DDR_CTL_75 — DDR-Controller Status & Control 75

Address: 4000 D12Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	RDLVL_EDGE	—	—	—	—	—	—	—	RDLVL_CS
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	RDLVL_GATE_REQ	—	—	—	—	—	—	—	RDLVL_REQ
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.78 DDR_CTL_75 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	RDLVL_EDGE	NA. The bit should be kept the initial value.	R/W
b23 to b17	Reserved	NA	R
b16	RDLVL_CS	NA. The bit should be kept the initial value.	R/W
b15 to b9	Reserved	NA	R
b8	RDLVL_GATE_REQ	NA	W
b7 to b1	Reserved	NA	R
b0	RDLVL_REQ	NA	W

6.4.1.77 DDR_CTL_76 — DDR-Controller Status & Control 76

Address: 4000 D130h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	RDLVL_GATE_REG_EN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	RDLVL_REG_EN	—	—	—	—	—	—	—	RDLVL_BEGIN_DELAY_EN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.79 DDR_CTL_76 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b17	Reserved	NA	R
b16	RDLVL_GATE_REG_EN	NA. The bit should be kept the initial value.	R/W
b15 to b9	Reserved	NA	R
b8	RDLVL_REG_EN	NA. The bit should be kept the initial value.	R/W
b7 to b1	Reserved	NA	R
b0	RDLVL_BEGIN_DELAY_EN	NA. The bit should be kept the initial value.	R/W

6.4.1.78 DDR_CTL_77 — DDR-Controller Status & Control 77

Address: 4000 D134h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	RDLVL_END_DELAY_0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_BEGIN_DELAY_0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.80 DDR_CTL_77 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	RDLVL_END_DELAY_0	NA	R
b15 to b0	RDLVL_BEGIN_DELAY_0	NA	R

6.4.1.79 DDR_CTL_78 — DDR-Controller Status & Control 78

Address: 4000 D138h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	RDLVL_OFFSET_DELAY_0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_MIDPOINT_DELAY_0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.81 DDR_CTL_78 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	RDLVL_OFFSET_DELAY_0	NA. The bit should be kept the initial value.	R/W
b15 to b0	RDLVL_MIDPOINT_DELAY_0	NA	R

6.4.1.80 DDR_CTL_79 — DDR-Controller Status & Control 79

Address: 4000 D13Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	RDLVL_DELAY_0							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_DELAY_0								—	—	—	—	—	—	—	RDLVL_OFFSET_DIR_0
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.82 DDR_CTL_79 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b8	RDLVL_DELAY_0	NA. The bit should be kept the initial value.	R/W
b7 to b1	Reserved	NA	R
b0	RDLVL_OFFSET_DIR_0	NA. The bit should be kept the initial value.	R/W

6.4.1.81 DDR_CTL_80 — DDR-Controller Status & Control 80

Address: 4000 D140h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	RDLVL_BEGIN_DELAY_1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_GATE_DELAY_0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.83 DDR_CTL_80 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	RDLVL_BEGIN_DELAY_1	NA	R
b15 to b0	RDLVL_GATE_DELAY_0	NA. The bit should be kept the initial value.	R/W

6.4.1.82 DDR_CTL_81 — DDR-Controller Status & Control 81

Address: 4000 D144h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	RDLVL_MIDPOINT_DELAY_1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_END_DELAY_1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.84 DDR_CTL_81 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	RDLVL_MIDPOINT_DELAY_1	NA	R
b15 to b0	RDLVL_END_DELAY_1	NA	R

6.4.1.83 DDR_CTL_82 — DDR-Controller Status & Control 82

Address: 4000 D148h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	RDLVL_OFFSET_DIR_1
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_OFFSET_DELAY_1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.85 DDR_CTL_82 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b17	Reserved	NA	R
b16	RDLVL_OFFSET_DIR_1	NA. The bit should be kept the initial value.	R/W
b15 to b0	RDLVL_OFFSET_DELAY_1	NA. The bit should be kept the initial value.	R/W

6.4.1.84 DDR_CTL_83 — DDR-Controller Status & Control 83

Address: 4000 D14Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	RDLVL_GATE_DELAY_1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_DELAY_1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.86 DDR_CTL_83 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	RDLVL_GATE_DELAY_1	NA. The bit should be kept the initial value.	R/W
b15 to b0	RDLVL_DELAY_1	NA. The bit should be kept the initial value.	R/W

6.4.1.85 DDR_CTL_84 — DDR-Controller Status & Control 84

Address: 4000 D150h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	RDLVL_END_DELAY_2															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_BEGIN_DELAY_2															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.87 DDR_CTL_84 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	RDLVL_END_DELAY_2	NA	R
b15 to b0	RDLVL_BEGIN_DELAY_2	NA	R

6.4.1.86 DDR_CTL_85 — DDR-Controller Status & Control 85

Address: 4000 D154h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	RDLVL_OFFSET_DELAY_2															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_MIDPOINT_DELAY_2															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.88 DDR_CTL_85 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	RDLVL_OFFSET_DELAY_2	NA. The bit should be kept the initial value.	R/W
b15 to b0	RDLVL_MIDPOINT_DELAY_2	NA	R

6.4.1.87 DDR_CTL_86 — DDR-Controller Status & Control 86

Address: 4000 D158h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	RDLVL_DELAY_2							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_DELAY_2								—	—	—	—	—	—	—	RDLVL_OFFSET_DIR_2
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.89 DDR_CTL_86 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b8	RDLVL_DELAY_2	NA. The bit should be kept the initial value.	R/W
b7 to b1	Reserved	NA	R
b0	RDLVL_OFFSET_DIR_2	NA. The bit should be kept the initial value.	R/W

6.4.1.88 DDR_CTL_87 — DDR-Controller Status & Control 87

Address: 4000 D15Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	AXI0_W_PRIORITY		—	—	—	—	—	—	AXI0_R_PRIORITY	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_GATE_DELAY_2															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.90 DDR_CTL_87 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b26	Reserved	NA	R
b25, b24	AXI0_W_PRIORITY	Priority of write commands from AXI port 0. 0 is the highest priority.	R/W
b23 to b18	Reserved	NA	R
b17, b16	AXI0_R_PRIORITY	Priority of read commands from AXI port 0. 0 is the highest priority.	R/W
b15 to b0	RDLVL_GATE_DELAY_2	NA. The bit should be kept the initial value.	R/W

6.4.1.89 DDR_CTL_88 — DDR-Controller Status & Control 88

Address: 4000 D160h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	AXI1_FIFO_TYPE_REG		—	—	—	—	—	—	AXI1_W_PRIORITY	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	AXI1_R_PRIORITY		—	—	—	—	—	—	AXI0_FIFO_TYPE_REG	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.91 DDR_CTL_88 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b26	Reserved	NA	R
b25, b24	AXI1_FIFO_TYPE_REG	Clock domain relativity between AXI port 1 and the controller. Should be set to 0.	R/W
b23 to b18	Reserved	NA	R
b17, b16	AXI1_W_PRIORITY	Priority of write commands from AXI port 1. 0 is the highest priority.	R/W
b15 to b10	Reserved	NA	R
b9, b8	AXI1_R_PRIORITY	Priority of read commands from AXI port 1. 0 is the highest priority.	R/W
b7 to b2	Reserved	NA	R
b1, b0	AXI0_FIFO_TYPE_REG	Clock domain relativity between AXI port 0 and the controller. Should be set to 0.	R/W

6.4.1.90 DDR_CTL_89 — DDR-Controller Status & Control 89

Address: 4000 D164h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	AXI3_R_PRIORITY		—	—	—	—	—	—	AXI2_FIFO_TYPE_REG	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	AXI2_W_PRIORITY		—	—	—	—	—	—	AXI2_R_PRIORITY	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.92 DDR_CTL_89 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b26	Reserved	NA	R
b25, b24	AXI3_R_PRIORITY	Priority of read commands from AXI port 3. 0 is the highest priority.	R/W
b23 to b18	Reserved	NA	R
b17, b16	AXI2_FIFO_TYPE_REG	Clock domain relativity between AXI port 2 and the controller. Should be set to 0.	R/W
b15 to b10	Reserved	NA	R
b9, b8	AXI2_W_PRIORITY	Priority of write commands from AXI port 2. 0 is the highest priority.	R/W
b7 to b2	Reserved	NA	R
b1, b0	AXI2_R_PRIORITY	Priority of read commands from AXI port 2. 0 is the highest priority.	R/W

6.4.1.91 DDR_CTL_90 — DDR-Controller Status & Control 90

Address: 4000 D168h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	PORT_ADDR_PROTECTION_EN	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	AXI3_FIFO_TYPE_REG		—	—	—	—	—	—	AXI3_W_PRIORITY	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.93 DDR_CTL_90 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	PORT_ADDR_PROTECTION_EN	Enable port address range protection logic and interrupt generation. Set to 1 to enable.	R/W
b23 to b16	Reserved	NA	R
b15 to b10	Reserved	NA	R
b9, b8	AXI3_FIFO_TYPE_REG	Clock domain relativity between AXI port 3 and the controller. Should be set to 0.	R/W
b7 to b2	Reserved	NA	R
b1, b0	AXI3_W_PRIORITY	Priority of write commands from AXI port 3. 0 is the highest priority.	R/W

6.4.1.92 DDR_CTL_[k] — Port0 Range[n] Start Address Setting Register (n = 0..15) (k = 91 + n × 2)

Address: 4000 D16Ch + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI0_START_ADDR_[n]	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI0_START_ADDR_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.94 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17 to b0	AXI0_START_ADDR_[n]	Start address of port 0 address range [n]	R/W

6.4.1.93 DDR_CTL_[k] — Port0 Range[n] End Address Setting Register (n = 0..15) (k = 92 + n × 2)

Address: 4000 D170h + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI0_END_ADDR_[n]	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI0_END_ADDR_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.95 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17 to b0	AXI0_END_ADDR_[n]	End address of port 0 address range [n]	R/W

6.4.1.94 DDR_CTL_[k] — Port1 Range[n] Start Address Setting Register (n = 0..15) (k = 123 + n × 2)

Address: 4000 D1ECh + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI1_START_ADDR_[n]	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI1_START_ADDR_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.96 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17 to b0	AXI1_START_ADDR_[n]	Start address of port 1 address range [n]	R/W

6.4.1.95 DDR_CTL_[k] — Port1 Range[n] End Address Setting Register (n = 0..15) (k = 124 + n × 2)

Address: 4000 D1F0h + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI1_END_ADDR_[n]	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI1_END_ADDR_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.97 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17 to b0	AXI1_END_ADDR_[n]	End address of port 1 address range [n]	R/W

6.4.1.96 DDR_CTL_[k] — Port2 Range[n] Start Address Setting Register (n = 0..15) (k = 155 + n × 2)

Address: 4000 D26Ch + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI2_START_ADDR_[n]	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI2_START_ADDR_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.98 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17 to b0	AXI2_START_ADDR_[n]	Start address of port 2 address range [n]	R/W

6.4.1.97 DDR_CTL_[k] — Port2 Range[n] End Address Setting Register (n = 0..15) (k = 156 + n × 2)

Address: 4000 D270h + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI2_END_ADDR_[n]	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI2_END_ADDR_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.99 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17 to b0	AXI2_END_ADDR_[n]	End address of port 2 address range [n]	R/W

6.4.1.98 DDR_CTL_[k] — Port3 Range[n] Start Address Setting Register (n = 0..15) (k = 187 + n × 2)

Address: 4000 D2ECh + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI3_START_ADDR_[n]	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI3_START_ADDR_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.100 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17 to b0	AXI3_START_ADDR_[n]	Start address of port 3 address range [n]	R/W

6.4.1.99 DDR_CTL_[k] — Port3 Range[n] End Address Setting Register (n = 0..14) (k = 188 + n × 2)

Address: 4000 D2F0h + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI3_END_ADDR_[n]	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI3_END_ADDR_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.101 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17 to b0	AXI3_END_ADDR_[n]	End address of port 3 address range [n]	R/W

6.4.1.100 DDR_CTL_218 — Port3 Range15 End Address Setting Register

Address: 4000 D368h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	AXI0_RANGE_PROT_BITS_0		—	—	—	—	—	—	AXI3_END_ADDR_15	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI3_END_ADDR_15															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.102 DDR_CTL_218 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b26	Reserved	NA	R
b25, b24	AXI0_RANGE_PROT_BITS_0	Allowed transaction types for port 0 address range 0. Set to 0 for privileged and secure only, set to 1 for secure (with or without privileged), set to 2 for privileged (with or without secure), or set to 3 for full access.	R/W
b23 to b18	Reserved	NA	R
b17 to b0	AXI3_END_ADDR_15	End address of port 3 address range 15.	R/W

6.4.1.101 DDR_CTL_[k] — Port0 Range[n] Protect Setting Register1 (n = 0..15) (k = 219 + n × 2)

Address: 4000 D36Ch + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	AXI0_RANGE_WID_CHECK_BITS_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI0_RANGE_RID_CHECK_BITS_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.103 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	AXI0_RANGE_WID_CHECK_BITS_[n]	ID check is not supported. Write 0xFFFF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W
b15 to b0	AXI0_RANGE_RID_CHECK_BITS_[n]	ID check is not supported. Write 0xFFFF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W

6.4.1.102 DDR_CTL_[k] — Port0 Range[n] Protect Setting Register2 (n = 0..14) (k = 220 + n × 2)

Address: 4000 D370h + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI0_RANGE_PROT_BITS_[n+1]	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	AXI0_RANGE_WID_CHECK_BITS_ID_LOOKUP_[n]				—	—	—	—	AXI0_RANGE_RID_CHECK_BITS_ID_LOOKUP_[n]			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.104 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17, b16	AXI0_RANGE_PROT_BITS_[n+1]	Allowed transaction types for port 0 address range [n+1]. Set to 0 for privileged and secure only, set to 1 for secure (with or without privileged), set to 2 for privileged (with or without secure), or set to 3 for full access.	R/W
b15 to b12	Reserved	NA	R
b11 to b8	AXI0_RANGE_WID_CHECK_BITS_ID_LOOKUP_[n]	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W
b7 to b4	Reserved	NA	R
b3 to b0	AXI0_RANGE_RID_CHECK_BITS_ID_LOOKUP_[n]	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W

6.4.1.103 DDR_CTL_250 — Port0 Range15 Protect Setting Register2

Address: 4000 D3E8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI1_RANGE_PROT_BITS_0	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	AXI0_RANGE_WID_CHECK_BITS_ID_LOOKUP_15				—	—	—	—	AXI0_RANGE_RID_CHECK_BITS_ID_LOOKUP_15			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.105 DDR_CTL_250 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17, b16	AXI1_RANGE_PROT_BITS_0	Allowed transaction types for port 1 address range 0. Set to 0 for privileged and secure only, set to 1 for secure (with or without privileged), set to 2 for privileged (with or without secure), or set to 3 for full access.	R/W
b15 to b12	Reserved	NA	R
b11 to b8	AXI0_RANGE_WID_CHECK_BITS_ID_LOOKUP_15	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W
b7 to b4	Reserved	NA	R
b3 to b0	AXI0_RANGE_RID_CHECK_BITS_ID_LOOKUP_15	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W

6.4.1.104 DDR_CTL_[k] — Port1 Range[n] Protect Setting Register1 (n = 0..15) (k = 251 + n × 2)

Address: 4000 D3ECh + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	AXI1_RANGE_WID_CHECK_BITS_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI1_RANGE_RID_CHECK_BITS_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.106 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	AXI1_RANGE_WID_CHECK_BITS_[n]	ID check is not supported. Write 0xFFFF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W
b15 to b0	AXI1_RANGE_RID_CHECK_BITS_[n]	ID check is not supported. Write 0xFFFF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W

6.4.1.105 DDR_CTL_[k] — Port1 Range[n] Protect Setting Register2 (n = 0..14) (k = 252 + n × 2)

Address: 4000 D3F0h + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI1_RANGE_PROT_BITS_[n+1]	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	AXI1_RANGE_WID_CHECK_BITS_ID_LOOKUP_[n]				—	—	—	—	AXI1_RANGE_RID_CHECK_BITS_ID_LOOKUP_[n]			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.107 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17, b16	AXI1_RANGE_PROT_BITS_[n+1]	Allowed transaction types for port 1 address range [n+1]. Set to 0 for privileged and secure only, set to 1 for secure (with or without privileged), set to 2 for privileged (with or without secure), or set to 3 for full access.	R/W
b15 to b12	Reserved	NA	R
b11 to b8	AXI1_RANGE_WID_CHECK_BITS_ID_LOOKUP_[n]	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W
b7 to b4	Reserved	NA	R
b3 to b0	AXI1_RANGE_RID_CHECK_BITS_ID_LOOKUP_[n]	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W

6.4.1.106 DDR_CTL_282 — Port1 Range15 Protect Setting Register2

Address: 4000 D468h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI2_RANGE_PROT_BITS_0	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	AXI1_RANGE_WID_CHECK_BITS_ID_LOOKUP_15				—	—	—	—	AXI1_RANGE_RID_CHECK_BITS_ID_LOOKUP_15			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.108 DDR_CTL_282 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17, b16	AXI2_RANGE_PROT_BITS_0	Allowed transaction types for port 2 address range 0. Set to 0 for privileged and secure only, set to 1 for secure (with or without privileged), set to 2 for privileged (with or without secure), or set to 3 for full access.	R/W
b15 to b12	Reserved	NA	R
b11 to b8	AXI1_RANGE_WID_CHECK_BITS_ID_LOOKUP_15	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W
b7 to b4	Reserved	NA	R
b3 to b0	AXI1_RANGE_RID_CHECK_BITS_ID_LOOKUP_15	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W

6.4.1.107 DDR_CTL_[k] — Port2 Range[n] Protect Setting Register1 (n = 0..15) (k = 283 + n × 2)

Address: 4000 D46Ch + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	AXI2_RANGE_WID_CHECK_BITS_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI2_RANGE_RID_CHECK_BITS_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.109 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	AXI2_RANGE_WID_CHECK_BITS_[n]	ID check is not supported. Write 0xFFFF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W
b15 to b0	AXI2_RANGE_RID_CHECK_BITS_[n]	ID check is not supported. Write 0xFFFF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W

6.4.1.108 DDR_CTL_[k] — Port2 Range[n] Protect Setting Register2 (n = 0..14) (k = 284 +n × 2)

Address: 4000 D470h + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI2_RANGE_PROT_BITS_[n+1]	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	AXI2_RANGE_WID_CHECK_BITS_ID_LOOKUP_[n]				—	—	—	—	AXI2_RANGE_RID_CHECK_BITS_ID_LOOKUP_[n]			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.110 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17, b16	AXI2_RANGE_PROT_BITS_[n+1]	Allowed transaction types for port 2 address range [n+1]. Set to 0 for privileged and secure only, set to 1 for secure (with or without privileged), set to 2 for privileged (with or without secure), or set to 3 for full access.	R/W
b15 to b12	Reserved	NA	R
b11 to b8	AXI2_RANGE_WID_CHECK_BITS_ID_LOOKUP_[n]	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W
b7 to b4	Reserved	NA	R
b3 to b0	AXI2_RANGE_RID_CHECK_BITS_ID_LOOKUP_[n]	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W

6.4.1.109 DDR_CTL_314 — Port2 Range15 Protect Setting Register2

Address: 4000 D4E8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI3_RANGE_PROT_BITS_0	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	AXI2_RANGE_WID_CHECK_BITS_ID_LOOKUP_15				—	—	—	—	AXI2_RANGE_RID_CHECK_BITS_ID_LOOKUP_15			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.111 DDR_CTL_314 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17, b16	AXI3_RANGE_PROT_BITS_0	Allowed transaction types for port 3 address range 0. Set to 0 for privileged and secure only, set to 1 for secure (with or without privileged), set to 2 for privileged (with or without secure), or set to 3 for full access.	R/W
b15 to b12	Reserved	NA	R
b11 to b8	AXI2_RANGE_WID_CHECK_BITS_ID_LOOKUP_15	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W
b7 to b4	Reserved	NA	R
b3 to b0	AXI2_RANGE_RID_CHECK_BITS_ID_LOOKUP_15	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W

6.4.1.110 DDR_CTL_[k] — Port3 Range[n] Protect Setting Register1 (n = 0..15) (k = 315 + n × 2)

Address: 4000 D4ECh + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	AXI3_RANGE_WID_CHECK_BITS_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	AXI3_RANGE_RID_CHECK_BITS_[n]															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.112 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	AXI3_RANGE_WID_CHECK_BITS_[n]	ID check is not supported. Write 0xFFFF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W
b15 to b0	AXI3_RANGE_RID_CHECK_BITS_[n]	ID check is not supported. Write 0xFFFF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W

6.4.1.111 DDR_CTL_[k] — Port3 Range[n] Protect Setting Register2 (n = 0..14) (k = 316 + n × 2)

Address: 4000 D4F0h + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AXI3_RANGE_PROT_BITS_[n+1]	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	AXI3_RANGE_WID_CHECK_BITS_ID_LOOKUP_[n]				—	—	—	—	AXI3_RANGE_RID_CHECK_BITS_ID_LOOKUP_[n]			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.113 DDR_CTL_[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b18	Reserved	NA	R
b17, b16	AXI3_RANGE_PROT_BITS_[n+1]	Allowed transaction types for port 3 address range [n+1]. Set to 0 for privileged and secure only, set to 1 for secure (with or without privileged), set to 2 for privileged (with or without secure), or set to 3 for full access.	R/W
b15 to b12	Reserved	NA	R
b11 to b8	AXI3_RANGE_WID_CHECK_BITS_ID_LOOKUP_[n]	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W
b7 to b4	Reserved	NA	R
b3 to b0	AXI3_RANGE_RID_CHECK_BITS_ID_LOOKUP_[n]	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range[n] of this port.	R/W

6.4.1.112 DDR_CTL_346 — Port3 Range15 Protect Setting Register2

Address: 4000 D568h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	AXI0_BDW							—	—	—	—	—	ARB_CMD_Q_THRESHOLD		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	AXI3_RANGE_WID_CHECK_BITS_ID_LOOKUP_15				—	—	—	—	AXI3_RANGE_RID_CHECK_BITS_ID_LOOKUP_15			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.114 DDR_CTL_346 Register Contents

Bit Position	Bit Name	Function	R/W
b31	Reserved	NA	R
b30 to b24	AXI0_BDW	Maximum bandwidth percentage for port 0.	R/W
b23 to b19	Reserved	NA	R
b18 to b16	ARB_CMD_Q_THRESHOLD	Threshold for command queue fullness related to overflow.	R/W
b15 to b12	Reserved	NA	R
b11 to b8	AXI3_RANGE_WID_CHECK_BITS_ID_LOOKUP_15	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range15 of this port.	R/W
b7 to b4	Reserved	NA	R
b3 to b0	AXI3_RANGE_RID_CHECK_BITS_ID_LOOKUP_15	ID check is not supported. Write 0xF when applying PORT_ADDR_PROTECTION_EN to Range15 of this port.	R/W

6.4.1.113 DDR_CTL_347 — DDR-Controller Status & Control 347

Address: 4000 D56Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	AXI1_BDW_OVFLOW	—	AXI1_BDW						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	AXI0_CURRENT_BDW							—	—	—	—	—	—	—	AXI0_BDW_OVFLOW
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.115 DDR_CTL_347 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	AXI1_BDW_OVFLOW	Port 1 behavior when bandwidth maximized. Set to 1 to allow overflow.	R/W
b23	Reserved	NA	R
b22 to b16	AXI1_BDW	Maximum bandwidth percentage for port 1.	R/W
b15	Reserved	NA	R
b14 to b8	AXI0_CURRENT_BDW	Current bandwidth usage percentage for port 0.	R
b7 to b1	Reserved	NA	R
b0	AXI0_BDW_OVFLOW	Port 0 behavior when bandwidth maximized. Set to 1 to allow overflow.	R/W

6.4.1.114 DDR_CTL_348 — DDR-Controller Status & Control 348

Address: 4000 D570h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	AXI2_CURRENT_BDW							—	—	—	—	—	—	—	AXI2_BDW_OVFLOW
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	AXI2_BDW							—	AXI1_CURRENT_BDW						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.116 DDR_CTL_348 Register Contents

Bit Position	Bit Name	Function	R/W
b31	Reserved	NA	R
b30 to b24	AXI2_CURRENT_BDW	Current bandwidth usage percentage for port 2.	R
b23 to b17	Reserved	NA	R
b16	AXI2_BDW_OVFLOW	Port 2 behavior when bandwidth maximized. Set to 1 to allow overflow.	R/W
b15	Reserved	NA	R
b14 to b8	AXI2_BDW	Maximum bandwidth percentage for port 2.	R/W
b7	Reserved	NA	R
b6 to b0	AXI1_CURRENT_BDW	Current bandwidth usage percentage for port 1.	R

6.4.1.115 DDR_CTL_349 — DDR-Controller Status & Control 349

Address: 4000 D574h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	CKE_STATUS		—	AXI3_CURRENT_BDW						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	AXI3_BDW_OVFLOW	—	AXI3_BDW						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.117 DDR_CTL_349 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b26	Reserved	NA	R
b25, b24	CKE_STATUS	The value of cke_status internal signal. Bit [0] is for CS0 and bit [1] is for CS1. 0: The memory is in power-down or self-refresh. 1: The memory is active.	R
b23	Reserved	NA	R
b22 to b16	AXI3_CURRENT_BDW	Current bandwidth usage percentage for port 3.	R
b15 to b9	Reserved	NA	R
b8	AXI3_BDW_OVFLOW	Port 3 behavior when bandwidth maximized. Set to 1 to allow overflow.	R/W
b7	Reserved	NA	R
b6 to b0	AXI3_BDW	Maximum bandwidth percentage for port 3.	R/W

6.4.1.116 DDR_CTL_350 — DDR-Controller Status & Control 350

Address: 4000 D578h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DLL_RST_ADJ_DLY								DLL_RST_DELAY							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DLL_RST_DELAY								—	—	—	—	—	—	—	MEM_RST_VALID
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.118 DDR_CTL_350 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	DLL_RST_ADJ_DLY	This parameter is invalid.	R/W
b23 to b8	DLL_RST_DELAY	This parameter is invalid.	R/W
b7 to b1	Reserved	NA	R
b0	MEM_RST_VALID	The value of mem_rst_valid internal signal. If this bit is set, the controller is able to regain control of the memory reset and cke signals.	R

6.4.1.117 DDR_CTL_351 — DDR-Controller Status & Control 351

Address: 4000 D57Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	TDFI_RDDATA_EN						—	—	TDFI_PHY_RDLAT					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	UPDATE_ERROR_STATUS							—	—	TDFI_PHY_WRLAT					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.119 DDR_CTL_351 Register Contents

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved	NA	R
b29 to b24	TDFI_RDDATA_EN	Holds the calculated DFI tRDDATA_EN timing parameter (in memory clock cycles), the maximum cycles between a read command and a dfi_rddata_en assertion.	R
b23, b22	Reserved	NA	R
b21 to b16	TDFI_PHY_RDLAT	Defines the DFI tPHY_RDLAT timing parameter (in memory clock cycles), the maximum cycles between a dfi_rddata_en assertion and a dfi_rddata_valid assertion.	R/W
b15	Reserved	NA	R
b14 to b8	UPDATE_ERROR_STATUS	Identifies the source of any DFI memory-controller-initiated or PHY-initiated update errors. Value of 1 indicates a timing violation of the associated timing parameter.	R
b7, b6	Reserved	NA	R
b5 to b0	TDFI_PHY_WRLAT	Holds the calculated DFI tPHY_WRLAT timing parameter (in memory clock cycles), the maximum cycles between a write command and a dfi_wrdata_en assertion.	R

6.4.1.118 DDR_CTL_352 — DDR-Controller Status & Control 352

Address: 4000 D580h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	TDFI_CTRLUPD_MAX													
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	TDFI_CTRLUPD_MIN				—	—	—	—	—	—	DRAM_CLK_DISABLE	
Value after reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Table 6.120 DDR_CTL_352 Register Contents

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved	NA	R
b29 to b16	TDFI_CTRLUPD_MAX	Defines the DFI tCTRLUPD_MAX timing parameter (in DFICLK clock cycles), the maximum cycles that dfi_ctrlupd_req can be asserted. If programmed to a non-zero, a timing violation will cause an interrupt and bit (1) set in the UPDATE_ERROR_STATUS parameter.	R/W
b15 to b12	Reserved	NA	R
b11 to b8	TDFI_CTRLUPD_MIN	Reports the DFI tCTRLUPD_MIN timing parameter (in DFICLK clock cycles), the minimum cycles that dfi_ctrlupd_req must be asserted.	R
b7 to b2	Reserved	NA	R
b1, b0	DRAM_CLK_DISABLE	Set value for the dfi_dram_clk_disable signal. Bit (0) controls cs0, bit (1) controls cs1, etc. Set each bit to 1 to disable.	R/W

6.4.1.119 DDR_CTL_353 — DDR-Controller Status & Control 353

Address: 4000 D584h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TDFI_PHYUPD_TYPE1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TDFI_PHYUPD_TYPE0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.121 DDR_CTL_353 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	TDFI_PHYUPD_TYPE1	Defines the DFI tPHYUPD_TYPE1 timing parameter (in DFICLK clock cycles), the maximum cycles that dfi_phyupd_req can assert after dfi_phyupd_ack for dfi_phyupd_type 1. If programmed to a non-zero, a timing violation will cause an interrupt and bit (3) set in the UPDATE_ERROR_STATUS parameter.	R/W
b15 to b0	TDFI_PHYUPD_TYPE0	Defines the DFI tPHYUPD_TYPE0 timing parameter (in DFICLK clock cycles), the maximum cycles that dfi_phyupd_req can assert after dfi_phyupd_ack for dfi_phyupd_type 0. If programmed to a non-zero, a timing violation will cause an interrupt and bit (2) set in the UPDATE_ERROR_STATUS parameter.	R/W

6.4.1.120 DDR_CTL_354 — DDR-Controller Status & Control 354

Address: 4000 D588h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TDFI_PHYUPD_TYPE3															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TDFI_PHYUPD_TYPE2															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.122 DDR_CTL_354 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	TDFI_PHYUPD_TYPE3	Defines the DFI tPHYUPD_TYPE3 timing parameter (in DFICLK clock cycles), the maximum cycles that dfi_phyupd_req can assert after dfi_phyupd_ack for dfi_phyupd_type 3. If programmed to a non-zero, a timing violation will cause an interrupt and bit (5) set in the UPDATE_ERROR_STATUS parameter.	R/W
b15 to b0	TDFI_PHYUPD_TYPE2	Defines the DFI tPHYUPD_TYPE2 timing parameter (in DFICLK clock cycles), the maximum cycles that dfi_phyupd_req can assert after dfi_phyupd_ack for dfi_phyupd_type 2. If programmed to a non-zero, a timing violation will cause an interrupt and bit (4) set in the UPDATE_ERROR_STATUS parameter.	R/W

6.4.1.121 DDR_CTL_355 — DDR-Controller Status & Control 355

Address: 4000 D58Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	TDFI_PHYUPD_RESP													
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.123 DDR_CTL_355 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved	NA	R
b15, b14	Reserved	NA	R
b13 to b0	TDFI_PHYUPD_RESP	Defines the DFI tPHYUPD_RESP timing parameter (in DFICLK clock cycles), the maximum cycles between a dfi_phyupd_req assertion and a dfi_phyupd_ack assertion. If programmed to a non-zero, a timing violation will cause an interrupt and bit (6) set in the UPDATE_ERROR_STATUS parameter.	R/W

6.4.1.122 DDR_CTL_356 — DDR-Controller Status & Control 356

Address: 4000 D590h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TDFI_CTRLUPD_INTERVAL															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TDFI_CTRLUPD_INTERVAL															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.124 DDR_CTL_356 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	TDFI_CTRLUPD_INTERVAL	Defines the DFI tCTRLUPD_INTERVAL timing parameter (in DFICLK clock cycles), the maximum cycles between dfi_ctrlupd_req assertions. If programmed to a non-zero, a timing violation will cause an interrupt and bit (0) set in the UPDATE_ERROR_STATUS parameter.	R/W

6.4.1.123 DDR_CTL_357 — DDR-Controller Status & Control 357

Address: 4000 D594h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	TDFI_DRAM_CLK_DISABLE				—	—	—	—	TDFI_CTRL_DELAY			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	WRLAT_ADJ						—	—	RDLAT_ADJ					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.125 DDR_CTL_357 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b28	Reserved	NA	R
b27 to b24	TDFI_DRAM_CLK_DISABLE	Defines the DFI tDRAM_CLK_DISABLE timing parameter (in DFICLK clock cycles), the delay between a dfi_dram_clock_disable assertion and the memory clock disable.	R/W
b23 to b20	Reserved	NA	R
b19 to b16	TDFI_CTRL_DELAY	Defines the DFI tCTRL_DELAY timing parameter (in DFICLK clock cycles), the delay between a DFI command change and a memory command.	R/W
b15, b14	Reserved	NA	R
b13 to b8	WRLAT_ADJ	Adjustment value for PHY write timing (memory clock cycles)	R/W
b7, b6	Reserved	NA	R
b5 to b0	RDLAT_ADJ	Adjustment value for PHY read timing (memory clock cycles)	R/W

6.4.1.124 DDR_CTL_358 — DDR-Controller Status & Control 358

Address: 4000 D598h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	TDFI_WRLVL_WW									
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TDFI_WRLVL_EN								—	—	—	—	TDFI_DRAM_CLK_ENABLE			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.126 DDR_CTL_358 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b26	Reserved	NA	R
b25 to b16	TDFI_WRLVL_WW	NA. The bit should be kept the initial value.	R/W
b15 to b8	TDFI_WRLVL_EN	NA. The bit should be kept the initial value.	R/W
b7 to b4	Reserved	NA	R
b3 to b0	TDFI_DRAM_CLK_ENABLE	Defines the DFI tDRAM_CLK_ENABLE timing parameter (in DFICLK clock cycles), the delay between a dfi_dram_clk_disable de-assertion and the memory clock enable.	R/W

6.4.1.125 DDR_CTL_359 — DDR-Controller Status & Control 359

Address: 4000 D59Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TDFI_WRLVL_RESP															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TDFI_WRLVL_RESP															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.127 DDR_CTL_359 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	TDFI_WRLVL_RESP	NA. The bit should be kept the initial value.	R/W

6.4.1.126 DDR_CTL_360 — DDR-Controller Status & Control 360

Address: 4000 D5A0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TDFI_WRLVL_MAX															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TDFI_WRLVL_MAX															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.128 DDR_CTL_360 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	TDFI_WRLVL_MAX	NA. The bit should be kept the initial value.	R/W

6.4.1.127 DDR_CTL_361 — DDR-Controller Status & Control 361

Address: 4000 D5A4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TDFI_WRLVL_RESPLAT								TDFI_WRLVL_DLL							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DFI_WRLVL_MAX_DELAY															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.129 DDR_CTL_361 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	TDFI_WRLVL_RESPLAT	NA. The bit should be kept the initial value.	R/W
b23 to b16	TDFI_WRLVL_DLL	NA. The bit should be kept the initial value.	R/W
b15 to b0	DFI_WRLVL_MAX_DELAY	NA. The bit should be kept the initial value.	R/W

6.4.1.128 DDR_CTL_362 — DDR-Controller Status & Control 362

Address: 4000 D5A8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TDFI_RDLVL_LOAD								TDFI_RDLVL_DLL							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TDFI_RDLVL_EN								TDFI_WRLVL_LOAD							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.130 DDR_CTL_362 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	TDFI_RDLVL_LOAD	NA. The bit should be kept the initial value.	R/W
b23 to b16	TDFI_RDLVL_DLL	NA. The bit should be kept the initial value.	R/W
b15 to b8	TDFI_RDLVL_EN	NA. The bit should be kept the initial value.	R/W
b7 to b0	TDFI_WRLVL_LOAD	NA. The bit should be kept the initial value.	R/W

6.4.1.129 DDR_CTL_363 — DDR-Controller Status & Control 363

Address: 4000 D5ACh

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	RDLVL_MAX_DELAY							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_MAX_DELAY								TDFI_RDLVL_RESPLAT							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.131 DDR_CTL_363 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b8	RDLVL_MAX_DELAY	NA. The bit should be kept the initial value.	R/W
b7 to b0	TDFI_RDLVL_RESPLAT	NA. The bit should be kept the initial value.	R/W

6.4.1.130 DDR_CTL_364 — DDR-Controller Status & Control 364

Address: 4000 D5B0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	TDFI_RDLVL_RR									
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_GATE_MAX_DELAY															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.132 DDR_CTL_364 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b26	Reserved	NA	R
b25 to b16	TDFI_RDLVL_RR	NA. The bit should be kept the initial value.	R/W
b15 to b0	RDLVL_GATE_MAX_DELAY	NA. The bit should be kept the initial value.	R/W

6.4.1.131 DDR_CTL_365 — DDR-Controller Status & Control 365

Address: 4000 D5B4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TDFI_RDLVL_RESP															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TDFI_RDLVL_RESP															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.133 DDR_CTL_365 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	TDFI_RDLVL_RESP	NA. The bit should be kept the initial value.	R/W

6.4.1.132 DDR_CTL_366 — DDR-Controller Status & Control 366

Address: 4000 D5B8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	RDLVL_RESP_MASK			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_RESP_MASK															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.134 DDR_CTL_366 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	NA	R
b23 to b20	Reserved	NA	R
b19 to b0	RDLVL_RESP_MASK	NA. The bit should be kept the initial value.	R/W

6.4.1.133 DDR_CTL_367 — DDR-Controller Status & Control 367

Address: 4000 D5BCh

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	RDLVL_EN	—	—	—	—	RDLVL_GATE_RESP_MASK			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_GATE_RESP_MASK															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.135 DDR_CTL_367 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	RDLVL_EN	NA. The bit should be kept the initial value.	R/W
b23 to b20	Reserved	NA	R
b19 to b0	RDLVL_GATE_RESP_MASK	NA. The bit should be kept the initial value.	R/W

6.4.1.134 DDR_CTL_368 — DDR-Controller Status & Control 368

Address: 4000 D5C0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	RDLVL_GATE_PREAMBLE_CHECK_EN	—	—	—	—	—	—	—	RDLVL_GATE_EN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.136 DDR_CTL_368 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved	NA	R
b15 to b9	Reserved	NA	R
b8	RDLVL_GATE_PREAMBLE_CHECK_EN	NA. The bit should be kept the initial value.	R/W
b7 to b1	Reserved	NA	R
b0	RDLVL_GATE_EN	NA. The bit should be kept the initial value.	R/W

6.4.1.135 DDR_CTL_369 — DDR-Controller Status & Control 369

Address: 4000 D5C4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	TDFI_RDLVL_MAX															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	TDFI_RDLVL_MAX															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.137 DDR_CTL_369 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	TDFI_RDLVL_MAX	NA. The bit should be kept the initial value.	R/W

6.4.1.136 DDR_CTL_370 — DDR-Controller Status & Control 370

Address: 4000 D5C8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	RDLVL_ERROR_STATUS													
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	RDLVL_GATE_DQ_0_COUNT				—	—	—	—	RDLVL_DQ_0_COUNT			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.138 DDR_CTL_370 Register Contents

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved	NA	R
b29 to b16	RDLVL_ERROR_STATUS	NA	R
b15 to b12	Reserved	NA	R
b11 to b8	RDLVL_GATE_DQ_0_COUNT	NA. The bit should be kept the initial value.	R/W
b7 to b4	Reserved	NA	R
b3 to b0	RDLVL_DQ_0_COUNT	NA. The bit should be kept the initial value.	R/W

6.4.1.137 DDR_CTL_371 — DDR-Controller Status & Control 371

Address: 4000 D5CCh

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	RDLVL_GATE_INTERVAL															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RDLVL_INTERVAL															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.139 DDR_CTL_371 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	RDLVL_GATE_INTERVAL	NA. The bit should be kept the initial value.	R/W
b15 to b0	RDLVL_INTERVAL	NA. The bit should be kept the initial value.	R/W

6.4.1.138 DDR_CTL_372 — DDR-Controller Status & Control 372

Address: 4000 D5D0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	OPTIMAL_RMODW_EN	MEMCD_RMODW_FIFO_PTR_WIDTH							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	MEMCD_RMODW_FIFO_DEPTH								—	—	—	—	—	TDFI_PHY_WRDATA		
Value after reset	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0

Table 6.140 DDR_CTL_372 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	OPTIMAL_RMODW_EN	NA. It should be kept the initial value.	R/W
b23 to b16	MEMCD_RMODW_FIFO_PTR_WIDTH	Reports the width of the controller read/modify/write FIFO pointer.	R
b15 to b8	MEMCD_RMODW_FIFO_DEPTH	Reports the depth of the controller read/modify/write FIFO.	R
b7 to b3	Reserved	NA	R
b2 to b0	TDFI_PHY_WRDATA	Defines the DFI tPHY_WRDATA timing parameter (in memory clock cycles), the maximum cycles between a dfi_wrdata_en assertion and a dfi_wrdata signal.	R/W

6.4.1.139 DDR_CTL_373 — DDR-Controller Status & Control 373

Address: 4000 D5D4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.141 DDR_CTL_373 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	Reserved	Reserved for future use. It should be kept the initial value.	R/W
b23 to b21	Reserved	NA	R
b20 to b16	Reserved	Reserved for future use. It should be kept the initial value.	R/W
b15 to b9	Reserved	NA	R
b8	Reserved	Reserved for future use. It should be kept the initial value.	R/W
b7 to b1	Reserved	NA	R
b0	Reserved	Reserved for future use. It should be set to 1.	R/W

6.4.1.140 DDR_CTL_374 — DDR-Controller Status & Control 374

Address: 4000 D5D8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	AXI3_ALL_STROBES_USED_ENABLE	—	—	—	—	—	—	—	AXI2_ALL_STROBES_USED_ENABLE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	AXI1_ALL_STROBES_USED_ENABLE	—	—	—	—	—	—	—	AXI0_ALL_STROBES_USED_ENABLE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.142 DDR_CTL_374 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	NA	R
b24	AXI3_ALL_STROBES_USED_ENABLE	Enables use of the AWALLSTRB signal for AXI port 3. Set to 1 to enable.	R/W
b23 to b17	Reserved	NA	R
b16	AXI2_ALL_STROBES_USED_ENABLE	Enables use of the AWALLSTRB signal for AXI port 2. Set to 1 to enable.	R/W
b15 to b9	Reserved	NA	R
b8	AXI1_ALL_STROBES_USED_ENABLE	Enables use of the AWALLSTRB signal for AXI port 1. Set to 1 to enable.	R/W
b7 to b1	Reserved	NA	R
b0	AXI0_ALL_STROBES_USED_ENABLE	Enables use of the AWALLSTRB signal for AXI port 0. Set to 1 to enable.	R/W

6.4.2 DDR PHY Register Description

6.4.2.1 FUNCCTRL — Function Control Register

Address: 4000 E000h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	MASKSDLOFS						
Value after reset	X	X	X	X	X	X	X	X	X	0	0	1	1	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	IFSEL		—	—	—	—	—	—	—	FUNCRSTB
Value after reset	X	X	X	X	X	X	0	1	X	X	X	X	X	X	X	0

Table 6.143 FUNCCTRL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b23	Reserved	NA	R
b22 to b16	MASKSDLOFS	Hi-Z mask circuit setting register.	R/W
b15 to b10	Reserved	NA	R
b9, b8	IFSEL	Voltage setting of interface. 2’b00: DVDDQ 1.8 V 2’b01: DVDDQ 1.5 V 2’b10: (Prohibited) 2’b11: (Prohibited)	R/W
b7 to b1	Reserved	NA	R
b0	FUNCRSTB	Reset setting in functional block. When resetting it, it is the same state as “PWRCTRL_DDRC.RSTN_B = 0” excluding the register. 0: Reset state 1: State of operation	R/W

6.4.2.2 DLLCTRL — MDLL Control Register

Address: 4000 E004h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	ASDLLOCK	MDLLOCK	MSATFG	—	MDACNTM	SDLYCTRL	DACNTUPD	—	—	MDACNT	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	MDACNT								—	—	HSLMODE	MSATMODE	DDMODE	MFSL		MDLLSTBY
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 6.144 DLLCTRL Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b27	Reserved	NA	R
b26	ASDLLOCK	Lock signal of Mater DLL. 1'b0: Unlock 1'b1: Lock	R
b25	MDLLOCK	Lock signal of Mater DLL. Do not use this bit, please use ASDLLOCK bit.	R
b24	MSATFG	Master DLL Saturation Flag 1'b0: No Saturate State (Delay Line control within the range) 1'b1: Saturate State (Maximum/Minimum control of Delay Line)	R
b23	Reserved	NA	R
b22	MDACNTM	Write of control code of Master DLL. When "1" is written, the control code of Master DLL is memorized in MDACNT. "0" is returned when reading it. It has high priority than a value written directly in MDACNT. (don't write simultaneously with bit 20)	W
b21	SDLYCTRL	Control selection of Slave DLL 1'b0: Controlled by Master DLL (default) 1'b1: Controlled by MDACNT	R/W
b20	DACNTUPD	Update of control code of Slave Delay. In case of SDLYCTRL = "1", the value of MDACNT is reflected in Slave Delay. When "1" is written, the control code is reflected in Slave Delay. "0" is returned when reading it. It is necessary to update it in case of the control by MDACNT. (don't write simultaneously with bit 22)	W
b19, b18	Reserved	NA	R
b17 to b8	MDACNT	Master DLL code monitor signal. When "1" is written in MDACNTM, DACNT [9:0] from Master DLL is memorized.	R/W
b7, b6	Reserved	NA	R
b5	HSLMODE	High-speed Lock Up mode setting. (reserved: Unimplemented) 1'b0: Detailed Lock Up mode (default) 1'b1: High-speed Lock Up mode (prohibited)	R/W
b4	MSATMODE	Saturate mode setting. 1'b0: OFF (self-reset mode) 1'b1: ON (Saturate mode: The delay control value stops at max/min value).	R/W
b3	DDMODE	Double Delay mode setting 1'b0: DDR3 1'b1: DDR2	R/W

Table 6.144 DLLCTRL Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b2, b1	MFSL	Frequency band setting. It should be set to 2'b10.	R/W
b0	MDLLSTBY	Master DLLSTBY setting After the frequency is changed, it is necessary to execute reset. 1'b0: Operation usually 1'b1: Reset (default)	R/W

6.4.2.3 ZQCALCTRL — ZQ Calibration Control Register

Address: 4000 E008h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	ZQCALRUN	ZQCALEND	ZQCALGAP	ZQCALPC			ZQCALPF				ZQCALNC			ZQCALNF		
Value after reset	0	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ZQCALNF	ZQCALINIT	—	—	—	—	ZQCALFREQ		ZQCALITVL				—	ZQCALMODE	ZQCALSTRV	ZQCALRSTB
Value after reset	X	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0

Table 6.145 ZQCALCTRL Register Contents

Bit Position	Bit Name	Function	R/W
b31	ZQCALRUN	Signal that indicates the state of ZQ calibration 1'b0: State of stop (default) 1'b1: The calibration is being executed.	R
b30	ZQCALEND	Signal that indicates the state of ZQ calibration end 1'b0: Calibration unfinished (default) 1'b1: Calibration end	R
b29	ZQCALGAP	Signal that indicates there is difference between ZQ calibration result and control code 1'b0: No difference (default) 1'b1: There is a difference.	R
b28 to b26	ZQCALPC	Pch calibration rough adjustment code output	R
b25 to b22	ZQCALPF	Pch calibration fine-tuning code output	R
b21 to b19	ZQCALNC	Nch calibration rough adjustment code output	R
b18 to b15	ZQCALNF	Nch calibration fine-tuning code output	R
b14	ZQCALINIT	ZQ calibration initialization end output 1'b0: Calibration unexecution (default) 1'b1: Calibration initialization end	R
b13 to b10	Reserved	NA	R
b9, b8	ZQCALFREQ	Setting sampling intervals of ZQ calibration. It should be set to 2'b01.	R/W
b7 to b4	ZQCALITVL	Setting ZQ calibration execution intervals 4'h0: Finished at one time. 2^ (ZQCALITVL+16) DFICLK clock cycle. The setting range is 4'h0 to 4'hA. Other setting is prohibited.	R/W
b3	Reserved	NA	R
b2	ZQCALMODE	ZQ calibration initial code setting 1'b0: No termination (default) 1'b1: Termination is used.	R/W
b1	ZQCALSTRV	ZQ calibration initial value setting 1'b0: Center value 1'b1: Last result (default)	R/W
b0	ZQCALRSTB	ZQ calibration circuit reset setting 1'b0: Reset (default) 1'b1: Reset release	R/W

6.4.2.4 ZQODTCTRL — ZQODT Control Register

Address: 4000 E00Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	CAPHASE		WRFIFOEN	FIFORPINIT		ZQDATA				ZQCK				ZQCMDAD		
Value after reset	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ZQCMDAD	—	SRDQ		SRCK		SRCMDAD		—	PHYODT		PHYODTEN	—	—	DRAMIF	
Value after reset	1	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1

Table 6.146 ZQODTCTRL Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31, b30	CAPHASE	Command/Address Output Phase setting The output phase of command/Address to DDR memory clock is set. 2'b10: 2/4 tCK setup else: Reserved (Prohibited)	R/W
b29	WRFIFOEN	I/F FIFO mode setting 1'b0: Prohibited 1'b1: FIFO use (default)	R/W
b28, b27	FIFORPINIT	I/F FIFO read pointer initializing. Setting to absorb the clock phase difference in DFICLK and PHY. 2'b01: ±1 DFICLK clock cycle else: Reserved	R/W
b26 to b23	ZQDATA	PHY Driver Impedance setting for Data (DQ,DM,DQS). It becomes ZDQ/{ZQDATA}. Range that can be set DDR3: 3 to 6 DDR2: 3 to 7 else: Reserved default=5 ZDQ is decided by ZQ automatic calibration and external resistance (DDR2 = 150Ω, DDR3 = 120Ω). The target value is DDR2:ZDQ = 300Ω, and DDR3:ZDQ = 240Ω.	R/W
b22 to b19	ZQCK	PHY Driver Impedance setting for CK. It becomes ZDQ/{ZQCK}. Range that can be set DDR3: 3 to 8 DDR2: 3 to 10 else: Reserved default = 5	R/W
b18 to b15	ZQCMDAD	PHY Driver Impedance setting for Command/Address. It becomes ZDQ/{ZQCMDAD}. Range that can be set DDR3: 3 to 8 DDR2: 3 to 10 else: Reserved default = 5	R/W
b14	Reserved	NA	R

Table 6.146 ZQODTCTRL Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b13, b12	SRDQ	Slew Rate setting for Data and DQS 2'b00: High (default) 2'b11: Low (prohibited) 2'b01: DQS signal mask circuit stop 2'b10: DQS signal mask circuit stop (prohibited)	R/W
b11, b10	SRCK	Slew Rate setting for CK The setting method is same as SRDQ.	R/W
b9, b8	SRCMDAD	Slew Rate setting for Command/Address The setting method is same as SRDQ.	R/W
b7	Reserved	NA	R
b6, b5	PHYODT	PHY ODT resistance setting 2'b00: OFF (Prohibited) 2'b01: Rtt_RD/1 2'b10: Rtt_RD/2 (default) 2'b11: Rtt_RD/3 Rtt_RD is determined by ZQ auto calibration and external resistance (DDR2 = 150Ω, DDR3 = 120Ω). The target value is DDR2:Rtt_RD = 150Ω, and DDR3:Rtt_RD = 120Ω.	R/W
b4	PHYODTEN	PHY ODT use setting 1'b0: ODT unused (prohibited) 1'b1: ODT use (default)	R/W
b3, b2	Reserved	NA	R
b1, b0	DRAMIF	DRAM I/F setting 2'b00: DDR2 2'b01: DDR3 (default) 2'b10: Prohibited 2'b11: Prohibited	R/W

6.4.2.5 RDCTRL — Read Control Register

Address: 4000 E010h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	PHYODTONT				PHYODTOFT				PDQODTONT				PDQODTOFT			
Value after reset	1	1	1	0	0	1	1	1	0	1	0	0	0	1	1	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	PHYBENONT				PHYBENOFT				PHYIENONT				PHYIENOFT			
Value after reset	1	1	1	0	0	1	1	1	0	1	0	0	0	1	1	0

Table 6.147 RDCTRL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b28	PHYODTONT	PHY DQS ODT ON Timing setting Effective at PHYODTEN = 1. The on timing of PHY ODT can be set at 0.5 tCK step (0 delay conditions). (PHYODTONT − 4) × 0.5 tCK before preamble 4'hE: 5.0 tCK before preamble (default)	R/W
b27 to b24	PHYODTOFT	PHY DQS ODT OFF Timing setting Effective at PHYODTEN = 1. The turning off timing of PHY ODT can be set at 0.5 tCK step (0 delay conditions). (PHYODTOFT + 2) × 0.5 tCK after postamble 4'h7: 4.5 tCK after postamble (default) The setting range can set 4'h0 to 4'hC. Other setting is prohibited.	R/W
b23 to b20	PDQODTONT	PHY DQ ODT ON Timing setting The setting method is same as PHYODTONT.	R/W
b19 to b16	PDQODTOFT	PHY DQ ODT OFF Timing setting. The setting method is same as PHYODTOFT.	R/W
b15 to b12	PHYBENONT	PHY DQS and DQ BEN ON Timing setting. ON timing of the bias enable of the receiver can be set at 0.5 tCK steps (0 delay conditions). (PHYBENONT − 4) × 0.5 tCK before preamble: DDR2/DDR3 4'hE: 5.0 tCK before preamble (DDR3: default)	R/W
b11 to b8	PHYBENOFT	PHY DQS and DQ BEN OFF Timing setting. The turning off timing of an enable bias of the receiver can be set at 0.5 tCK steps (0 delay conditions). (PHYBENOFT + 2) × 0.5 tCK after postamble: DDR2/DDR3. 4'h7: 4.5 tCK after postamble (DDR3: default) The setting range can set 4'h0 to 4'hC. Other setting is prohibited.	R/W
b7 to b4	PHYIENONT	PHY DQS and DQ Input enable ON Timing setting. Inputting enable ON timing of the receiver can be set at 0.5 tCK step (0 delay conditions). It is necessary to turn on more than 1 ns ahead of the 1st rise of DQS. (PHYIENONT − 4) × 0.5 tCK before preamble: DDR2/DDR3. 4'h4: Simultaneous with preamble (DDR3: default)	R/W
b3 to b0	PHYIENOFT	PHY DQS and DQ Input enable OFF Timing setting. Enable turning off timing for the input of the receiver can be set at 0.5 tCK step. (0 delay conditions). It is necessary to consider Flight Time. (PHYIENOFT + 2) × 0.5 tCK after postamble: DDR2/DDR3. 4'h6: 4.0 tCK after postamble (default). The setting range can set 4'h0 to 4'hC. Other setting is prohibited.	R/W

6.4.2.6 RDTMG — READ Timing Control Register

Address: 4000 E014h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	RDENVALID				—	—	WDOMODE	
Value after reset	X	X	X	X	X	X	X	X	1	0	1	1	X	X	0	0

Table 6.148 RDTMG Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved	NA	R
b7 to b4	RDENVALID	Read Data transfer setting. Waiting time (DFICLK) from RDDATAEN = “1” to RDDATAVALID = “1” is set. It is necessary to match tphy_rdlat. RDENVALID+4 DFICLK 4’hB: 15 DFICLK (default) (WRFIFOEN=1’b1, FIFORPINIT=2’b01) 0 set is prohibited.	R/W
b3, b2	Reserved	NA	R
b1, b0	WDOMODE	Command DQ output mode setting 2’b00: Normal mode (default). Others: Reserved (prohibited)	R/W

6.4.2.7 FIFOINIT — FIFO Initialization Register

Address: 4000 E018h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	RDPTINITEXE	—	—	—	—	—	—	—	WRPTINITEXE
Value after reset	X	X	X	X	X	X	X	0	X	X	X	X	X	X	X	0

Table 6.149 FIFOINIT Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b9	Reserved	NA	R
b8	RDPTINITEXE	Read FIFO pointer initialization. The initialization of the FIFO pointer of Read is executed. Initialization is executed by writing “1”. “0” is returned when reading it.	W
b7 to b1	Reserved	NA	R
b0	WRPTINITEXE	Write FIFO pointer initialization. The initialization of the pointer of the value of FIFORPINIT is executed. Initialization is executed by writing “1”. “0” is returned when reading it.	W

6.4.2.8 OUTCTRL — Output Control Register

Address: 4000 E01Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	MBL		—	—	—	MRL				
Value after reset	X	X	X	X	X	X	1	0	X	X	X	0	1	0	1	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	MWL					—	—	DISOUT		—	RESETBOE	CKEODTOE	ADCMDOE
Value after reset	X	X	X	0	1	0	0	0	X	X	0	0	X	1	1	0

Table 6.150 OUTCTRL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b26	Reserved	NA	R
b25, b24	MBL	Only for evaluation. Don't set value other than reset value	R/W
b23 to b21	Reserved	NA	R
b20 to b16	MRL	Only for evaluation. Don't set value other than reset value	R/W
b15 to b13	Reserved	NA	R
b12 to b8	MWL	Only for evaluation. Don't set value other than reset value	R/W
b7, b6	Reserved	NA	R
b5, b4	DISOUT	Output mode setting at DFIDATABYTEDISABLE 2'b00: Floating 2'b01: L Output. (However DDR_DQS_N H output); else: Prohibited	R/W
b3	Reserved	NA	R
b2	RESETBOE	Outputting enable setting of DDR_RESET_N 1'b0: Floating 1'b1: Output (default)	R/W
b1	CKEODTOE	Outputting enable setting of DDR_CLKEN and DDR_ODT 1'b0: Floating 1'b1: Output (default)	R/W
b0	ADCMDOE	Address and Command output enable setting. Output controls of address command other than DDR_CLKEN, DDR_ODT, and DDR_RESET_N 1'b0: Floating 1'b1: Output	R/W

6.4.2.9 WLCTRL1 — Write Leveling Control Register 1

Address: 4000 E040h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	WLEN	WLAUTO	—	WLVEND	—	WLSTATE		WLSTR	—	—	—	—	—	—	—	—
Value after reset	0	0	X	0	X	0	0	0	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	WL2OFS							—	WL1OFS						
Value after reset	X	0	0	0	0	0	0	0	X	0	0	0	0	0	0	0

Table 6.151 WLCTRL1 Register Contents

Bit Position	Bit Name	Function	R/W
b31	WLEN	Write Leveling function use setting. It is necessary to write “1” in this bit before initialization of the FIFO pointer when Write Leveling is used. 1’b0: Unused (default) 1’b1: Use (FIFO I/F mode limitation)	R/W
b30	WLAUTO	Setting of Write Leveling execution mode (Reserved mode) 1’b0: Manual setting (default) 1’b1: Automatic setting (Prohibited)	R/W
b29	Reserved	NA	R
b28	WLVEND	Write Leveling End Flag. 1’b0: Not end 1’b1: End	R
b27	Reserved	NA	R
b26, b25	WLSTATE	Write Leveling status setting (Prohibited)	R/W
b24	WLSTR	Write Leveling timing adjustment. The timing is readjusted with the set value by writing “1”. “0” is returned when reading it. Please write to this register with changing only WLSTR to “1” after waiting for 50 clocks (DFICLK-converted) after setting each item in the state of “0”. Please wait for 50 DFICLK or more after it executes it.	W
b23 to b15	Reserved	NA	R
b14 to b8	WL2OFS	The second Byte Write Leveling manual operation offset setting. WL2OFS[6] = 1’b1 90-90/32 × WL2OFS[5:0] [degree] (0 to 90 degree: 0 ≤ WL2OFS[5:0] ≤ 32) WL2OFS[6] = 1’b0 90+90/32 × WL2OFS[5:0] [degree] (90 to 135 degree: 0 ≤ WL2OFS[5:0] ≤ 16)	R/W
b7	Reserved	NA	R
b6 to b0	WL1OFS	The first Byte Write Leveling manual operation offset setting. WL1OFS[6] = 1’b1 90-90/32 × WL1OFS[5:0] [degree] (0 to 90 degree: 0 ≤ WL1OFS[5:0] ≤ 32) WL1OFS[6] = 1’b0 90+90/32 × WL1OFS[5:0] [degree] (90 to 135 degree: 0 ≤ WL1OFS[5:0] ≤ 16)	R/W

6.4.2.10 DQCALOFS1 — DQS Offset Setting

Address: 4000 E0E8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	B2RSSAT	B1RSSAT	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	DQCAL2OFS							—	DQCAL1OFS						
Value after reset	X	0	0	0	0	0	0	0	X	0	0	0	0	0	0	0

Table 6.152 DQCALOFS1 Register Contents

Bit Position	Bit Name	Function	R/W
b31	B2RSSAT	SDLY Saturation Flag output for 2nd byte Read 1'b0: No Saturate State 1'b1: Saturate State	R
b30	B1RSSAT	SDLY Saturation Flag output for 1st byte Read 1'b0: No Saturate State 1'b1: Saturate State	R
b29 to b15	Reserved	NA	R
b14 to b8	DQCAL2OFS	2nd byte DQS offset setting DQCAL2OFS[6] = 1'b1 90-90/32 × DQCAL2OFS[5:0] [degree] (0 to 90 degree: 0 ≤ DQCAL2OFS[5:0] ≤ 32) DQCAL2OFS[6] = 1'b0 90+90/32 × DQCAL2OFS[5:0] [degree] (90 to 180 degree: 0 ≤ DQCAL2OFS[5:0] ≤ 32)	R/W
b7	Reserved	NA	R
b6 to b0	DQCAL1OFS	1st byte DQS offset setting DQCAL1OFS[6] = 1'b1 90-90/32 × DQCAL1OFS[5:0] [degree] (0 to 90 degree: 0 ≤ DQCAL1OFS[5:0] ≤ 32) DQCAL1OFS[6] = 1'b0 90+90/32 × DQCAL1OFS[5:0] [degree] (90 to 180 degree: 0 ≤ DQCAL1OFS[5:0] ≤ 32)	R/W

6.5 Operation

6.5.1 Address Mapping

The DDR Controller automatically maps user addresses to the DRAM memory in a contiguous block. Addressing starts at user address 0 and ends at the highest available address according to the size and number of DRAM memories present. This mapping is dependent on how the parameters in the internal DDR Controller registers are programmed.

The mapping of the address space to the internal data storage structure of the DRAM memories is based on the actual size of the DRAM memories available. The size is stored in user-programmable parameters that must be initialized at power up. Certain DRAM memories allow for different mapping options to be chosen, while other DRAM memories depend on the memory burst length chosen.

6.5.1.1 DDR SDRAM Address Mapping Options

The address structure of DDR SDRAM memories contains five fields. Each of these fields can be individually addressed when accessing the DRAM. The address map for this DDR Controller is ordered as follows:

Chip Select — Row — Bank — Column — Datapath

The actual widths of the fields can be changed if the memory address width parameters (COL_DIFF, BANK_DIFF and ROW_DIFF) are programmed differently.

CAUTION

The maximum address range of the DDR memory controller is limited to 0x00000000 – 0x7FFFFFFF internally. On RZ/N1, the DDR memory is mapped to 0x80000000 − 0xFFFFFFFF.

6.5.1.2 Maximum Address Space

The maximum user address range is determined by the width of the memory datapath, the number of chip select pins, and the address space of the DRAM memory. The maximum amount of memory can be calculated by the following formula:

$$MaxMemBytes = ChipSelects \times 2^{Address} \times NumBanks \times DPWidthBytes$$

Figure 6.2 Maximum Amount of Memory Formula

Chip Selects = 2
Memory Address = 16 + 11 (Row + Column)
Number of Banks Per Chip Select = 8
Memory Datapath Width in Bytes = 2 bytes

CAUTION

The maximum address range of the DDR memory controller is limited to 0x00000000 – 0x7FFFFFFF internally. On RZ/N1, the DDR memory is mapped to 0x80000000 − 0xFFFFFFFF.

As a result, the maximum accessible memory area is 2 GB.

6.5.1.3 Memory Mapping to Address Space

The maximum allowable address space and mapping into the DRAM memories for the DDR Controller is shown in **Figure 6.3, Maximum Memory Map**.

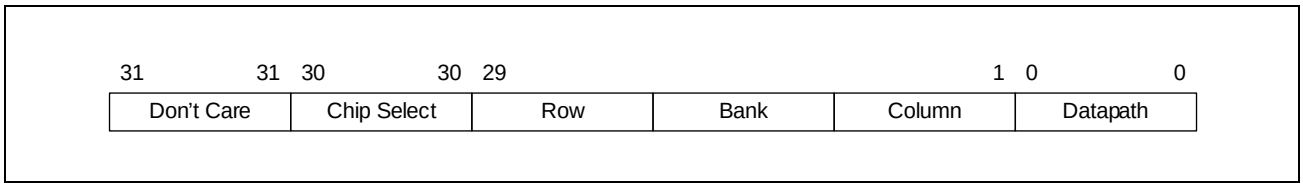


Figure 6.3 Maximum Memory Map

CAUTION

Bit [31] **must** not be used as it is not considered within the DDR controller. The maximum allowed Address is 0x7FFFFFFF.

The "COL_DIFF" and "ROW_DIFF" parameters can each range from the maximum available values for the controller to seven bits smaller than the maximum available values. This allows the DDR Controller to function with a wide variety of memory sizes.

The settings for the "COL_DIFF" and "ROW_DIFF" parameters control how the address map is used to decode the user address to the DRAM chip selects and row and column addresses. The "BANK_DIFF" parameter controls the DRAM bank address information. It is assumed that the values in these parameters never exceed the maximum values configured.

For example, if the controller is wired to memories with 13 row pins and 9 column bits, the maximum accessible memory space would be reduced. The accessible memory space for this configuration is 128 MB.

The address map for this configuration is shown in **Figure 6.4, Alternate Memory Map**. Note that address bits 27 through 31 are listed as "don't care" bits. These bits are ignored when the controller generates the address to the DRAM memories, but they are used to verify that the address lies within the usable address range of the controller. Therefore, the user should drive these bits to 0 to avoid the controller interpreting the command as being out-of-range and causing one or both of the out-of-range interrupts (bits 1 or 2) to be set to 1 in the "INT_STATUS" parameter.

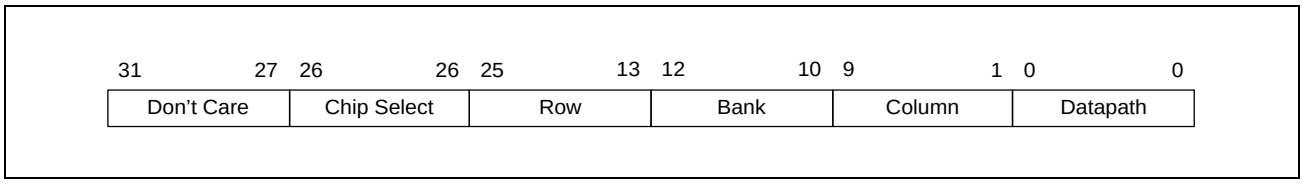


Figure 6.4 Alternate Memory Map

CAUTION

The Chip Select, Row, Bank, and Column fields are used to address an entire memory word, and the Datapath bits are used to address individual bytes within the word. For example, for a read starting at byte address 0x1, the Datapath bits must be defined as 1'b1 in order to address this byte directly. Reads and writes are memory word-aligned if all the Datapath bits are 0.

6.5.2 AXI Interface Ports

The DDR Controller contains 4 internal AXI ports which communicate on the AXI bus. Each AXI port uses an AXI interface block to connect to the Controller core. An arbitration scheme is responsible for arbitrating requests from the ports and sending requests to the Controller core. Each transaction received at the Arbiter logic has an associated priority, which works with each port's arbitration logic to determine how ports issue requests to the Controller core. This DDR Controller supports the Bandwidth Allocation/Priority Round-Robin arbitration scheme.

The Arbiter logic routes read data from the Controller core to the appropriate port. The requesting port is assumed able to receive the data. Write data from each port is connected directly to its own write data interface in the Controller core, allowing the ports to independently pass write data to the Controller core buffers.

6.5.2.1 Arbitration Scheme

The bandwidth allocation scheme is an extension of simple round-robin arbitration. It is based on the priority of the requests and is influenced by the actual bandwidth consumed by the port inside the Controller core.

Priority round-robin arbitration is a complex arbitration scheme. In order to understand its operation, each concept must be first understood individually. The following Sections describes the various components of priority round-robin arbitration.

6.5.2.2 Understanding Round-Robin Arbitration

Round-robin operation is a simple form of arbitration which offers each port an opportunity to issue a command. This scheme uses a counter that rotates through the port numbers, incrementing every time a port request is granted from any port.

If the port that the counter is referencing has an active request, and the command queue is not full, then this request will be sent to the Controller core. If there is not an active request for that port, then the port will be skipped, and the next port will be checked. The counter will increment by one whenever any request has been processed, regardless of which port's request was arbitrated.

Round-robin operation ensures that each port's requests can be successfully arbitrated into the Controller core every N cycles, where N is the number of ports in the DDR Controller. No port will ever be locked out, and any port can have its requests serviced on every cycle as long as all other ports are quiet and the command queue is not full.

An example of the round-robin scheme is shown in the following **Table 6.153, Round-Robin Operation Example**. Cycles 0, 2 and 6 show the system behavior when the command queue is full. Cycle 8 and 11 show the system behavior when the port addressed by the arbitration counter does not have an active request. In particular, note cycle 11: The port addressed by the arbitration counter (0) is not requesting, so the counter scans through the other ports, in incrementing order, to find an active request. Port 2 is requesting and therefore wins arbitration, but the counter only increments to port 1 which was the next port in the sequence. All other cycles show normal behavior.

Table 6.153 Round-Robin Operation Example

Cycle	Port Addressed by the Arbitration Counter	Ports Requesting				Command Queue Full?	Arbitration Winner	Value of Counter at Next Cycle
		Port 0	Port 1	Port 2	Port 3			
0	0	Y	Y	Y	Y	Yes	None	0
1	0	Y	Y	Y	Y	No	P0	1
2	1		Y	Y	Y	Yes	None	1
3	1	Y	Y	Y	Y	No	P1	2
4	2	Y		Y	Y	No	P2	3
5	3	Y			Y	No	P3	0
6	0	Y		Y		Yes	None	0
7	0	Y		Y		No	P0	1
8	1			Y		No	P2	2
9	2			Y	Y	No	P2	3
10	3	Y			Y	No	P3	0
11	0			Y		No	P2	1

6.5.2.3 Understanding Port Priority

Priorities are associated with a port and each port has separate priority parameter for reads and writes. These values are stored into the programmable parameters “AXIY_R_PRIORITY” and “AXIY_W_PRIORITY” at controller initialization (where Y represents the AXI port number [0-3]). Internally, the ports are organized into priority groups based on their priority setting. All ports within a priority group are treated equally for arbitration unless a port has exceeded its allocated bandwidth. The priority value is also used by the placement logic inside the Controller core when filling the command queue.

CAUTION

A situation may arise where multiple ports are defined with a certain priority, but only a subset of these ports are actively requesting. If this occurs, the ports with active requests may not win arbitration equally. Rather, the behavior is dependent on the port activity and the organization of active ports within the arbitration order. Refer to **Section 6.5.2.8(1), Example with Four Ports, One Priority Level and No Bandwidth Consideration** for more information on this scenario.

A priority value of 0 is highest priority, and a priority value of (decimal) 3 is the lowest priority in the DDR Controller. The user may program at priority level 0; however, it is best to reserve this priority value so that the placement queue can elevate to this level through aging.

6.5.2.4 Understanding Port Bandwidth

Each port has an associated bandwidth limit that sets the maximum percentage of the Controller core bandwidth that the port is allowed to use. Once this level is reached, the Arbiter will no longer accept requests from this port until the bandwidth usage drops below the threshold. This scheme allows for the bandwidth to be shared between the ports. If required, an overflow option allows the port to continue to receive requests after the bandwidth limit has been reached. Refer to **Section 6.5.2.6, Understanding Port Bandwidth Overflow** for more information on this option.

The bandwidth limits are stored in the programmable bandwidth parameters “AXIY_BDW” for each port Y at DDR Controller initialization. The current bandwidth parameters “AXIY_CURRENT_BDW” parameters are used to track the actual bandwidth utilized as computed by the bandwidth calculation module inside the Arbiter.

Port bandwidth is computed by counting the number of cycles that the Controller core is busy actively processing that port’s request in each 100 cycles period, referred to as the statistics window. In the DDR Controller, 10 counters are used for this computation. The counters track the number of active cycles in each statistics window, generating a moving average bandwidth value for each port. This is the actual bandwidth utilized value saved in the current bandwidth parameters “AXIY_CURRENT_BDW” for each port Y at DDR Controller. The values in the current bandwidth parameters are updated every 10 cycles (the statistics reporting time) with the actual bandwidth used in the last 100 cycles.

Note that if all ports are assigned 100% or greater bandwidth, then bandwidth usage will not factor and arbitration will be purely based on priority.

The Controller core is defined as actively processing for a port if any of the following situations occur:

- The Controller core is ready to transfer write data from the port to memory, but the data has not arrived from the port.
- The Controller core is holding the port’s command and is ready to transfer to memory, but is waiting to open a bank, pre-charge a bank, or some other memory-related action.
- The Controller core is actively transferring data from the port to memory.
- The Controller core is ready to transfer read data from memory to the port, but the port is busy and unable to accept the data.

The following **Figure 6.5, System Bandwidth Example** shows bandwidth usage for a 4-port system with a 100-cycle calculation window and 10 counters.

Bandwidth Cycle Window	Number of Active Cycles out of 10 Cycles			
	Port 0	Port 1	Port 2	Port 3
0-9	5	5	0	0
10-19	3	1	2	4
20-29	7	0	1	2
30-39	1	5	2	2
40-49	4	0	2	3
50-59	5	1	0	1
60-69	1	3	4	2
70-79	3	1	0	3
80-89	0	3	2	2
90-99	2	1	2	4
100-109	3	4	2	0
110-119	0	0	5	5
120-129	0	0	5	5
130-139	3	3	3	1
140-149	3	3	4	0
150-159	0	0	7	1
160-169	2	1	6	1
170-179	1	2	3	2
180-189	3	0	3	3
190-199	3	1	0	0

Figure 6.5 System Bandwidth Example

For this system, the bandwidth will be monitored over each 100 cycles. The bandwidth calculation parameters will be updated every 10 cycles with the bandwidth usage of the last 100 cycles as shown graphically in **Figure 6.5, System Bandwidth Example**. The bandwidth totals are shown in **Table 6.154, System Bandwidth Usage Example**. Note that in the system, these values would be stored in the current bandwidth parameters “AXIY_CURRENT_BDW” parameters after each calculation.

Table 6.154 System Bandwidth Usage Example

Counter Number	Cycles Counting	Calculated Usage			
		Port 0	Port 1	Port 2	Port 3
0	0-99	31 / 100 = 31%	20 / 100 = 20%	15 / 100 = 15%	23 / 100 = 23%
1	10-109	29 / 100 = 29%	19 / 100 = 19%	17 / 100 = 17%	23 / 100 = 23%
2	20-119	26 / 100 = 26%	18 / 100 = 18%	20 / 100 = 20%	24 / 100 = 24%
3	30-129	19 / 100 = 19%	18 / 100 = 18%	24 / 100 = 24%	27 / 100 = 27%
4	40-139	21 / 100 = 21%	16 / 100 = 16%	25 / 100 = 25%	26 / 100 = 26%
5	50-149	20 / 100 = 20%	19 / 100 = 19%	27 / 100 = 27%	23 / 100 = 23%
6	60-159	15 / 100 = 15%	18 / 100 = 18%	34 / 100 = 34%	23 / 100 = 23%
7	70-169	16 / 100 = 16%	16 / 100 = 16%	36 / 100 = 36%	22 / 100 = 22%
8	80-179	14 / 100 = 14%	17 / 100 = 17%	39 / 100 = 39%	21 / 100 = 21%
9	90-189	17 / 100 = 17%	14 / 100 = 14%	40 / 100 = 40%	22 / 100 = 22%
0	100-199	18 / 100 = 18%	14 / 100 = 14%	38 / 100 = 38%	18 / 100 = 18%

6.5.2.5 Understanding Port Bandwidth Hold-Off

When the bandwidth used by a port exceeds its specified limit, that port is held off from subsequent arbitration decisions for a period of time known as the statistics reporting time. This causes a period of inactivity from that port, allowing the actual bandwidth used for that port to fall below the threshold. Since the bandwidth used is updated every ten cycles, the minimum hold-off period for this system is ten cycles.

After every 10 cycles (the statistics reporting time), the DDR Controller will update the current bandwidth parameters with the bandwidth usage of the last 100 cycles. For any port Y, if the current bandwidth parameter value “AXIY_CURRENT_BDW” meets or exceeds the allocated value “AXIY_BDW”, the Arbiter will not accept requests from this port for the next 10 cycles starting at the next statistics reporting time. After 10 cycles, the current bandwidth parameters will be updated and the DDR Controller must evaluate the usage again.

The user must keep the following considerations in mind when using this system:

- The statistics reporting time is 10 cycles. If the port was not being held off at the start of each 10-cycle window, it may issue up to 10 commands during the next 10 cycles until the bandwidth usage is evaluated again. Bandwidth usage is NOT calculated on each cycle.
- The statistics window is 100 clock cycles. Only the last 100 clock cycles are considered for bandwidth usage. Anything older than 100 cycles is not relevant.
- Hysteresis may alter the bandwidth usage. Commands enter the command queue once they have won arbitration, but commands sitting in the command queue are not calculated in the port bandwidth usage until they reach the top of the command queue and start to execute.

The examples in the previous section, **Figure 6.5, System Bandwidth Example** and **Table 6.154, System Bandwidth Usage Example**, represent a system where all ports are defined with a bandwidth allocation of 100% (or greater) which means that ports are never held off from winning arbitration due to bandwidth usage. The same system will now be evaluated with the following bandwidth limits:

axi0_bdw = 20%
axi1_bdw = 10%
axi2_bdw = 30%
axi3_bdw = 50%

Port 3 never exceeds the bandwidth allocation based on current traffic. However, Ports 0, 1 and 2 do exceed bandwidth and therefore they will be held off. The cycles marked in asterisks (*) in **Figure 6.6, System Bandwidth Example with Bandwidth Exceeded** are hold-off periods for the port.

Bandwidth Cycle Window	Number of Active Cycles			
	Port 0	Port 1	Port 2	Port 3
0-9	5	5	0	0
10-19	3	1	2	4
20-29	7	0	1	2
30-39	1	5	2	2
40-49	4	0	2	3
50-59	5	1	0	1
60-69	1	3	4	2
70-79	3	1	0	3
80-89	0	3	2	2
90-99	2	1	2	4
Bandwidth %, Cycles 0-99	31 %	20 %	15 %	23 %
100-109	0 *	0 *	2	0
Bandwidth %, Cycles 10-109	26 %	15 %	17 %	23 %
110-119	0 *	0 *	5	5
Bandwidth %, Cycles 20-119	23 %	14 %	20 %	24 %
120-129	0 *	0 *	5	5
Bandwidth %, Cycles 30-129	16 %	14 %	24 %	27 %
130-139	3	0 *	3	1
Bandwidth %, Cycles 40-139	18 %	9 %	25 %	26 %
140-149	3	3	4	0
Bandwidth %, Cycles 50-149	17 %	12 %	27 %	23 %
150-159	0	0 *	7	1
Bandwidth %, Cycles 60-159	12 %	11 %	34 %	23 %
160-169	2	0 *	0 *	1
Bandwidth %, Cycles 70-169	13 %	8 %	30 %	22 %
170-179	1	2	0 *	2
Bandwidth %, Cycles 80-179	11 %	9 %	30 %	21 %
180-189	3	0	0 *	3
Bandwidth %, Cycles 90-189	14 %	6 %	28 %	22 %

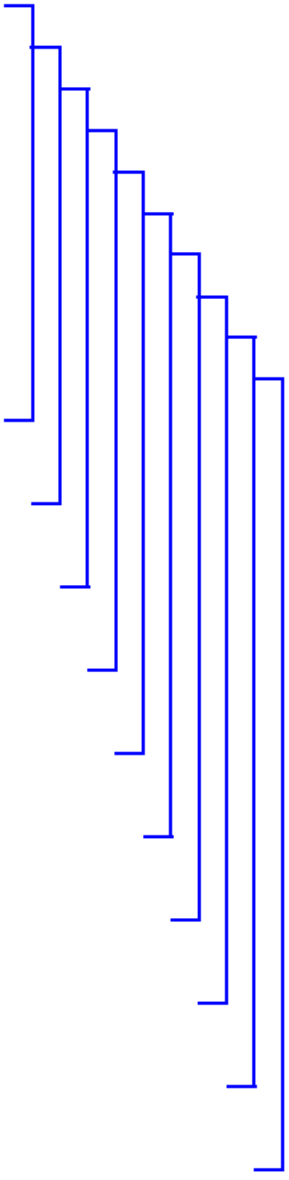


Figure 6.6 System Bandwidth Example with Bandwidth Exceeded

6.5.2.6 Understanding Port Bandwidth Overflow

The bandwidth hold-off scheme described in **Section 6.5.2.5, Understanding Port Bandwidth Hold-Off** is designed to constrain individual ports (especially ports programmed at higher priority) from overtaking all available bandwidth and locking out other ports. However, this can have its drawbacks.

Consider a situation where only one port has been actively requesting and has therefore used all of its available bandwidth. The bandwidth hold-off will prevent additional requests from being accepted, even though no other ports are requesting. The command queue will sit empty for several cycles while the hold-off is cleared. This is obviously wasted DDR Controller bandwidth and is detrimental to overall system performance. A bandwidth hold-off override function has been incorporated for such a situation in the bandwidth overflow parameters “AXIY_BDW_OVFLOW”.

A port will be allowed to exceed its allocated bandwidth when all of these conditions are true:

(1) The bandwidth overflow parameter “AXIY_BDW_OVFLOW” is set to 1 for port Y.

(2) No other port, whose bandwidth has not been exceeded, is requesting at the same priority level.

(3) The command queue has less than the number of entries specified in the “ARB_CMD_Q_THRESHOLD” parameter.

This last condition is a preventative measure to maintain latency requirements for ports programmed at higher priority. When a port is allowed to exceed bandwidth, it may fill the command queue with transactions. If this occurs, and a higher priority port starts requesting, then there will be no room in the command queue for the new requests. This means that the higher priority port will actually be held off for potentially several cycles. In this situation, even though the Controller core bandwidth is being utilized well, the latency requirements of the higher priority port are not being met. The “ARB_CMD_Q_THRESHOLD” parameter is used to limit the bandwidth overflow and prevent this condition. It ensures that a certain number of slots remain available in the command queue for other ports. As a result, bandwidth overflow will be allowed as long as there are less than “ARB_CMD_Q_THRESHOLD” number of entries in the command queue.

6.5.2.7 Priority Round-Robin Arbitration Summary

The Controller priority round-robin arbitration system combines the concepts of round-robin operation, priority, port bandwidth, port bandwidth hold-off and port bandwidth overflow. The incoming commands are separated into priority groups based on the priority of the associated port for that type of command. Within each priority group, the Arbiter evaluates the requesting ports, the command queue, the priority of the requests, the bandwidth being used and the overflow option to determine the winner of the arbitration. The order of steps is as follows:

(1) Is the command queue full?
Yes: No further action is taken.
No: Review the ports.

(2) Within each priority group with at least one active request, select a request for evaluation for that priority group. Each priority group uses a round-robin arbitration counter that will address a port for arbitration.

(3) Evaluate the highest priority group with a selected command. Has the bandwidth allocation for this port been exceeded?
No: This request wins arbitration. Skip to step 7.
Yes: Continue to step 4 to check the bandwidth overflow status.

(4) Is bandwidth overflow enabled?
No: Skip to step 6.
Yes: Continue to step 5 to check the conditions for overflow.

(5) Are the overflow conditions listed in **Section 6.5.2.6, Understanding Port Bandwidth Overflow** met?
No: Continue to step 6.
Yes: This request wins arbitration. Skip to step 7.

(6) The selected command has failed evaluation. The Arbiter will select another command for evaluation.

- If there is another command at the same priority group that has not been evaluated, the next active request (in cyclical port order) will be selected for evaluation for that priority group. Return to step 3 with that command.
- If there are no other commands at that priority group to evaluate, examine the next highest priority group, and return to step 3 with that command.

(7) Once a request wins arbitration, it is processed into the command queue and the round-robin counter is updated to the next port in the circular queue. If any request is processed for a priority group, the round-robin arbitration counter for that priority group will be incremented by one. The counter will never increment by more than one, regardless of how many commands were evaluated or which port's command was finally accepted.

6.5.2.8 Arbitration Examples

(1) Example with Four Ports, One Priority Level and No Bandwidth Consideration

As a very simple example, consider a system where all ports are programmed to accept all commands at a single priority level. The arbitration counter will address all ports defined at this priority level in cyclical fashion, regardless of the status of their activity. At each cycle, the arbitration counter will address a particular port. If that port is not requesting, the DDR Controller looks for the next port (in circular fashion) making a request and that port wins arbitration. With each successful arbitration, the arbitration counter increments by 1 and on the next cycle, the next port will be given an opportunity to win arbitration.

To demonstrate arbitration behavior, consider the following system:

- Four ports.
- All ports request only at priority 1.
- All ports have bandwidth allocations of 100% and the bandwidth overflow bit is set to 1. (Bandwidth will not affect arbitration.)

An example with these settings is shown in **Table 6.155, Priority Round-Robin with One Priority Level**. In cycles 1-4, port 0 and port 1 are always requesting, and port 2 and port 3 are quiet. When the arbitration counter addresses port 0 or 1, these ports win arbitration. When the arbitration counter addresses ports 2, since there is not an active request on that port, the next port in round-robin order that is requesting (port 0) wins arbitration. Similarly, port 0 wins arbitration when port 3 is addressed by the arbitration counter.

Cycles 5 and 6 demonstrate that port 0 or port 1 could win arbitration on any cycle. Even though the arbitration count is addressing port 0 in cycle 5, port 1 wins arbitration, and vice-versa for cycle 6.

In situations where not all ports are requesting, the distribution of arbitration winners across ports is not a standard formula; it is entirely dependent on the organization of requesting ports in the arbitration order. For example, in cycles 1-4, only ports 0 and 1 are requesting. These cycles may seem to suggest that the lowest-numbered port wins arbitration (N-1)/N times, where N represents the number of ports in the priority level. However, this is only a coincidence. Cycles 7-10, where only port 0 and port 3 are requesting, show a scenario in which the higher-numbered port wins arbitration more often. And cycles 11-14, where port 0 and port 2 are requesting, show an equitable distribution of arbitration wins. Therefore, no assumptions should be made on arbitration wins.

If a user wishes to provide each port with an equal opportunity for arbitration, only ports that are requesting at a priority level should be programmed with that priority value.

Table 6.155 Priority Round-Robin with One Priority Level

Cycle	Port Addressed by the Arbitration Counter	Ports Requesting				Command Queue Full?	Arbitration Winner	Value of Counter at Next Cycle
		Port 0	Port 1	Port 2	Port 3			
0	0	Y	Y			Yes	None	0
1	0	Y	Y			No	P0	1
2	1	Y	Y			No	P1	2
3	2	Y	Y			No	P0	3
4	3	Y	Y			No	P0	0
5	0		Y			No	P1	1
6	1	Y				No	P0	2
7	2	Y			Y	No	P3	3
8	3	Y			Y	No	P3	0
9	0	Y			Y	No	P0	1
10	1	Y			Y	No	P3	2
11	2	Y		Y		No	P2	3
12	3	Y		Y		No	P0	0
13	0	Y		Y		No	P0	1
14	1	Y		Y		No	P2	2

(2) Example with Four Ports, Two Priority Levels and No Bandwidth Consideration

To demonstrate arbitration behavior, consider the following system:

- Four ports.
- Ports 0 and 1 request only at priority 1. Ports 2 and 3 request only at priority 2.
- All ports have bandwidth allocations of 100% and the bandwidth overflow bit is set to 1. (Bandwidth will not affect arbitration.)

An example with these settings is shown in **Table 6.156, Priority Round-Robin without Bandwidth Consideration**. Note that priority 2 requests will only win arbitration when there are no priority 1 requests. Also note that each arbitration counter only increments, and always increments, when a request of that priority is processed. Cycle 5 demonstrates the “always” condition of counter incrementing: even though the arbitration was won by the other port of highest priority instead of the one in the counter, the counter still increments.

Table 6.156 Priority Round-Robin without Bandwidth Consideration

Cycle	Port Addressed by the Arbitration Counter		Ports Requesting				Command Queue Full?	Arbitration Winner	Value of Counter at Next Cycle	
	PG 1*1	PG 2*2	Port 0	Port 1	Port 2	Port 3			PG 1*1	PG 2*2
0	0	2	Y	Y	Y	Y	Yes	None	0	2
1	0	2	Y	Y	Y	Y	No	P0	1	2
2	1	2		Y	Y	Y	No	P1	0	2
3	0	2	Y		Y	Y	No	P0	1	2
4	1	2			Y	Y	No	P2	1	3
5	1	3	Y			Y	No	P0	0	3
6	0	3				Y	No	P3	0	2
7	0	2					No	None	0	2
8	0	2			Y	Y	Yes	None	0	2
9	0	2	Y		Y	Y	No	P0	1	2
10	1	2			Y	Y	No	P2	1	3
11	1	3				Y	No	P3	1	2

Note 1. PG1 = Priority Group 1

Note 2. PG2 = Priority Group 2

(3) Example with Four Ports, Two Priority Levels and Bandwidth Consideration

The example shown in **Section 6.5.2.8(2), Example with Four Ports, Two Priority Levels and No Bandwidth Consideration** was a very simplified case without considering bandwidth. However, in most cases, bandwidth will factor into the arbitration. Consider a system in which the allocated bandwidth is less than 100%. This system is shown in **Table 6.157, Priority Round-Robin with Bandwidth Consideration**. The “Bandwidth Held Off” column indicates if the bandwidth was held off for any port in that cycle. (Note cycles 3 and 11.) Even though the priority group 1 arbitration counter is pointing to port 0 in both cases, because the port is held off, port 0 does not win arbitration. A lower priority port, from priority group 2, wins arbitration instead. In this simplified example, every statistics reporting time is represented as 1 cycle, which allows port 0 to win arbitration on the cycle after hold-off (cycles 4 and 12). In the DDR Controller, the statistics reporting time is 10 cycles, so port 0 would actually be held off for 10 cycles.

Table 6.157 Priority Round-Robin with Bandwidth Consideration

	Port Addressed by the Arbitration Counter		Ports Requesting				Command Queue Full?	Bandwidth Held Off?	Arbitration Winner	Value of Counter at Next Cycle	
Cycle	PG 1*1	PG 2*2	Port 0	Port 1	Port 2	Port 3				PG 1*1	PG 2*2
0	0	2	Y	Y	Y	Y	Yes	No	None	0	2
1	0	2	Y	Y	Y	Y	No	No	P0	1	2
2	1	2		Y	Y	Y	No	No	P1	0	2
3	0	2	Y		Y	Y	No	Yes, Port0	P2	0	3
4	0	3	Y			Y	No	No	P0	1	3
5	1	3	Y			Y	No	No	P0	0	3
6	0	3				Y	No	No	P3	0	2
7	0	2					No	No	None	0	2
8	0	2			Y	Y	Yes	No	None	0	2
9	0	2	Y		Y	Y	No	No	P0	1	2
10	1	2	Y		Y	Y	No	No	P0	0	2
11	0	2	Y		Y	Y	No	Yes, Port0	P2	0	3
12	0	3	Y			Y	No	No	P0	1	3
13	1	3				Y	No	No	P3	1	2

Note 1. PG1 = Priority Group 1

Note 2. PG2 = Priority Group 2

6.5.2.9 Programming for Priority Round-Robin Arbitration

The priority round-robin arbitration scheme requires the use of several programmable parameters: AXIY_BDW, AXIY_BDW_OVFLOW, AXIY_CURRENT_BDW, AXIY_R_PRIORITY, AXIY_W_PRIORITY and ARB_CMD_Q_THRESHOLD. Since these parameters work together, there are trade-offs for the settings.

CAUTION

All of the arbitration parameters must be programmed prior to setting the start parameter to 1. Any subsequent changes to the arbitration parameters may result in unpredictable system operation.

Command priority, port bandwidth and bandwidth overflow play an important role in arbitration. Priority is the most important factor since commands are first sorted into priority groups based on their priority settings. In a multi-port system, all ports with tight latency requirements should be assigned higher priority values (lower numbers). Note that the priority of the command affects both arbitration and placement into the command queue. As a result, it is more complicated to meet the latency requirements of all ports. For high priority commands or ports with low allocated bandwidths, bandwidth overflow may be useful. To mimic simple round-robin arbitration, program all ports to have the same priority and full bandwidth allocations.

CAUTION

The bandwidth parameter “AXIY_BDW” is a 7-bit parameter. Any programmed value greater than 0x64 is interpreted as not having a maximum bandwidth allowance.

6.5.3 Port Protection Option

Each AXI port contains an address protection option to set up regions of the memory map that will be protected. If the “PORT_ADDR_PROTECTION_EN” parameter is set to 1, all incoming addresses and access types will be checked against valid address ranges and types in order to protect each port’s specified memory space. When the “PORT_ADDR_PROTECTION_EN” parameter is cleared to 0, no addresses will be checked.

When enabled, if the command does not match the region information in address, transaction type or protection level, a port out-of-range error will be logged. The failing transaction will process as a flushed write or a read returning zeroes. In addition, one or both of the out-of-range interrupts (bits 1 or 2) will be set in the “INT_STATUS” parameter and the error signature will be saved. A bus error will be reported on the AXI interface.

The number of address regions is 16 for each port. Refer to **Table 6.158, AXI Port Address Region Setting** for the number of address regions.

Table 6.158 AXI Port Address Region Setting

Port Number	Number of Address Regions
Port 0	16
Port 1	16
Port 2	16
Port 3	16

The system will default to restricting any type of access for the entire memory. For any one port, valid regions must be set up with unique addresses such that the regions do not overlap and with IDs and protection levels that will be supported. Failure to change any of the ID check and protection level parameters prior to enabling protection will result in all commands failing. For ports that will not be using port protection, the user must define at least one region that will accept all read and write cycles, for any address. Therefore, in this case, at least 1 region must be defined with the following characteristics:

AXIY_START_ADDR_Z = All zeros (min address)
AXIY_END_ADDR_Z = All ones (max address)
AXIY_RANGE_PROT_BITS_Z = 2’b11
AXIY_RANGE_RID_CHECK_BITS_Z = 0xFFFF
AXIY_RANGE_WID_CHECK_BITS_Z = 0xFFFF
AXIY_RANGE_RID_CHECK_BITS_ID_LOOKUP_Z = 0xF
AXIY_RANGE_WID_CHECK_BITS_ID_LOOKUP_Z = 0xF

CAUTION

ID check is Not supported.

All bits related to ID_CHECK settings should be set to 1.

For each parameter, Y represents the port number and Z differentiates between the various regions. Note that it is acceptable for address regions for one port to overlap access regions for another port.

The minimum granularity of an address region is 16K. Bit [13] of the incoming system address is used as the lowest bit for comparison. Internally, the Controller compares bit X of the address, where X is the defined as the 14th bit, left shifted by the number of bits used for the datapath. The user may define regions as any multiple of 16K, up to 1/2 of the memory space.

Port protection works by checking incoming requests against the valid address ranges. If enabled by setting the PORT_ADDR_PROTECTION_EN parameter to 1, an incoming command is checked based on the restrictions defined in the port protection parameters. ALL of the following tests must pass for a command to remain valid:

- Address Check:

  The starting and ending addresses must both fall within a single region Z as defined by the “AXIY_START_ADDR_Z” and “AXIY_END_ADDR_Z” parameters. If both starting and ending addresses do not fall within a single region Z, the command will fail.

- Protection Check:

  The relevant protection signal “axiY_ARPROT or axiY_AWPROT” must match the settings for the associated “AXIY_RANGE_PROT_BITS_Z” parameter. The “AXIY_RANGE_PROT_BITS_Z” parameter defines the restrictions for region Z as privileged & secure access, privileged access, secure access or full access. If the protection signal does not match the settings of region Z, the command will fail.

- Set all bits 1 to following parameters

  “AXIY_RANGE_RID_CHECK_BITS_Z”
  “AXIY_RANGE_WID_CHECK_BITS_Z”
  “AXIY_RANGE_RID_CHECK_BITS_ID_LOOKUP_Z”
  “AXIY_RANGE_WID_CHECK_BITS_ID_LOOKUP_Z”

Commands that fail are processed through the controller, but do not corrupt memory and do not return valid data to the user interface. A failing write command will be processed internally as a flushed write command. In this case, the write data is cleared out of the controller FIFOs but the data stored in DRAM memory will NOT change.

A failing read will read the appropriate number of bytes in the memory. However, this data will be ignored. Since the data will be ignored, no ECC detection or correction is attempted. Therefore, a read command error will not trigger an ECC error interrupt or save the erroneous data.

In the event of a failure, the port command error interrupt (bit 7) will be set in the INT_STATUS parameter, and reported to the Global Interrupt Controller on the DDRC_Int signal (if the associated bit is not masked in the INT_MASK parameter). The error signature will be logged in the PORT_CMD_ERROR_ADDR and PORT_CMD_ERROR_TYPE.

6.5.4 Command Queue with Placement Logic

The Controller core contains a command queue that accepts commands from the Arbiter. This command queue uses a placement algorithm to determine the order that commands will be placed into the command queue. The placement logic follows many rules to determine where new commands should be inserted into the queue, relative to the contents of the command queue at the time. Placement is determined by considering address collisions, source collisions, data collisions, command types and priorities. In addition, the placement logic attempts to maximize efficiency of the Controller core through command grouping, write-to-read splitting and bank splitting.

Many of the rules used in placement may be individually enabled/disabled. In addition, the command queue may be disabled by clearing the PLACEMENT_EN parameter, resulting in an in-line queue that services requests in the order they are received. If the PLACEMENT_EN parameter is cleared to 0 and the IN_ORDER_ACCEPT parameter is set to 1, the placement algorithm will be ignored.

6.5.4.1 Rules of the Placement Algorithm

The factors affecting command placement all work together to identify where a new command fits into the execution order. They are listed in order of importance.

(1) Address Collision/Data Coherency Violation

The order in which read and write commands are processed in the controller is critical to proper system behavior. While reads and writes to different addresses are independent and may be re-ordered without affecting system performance, reads and writes that access the same address are significantly related. If the port requests a read after a write to the same address, then repositioning the read before the write would return the original data, not the changed data. Similarly, if the read was requested ahead of the write but accidentally positioned after the write, then the read would return the new data, not the original data prior to being overwritten. These are significant data coherency mistakes.

To avoid address collisions, reads or writes that access the same chip select, bank and row as a command already in the command queue will be inserted into the command queue after the original command, even if the new command is of a higher priority. This rule is ignored when comparing a new read command to an existing read. Even if an address collision occurs between these reads, there is no data integrity issue and the data may be returned in any order.

Address collision checking may be enabled/disabled through the ADDR_CMP_EN parameter and should only be disabled if the system can guarantee coherency of reads and writes.

(2) Priority

Priorities are used to distinguish important commands from less important commands. Each command is given a priority based on the command type through the programmable parameters “AXIY_R_PRIORITY” and “AXIY_W_PRIORITY” (where Y represents the port number).

The placement algorithm will attempt to place higher priority commands ahead of lower priority commands, as long as they have no address collisions. Higher priority commands will be placed lower in the command queue if they access the same address, are from the same requestor or use the same buffer as lower priority commands already in the command queue.

Priority checking is enabled through the “PRIORITY_EN” parameter.

(3) Bank Splitting

Before accesses can be made to two different rows within the same bank, the first active row must be closed (pre-charged) and the new row must be opened (activated). Both activities require some timing overhead; therefore, for optimization, the placement logic will attempt to insert the new command into the command queue such that commands

to other banks may execute during this timing overhead. The placement of the new commands will still follow priority and address collision rules.

Bank splitting is enabled through the “BANK_SPLIT_EN” parameter.

(4) Write-to-Read Splitting

When a read command follows a write command to the same chip select, there is some timing overhead to switch command types. For optimization, the placement logic will attempt to insert the new command into the command queue to separate two commands addressing the same chip select of different types where the write is going to execute before the read. The placement of the new commands will still follow priority and address collision rules. Write-to-read splitting is enabled through the “W2R_SPLIT_EN” parameter.

(5) Read/Write Grouping

The memory suffers a small timing overhead when switching from read to write mode. For efficiency, the placement logic will attempt to place a new read command sequentially with other read commands in the command queue, or a new write command sequentially with other write commands in the command queue. Grouping will only be possible if no priority or address collision rules are violated.

Read/write grouping is enabled through the “RW_SAME_EN” parameter.

(a) Bank Conflicts and Read/Write Grouping

If the new command addresses the same chip select and same bank, but a different row, as a command currently in the command queue, these commands are considered to have a bank conflict. As described in **Section 6.5.4.1(3), Bank Splitting**, the placement logic will attempt to separate commands with bank conflicts. For this controller, certain placements are prohibited for read/write grouping to support ideal bank splitting.

These checks are controlled through the “DISABLE_RW_GROUP_W_BNK_CONFLICT” parameter. If bit [0] of this parameter is set to 1, a new command will be prohibited from placement in the entry directly before or directly after the command with a bank conflict. If bit [1] of this parameter is also set to 1, the new command will also be prohibited from being placed two entries before or two entries after the command with a bank conflict. The following table shows a simplified command queue.

Table 6.159 Simple Command Queue Example

Entry	Read/Write	Bank	Row
0	Rd	0	0
1	Rd	0	0
2	Rd	0	0
3	Rd	1	0
4	Rd	0	0
5	Rd	0	0
...			

For this example, a new entry is received that is a READ to BANK 1, Row 1. Assume that no priority or address collision rules are violated. This new command would have a bank conflict with Entry 3.

- If DISABLE_RW_GROUP_W_BNK_CONFLICT[0] = 1, the new command could not be placed immediately before or after the conflicting command. If the command was placed into entry 3, entries 3-5 would be moved to entries 4-6, and the bank conflicts would occur between entries 3 and 4. If the command was placed into entry 4, entries 4-5 would be moved to entries 5-6 and the bank conflict would still occur between entries 3 and 4. Therefore, entries 3 and 4 are prohibited for placement.

- If DISABLE_RW_GROUP_W_BNK_CONFLICT[1] = 1, the new command could not be placed two entries before or two entries after the conflicting command. If the command was placed into entry 2, entries 2-5 would move to entries 3-6, and the bank conflict would occur between entries 2 and 4. If the command was placed into entry 5, the bank conflict would occur with between entries 3 and 5. Therefore, entries 2 and 5 are also prohibited for placement.
- Therefore, if the “DISABLE_RW_GROUP_W_BNK_CONFLICT” parameter was set to 2’b11, the new entry could only be placed at entry 0, 1 or 6, allowing at least 2 commands in between the conflicting commands.

CAUTION

It is not meaningful to set bit [1] of the “DISABLE_RW_GROUP_W_BNK_CONFLICT” parameter without bit [0].

(b) Chip Select Grouping with Read/Write Grouping

When attempting to group read and write commands, the placement logic will also consider the chip select for the commands. If possible, read commands will be grouped with read commands to the same chip select, and write commands with write commands to the same chip select. If chip select grouping is not possible, commands will still be grouped by command type if the “RW_SAME_EN” parameter is set to 1. If read/write grouping is disabled (“RW_SAME_EN” is cleared to 0), chip select grouping will have no effect.

Chip select grouping is enabled through the “CS_SAME_EN” parameter.

(c) Page Grouping with Read/Write Grouping

When attempting to group read and write commands, the placement logic will also consider the page for the commands. If possible, read commands will be grouped with read commands to the same page, and write commands with write commands to the same page. If page grouping is not possible, commands will still be grouped by command type if the “RW_SAME_EN” parameter is set to 1. If read/write grouping is disabled (RW_SAME_EN is cleared to 0), page grouping will have no effect.

Page grouping is enabled through the “RW_SAME_PAGE_EN” parameter.

6.5.4.2 Command Execution Order After Placement

Once a command has been placed in the command queue, selection logic will be used to determine how to pull commands from the queue for execution. This logic may be disabled by setting the “IN_ORDER_ACCEPT” parameter to 1, resulting in the command queue executing the commands in the order that they are placed relatively in the command queue. If the “IN_ORDER_ACCEPT” parameter is cleared to 0, the selection logic will be utilized. Regardless of the setting of this parameter, high-priority command swapping and command aging are provided which may affect commands after they have been placed into the command queue.

(1) Command Selection Logic

On each clock cycle, the selection logic will scan the top 4 entries of the command queue to determine which command to execute. This value is defined at configuration. Commands are considered for execution based on bank readiness, availability of at least 1 burst of data (writes), availability of storage for at least 1 burst of data (reads), bus turnaround timing (JEDEC-specified and programmable) and conflicts. Similar to the placement rules, a command will not be executed before a command that was placed ahead of it in the command queue if there are any address or bank conflicts.

All placement rules mentioned in this chapter are followed by the selection logic other than priority. It is possible that lower priority commands may be executed ahead of higher priority commands if the higher priority commands are not ready to execute, provided that there are no conflicts with commands ahead in the command queue. The controller will also not execute a read/modify/write sequence before another read/modify/write sequence ahead of it in the command queue due to limited storage in the Controller core.

The selection feature is disabled through the “IN_ORDER_ACCEPT” parameter. If this parameter is set to 1, only the top entry of the command queue will be considered for execution. **Figure 6.7, Selection Logic** shows the command selection logic relative to the rest of the placement logic.

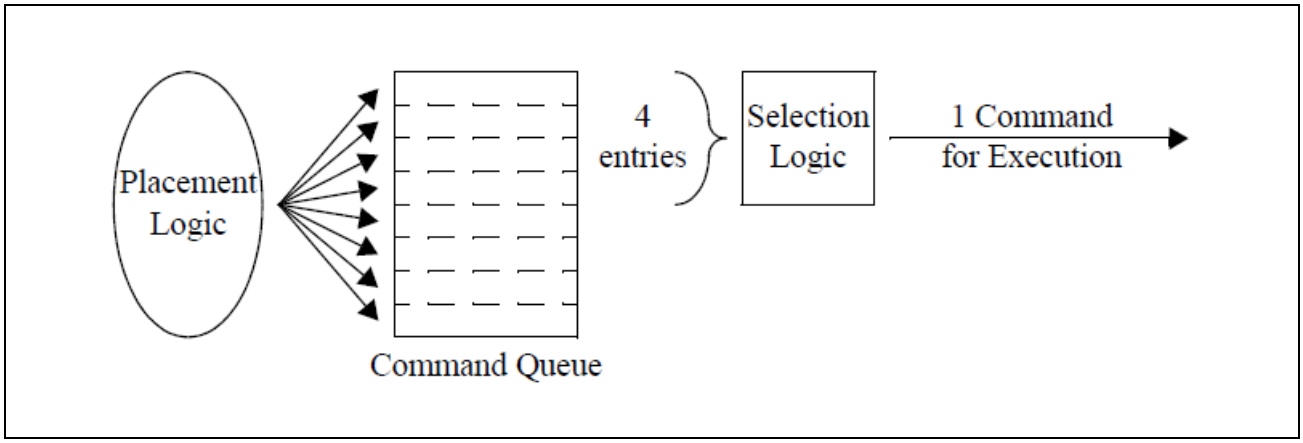


Figure 6.7 Selection Logic

(2) High-Priority Command Swapping

Commands are assigned priority values to ensure that critical commands are executed more quickly in the controller than less important commands. Therefore, it is desirable that high-priority commands pass into the Controller core as soon as possible. The placement algorithm takes priority into account when determining the order of commands, but still allows a scenario in which a high-priority command sits waiting in the command queue while another command, perhaps of a lower priority, is in process.

The high-priority command swapping feature allows this new high-priority command to be executed more quickly. If the user has enabled the swapping function through the “SWAP_EN” parameter, then the behavior of the swapping logic will be dependent on the value of the “IN_ORDER_ACCEPT” parameter. If the command queue must be executed in order (IN_ORDER_ACCEPT = 1), the entry at the top of the command queue will be compared with the current command in progress. If the selection logic is being used (IN_ORDER_ACCEPT = 0), the top 4 entries of the command queue will be compared with the current command in progress to determine which command may need to be executed first. If the selected command queue entry is of a higher priority (not the same priority) and it does not have an address conflict with the current command being executed, then the original command will be interrupted.

If the command is to be interrupted, it will be halted after completing the current burst, stored and placed at the top of the command queue, and the new command will begin executing. As long as the command queue is not full, new commands may continue to be inserted into the command queue based on the placement rules, even at the top of the queue ahead of the interrupted command. The selection logic will determine the command to execute next. Whenever the interrupted command is resumed, it will start from the point at which it was interrupted.

Note that priority 0 commands will never be interrupted, so the user should set any commands that should not be interrupted to priority 0.

(3) Command Aging

Since commands can be inserted ahead of existing commands in the command queue, the situation could occur where a low priority command remains at the bottom of the queue indefinitely. To avoid such a lockout condition, aging counters have been included in the placement logic that measure the number of cycles that each command has been waiting. If an aging counter hits its maximum, the priority of the associated command will be decremented by one (lower priority commands are executed first). This increases the likelihood that this command will be picked by the selection logic and execute. Note that this command does not move relative positions in the command queue when it ages; the new priority will be considered when placing new commands into the command queue.

Aging is controlled through a master aging counter and command aging counters associated with each command in the command queue. The AGE_COUNT and “COMMAND_AGE_COUNT” parameters hold the initial values for each of these counters, respectively. When the master counter counts down the “AGE_COUNT” parameter value, a signal is sent to the command aging counters to decrement. When the command aging counters have expired, the priority of the associated command is decremented and the counter is reset. Therefore, a command does not age by a priority level until the total elapsed cycles has reached the product of (AGE_COUNT + 1) and (COMMAND_AGE_COUNT + 1). The maximum number of cycles that any command can wait in the command queue until reaching the top priority level is the product of (AGE_COUNT + 1), (COMMAND_AGE_COUNT + 1), and the number of priority levels in the system.

If command swapping is enabled, it is possible that a command in progress could be interrupted by a higher priority command in the command queue. This situation could arise if a new higher-priority command arrives and is placed while the current command is in progress, or by a command in the queue aging to a lower priority while the current command is in progress. An interrupted command will be placed at the top of the command queue. Command aging will always be enabled.

6.5.4.3 ACT Request Control

The DDR Controller provides a means to limit which commands of the command queue may issue ACT requests. This can be used to prevent situations in which an ACT is issued for a command in the queue, and before that command can be executed, a new command is placed ahead of it which accesses the same bank but a different row. This would require a PRE-ACT sequence that may have been avoided if the first ACT was never issued.

This functionality is controlled through the “NUM_Q_ENTRIES_ACT_DISABLE” parameter, which specifies the number of entries of the command queue in which ACT requests are not allowed. For this 8-deep command queue, the entries are numbered 0-7, where entry 0 is the command next to execute.

Table 6.160 Programming of NUM_Q_ENTRIES_ACT_DISABLE Parameter

Value	Effect
3’b000	ACT request can occur from entries 0-7
3’b001	ACT request can occur from entries 0-6
3’b010	ACT request can occur from entries 0-5
3’b011	ACT request can occur from entries 0-4
3’b100	ACT request can occur from entries 0-3
3’b101	ACT request can occur from entries 0-2
3’b110	ACT request can occur from entries 0-1
3’b111	ACT request can occur from entries 0

6.5.5 DRAM Command Processing

The DRAM command processing logic is used to process the commands in the command queue. The logic organizes the commands to the memories in such a way that data throughput is maximized. Bank opening and closing cycles are used for data transfers.

The logic uses a variety of factors to determine when to issue bank open and close commands. The logic reviews the entire command queue for look-ahead of which banks are to be accessed in the future. The timing is then set to meet the “TRC” and “TRAS_MIN” timing parameters of the memories, values which were programmed into the controller on initialization. This flexibility allows the controller to be tuned to extract the maximum performance out of memories. The parameters that relate to DRAM memory protocol are listed in the **Section 6.4.1, DDR Controller Register Description**.

6.5.6 ECC Function

The DDR Controller provide an optional error reporting and correcting circuitry that can be used to verify data in memory and correct memory errors if they occur. The logic will check for errors in both the data and the check code on all read transactions. ECC, or error checking and correcting, is the process of detecting bit errors in the memory data and if possible, correcting them. This function can confirm the accuracy of data and remove or at least identify bit errors. ECC works by storing unique “check codes” in memory. A check code is a mathematical description of the information in an aligned segment of memory known as an “ECC data word”. The check code is always related to the entire ECC data word, and is used inside the DDR Controller on all memory reads to control data accuracy. Check codes are not input from, or output to, the user interface.

An ECC data word cannot start and end at any random address; these words are memory aligned to their size. The starting addresses of ECC data words are defined as ECC word boundaries and the alignment of user transactions to these boundaries determines how transactions are processed inside the DDR Controller. If the user does not wish to use the ECC option, the ECC module may be disabled by setting the control parameter “ECC_EN” to 0. This DDR Controller supports a 32-bit ECC data word size. A 7-bit check code is maintained for each 32-bit memory area. ECC word boundaries fall on each 4-byte address (0xN0, 0xN4, 0xN8,0xNC).

6.5.6.1 ECC Error Types

An ECC error is defined in the DDR Controller as “correctable” or “un-correctable”. A correctable error is a single bit error in the check code or the data. The controller uses the syndrome to determine which bit is erroneous and is able to correct the error. An un-correctable error is a double bit error in the check code and/or the data. The controller is able to identify that 2 bits in the check code and/or the data are incorrect, but cannot determine exactly which 2 bits are erroneous, and therefore cannot correct the error.

CAUTION

The DDR Controller can detect single bit and double bit errors. Errors in more than 2 bits of the check code and/or the data will cause the controller to behave unpredictably.

6.5.6.2 Features of the ECC Logic

The Controller ECC implementation provides the following features:

- Internal controls to disable individual I/O pads or to disable the ECC module

  The DDR Controller provides disable signals to ignore certain or all bits of the memory datapath. This is useful with the Controller reduced datapath feature, with specific ECC memories, or when there is no ECC memory available. The user may disable the ECC module completely by programming the “ECC_EN” parameter to 0. In this mode, the controller will not compute and compare check codes, nor attempt to store the check codes to memory.

- Register storage of the ECC signature on ECC errors

  Two sets of parameters (one set for correctable errors, one set for un-correctable errors) store the address, data, and syndrome associated with the ECC error on a read command.

- Interrupt generation

  Four interrupt bits in the register array communicate ECC status to the user for read commands.

- Automatic correction of single-bit errors

  If single bit error occurs, the erroneous bit will be flipped automatically and accurate data/check codes will be written to memory (write transactions) or returned to the user interface (read transactions). Note that corrections made on read data will not be reflected to memory.

- Memory integrity preservation through ECC scrubbing

  ECC scrubbing will ensure that the entire combined ECC data word will be written to memory to preserve memory integrity. ECC scrubbing does not automatically correct memory errors on read operations, but does allow a simple way for the user to correct errors in memory. If a correctable error is detected on a read, and ECC reporting and correction is enabled, the controller will fix the data prior to returning it to the user interface and flag the existence of the error in memory. However, the error still exists in the memory.

  With ECC scrubbing, the user may respond to a correctable error by issuing a write to this memory location with all bytes masked. This will trigger a read/modify/write operation where the controller will read in the data from memory, discover and correct the error and then write the data and the check code to memory. Since all bytes of the data are written to memory, the error is overwritten - or scrubbed - from memory. ECC scrubbing requires that ECC correction and reporting are both enabled.

- Automatic corruption of ECC codes for read data errors

  The read data is verified for ECC prior to combining the read data with the new write data and calculating a new ECC. If an un-correctable error is detected on the read data, it is likely that this address space contains bad memory. As a result, the Controller will automatically corrupt the check code for this ECC data word so that all future accesses to this area will result in an error. This feature may be disabled. If disabled, the error will be ignored and new data and a new check code will be written to memory. A future access to this address may or may not result in an un-correctable error.

- ECC error forcing

  The user can force a specific check code to be written into memory for diagnostic purposes or for flagging a particular memory address as erroneous for future accesses. When a write check is initiated, the programmed value will be XOR’ed with the generated ECC bits and written to memory. When this ECC data word is read back, the ECC error event will be detected.

6.5.6.3 ECC Control

ECC functionality is controlled through a parameter named “ECC_EN”. This parameter enables ECC and sets the reporting and correcting behavior. To enable the ECC function, set “REDUC” to 1 to enable the half datapath feature. The rest of the datapath is used for the ECC check code, so the data area is halved.

(1) Error Signature Parameters

Register storage is provided for the failing address, data, and syndrome for ECC single bit and double bit errors for read operations. There is one set of parameters for each type of error.

Note that the controller is only able to report errors within a half 64-bit-defined-word range. Therefore, if an error occurs, the address reported in the “ECC_C_ADDR” or “ECC_U_ADDR” parameter will be the previous 32-bit boundary of the erring location.

The signature information will be stored until the user reads these register locations. Subsequent errors that occur before the error reporting parameters are read will trigger an interrupt, but their error signature will be lost.

(2) Interrupt Status Bits

The “INT_STATUS” parameter indicates the status of all interrupts in the DDR Controller. The parameter is cleared on reset of the controller. There are 4 bits in the “INT_STATUS” parameter relating to ECC.

Bits [4] and [6] will only be set if a subsequent ECC error occurs before the initial ECC error has been acknowledged by setting the associated bit in the “INT_ACK” parameter to 1. When any of these errors occur, the associated bit will be set in the “INT_STATUS” parameter. If the associated interrupt is not masked, the DDRC_Int signal will also be triggered to the Global Interrupt Controller. An interrupt is masked if the associated bit is set to 1 in the “INT_MASK” parameter.

6.5.6.4 Syndromes

On read commands, the controller will retrieve ECC data words from memory and their associated check codes. The Controller will generate a check code based on the data that it has read from memory. The checking function will XOR this value with the check code read from memory. The result of the XOR operation between a generated check code and a stored check code is known as a syndrome. If the syndrome is anything other than 0x00, an ECC error event has occurred. Incorrect bits could be located in the data or in the saved check code. In either case, the data and the check code do not match. The syndrome identifies which bit (on a single bit error) of the ECC data or check code is incorrect. If a syndrome does not deterministically identify which particular data bit which must be flipped, the error is a double bit or multi-bit error and cannot be corrected.

The following **Table 6.161, 32-Bit ECC Syndromes** displays the syndromes that correspond to single bit errors. If the syndrome is 0x00, then there was no ECC error. Any values other than the ones shown relate to double bit or multi-bit errors.

Table 6.161 32-Bit ECC Syndromes

Syndrome	Incorrect Bit	Syndrome	Incorrect Bit	Syndrome	Incorrect Bit
0x00	No Error	0x20	Check [5]	0x57	Data [11]
0x01	Check [0]	0x23	Data [23]	0x58	Data [10]
0x02	Check [1]	0x25	Data [22]	0x5B	Data [9]
0x04	Check [2]	0x26	Data [21]	0x5E	Data [8]
0x08	Check [3]	0x29	Data [20]	0x62	Data [7]
0x0B	Data [31]	0x2A	Data [19]	0x64	Data [6]
0x0E	Data [30]	0x2C	Data [18]	0x67	Data [5]
0x10	Check [4]	0x31	Data [17]	0x68	Data [4]
0x13	Data [29]	0x34	Data [16]	0x6B	Data [3]
0x15	Data [28]	0x40	Check [6]	0x6D	Data [2]
0x16	Data [27]	0x4A	Data [15]	0x70	Data [1]
0x19	Data [26]	0x4F	Data [14]	0x75	Data [0]
0x1A	Data [25]	0x52	Data [13]		
0x1C	Data [24]	0x54	Data [12]		

CAUTION

Any other syndrome than the above table relate to double bit or multi-bit errors.

6.5.6.5 Command Processing when ECC is Enabled

A read command to the DDR Controller will always result in reading complete ECC data words of information, and their associated check codes, from the DRAM memory. The information will first be read into the Controller core where a check code will be generated for each read ECC data word. This generated code will be compared with the check code read from memory. If the two check codes do not match, then an error has occurred. Entire ECC data words containing the starting and ending addresses will be returned to the requestor; the requestor must ignore bytes that are not relevant.

A write command to the DDR Controller will trigger an internal read command before the write occurs. A read will be required because this allows the controller knowledge of the entire ECC data word for accurate check code generation. Once the data has been read into the controller core, a check code will be computed on the read data. Errors will be reported or corrected if possible. Then the read data will be combined with the new write data to create a new ECC data word.

A check code will be generated on the new ECC data word, whether all the data is new or only some of it. This code will be stored in the memories along with the new ECC data word and will be available to the controller when that data is required by the user interface.

The behavior of the ECC logic on each of the commands is shown in the following **Table 6.162, ECC Functionality on Various Transaction Types**.

Table 6.162 ECC Functionality on Various Transaction Types (1/2)

Type of Command	Starting Address	Ending Address	Internal Command	Check Code		
				Generated?	Compared?	Saved?
Read	Any	Any	Reads of Complete ECC Data Words	Yes	Yes	No
Write	Aligned	Aligned	Writes of Complete ECC Data Words	Yes	No	Yes
Write	Aligned	Unaligned	1 Read prior to the Writes. All writes of complete ECC data words.			
			Read of the ECC data word containing the ending address.	Yes	Yes	No
			Writes of complete ECC data words of new write data.	Yes	No	Yes
			Final ECC data word written is a combination of read data and new write data (RMW).	Yes	No	Yes
Write	Unaligned	Aligned	1 Read prior to the Writes. All writes of complete ECC data words.			
			Read of the ECC data word containing the starting address.	Yes	Yes	No
			First ECC data word written is a combination of read data and new write data (RMW).	Yes	No	Yes
			Writes of complete ECC data words of new write data.	Yes	No	Yes
Write	Unaligned	Unaligned	2 Reads prior to the Writes. All writes of complete ECC data words.			
			Read of the ECC data word containing the starting address.	Yes	Yes	No
			First ECC data word written is a combination of first word read data and new write data (RMW).	Yes	No	Yes
			Writes of complete ECC data words of new write data.	Yes	No	Yes
			Read of the ECC data word containing the ending address.	Yes	Yes	No
			Final ECC data word written is a combination of last word read data and new write data (RMW).	Yes	No	Yes

Table 6.162 ECC Functionality on Various Transaction Types (2/2)

Type of Command	Starting Address	Ending Address	Internal Command	Check Code		
				Generated?	Compared?	Saved?
Masked Write	Any	Any	Masked Writes are limited to 1 64-bit-defined-word in length. Therefore there will be 2 RMW operations. All writes of complete ECC data words.			
			Read of the ECC data word containing the starting address.	Yes	Yes	No
			First ECC data word written is a combination of first word read data and new write data (RMW).	Yes	No	Yes
			Read of the ECC data word containing the ending address - if applicable.	Yes	Yes	No
			Final ECC data word written is a combination of last word read data and new write data (RMW) - if applicable.	Yes	No	Yes

6.5.6.6 ECC and Read Operations

If ECC is disabled (the “ECC_EN” parameter is cleared to 0), read requests will pull data from memory and return the data to the user interface. No ECC checking will occur and no error information will be saved or reported. If ECC is enabled (the “ECC_EN” parameter is set to 1), ECC behavior will follow the information shown in **Table 6.162, ECC Functionality on Various Transaction Types**. On all reads, even read requests that do not span an entire ECC data word, the entire ECC data word that contains the starting address will be read to the DDR Controller core and returned to the user. If the request did not encompass the entire ECC data word, the user will need to ignore the extra bytes. The stored check code will also be read to the controller core. The entire ECC data word and the stored check code are required for ECC verification. The internal logic will generate a check code for this data and XOR the generated code with the value read from memory. The result of the XOR operation between a generated check code and a stored check code is known as a syndrome. The action of the Controller is dependent on the syndrome:

- The syndrome is 0x00

  The computed check code and the stored check code match and therefore the data is accurate. The Controller will send the data to the user interface. No errors occurred, so the ECC error parameters will not be updated, no interrupts will occur and the user-interface signals will not be asserted.

- The syndrome indicates a single-bit error

  The error is in the data or the check code, but the Controller is aware of exactly which bit is erroneous by the value of the syndrome. Even though the bit can and will be corrected, the error will still be flagged as a “correctable” error. The Controller will store the address, data, and syndrome in the ECC correctable error parameters, and either the first correctable ECC error interrupt (bit [3]) or the second correctable ECC error interrupt (bit [4]) will be set in the “INT_STATUS” parameter. The Controller will use the syndrome information to correct the erroneous bit and then the correct data will be sent to the user interface. The erroneous data or check code bit in memory will not be changed.

- The syndrome indicates a multi-bit error

  The error is in the data or the check code, but the Controller cannot identify which bits are erroneous. The Controller will store the address, data, and syndrome in the ECC un-correctable error parameters, and either the first un-correctable ECC error interrupt (bit [5]) or the second un-correctable ECC error interrupt (bit [6]) will be set in the “INT_STATUS” parameter. The incorrect data will be sent to the user interface. In addition, an error response will be sent on the “axiY_RRESP” signal. For default transfers, the error will be sent with the beats that caused the error. If the error was associated with a narrow transfer, then the error will be sent with each beat of the erroneous data word.

6.5.6.7 ECC and Write Operations

If ECC is disabled (the “ECC_EN” parameter is cleared to 0), write requests will write the data from memory. If a read/modify/write operation is required, data will be pulled from memory, combined with the new data and then written to memory. ECC check codes will not be read from memory, no ECC checking will occur and no error information will be saved or reported. If ECC is enabled (the “ECC_EN” parameter is set to 1), all ECC writes are completed as read/modify/write operations. Since the value of the check code depends on the entire ECC word, and it is possible that only some of the bytes of the ECC word are new data, the controller cannot compute a check code on this data. As a result, these types of operations are performed as read/modify/write operations. A read/modify/write operation consists of these steps:

- Reading the ECC Data Word

  The ECC data word containing the write data address, and the corresponding check code, will be read to the DDR Controller core.

- Validating the Data

  The controller will generate a check code from the read data and XOR this value with the check code read from memory to determine the syndrome. The action of the Controller is dependent on the syndrome:

  - The syndrome is 0x00. The computed check code and the stored check code match and therefore the data is accurate.
  - The syndrome indicates a single-bit error. The error is in the data or the check code, but the Controller is aware of exactly which bit is erroneous by the value of the syndrome. Even though the bit can and will be corrected, the error will still be flagged as a “correctable” error. The Controller will store the address, data, and syndrome in the ECC correctable error parameters, and either the first correctable ECC error interrupt (bit [3]) or the second correctable ECC error interrupt (bit [4]) will be set in the “INT_STATUS” parameter. The Controller will use the syndrome information to correct the erroneous bit.
  - The syndrome indicates a multi-bit error. The error is in the data or the check code, but the Controller cannot identify which bits are erroneous. The Controller will store the address, data, and syndrome in the ECC un-correctable error parameters, and either the first un-correctable ECC error interrupt (bit [5]) or the second un-correctable ECC error interrupt (bit [6]) will be set in the “INT_STATUS” parameter. The Controller will automatically corrupt the check code for this ECC data word if ECC corruption is enabled (the “ECC_DISABLE_W_UC_ERR” parameter is cleared to 0). This ensures that a future access to this location will reveal an ECC error. If the “ECC_DISABLE_W_UC_ERR” parameter is set to 1, the check code will not be corrupted and the un-correctable error will be lost.

- Combining the Data

  The corrected or un-corrected 64-bit-defined-word will be combined with the new write data. If the check code was intentionally corrupted during the read phase, the new check code will also be corrupted. Otherwise, the controller will calculate a new check code based on the combined data.

- Writing the Information

  The new check code [accurate or corrupted] and the entire 64-bit-defined-word will be written to memory.

6.5.6.8 Automatic ECC Corruption

For ECC accuracy, data is verified for ECC during the read phase of a write prior to modifying and writing. If an un-correctable error is detected on the read data, it is likely that this address space contains bad memory. As a result, the Controller will automatically corrupt the check code for this ECC data word so that all future accesses to this area will result in an error. The corruption occurs during both the read and the write phases of the write transaction. If desired, this feature may be disabled by setting the “ECC_DISABLE_W_UC_ERR” parameter. If disabled, the un-correctable error will be ignored. The read data and write data will be combined, and a check code will be calculated to match the new ECC data word. New data and the new check code will be written to memory. A future access to this address may or may not result in an un-correctable error.

6.5.6.9 Forcing an ECC Error Event

There are situations where the user may wish to force an ECC error. This could be used for testing purposes or to tag a particular memory location as erroneous. The latter could be used when an un-correctable error occurs on a write operation and the “ECC_DISABLE_W_UC_ERR” parameter was accidentally set to 1 at the time. By having corruption disabled, the ECC error was lost. By forcibly corrupting the check code in memory, an ECC error will occur on subsequent reads of this location. The procedure for forcing an ECC event is as follows:

(1) Set the “ECC_EN” parameter to 1. This will enable ECC checking.

(2) Ensure that no writes to the controller are pending.

(3) Write a value to the “XOR_CHECK_BITS” parameter that will trigger an ECC event once that word is read. Use the syndromes listed in **Section 6.5.6.4, Syndromes** to program the “XOR_CHECK_BITS” parameter. Each byte of the “XOR_CHECK_BITS” parameter controls the ECC event forcing for a separate 64-bit-defined-word space. **Table 6.163, Parameter XOR_CHECK_BITS Mapping** shows how each byte of the “XOR_CHECK_BITS” parameter maps to 64-bit-defined-word bits. For example, to force a single bit correctable error on bit 0 of the 64-bit-defined-word space shown, write 0x75 into that byte of the “XOR_CHECK_BITS” parameter. To force a double bit un-correctable error for the 64-bit-defined-word space, write 0x03 into that byte of the “XOR_CHECK_BITS” parameter.

(4) Assert the FWC (force write check) parameter using the register interface.

(5) Execute a write command to the controller for an aligned 64-bit-defined-word.

(6) The next read command to the same address will force the ECC error. This action also automatically clears the “FWC” parameter bit to 0.

(7) Depending on the programming of the “XOR_CHECK_BITS” parameter, a single bit, double bit or multi-bit ECC error will occur. For single bit and double bit errors, the appropriate bit in the “INT_STATUS” parameter will be set to 1 and the ECC error signature parameters will be filled with the relevant information. Refer to **Table 6.163, Parameter XOR_CHECK_BITS Mapping** for details on how to force a bad ECC syndrome.

Table 6.163 Parameter XOR_CHECK_BITS Mapping

Mapping	Description
XOR_CHECK_BITS [13:7] maps to 64-bit-defined-word [63:32] XOR_CHECK_BITS [6:0] maps to 64-bit-defined-word [31:0]	This value will be XOR’ed with the generated ECC bits and then written to memory when the FWC bit is also set to 1. When this 64-bit-defined-word is read back, an ECC event will be detected. For codes to force a particular event, refer to the ECC syndromes table.

6.5.6.10 Clearing a Reported ECC Event

To clear a reported ECC event, the user should follow these steps:

- Read the ECC data, address and syndrome parameters to determine where the event occurred.
- Set the associated bit in the “INT_ACK” parameter to 1. This clears the ECC interrupt as well as the ECC event parameters, and allows future events to be captured.

6.5.7 Low Power Control Management

In many applications, it is desirable to minimize the power consumption of the DDR Controller and the memories. The DDR Controller provides various user-configurable low power options to address power savings. The low power logic is located in the Low Power Control (LPC) module.

6.5.7.1 Low Power States

CAUTION

- Transitions will only be made into deeper low power states. If the user requires to switch to a higher power state, the current state must be exited and then the new state entered.
- Low power state transitions that may be completed without a low power exit will be performed when possible. The system will include low power exits if necessary when switching from one low power state to another.

(1) Active Power-Down

The controller sets the memories into power-down while any row is active in the bank. This state reduces the overall power consumption of the system, but has the least effect of all the low power states. In this state, the controller and memory clocks are fully operational, but the CKE input bit to the memories is de-asserted. If entry into the “Active Power-Down” state was requested, the memory will enter either active or pre-charge power-down mode depending on the state of the rows. If there are no open rows, pre-charge power-down mode will be entered.

The controller will continue to monitor memory refresh needs and will automatically bring the memory out of power-down to perform these refreshes. When a refresh is required, the CKE input bit to the memories will be re-enabled. This action brings the memories out of power-down. Memory that has been transitioned into active power-down mode will automatically transition to pre-charge power-down mode after the controller performs a refresh since the refresh process includes a pre-charge all command.

For DDR2, active power-down mode supports both fast and slow exit modes depending on how the memory mode registers are programmed. If the active power-down bit (MR [A12]) is cleared to 0 (fast exit), the timing parameter TXARD will define the exit time from this low power state to a read command. If the active power-down bit is set to 1 (slow exit), the timing parameter TXARDS will be used.

(2) Active Power-Down with Memory Clock Gating

CAUTION

This low power state is NOT supported for standard DDR2 or DDR3 memories. When set into this state, the controller will attempt to place the memories in power-down and gate off the memory clock. The memory will function unpredictably and may hang.

(3) Pre-Charge Power-Down

The controller sets the memories into power-down once all banks are idle. If any rows are active, prior to issuing the power-down mode command, the controller will issue a pre-charge all command. If any NVM memories are present in the system, the pre-charge command is not relevant and will not be sent to the NVM memories.

The controller will continue to monitor memory refresh needs and will automatically bring the memory out of power-down to perform these refreshes. When a refresh is required, the CKE input bit to the memories will be re-enabled. This action brings the memories out of power-down. Once the refresh has been completed, the memories will be returned to pre-charge power-down mode following the de-assertion the CKE input bit.

For DDR3, pre-charge power-down mode supports both fast and slow exit modes depending on how the memory mode registers are programmed. If the pre-charge power-down bit (MR0 [A12]) is cleared to 0 (slow exit), the timing parameter “TXPDLL” will define the exit time from this low power state to a read command. If the pre-charge power-down bit is set to 1 (fast exit), the timing parameter “TPDEX” will be used.

(4) Pre-Charge Power-Down with Memory Clock Gating

CAUTION

This low power state is NOT supported for standard DDR2 or DDR3 memories. When set into this state, the controller will attempt to place the memories in power-down and gate off the memory clock. The memory will function unpredictably and may hang.

(5) Self-Refresh

The controller sets the memories into self-refresh mode. In this low power mode, the controller and memory clocks are fully operational and the CKE input bit to the memories is de-asserted. Since the memory automatically refreshes its contents, the controller does not need to send explicit refreshes to the memory.

(6) Self-Refresh with Memory Clock Gating

The controller sets the memories into self-refresh and gates off the clock to the memories. Before the memories are removed from self-refresh, the clock will be gated on again.

(7) Self-Refresh with Memory and Controller Clock Gating

This is the deepest low power state of the controller. The controller sets the memories into self-refresh and gates off the clock to the memories. In addition, the clock to the controller will be gated off (except the AHB Register interface).

Before the memories are removed from self-refresh, the controller and memory clocks will be gated on. If automatic exit from this state is enabled, a new transaction that addresses a memory device in this state will wake up the memory to process the transaction.

CAUTION

- This state should not be entered when a read or write command is being processed. When using the software programmable interface, the user should ensure that the controller is idle before requesting entry into this state by checking the “CONTROLLER_BUSY” parameter. When using the automatic interface, the DDR Controller will complete this check.
- When the controller clock is gated through the low power control module, writes to any of the programmable command parameters - parameters that result in command execution within the controller - will be ignored and not result in execution of the associated commands. Any commands issued while the controller clock is gated may or may not be executed once the controller clock is un-gated. In general, command registers should not be programmed during controller clock gating as the commands will have undefined results.

6.5.7.2 Management of the Low Power Control Module

(1) Interfaces

There are two means to manage the LPC module:

- Software Programmable Interface

  This interface uses programmable parameters in the DDR Controller registers and status reporting to manage low power command activity.

- Automatic Interface

  This interface supports automatic entry and exit for all low power states with separate enables and counters for each low power state.

Table 6.164 Low Power State Management

	Interfaces	
Low Power State	Software Programmable	Automatic
Normal Mode	—	—
Active Power-Down	Yes	Yes
Pre-Charge Power-Down	Yes	Yes
Self-Refresh	Yes	Yes
Self-Refresh with Memory Clock Gating	Yes	Yes
Self-Refresh with Memory and Controller Clock Gating	Yes	Yes
Shutdown	No	No

CAUTION

Memory clock gating with power-down is NOT supported for DDR2 and DDR3 memories.

(2) Low Power Arbiter

Since the LPC module may be managed by multiple masters, an arbitration scheme has been implemented that only allows one interface (software programmable, automatic) to execute low power commands at a time. Each interface may maintain ownership of the LPC module as long as it maintains its request.

The software programmable interface and automatic interfaces generate request signals with a command to win arbitration of the LPC module. They lose arbitration after their request has completed. The software programmable interface does include a lock option to hold arbitration for another command. This lock is not failsafe.

Commands will only be executed once the low power arbiter grants control to the interface associated with the operation and only when the LPC module is not executing a command. The interfaces have the following priorities:

(1) Software programmable interface
(2) Automatic interface

6.5.7.3 Software Programmable Interface

The software programmable interface provides a single programmable parameter “LP_CMD” which is used to request entry into or exit from any of the supported low power states. When the user programs the “LP_CMD” parameter with a supported value, if the module is idle “LP_ARB_STATE = 4’b0000”, the software programmable interface will win arbitration of the LPC module. This will be reflected in the “LP_ARB_STATE” parameter being set to 4’b0001 (or 4’b1001 if the “LOCK” bit (bit 7) was set when the command was requested). If the arbitration status is already defined for the software programmable interface “LP_ARB_STATE = 4’b0001 or 4’b1001”, the command will continue without any change to this parameter. If any other interface has won arbitration, the software command will remain pending until it can execute. There is no way to abort the command once it has been issued.

When the interface wins arbitration, the command defined in the “LP_CMD” parameter will be passed to the LPC module. The “LP_CMD” parameter is encoded to initiate an entry or exit, and to specify the low power state requested. Once the command is accepted, the “VALID” bit (bit 5) of the “LP_STATE” parameter will be cleared to 0 and the command processed.

The action taken by the LPC module depends on the low power state request and the state of the LPC module at the time that the request is received. If the memory is currently in a low power state and the exit bit is set in the command request, the LPC module will trigger an exit of the current low power state. If the entry bit is set, the LPC module will review the specified low power state. If the memory is not in a low power state, or the request is for a deeper low power state than the current state, the LPC module will trigger an entry into the new low power state. If the request is for the same or a lesser low power state than the current state, no action will be taken by the LPC module.

Once the command has completed (or if no action is taken), the “VALID” bit (bit 5) of the “LP_STATE” parameter will be set to 1, the new state will be reflected in the low power state bits (bits [4:0] of the “LP_STATE” parameter), and the low power command complete interrupt (bit 9) will be set to 1 in the INT_STATUS parameter. If the command was issued without the “LOCK” bit set “LP_CMD[7]”, the arbitration parameter will be reset to idle “LP_ARB_STATE = 4’b0000”. If the command was issued with the “LOCK” bit set “LP_CMD[7]”, the low power arbiter will hold the arbitration in software programmable interface control with the lock indicator set “LP_ARB_STATE = 4’b1001”. If a high-priority request breaks the software programmable interface’s lock, the system may require a system reset.

While the “VALID” bit (bit 5) of the “LP_STATE” parameter is cleared to 0, no additional commands will be accepted into the LPC module. Any writes to the “LP_CMD” parameter will result in unpredictable behavior.

(1) Lock Option

The software programmable interface provides the option to lock the arbitration. Since the LPC module may be modifying the low power state of the memory at any moment, reading the “LP_STATE” parameter even immediately prior to issuing a low power command through the software programmable interface does not guarantee that the low power state of the system has not changed since the parameter was read. One way to be sure that the state does not change from a read of the “LP_STATE” parameter to the execution of an “LP_CMD” is to lock the LPC module from access by any other interfaces. This can be done by issuing a lock without an entry into any low power state “LP_CMD = 8’b100_000_00”. This will gain control of the LPC module, define a specific state and set the low power command complete interrupt (bit 9) in the INT_STATUS parameter to 1. The only difference between this command and any other low power command issued from the software programmable interface is that at the end of the command, the “LP_ARB_STATE” parameter is not reset to idle (4’b0000) but instead remains at 4’b1001 for software programmable interface control with a lock. Based on the current state, the software programmable interface can then enter any other low power state or exit the low power state.

The lock option may also be issued along with any low power entry or exit command “LP_CMD = 8’b100_xxx_10 or 8’b100_000_01”. This command is also processed in the same way as any other low power command, with the only difference that the “LP_ARB_STATE” is not reset to idle at the end of the command. However, note that the only way to remove the lock is to issue another low power entry or exit command with the lock bit cleared.

(2) Software Programmable Interface Commands

The “LP_CMD” parameter is used to issue low power commands to the LPC module through the software programmable interface. There is no strobe associated with the software programmable interface and writes to this parameter will be sent to the LPC module for arbitration and execution. For detailed description refer please to the detailed **Section 6.4.1.27, DDR_CTL_26 — DDR-Controller Status & Control 26**.

Three low power modes are supported for the software programmable interface with the following bit (LP_CMD[4:2]) settings:

- 3’b000 = Active Power-Down

CAUTION

If entry into either of the “Active Power-Down” was requested, the memory will enter either active or pre-charge power-down mode depending on the state of the rows. If not all rows are closed at the time, the memories will enter active power-down mode. If there are no open rows, the memories will enter pre-charge power-down mode. To ensure that all rows are closed, the user should specify a “Pre-Charge Power-Down”.

- 3’b001 = Pre-Charge Power-Down
- 3’b010 = Self-Refresh

Table 6.165 Supported Software Programmable Interface Commands

Parameter “LP_CMD”	Description
8’b000_000_01	Exit any low power state
8’b000_000_10	“Active Power-Down” Entry
8’b000_001_10	“Pre-Charge Power-Down” Entry
8’b000_010_10	“Self-Refresh Entry”
8’b001_010_10	“Self-Refresh with Memory Clock Gating” Entry
8’b011_010_10	“Self-Refresh with Memory and Controller Clock Gating” Entry ***Note*)** This state should not be entered when a read or write command is being processed. The user should ensure that the controller is idle before requesting entry into this state by checking the CONTROLLER_BUSY parameter.
8’b100_000_00	Set the lock (no low power command)
8’b100_000_01	Exit any low power state, with lock
8’b100_000_10	“Active Power-Down” Entry, with lock
8’b100_001_10	“Pre-Charge Power-Down Entry”, with lock
8’b100_010_10	“Self-Refresh Entry”, with lock
8’b101_010_10	“Self-Refresh with Memory Clock Gating” Entry, with lock
8’b111_010_10	“Self-Refresh with Memory and Controller Clock Gating” Entry, with lock ***Note*)** This state should not be entered when a read or write command is being processed. The user should ensure that the controller is idle before requesting entry into this state by checking the CONTROLLER_BUSY parameter.

6.5.7.4 Automatic Interface

The LPC module supports automatic entry into each of the low power states based on programmable enables and idle state monitors. Each low power mode (not state) has a separate enable bit and counter as shown in **Table 6.166, Low Power State Management**. As with the software programmable interface, the automatic interface must win arbitration of the LPC module to issue requests. The automatic interface has the lowest priority for arbitration.

When the controller is idle, each of the timing counters that are enabled begin counting down the cycles of inactivity. Idle time requires that no read or write commands are executing or pending in the command queue or any of the ports. For the power-down states, idle time begins when no commands are waiting to be sent to memory and the counter decrements for each cycle of controller inactivity. For the self-refresh states, idle time begins only after all the read data for outstanding read commands has been retrieved; this restriction ensures that all read data is received even if the automatic request includes gating off the controller clock. The self-refresh counters decrement for each long count of inactivity.

(1) Automatic Entry

If any of the counters expire, the automatic interface will request arbitration. The timing counters are initially loaded with the values in the associated parameters, and will only decrement if the associated “LP_AUTO_ENTRY_EN” parameter bit is set to 1 and the counters are loaded with a non-zero value. If no other interface has control of the LPC module “LP_ARB_STATE = 4’b0000 or 4’b0011”, the automatic interface will win arbitration and the specified low power state will be compared to the current state of the memories. If the memory is already in a low power state, and the expired counter is associated with a deeper low power state than the current state, the LPC module will trigger an entry into the new low power state. If the expired counter is associated with a higher power usage state, the counter expiration will be ignored.

If a counter expires while the LPC module is being controlled by the software programmable interface, the automatic request will remain pending until the arbiter returns to idle “LP_ARB_STATE = 4’b0000” and the automatic interface is granted control of the LPC module. A pending request will be cancelled if a read or write command enters the command queue, resetting the automatic counters to the programmed values.

If the power-down counter expires, the DDR Controller will evaluate the state of the memory and the setting of the “LP_AUTO_MEM_GATE_EN” parameter to determine which low power state to enter. If not all rows are closed at the time, the memories will enter active power-down mode. If there are no open rows, the memories will enter pre-charge power-down mode. The memory clock will be gated off in either mode if bit [0] is set to 1 in the “LP_AUTO_MEM_GATE_EN” parameter. Memory that has been transitioned into active power-down mode will automatically transition to pre-charge power-down mode when the controller performs a refresh since the refresh process includes a pre-charge all command.

Multiple automatic low power idle counters may expire at the same time. When this happens, the counter associated with the deepest low power state will be entered. No state change will occur if current low power state is deeper than the states associated with any of the expired counters.

Since the controller only supports entering deeper low power states, it is meaningful to program progressively larger idle times for the deeper low power states. The parameters are shown in **Table 6.166, Low Power State Management** from least power savings to most power savings. The associated counters will only count idle cycles if the “LP_AUTO_ENTRY_EN” parameter bit is set to 1, and if the associated parameter is a non-zero value.

Table 6.166 Low Power State Management

Low Power State	Enabling Parameter	Counter Parameter
Active Power-Down	LP_AUTO_ENTRY_EN [0] = 1'b1 LP_AUTO_MEM_GATE_EN [0] = 1'b0	LP_AUTO_PD_IDLE
Pre-Charge Power-Down	LP_AUTO_ENTRY_EN [0] = 1'b1 LP_AUTO_MEM_GATE_EN [0] = 1'b0	LP_AUTO_PD_IDLE
Self-Refresh	LP_AUTO_ENTRY_EN [1] = 1'b1 LP_AUTO_MEM_GATE_EN [1] = 1'b0	LP_AUTO_SR_IDLE
Self-Refresh with Memory Clock Gating	LP_AUTO_ENTRY_EN [1] = 1'b1 LP_AUTO_MEM_GATE_EN [1] = 1'b1	LP_AUTO_SR_IDLE
Self-Refresh with Memory and Controller Clock Gating	LP_AUTO_ENTRY_EN [2] = 1'b1	LP_AUTO_SR_MC_GATE_IDLE

CAUTION

When the DDR Controller issues a power-down mode entry, the DRAMs will enter one of the power-down states depending on the state of the rows and the setting of the "LP_AUTO_MEM_GATE_EN[0]" parameter bit at that time.

(2) Automatic Exit

The automatic interface also supports automatic exit from a low power state if the system requires, with separate enable bits for each low power state. During an automatic exit, all of the idle counters are reset to their programmed values and the memories are returned to normal operation. When a new read or write command enters the command queue, if the current low power state's associated "LP_AUTO_EXIT_EN" parameter bit is set to 1, the automatic interface will request arbitration. If no other interface has control of the LPC module "LP_ARB_STATE = 4'b0000 or 4'b0011", the automatic interface will win arbitration and an exit low power command be triggered. If the current state is not enabled for automatic exit in the "LP_AUTO_EXIT_EN" parameter, the LPC module will not exit low power.

If the LPC module is being controlled by the software programmable interface when the new read or write command appears, the automatic request will remain pending until the arbiter returns to idle "LP_ARB_STATE = 4'b0000" and the automatic interface is granted control of the LPC module.

Only new read or write commands will cause the counters to be reloaded to their programmed values and trigger an exit. Other commands, including MRR, MRW, low power entry and exit commands, register accesses, refresh, ZQ, etc., do not reset the counters and do not trigger an exit from low power. When the controller is idle and the memories have automatically entered a low power state, these commands may be prevented from executing.

It is generally expected that any state that is defined for automatic entry "LP_AUTO_ENTRY_EN" should also be enabled for automatic exit "LP_AUTO_EXIT_EN". Automatic exit may also be used for low power states that are not defined for automatic entry, but are expected to be entered manually through the software programmable interface "LP_CMD". If it is desirable to use both manual and automatic entry/exit into the same low power state, then the user may need to re-program the "LP_AUTO_EXIT_EN" parameter prior to issuing a request through the software programmable interface.

(3) Automatic Interface Management

The LPC module may be manipulated through programmable parameters in the DDR Controller registers.

6.6 Usage Notes

6.6.1 Simplified DDR Initialization

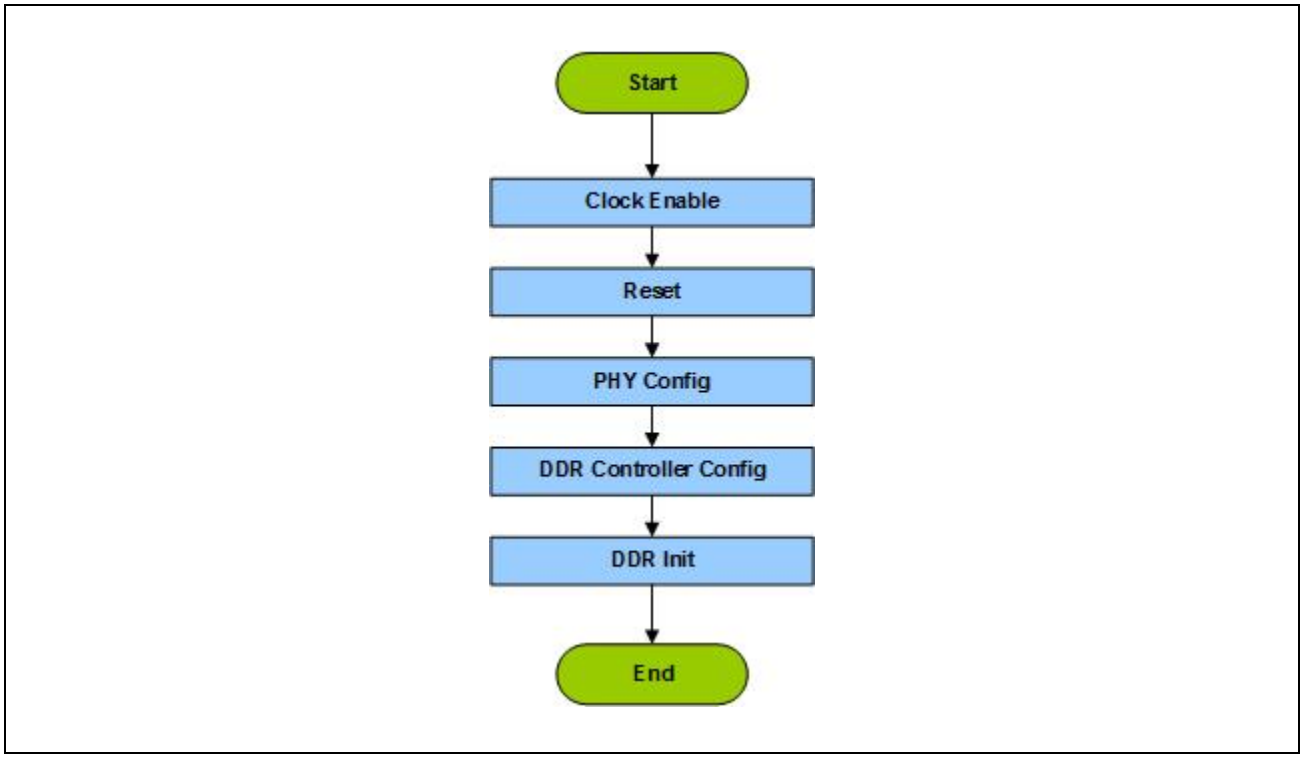


Figure 6.8 Simplified Init Flow

6.6.2 DDR Initialization Example

(1) Clock Enable

Enable AXI Interconnection (NoC) Clock-A (DDR_XCLK) and DDR (Ctrl/PHY) Clock-B (DDR_DFICLK):
{write} PWRCTRL_DDRC.CLKEN_B = 1
{write} PWRCTRL_DDRC.CLKEN_A = 1

(2) Reset

Request idle state for AXI interconnection and wait for acknowledge
{write} PWRCTRL_DDRC.MIREQ_A = 1
{wait} PWRSTAT_DDRC.MISTAT_A = 1

Assert Reset to AXI Interconnection (NoC) Reset-A and DDR (Ctrl/PHY) Reset-B
{write} PWRCTRL_DDRC.RSTN_A = 0
{write} PWRCTRL_DDRC.RSTN_B = 0

Assert Soft-Reset of DDR (PHY) and pre-define Voltage setting and Hi-Z mask
{write} FUNCCTRL = 0x00180100 (DDR3)
{write} FUNCCTRL = 0x00200000 (DDR2)

De-Assert Reset to AXI Interconnection (NoC) Reset-A and DDR (Ctrl/PHY) Reset-B
{write} PWRCTRL_DDRC.RSTN_B = 1
{write} PWRCTRL_DDRC.RSTN_A = 1

Request active state for AXI interconnection and wait for acknowledge
{write} PWRCTRL_DDRC.MIREQ_A = 0
{wait} PWRSTAT_DDRC.MISTAT_A = 0

Assert Soft-Reset of DLL (PHY) and pre-define frequency band setting
{write} DLLCTRL = 0x00000005 (DDR3)
{write} DLLCTRL = 0x0000000D (DDR2)

(3) DDRPHY Configuration

ZQCALCTRL register setting
{write} ZQCALCTRL = 0x00000186 (Use termination)
{write} ZQCALCTRL = 0x00000182 (No termination)

ZQODTCTRL register setting
{write} ZQODTCTRL = 0xAB330031 (DDR3)
{write} ZQODTCTRL = 0xAB330070 (DDR2)

RDCTRL register setting
{write} RDCTRL = 0xB545B544 (DDR3)
{write} RDCTRL = 0x94449443 (DDR2)

RDTMG register setting
{write} RDTMG = 0x000000B0 (DDR3)
{write} RDTMG = 0x000000A0 (DDR2)

Enable output for DDR_CLKEN, DDR_ODT and DDR_RESET_N
{write} OUTCTRL = 0x020A0806

Write Leveling Settings
{write} WLCTRL1 = 0x80005556 (DDR3)
{write} WLCTRL1 = 0x80005C5D (DDR2)

DQCALOFS1 register setting
{write} DQCALOFS1 = 0x00004545

De-Assert Soft-Reset of DLL (PHY) and define frequency band setting
{write} DLLCTRL = 0x00000004 (DDR3)
{write} DLLCTRL = 0x0000000C (DDR2)

De-Assert Soft-Reset of DDR (PHY) and define Voltage setting and Hi-Z mask
{write} FUNCCTRL = 0x00180101 (DDR3)
{write} FUNCCTRL = 0x00200001 (DDR2)

Initialize DDR (PHY) FIFO-pointer
{write} FIFOINIT = 0x00000101

Configuration for ZQ calibration
{write} ZQCALCTRL = 0x00000187 (Use termination)
{write} ZQCALCTRL = 0x00000183 (No termination)

Wait more than 200 us, or wait until MDLL locked and ZQ calibration completed
{wait} DLLCTRL.ASDLLOCK = 1
{wait} ZQCALCTRL.ZQCALEND = 1

Enable output also for Address and Command
{write} OUTCTRL = 0x020A0807

Wait 200 us or more

(4) DDR (Ctrl) initialization

DDR Controller configuration
{write} DDR_CTL_00
{write} :
{write} DDR_CTL_87

AXI port setting
{write} DDR_CTL_87
{write} :
{write} DDR_CTL_349
{write} DDR_CTL_374

PHY setting
{write} DDR_CTL_350
{write} :
{write} DDR_CTL_372

Start Initialization
{write} DDR_CTL_00.START = 1

Wait initialization complete
{wait} DDR_CTL_56.INT_STATUS[20] = 1
{wait} DDR_CTL_56.INT_STATUS[8] = 1

Wait tXPR (DDR3) or 400 ns (DDR2)
Initialize DDR memory
Write Leveling Settings
{write} WLCTRL1 = 0x81005556 (DDR3)
{write} WLCTRL1 = 0x81005C5D (DDR2)

Section 7 NAND Flash Controller

7.1 Overview

The NAND Flash Controller supports the functionality of the devices described in the ONFI 2.2 specifications, as well as older devices compatible with the ONFI 1.x specifications. The integrated BCH Error Correction Code (ECC) algorithm allows up to 32-bit error correction.

The correction ability and subpage size of the BCH component are programmable. The number of banks and the number of devices per bank that can be handled by the controller are one and four respectively.

- NAND interface with 8-bit bus width
- Compatible with ONFI 1.x and 2.x standards
  (RZ/N1 NAND Boot mode supports ONFI compliant memories only)
- Support for asynchronous mode
- 4 chip selects
- Support for 4 NAND Flash device per bank (4 Chip Selects)
- Write protection
- Programmable address cycle (0/1/2/3/4/5)
- Protected area size
- Integrated DMA
- Support for 256 B, 512 B, 2 KB, 1 KB, 4 KB, 8 KB, 16 KB pages
- BCH ECC algorithm for multiple errors (Error detection and data correction)
  - ECC data block size: 256 B, 512 B, 1024 B
  - ECC correction capability: 2, 4, 8, 16, 24, 32 bits errors
- Bad Block Management (BBM)

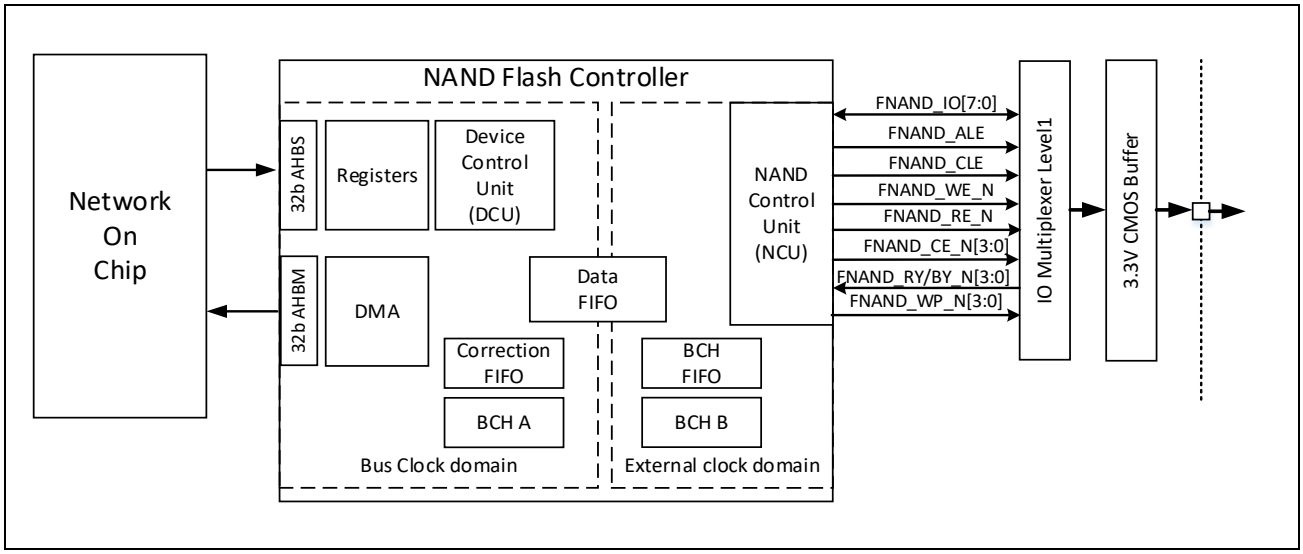


Figure 7.1 NAND Flash Controller Interfaces and Connections

7.2 Signal Interfaces

Signal Name	Input Output	Description
Clock		
NAND_HCLK	Input	Internal bus clock (AHB)
NAND_ECLK	Input	External Interface clock
Interrupt		
NAND_Int	Output	Level sensitive interrupt, Active High
External signal		
FNAND_CE_N[3:0]	Output	Chip Enable, Active Low
FNAND_IO[7:0]	I/O	Data
FNAND_CLE	Output	Command Latch Enable
FNAND_ALE	Output	Address Latch Enable
FNAND_RE_N	Output	Read Enable, Active Low
FNAND_WE_N	Output	Write Enable, Active Low
FNAND_WP_N[3:0]	Output	Write Protect/Reset
FNAND_RY/BY_N[3:0]	Input	Ready/Busy (RnB), Ready=High, Busy=Low

7.3 Register Map

Table 7.1 NAND Flash Controller Register Map

Address	Register Symbol	Register Name
4010 2000h	COMMAND	Command Register
4010 2004h	CONTROL	CONTROL Register
4010 2008h	STATUS	STATUS Register
4010 200Ch	STATUS_MASK	STATUS_MASK Register
4010 2010h	INT_MASK	INT_MASK Register
4010 2014h	INT_STATUS	INT_STATUS Register
4010 2018h	ECC_CTRL	ECC Control Register
4010 201Ch	ECC_OFFSET	ECC Offset Register
4010 2020h	ECC_STAT	ECC Status Register
4010 2024h	ADDR0_COL	Column Address 0 Register
4010 2028h	ADDR0_ROW	Row Address 0 Register
4010 202Ch	ADDR1_COL	Column Address 1 Register
4010 2030h	ADDR1_ROW	Row Address 1 Register
4010 2034h	PROTECT	Protect Register
4010 2038h	FIFO_DATA	FIFO Data Register
4010 203Ch	DATA_REG	Data Register
4010 2040h	DATA_REG_SIZE	DATA_REG_SIZE Register
4010 2044h + 4h × n	DEV[n]_PTR (n = 0..3)	Device [n] Remap Pointer Register
4010 2064h	DMA_ADDR	DMA Address Register
4010 206Ch	DMA_CNT	DMA Counter Register
4010 2070h	DMA_CTRL	DMA Control Register
4010 2074h	BBM_CTRL	BBM Control Register
4010 2080h	MEM_CTRL	Memory Devices Control Register
4010 2084h	DATA_SIZE	Data Size Register
4010 2088h	TIMINGS_ASYN	Asynchronous Mode Timings Register
4010 2090h	TIME_SEQ_0	Command Sequence Timing Register 0
4010 2094h	TIME_SEQ_1	Command Sequence Timing Register 1
4010 2098h	TIME_GEN_SEQ_0	Generic Command Sequence Register 0
4010 209Ch	TIME_GEN_SEQ_1	Generic Command Sequence Register 1
4010 20A0h	TIME_GEN_SEQ_2	Generic Command Sequence Register 2
4010 20B0h	FIFO_INIT	FIFO Init Register
4010 20B4h	FIFO_STATE	FIFO Status Register
4010 20B8h	GEN_SEQ_CTRL	Generic Sequence Register
4010 20BCh	MLUN	LUN Configuration Register
4010 20C0h + 4h × n	DEV[n]_SIZE (n = 0..3)	Device [n] BBM Record Counter Register
4010 2114h	DMA_TLVL	DMA Trigger Level Register
4010 2124h	CMD_MARK	CMD ID Initial Value
4010 2128h	LUN_STATUS_0	LUN Status Register
4010 2134h	TIME_GEN_SEQ_3	Generic Command Sequence Register 3
4010 2148h	INT_STAT	Internal Status Register
4010 214Ch	ECC_CNT	ECC Error Counter
4010 2150h	PARAM_REG	PARAMETER Register

7.4 Register Description

7.4.1 COMMAND — Command Register

The write of the command sequence code to this register triggers the programmed command sequence execution as soon as it is possible. If the execution cannot be started immediately, then the transfer to this register is prolonged by the series of the WAIT responses best suited for the selected system bus. Each command sequence can trigger the interrupt when it is completed.

Address: 4010 2000h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	cmd_2								cmd_1_cmd_3							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	cmd_0								data_sel	input_sel	cmd_seq					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.2 COMMAND Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	cmd_2	Code of the third command in a sequence	R/W
b23 to b16	cmd_1_cmd_3	Code of the second command in a sequence	R/W
b15 to b8	cmd_0	Code of the first command in a sequence	R/W
b7	data_sel	Data or FIFO selection flag 0: The FIFO module selected 1: The DATA register selected	R/W
b6	input_sel	Input module selection flag 0: Select the AHBS module as input 1: Select the DMA module as input	R/W
b5 to b0	cmd_seq	Command code	R/W

7.4.2 CONTROL — CONTROL Register

Address: 4010 2004h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	auto_read_stat_en	mlun_en	small_block_en	—	—	—	addr1_auto_incr	addr0_auto_incr
Value after reset	X	X	X	X	X	X	X	X	0	0	0	X	X	X	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	prot_en	bbm_en	io_width	—	—	—	—	block_size		ecc_en	int_en	—	ecc_block_size		read_status_en
Value after reset	X	0	0	0	X	X	X	X	0	0	0	0	X	0	0	0

Table 7.3 CONTROL Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved		R
b23	auto_read_stat_en	Auto Read Status mode enable If active, the controller reads the status after the PROGRAM PAGE and BLOCK ERASE commands. It can trigger interrupt. The ERROR_MASK field in the STATUS_MASK register must be configured when this feature is enabled. 0: Auto Read Status mode disabled 1: Auto Read Status mode enabled	R/W
b22	mlun_en	Multi LUN mode enable 0: Multi LUN mode disabled 1: Multi LUN mode enabled	R/W
b21	small_block_en	Enable small block mode In this mode, the controller sends only a single byte as the column address instead of two bytes, as done in the big block NAND Flash devices. 0: Big block mode enabled 1: Small block mode enabled ***Caution*)** Only devices that allow to disable CE signal when device is in the busy state are supported.	R/W
b20 to b18	Reserved		R
b17	addr1_auto_incr	Address auto increment for row address register 1 0: Auto increment disabled 1: Auto increment enabled When this bit is set, sending any command sequence using address register 1 causes the increment of address register 1	R/W
b16	addr0_auto_incr	Address auto increment for row address register 0 0: Auto increment disabled 1: Auto increment enabled When this bit is set, sending any command sequence using address register 0 causes the increment of address register 0	R/W
b15	Reserved		R
b14	prot_en	Protect mechanism enable 0: Protect disabled 1: Protect enabled	R/W
b13	bbm_en	Bad Block Management enable For more details, see **Section 7.5.6, Remapping Mechanism**	R/W
b12	io_width	NAND Flash I/O width select 0: 8 bits 1: Reserved	R/W

Table 7.3 CONTROL Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b11 to b8	Reserved		R
b7, b6	block_size	Block size select 2'b00: 32 pages per block 2'b01: 64 pages per block 2'b10: 128 pages per block 2'b11: 256 pages per block	R/W
b5	ecc_en	Hardware ECC support enable 0: ECC disabled 1: ECC enabled Hardware ECC can be used only when m × (ECC_BLOCK_SIZE) ≤ DATA_SIZE ≤ m × (ECC_BLOCK_SIZE + 32), where m is 1, 2, 3,….	R/W
b4	int_en	Interrupt enables 0: Interrupts disabled 1: Interrupts enabled	R/W
b3	Reserved		R
b2, b1	ecc_block_size	The ECC block size 2'b00: 256 bytes 2'b01: 512 bytes 2'b10: 1024 bytes 2'b11: Not available The ECC block size can be changed only when all memory devices are ready.	R/W
b0	read_status_en	Automatically check RnB lines status The STATE_MASK field in the STATUS_MASK register must be configured when this feature is enabled. 0: The controller checks RnB lines 1: The Controller sends READ STATUS commands ***Caution*)** Automatically sent READ STATUS command is only available in devices compatible with ONFI 1.0	R/W

7.4.3 STATUS — STATUS Register

This register stores the NAND Flash Controller and connected device status flags. These flags can be used by the system to obtain the current controller internal state.

The CTRL_STAT flag is set after the controller starts to execute the requested command for the selected NAND Flash device, and it is active while the command execution is not completed. Command execution can be divided into two phases. In the first phase, the command sequence is executed at the moment when the NAND Flash device goes into the busy state. After that, the controller stores information about the pending operation on the selected device, but is also able to execute a new command on any other device. In the second phase, the controller automatically finishes the pending command execution based on the previously stored data. As long as this flag is set, the controller does not accept any new commands.

The MEM[n]_ST flags correspond to the NAND Flash device with the same index value. The flags give information about the NAND Flash devices' state.

Address: 4010 2008h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	cmd_id							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	data_reg_st	datasize_error_st	ctrl_stat	—	—	—	—	mem3_st	mem2_st	mem1_st	mem0_st
Value after reset	X	X	X	X	X	0	0	0	X	X	X	X	1	1	1	1

Table 7.4 STATUS Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved		R
b23 to b16	cmd_id	Command id field	R
b15 to b11	Reserved		R
b10	data_reg_st	Data Reg status flag Resetting of this flag is possible only by reading the data from the DATA_REG register. 1: Data in DATA_REG is available 0: Data in DATA_REG is not available	R
b9	datasize_error_st	The data size value error When the ECC is enabled, this bit indicates incorrect value in the DATA_SIZE register 0: Correct value 1: Incorrect value	R
b8	ctrl_stat	The main controller status bit 0: Controller ready 1: Controller busy	R
b7 to b4	Reserved		R
b3	mem3_st	Device 3 status flag 1: Device ready 0: Device busy	R
b2	mem2_st	Device 2 status flag 1: Device ready 0: Device busy	R

Table 7.4 STATUS Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b1	mem1_st	Device 1 status flag 1: Device ready 0: Device busy	R
b0	mem0_st	Device 0 status flag 1: Device ready 0: Device busy	R

7.4.4 STATUS_MASK — STATUS_MASK Register

The state_mask field is used to mark the ready/busy bits in the NAND Flash device status byte. This field is used during the internal read status operation. In the case of ONFI, the user must mask all fields except RDY or ARDY (depending on the application).

The error_mask field is used to mask unused fields when the controller automatically reads the status of the NAND Flash memory device. In the case of ONFI, the user must mask all fields except FAIL or FAILC (depending on the application).

Address: 4010 200Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	error_mask								state_mask							
Value after reset	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Table 7.5 STATUS_MASK Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved		R
b15 to b8	error_mask	Error mask Used to mask the error bits if automatic read status feature is enabled. 0: Masked 1: Unmasked	R/W
b7 to b0	state_mask	State mask Used to mask status bits when the read status command is used to obtain the NAND Flash status 0: Masked 1: Unmasked	R/W

7.4.5 INT_MASK — INT_MASK Register

This register allows masking of the selected interrupts source in the NAND Flash Controller. The masked interrupts still set the appropriate bits in the status register, but those changes do not trigger the interrupt.

Address: 4010 2010h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	ecc_int3_en	ecc_int2_en	ecc_int1_en	ecc_int0_en	—	—	—	—	stat_err_int3_en	stat_err_int2_en	stat_err_int1_en	stat_err_int0_en
Value after reset	X	X	X	X	0	0	0	0	X	X	X	X	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	mem3_rdy_int_en	mem2_rdy_int_en	mem1_rdy_int_en	mem0_rdy_int_en	—	pg_sz_err_int_en	—	—	dma_int_en	data_reg_int_en	cmd_end_int_en	prot_int_en
Value after reset	X	X	X	X	0	0	0	0	X	0	X	X	0	0	0	0

Table 7.6 INT_MASK Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b28	Reserved		R
b27	ecc_int3_en	Enable the interrupt from the ECC module status for device 3 0: Interrupt disabled 1: Interrupt enabled	R/W
b26	ecc_int2_en	Enable the interrupt from the ECC module status for device 2 0: Interrupt disabled 1: Interrupt enabled	R/W
b25	ecc_int1_en	Enable the interrupt from the ECC module status for device 1 0: Interrupt disabled 1: Interrupt enabled	R/W
b24	ecc_int0_en	Enable the interrupt from the ECC module status for device 0 0: Interrupt disabled 1: Interrupt enabled	R/W
b23 to b20	Reserved		R
b19	stat_err_int3_en	Enable flag if most recently finished operation on the Memory device 3 failed applied to PROGRAM PAGE and BLOCK ERASE operations. It is not valid following a READ-series operation 0: Interrupt disabled 1: Interrupt enabled	R/W
b18	stat_err_int2_en	Enable flag if most recently finished operation on the Memory device 2 failed applied to PROGRAM PAGE and BLOCK ERASE operations. It is not valid following a READ-series operation 0: Interrupt disabled 1: Interrupt enabled	R/W
b17	stat_err_int1_en	Enable flag if most recently finished operation on the Memory device 1 failed applied to PROGRAM PAGE and BLOCK ERASE operations. It is not valid following a READ-series operation 0: Interrupt disabled 1: Interrupt enabled	R/W
b16	stat_err_int0_en	Enable flag if most recently finished operation on the Memory device 0 failed applied to PROGRAM PAGE and BLOCK ERASE operations. It is not valid following a READ-series operation 0: Interrupt disabled 1: Interrupt enabled	R/W
b15 to b12	Reserved		R

Table 7.6 INT_MASK Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b11	mem3_rdy_int_en	The memory device 3 is ready for the new command 0: Interrupt disabled 1: Interrupt enabled	R/W
b10	mem2_rdy_int_en	The memory device 2 is ready for the new command 0: Interrupt disabled 1: Interrupt enabled	R/W
b9	mem1_rdy_int_en	The memory device 1 is ready for the new command 0: Interrupt disabled 1: Interrupt enabled	R/W
b8	mem0_rdy_int_en	The memory device 0 is ready for the new command 0: interrupt disabled 1: interrupt enabled	R/W
b7	Reserved		R
b6	pg_sz_err_int_en	Data size error occur 0: Interrupt disabled 1: Interrupt enabled	R/W
b5, b4	Reserved		R
b3	dma_int_en	DMA transfer ended 0: Interrupt disabled 1: Interrupt enabled	R/W
b2	data_reg_int_en	data in data_reg is available 0: Interrupt disabled 1: Interrupt enabled	R/W
b1	cmd_end_int_en	Command sequence ended 0: Interrupt disabled 1: Interrupt enabled	R/W
b0	prot_int_en	Erase or Write protected area interrupt enable 0: Interrupt disabled 1: Interrupt enabled	R/W

7.4.6 INT_STATUS — INT_STATUS Register

This register stores the NAND Flash Controller interrupt flags. If the given bit has the logical ‘0’ value, then the corresponding interrupt condition is not met. If the given bit has the logical ‘1’ value, then the corresponding interrupt condition is met.

Address: 4010 2014h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	ecc_int3_fl	ecc_int2_fl	ecc_int1_fl	ecc_int0_fl	—	—	—	—	stat_err_int3_fl	stat_err_int2_fl	stat_err_int1_fl	stat_err_int0_fl
Value after reset	X	X	X	X	0	0	0	0	X	X	X	X	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	mem3_rdy_int_fl	mem2_rdy_int_fl	mem1_rdy_int_fl	mem0_rdy_int_fl	—	pg_sz_err_int_fl	—	—	dma_int_fl	data_reg_int_fl	cmd_end_int_fl	prot_int_fl
Value after reset	X	X	X	X	0	0	0	0	X	0	X	X	0	0	0	0

Table 7.7 INT_STATUS Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b28	Reserved		R
b27	ecc_int3_fl	Interrupt flag in the ECC module is set for device 3	R/W
b26	ecc_int2_fl	Interrupt flag in the ECC module is set for device 2	R/W
b25	ecc_int1_fl	Interrupt flag in the ECC module is set for device 1	R/W
b24	ecc_int0_fl	Interrupt flag in the ECC module is set for device 0	R/W
b23 to b20	Reserved		R
b19	stat_err_int3_fl	Interrupt flag for most recently finished operation on the device 3 failed applies to PROGRAM PAGE and BLOCK ERASE operations. It is not valid following a READ-series operation	R/W
b18	stat_err_int2_fl	Interrupt flag for most recently finished operation on the device 2 failed applies to PROGRAM PAGE and BLOCK ERASE operations. It is not valid following a READ-series operation	R/W
b17	stat_err_int1_fl	Interrupt flag for most recently finished operation on the device 1 failed applies to PROGRAM PAGE and BLOCK ERASE operations. It is not valid following a READ-series operation	R/W
b16	stat_err_int0_fl	Interrupt flag for most recently finished operation on the device 0 failed applies to PROGRAM PAGE and BLOCK ERASE operations. It is not valid following a READ-series operation	R/W
b15 to b12	Reserved		R
b11	mem3_rdy_int_fl	The memory device 3 is ready for the new command interrupt flag	R/W
b10	mem2_rdy_int_fl	The memory device 2 is ready for the new command interrupt flag	R/W
b9	mem1_rdy_int_fl	The memory device 1 is ready for the new command interrupt flag	R/W
b8	mem0_rdy_int_fl	The memory device 0 is ready for the new command interrupt flag	R/W
b7	Reserved		R
b6	pg_sz_err_int_fl	Data size error occur interrupt flag	R/W
b5, b4	Reserved		R
b3	dma_int_fl	DMA transfer ended interrupt flag	R/W
b2	data_reg_int_fl	data in data_reg is available interrupt flag	R/W
b1	cmd_end_int_fl	Command sequence ended interrupt flag	R/W
b0	prot_int_fl	Erase or Write protected area interrupt flag	R/W

7.4.7 ECC_CTRL — ECC Control Register

This register stores all configuration parameters required by the ECC. The ECC_CAP field is only available if the BCH ECC module with software configurable correction factor is selected.

Address: 4010 2018h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	ecc_sel	
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	err_threshold						—	—	—	—	—	ecc_cap		
Value after reset	X	X	0	0	0	0	0	0	X	X	X	X	X	0	0	0

Table 7.8 ECC_CTRL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b18	Reserved		R
b17, b16	ecc_sel	The ECC interrupt source select This signal selects the ECC module flag that will be used as a source for the interrupt signal: 2'b00: Select ECC_ERROR (correctable error) flag as interrupt source. 2'b01: Select ECC_UNC (uncorrectable error) flag as interrupt source. 2'b1x: Select ECC_OVER (acceptable errors level overflow) flag as interrupt source.	R/W
b15, b14	Reserved		R
b13 to b8	err_threshold	The acceptable errors level The value of this field contains the number of errors that is acceptable for software. This field must be initialized by software.	R/W
b7 to b3	Reserved		R
b2 to b0	ecc_cap	Correction ability. The correction ability can be changed only when all memory devices are ready. 3'b000: 2 3'b001: 4 3'b010: 8 3'b011: 16 3'b100: 24 3'b101: 32 Values other than those above select the 32 correction ability.	R/W

7.4.8 ECC_OFFSET — ECC Offset Register

This register stores the offset value from the beginning of the page to the place where correction words will be stored.

The value of this register must be bigger than the value in the DATA_SIZE register.

In small block mode, the value in the ECC_OFFSET is ignored, and the correction words are located in the NAND Flash memory device just behind the data.

Address: 4010 201Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ecc_offset															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.9 ECC_OFFSET Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved		R
b15 to b0	ecc_offset	Correction words block offset	R/W

7.4.9 ECC_STAT — ECC Status Register

This register stores all the ECC module status information.

Address: 4010 2020h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	ecc_over_3	ecc_over_2	ecc_over_1	ecc_over_0
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	ecc_unc_3	ecc_unc_2	ecc_unc_1	ecc_unc_0	—	—	—	—	ecc_error_3	ecc_error_2	ecc_error_1	ecc_error_0
Value after reset	X	X	X	X	0	0	0	0	X	X	X	X	0	0	0	0

Table 7.10 ECC_STAT Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b20	Reserved		R
b19	ecc_over_3	The Memory device 3 acceptable errors level overflow The bit is set when the number of errors is bigger than the value of declared ERR_THRESHOLD bits.	R/W
b18	ecc_over_2	The Memory device 2 acceptable errors level overflow The bit is set when the number of errors is bigger than the value of declared ERR_THRESHOLD bits.	R/W
b17	ecc_over_1	The Memory device 1 acceptable errors level overflow The bit is set when the number of errors is bigger than the value of declared ERR_THRESHOLD bits.	R/W
b16	ecc_over_0	The Memory device 0 acceptable errors level overflow The bit is set when the number of errors is bigger than the value of declared ERR_THRESHOLD bits.	R/W
b15 to b12	Reserved		R
b11	ecc_unc_3	The Memory device 3 uncorrectable error flag The bit is set when the uncorrectable errors occur during the read operation.	R/W
b10	ecc_unc_2	The Memory device 2 uncorrectable error flag The bit is set when the uncorrectable errors occur during the read operation.	R/W
b9	ecc_unc_1	The Memory device 1 uncorrectable error flag The bit is set when the uncorrectable errors occur during the read operation.	R/W
b8	ecc_unc_0	The Memory device 0 uncorrectable error flag The bit is set when the uncorrectable errors occur during the read operation.	R/W
b7 to b4	Reserved		R
b3	ecc_error_3	The Memory device 3 correctable error flag The bit is set when the correctable errors occur during the read operation.	R/W
b2	ecc_error_2	The Memory device 2 correctable error flag The bit is set when the correctable errors occur during the read operation.	R/W
b1	ecc_error_1	The Memory device 1 correctable error flag The bit is set when the correctable errors occur during the read operation.	R/W
b0	ecc_error_0	The Memory device 0 correctable error flag The bit is set when the correctable errors occur during the read operation.	R/W

7.4.10 ADDR0_COL — Column Address 0 Register

Address: 4010 2024h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	addr0_col															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.11 ADDR0_COL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved		R
b15 to b0	addr0_col	Column address For detailed description refer to **Section 7.8.1, ADDR[n]_COL and ADDR[n]_ROW Registers**.	R/W

7.4.11 ADDR0_ROW — Row Address 0 Register

Address: 4010 2028h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	addr0_row							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	addr0_row															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.12 ADDR0_ROW Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved		R
b23 to b0	addr0_row	Row address For detailed description refer to **Section 7.8.1, ADDR[n]_COL and ADDR[n]_ROW Registers**.	R/W

7.4.12 ADDR1_COL — Column Address 1 Register

Address: 4010 202Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	addr1_col															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.13 ADDR1_COL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved		R
b15 to b0	addr1_col	Column address For detailed description refer to **Section 7.8.1, ADDR[n]_COL and ADDR[n]_ROW Registers**.	R/W

7.4.13 ADDR1_ROW — Row Address 1 Register

Address: 4010 2030h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	addr1_row							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	addr1_row															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.14 ADDR1_ROW Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved		R
b23 to b0	addr1_row	Row address For detailed description refer to **Section 7.8.1, ADDR[n]_COL and ADDR[n]_ROW Registers**.	R/W

7.4.14 PROTECT — Protect Register

The NAND Flash Controller allows the defining of the area that will be protected against any modifications.

The protected area is a space which cannot be erased or overwritten. Any attempt to erase/overwrite this space always ends with an error. Since write and erase processes have constraints (only Page can be written and only Block can be erased), the protected area can be defined with Block-size precision. This register's lower bits [15:0] define the beginning address of the protected area and are related to the NAND Flash memory block address bits of ADDR0_ROW and ADDR1_ROW registers. This register higher bits [31:16] define the ending address of the protected area and are related to the NAND Flash memory block address bits of ADDR0_ROW and ADDR1_ROW registers. Please refer to **Section 7.8.2, Protect Register (PROTECT)** how this register fields are used to define the protected area.

Address: 4010 2034h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	prot_up															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	prot_down															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.15 PROTECT Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	prot_up	Protect area upper limit	R/W
b15 to b0	prot_down	Protect area lower limit	R/W

7.4.15 FIFO_DATA — FIFO Data Register

This register is used as an entry point to the FIFO module for the AHBS module. The CPU can access the FIFO module by reading or writing to this register in the same way as it accesses any other registers. The FIFO module works on 32-bit words, thus when this register is accessed from the narrower bus, then:

- For the read operation, the access to the lowest byte triggers the word read from the FIFO module. If the requested data is narrower than the FIFO word size, then the read word is stored for further access. If the read request does not strobe the lowest byte, then the previously stored data is used instead, triggering new access to the FIFO.
- For the write operation, the situation is almost the same. Only the request that strobes the lowest byte triggers the write access to the FIFO module. Any other requests cause only write to the temporary register used for further access to the FIFO module.

Address: 4010 2038h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	fifo_data															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	fifo_data															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.16 FIFO_DATA Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	fifo_data	The data FIFO interface register	R/W

7.4.16 DATA_REG — Data Register

This register is used for storage of the data that is read in the registered mode. The registered mode is allowed in the read direction only.

Address: 4010 203Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	data_reg															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	data_reg															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.17 DATA_REG Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	data_reg	The data register	R

7.4.17 DATA_REG_SIZE — DATA_REG_SIZE Register

This register allows the selection of data size in the registered work mode. The data size in the registered mode is limited to four bytes.

Address: 4010 2040h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	data_reg_size	
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0	0

Table 7.18 DATA_REG_SIZE Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b2	Reserved		R
b1, b0	data_reg_size	selection of the number of valid bytes in the DATA register: 2'b00: Single byte valid 2'b01: Two lower bytes valid 2'b10: Three lower bytes valid 2'b11: All four bytes valid	R/W

7.4.18 DEV[n]_PTR — Device [n] Remap Pointer Register (n = 0..3)

The bad block management mechanism implemented in the controller uses the tables in the system memory to store the remapping records. Each device in the bank requires a separate table. This register stores the table of the remapping records address for the device [n].

Address: 4010 2044h + 4h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	ptr_addr															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ptr_addr															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.19 DEV[n]_PTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	ptr_addr	The remap table pointer The field contains an address of the remap table in the system memory.	R/W

7.4.19 DMA_ADDR — DMA Address Register

This register defines the DMA base address. The two least significant bits are ignored.

Address: 4010 2064h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	dma_addr_l															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	dma_addr_l															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.20 DMA_ADDR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	dma_addr_l	The DMA address	R/W

7.4.20 DMA_CNT — DMA Counter Register

This register defines the number of bytes that will be transferred by the DMA module. The register remains unchanged during the transfer process.

Address: 4010 206Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	cnt_init															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	cnt_init															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.21 DMA_CNT Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	cnt_init	Bytes counter initial value The field contains data page length in bytes (0000_0004h – FFFF_FFFCh). The number of the bytes has to be divided by 4.	R/W

7.4.21 DMA_CTRL — DMA Control Register

Address: 4010 2070h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	dma_start	—	dma_mode	dma_burst			err_flag	dma_ready
Value after reset	X	X	X	X	X	X	X	X	0	X	0	0	0	0	0	0

Table 7.22 DMA_CTRL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7	dma_start	The DMA start Set bit DMA_START to start DMA when the command sequence is sent to the NAND Flash memory.	R/W
b6	Reserved		R
b5	dma_mode	The DMA work mode 0: The register-s managed mode 1: The Scatter-Gather mode	R/W
b4 to b2	dma_burst	The DMA burst type These bits define the main transfer type used by the DMA to precede the requested transfer. 3’b000: Incrementing burst of four transfers (address increment, fixed length) 3’b001: Address constant burst of sixteen transfers (fixed length) 3’b010: Single transfer (address increment) 3’b011: Burst of unspecified length (address increment) 3’b100: Incrementing burst of eight transfers (address increment, fixed length) 3’b101: Incrementing burst of sixteen transfers (address increment, fixed length)	R/W
b1	err_flag	The DMA error flag The flag is set when a transmission error occurs during the DMA transfer.	R
b0	dma_ready	The DMA ready flag The flag is set when transfer is completed.	R

7.4.22 BBM_CTRL — BBM Control Register

Address: 4010 2074h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	rmp_init
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

Table 7.23 BBM_CTRL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b1	Reserved		R
b0	rmp_init	Remap initial flag If set this flag force the BBM module to reread the remapping table after it was updated by software. This flag is set by software and cleared by hardware.	R/W

7.4.23 MEM_CTRL — Memory Devices Control Register

This register stores the set of configuration parameters that are used to select the destination NAND Flash device for the current transfer and the write protect signal for each device.

Address: 4010 2080h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	mem3_wp	mem2_wp	mem1_wp	mem0_wp	—	—	—	—	—	—	mem_ce	
Value after reset	X	X	X	X	0	0	0	0	X	X	X	X	X	X	0	0

Table 7.24 MEM_CTRL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b12	Reserved		R
b11	mem3_wp	The memory device 3 WP line The setting value is output to FNAND_WP_N[3].	R/W
b10	mem2_wp	The memory device 2 WP line The setting value is output to FNAND_WP_N[2].	R/W
b9	mem1_wp	The memory device 1 WP line The setting value is output to FNAND_WP_N[1].	R/W
b8	mem0_wp	The memory device 0 WP line The setting value is output to FNAND_WP_N[0].	R/W
b7 to b2	Reserved		R
b1, b0	mem_ce	The memory device select field 2'b00: Memory device 0 2'b01: Memory device 1 2'b10: Memory device 2 2'b11: Memory device 3	R/W

7.4.24 DATA_SIZE — Data Size Register

This register stores the value of the data block size. The data size value is remembered as the number of bytes per transferred block, but its size must be declared as the multiple of the chosen NAND Flash word size. The unused bits for the given word size configuration are ignored and replaced with zeros.

Address: 4010 2084h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	data_size														
Value after reset	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.25 DATA_SIZE Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b15	Reserved		R
b14 to b0	data_size	Data size	R/W
		***Caution*)** Write the proper value when ECC is enabled: (ECC_BLOCK_SIZE) × m ≤ data_size ≤ (ECC_BLOCK_SIZE +32) × m, where m is 1,2,3…	

7.4.25 TIMINGS_ASYN — Asynchronous Mode Timings Register

The NAND Flash Controller is intended for use with a wide range of host clock rates. To maximize flexibility, some timing parameters are configurable. Two waveform configuration parameters are defined in this register. The value generated by the controller equals the minimum value written into the register, increased by 1. All the timings are generated using the NAND_ECLK clock signal.

Please refer to **Section 7.8.3, Asynchronous Mode Timings Register (TIMINGS_ASYN)** how timing parameters are mapped to the NAND Flash interface.

Address: 4010 2088h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	trwh				trwp			
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 7.26 TIMINGS_ASYN Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7 to b4	trwh	RE_N or WE_N high hold time	R/W
b3 to b0	trwp	RE_N or WE_N pulse width	R/W

7.4.26 TIME_SEQ_0 — Command Sequence Timing Register 0

The NAND Flash Controller is intended for use with a wide range of host clock rates. To maximize flexibility, some timing parameters are configurable. Some waveform configuration parameters are defined in this register. The value generated by the controller equals the minimum value written into the register, increased by 1. All the timings are generated using the NAND_ECLK clock signal.

Address: 4010 2090h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	twhr						—	—	trhw					
Value after reset	X	X	0	0	0	0	0	0	X	X	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	tadl						—	—	tccs					
Value after reset	X	X	0	0	0	0	0	0	X	X	0	0	0	0	0	0

Table 7.27 TIME_SEQ_0 Register Contents

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved		R
b29 to b24	twhr	WE_N high to RE_N low time	R/W
b23, b22	Reserved		R
b21 to b16	trhw	RE_N high to WE_N low time	R/W
b15, b14	Reserved		R
b13 to b8	tadl	ALE to data start time	R/W
b7, b6	Reserved		R
b5 to b0	tccs	Change column setup	R/W

7.4.27 TIME_SEQ_1 — Command Sequence Timing Register 1

The NAND Flash Controller is intended for use with a wide range of host clock rates. To maximize flexibility, some timing parameters are configurable. Some waveform configuration parameters are defined in this register. The value generated by the controller equals the minimum value written into the register, increased by 1. All the timings are generated using the NAND_ECLK clock signal.

Please refer to **Section 7.8.4, Command Sequence Timing Register 1 (TIME_SEQ_1)** how the tWW timing parameter is mapped to the NAND Flash interface.

Address: 4010 2094h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	tww					
Value after reset	X	X	X	X	X	X	X	X	X	X	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	trr						—	—	twb					
Value after reset	X	X	0	0	0	0	0	0	X	X	0	0	0	0	0	0

Table 7.28 TIME_SEQ_1 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b22	Reserved		R
b21 to b16	tww	Delay time between WP_N change and command drive This field is used to parameterize the tWW delay.	R/W
b15, b14	Reserved		R
b13 to b8	trr	RnB high to RE_N low TRR time period from rising edge on RnB input line to the moment when the read enable signal can be asserted.	R/W
b7, b6	Reserved		R
b5 to b0	twb	tWB delay Time period measured from rising edge of the WE_N signal to a falling edge on the RnB line or the NAND Flash device SR[6] low.	R/W

7.4.28 TIME_GEN_SEQ_0 — Generic Command Sequence Register 0

The NAND Flash Controller is intended for use with a wide range of host clock rates. To maximize flexibility, some timing parameters are configurable. Some waveform configuration parameters for the Generic Sequence are defined in this register. The value generated by the controller equals the minimum value written into the register, increased by 1. All the timings are generated using the NAND_ECLK clock signal. For more detail, refer to **Section 7.5.3, Generic Command Sequence**.

Address: 4010 2098h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	t0_d3						—	—	t0_d2					
Value after reset	X	X	0	0	0	0	0	0	X	X	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	t0_d1						—	—	t0_d0					
Value after reset	X	X	0	0	0	0	0	0	X	X	0	0	0	0	0	0

Table 7.29 TIME_GEN_SEQ_0 Register Contents

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved		R
b29 to b24	t0_d3	Command to data time The time between sending a command and data transferring	R/W
b23, b22	Reserved		R
b21 to b16	t0_d2	Command to delay time The time between sending a command to the NAND Flash memory device and waiting until the memory is ready	R/W
b15, b14	Reserved		R
b13 to b8	t0_d1	Command to command time The time between two subsequent commands sent to the NAND Flash memory device.	R/W
b7, b6	Reserved		R
b5 to b0	t0_d0	Command to address time The time between sending a command and sending the address to the NAND Flash memory device.	R/W

7.4.29 TIME_GEN_SEQ_1 — Generic Command Sequence Register 1

The NAND Flash Controller is intended for use with a wide range of host clock rates. To maximize flexibility, some timing parameters are configurable. Some waveform configuration parameters for the Generic Sequence are defined in this register. The value generated by the controller equals the minimum value written into the register, increased by 1. All the timings are generated using the NAND_ECLK clock signal. For more detail, refer to **Section 7.5.3, Generic Command Sequence**.

Address: 4010 209Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	t0_d7						—	—	t0_d6					
Value after reset	X	X	0	0	0	0	0	0	X	X	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	t0_d5						—	—	t0_d4					
Value after reset	X	X	0	0	0	0	0	0	X	X	0	0	0	0	0	0

Table 7.30 TIME_GEN_SEQ_1 Register Contents

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved		R
b29 to b24	t0_d7	Address to data time The time between sending the address and data transferring	R/W
b23, b22	Reserved		R
b21 to b16	t0_d6	Address to delay time The time between sending the address to the NAND Flash memory device and waiting until the memory is ready	R/W
b15, b14	Reserved		R
b13 to b8	t0_d5	Address to address time The time between two subsequent addresses sent to the NAND Flash memory device.	R/W
b7, b6	Reserved		R
b5 to b0	t0_d4	Address to command time The time between sending the address and the command to the NAND Flash memory device	R/W

7.4.30 TIME_GEN_SEQ_2 — Generic Command Sequence Register 2

The NAND Flash Controller is intended for use with a wide range of host clock rates. To maximize flexibility, some timing parameters are configurable. Some waveform configuration parameters for the Generic Sequence are defined in this register. The value generated by the controller equals the minimum value written into the register, increased by 1. All the timings are generated using the NAND_ECLK clock signal. For more detail, refer to **Section 7.5.3, Generic Command Sequence**.

Address: 4010 20A0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	t0_d11						—	—	t0_d10					
Value after reset	X	X	0	0	0	0	0	0	X	X	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	t0_d9						—	—	t0_d8					
Value after reset	X	X	0	0	0	0	0	0	X	X	0	0	0	0	0	0

Table 7.31 TIME_GEN_SEQ_2 Register Contents

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved		R
b29 to b24	t0_d11	Data to delay time The time between data transferring and waiting until the memory is ready.	R/W
b23, b22	Reserved		R
b21 to b16	t0_d10	Data to command time The time between data transferring and sending a command to the NAND Flash memory device.	R/W
b15, b14	Reserved		R
b13 to b8	t0_d9	Delay to command time The time between waiting until the memory is ready and sending a command to the NAND Flash memory device.	R/W
b7, b6	Reserved		R
b5 to b0	t0_d8	Delay to data time The time between waiting until the memory is ready and data transferring.	R/W

7.4.31 FIFO_INIT — FIFO Init Register

The setting of this bit causes the flushing of FIFO. It is not necessary to set this bit before sending each command to the NAND Flash memory device. This feature is reserved only for a situation where previous FIFO content must be purged before a new operation.

Address: 4010 20B0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	fifo_init
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

Table 7.32 FIFO_INIT Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b1	Reserved		R
b0	fifo_init	The FIFO init request	W

7.4.32 FIFO_STATE — FIFO Status Register

Address: 4010 20B4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	df_w_empty	df_r_full	cf_accept_w	cf_accept_r	cf_full	cf_empty	df_w_full	df_r_empty
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 7.33 FIFO_STATE Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7	df_w_empty	The FIFO empty state bit If this flag is set, there is no data in the FIFO available. This flag is valid for the write direction.	R
b6	df_r_full	The FIFO full state bit If this flag is set, there is no free space for the data in FIFO available. This flag is valid for the read direction.	R
b5	cf_accept_w	The Command FIFO accept flag - write direction If this flag is set then next write access will finish with any additional delay.	R
b4	cf_accept_r	The Command FIFO accept flag - read direction This is informational flag. If it is set then read transfer on the CMD FIFO internal interface will be accepted with any additional delay.	R
b3	cf_full	The Command FIFO full flag If this flag is set, there is no free space for the data in the actual command FIFO. It can’t be used to check if next transfer will be accepted.	R
b2	cf_empty	The Command FIFO empty flag If this flag is set, there is no data in the actual command FIFO. It can’t be used to check if next transfer will be accepted.	R
b1	df_w_full	The FIFO full state bit If this flag is set, there is no free space for the data in FIFO available. This flag is valid for the write direction.	R
b0	df_r_empty	The FIFO empty state bit If this flag is set, there is no data in the FIFO available. This flag is valid for the read direction.	R

7.4.33 GEN_SEQ_CTRL — Generic Sequence Register

This register is used to parameterize the generic command sequences. For more detail, refer to **Section 7.5.3, Generic Command Sequence**.

Address: 4010 20B8h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	cmd3							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	imd_seq	delay_en		data_en	row_a1		row_a0		col_a1		col_a0		cmd3_en	cmd2_en	cmd1_en	cmd0_en
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.34 GEN_SEQ_CTRL Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved		R
b23 to b16	cmd3	Command 3 code value	R/W
b15	imd_seq	Enable immediate command execution This bit allows the command sequence to be executed without checking the selected target state. 0: Feature disabled 1: Feature enabled	R/W
b14, b13	delay_en	Enable the busy 0 or 1 phase This bit allows enabling or disabling the presence of the “busy” phase in the universal command sequence. 2’b00: disable both delays 2’b01: Enable DELAY0 2’b10: Enable DELAY1 2’b11: Disable both delays	R/W
b12	data_en	Enable data part sequence This bit allows enabling or disabling the data phase of the universal command sequence. 0: Disable data phase 1: Enable data phase	R/W
b11, b10	row_a1	Row address cycles Number of the row address bytes sent to NAND Flash device. 2’b00: 0 address cycles 2’b01: 1 address cycles 2’b10: 2 address cycles 2’b11: 3 address cycles	R/W
b9, b8	row_a0	Row address cycles Number of the row address bytes sent to NAND Flash device. 2’b00: 0 address cycles 2’b01: 1 address cycles 2’b10: 2 address cycles 2’b11: 3 address cycles	R/W

Table 7.34 GEN_SEQ_CTRL Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b7, b6	col_a1	Column address cycles Number of the column address bytes sent to NAND Flash device. 2'b00: 0 address cycles 2'b01: 1 address cycles 2'b10: 2 address cycles 2'b11: Not available	R/W
b5, b4	col_a0	Column address cycles Number of the column address bytes sent to NAND Flash device. 2'b00: 0 address cycles 2'b01: 1 address cycles 2'b10: 2 address cycles 2'b11: Not available	R/W
b3	cmd3_en	Enable command 3 phase This bit allows enabling or disabling the presence of the "command 3" phase in the universal command sequence. 1: Enable 0: Disable	R/W
b2	cmd2_en	Enable command 2 phase This bit allows enabling or disabling the presence of the "command 2" phase in the universal command sequence. 1: Enable 0: Disable	R/W
b1	cmd1_en	Enable command 1 phase This bit allows enabling or disabling the presence of the "command 1" phase in the universal command sequence. 1: Enable 0: Disable	R/W
b0	cmd0_en	Enable command 0 phase This bit allows enabling or disabling the presence of the "command 0" phase in the universal command sequence. 1: Enable 0: Disable	R/W

7.4.34 MLUN — LUN Configuration Register

This register contains the LUN address offset bits and number of available LUNs.

Address: 4010 20BCh

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	mlun_sel		—	—	—	—	—	mlun_idx		
Value after reset	X	X	X	X	X	X	0	0	X	X	X	X	X	0	0	0

Table 7.35 MLUN Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b10	Reserved		R
b9, b8	mlun_sel	LUN number 2'b00: Two LUN-s. 2'b01: Four LUN-s.	R/W
b7 to b3	Reserved		R
b2 to b0	mlun_idx	The LUN address offset The content of this field is used as the bit offset value from the bit zero of the last address byte. This fields point to the bit index in the address that identify active LUN.	R/W

7.4.35 DEV[n]_SIZE — Device [n] BBM Record Counter Register (n = 0..3)

The bad block management mechanism implemented in the controller uses the tables in the system memory to store the remapping records. Each table can store a variable number of records depending on the number of bad blocks in the given NAND Flash device. This stores the number of records in the table for device [n].

Address: 4010 20C0h + 4h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	dev_size											
Value after reset	X	X	X	X	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.36 DEV[n]_SIZE Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b12	Reserved		R
b11 to b0	dev_size	Number of record for device [n]	R/W

7.4.36 DMA_TLVL — DMA Trigger Level Register

This register allows the setting of the data FIFO occupancy level that will trigger the DMA module.

Address: 4010 2114h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	dma_tlvl							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 7.37 DMA_TLVL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7 to b0	dma_tlvl	The DMA trigger level The trigger level is counted using the 32-bit words as entity.	R/W

7.4.37 CMD_MARK — CMD ID Initial Value

Address: 4010 2124h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	cmd_id							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 7.38 CMD_MARK Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7 to b0	cmd_id	CMD ID initial value	W

7.4.38 LUN_STATUS_0 — LUN Status Register

The LUN_STATUS_0 register allow to access the LUN status information for devices 0-3. Each bit of the LUN status field contain the status of single LUN of device. The busy status is marked as logical ‘0’, the ready state is marked as logical ‘1’.

Address: 4010 2128h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	mem3_lun								mem2_lun							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	mem1_lun								mem0_lun							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.39 LUN_STATUS_0 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	mem3_lun	The memory 3 LUN-s status field	R
b23 to b16	mem2_lun	The memory 2 LUN-s status field	R
b15 to b8	mem1_lun	The memory 1 LUN-s status field	R
b7 to b0	mem0_lun	The memory 0 LUN-s status field	R

7.4.39 TIME_GEN_SEQ_3 — Generic Command Sequence Register 3

The NAND Flash Controller is intended for use with a wide range of host clock rates. To maximize flexibility, some timing parameters are configurable. Some waveform configuration parameters for the Generic Sequence are defined in this register. The value generated by the controller equals the minimum value written into the register, increased by 1. All the timings are generated using the NAND_ECLK clock signal. For more detail, refer to **Section 7.5.3, Generic Command Sequence**.

Address: 4010 2134h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	t0_d12					
Value after reset	X	X	X	X	X	X	X	X	X	X	0	0	0	0	0	0

Table 7.40 TIME_GEN_SEQ_3 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b6	Reserved		R
b5 to b0	t0_d12	Data to sequence end time The time between the data transferring phase and sequence end. The sequence is ended when any other sequence phases after the data transfer phase are not enabled.	R/W

7.4.40 INT_STAT — Internal Status Register

The internal operations status register stores the status values read for the internal status sequences send to the NAND Flash device. The higher two bits identify the sequence that was used to get value for the lower field (bits 7 to 0).

Address: 4010 2148h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	seq_id		stat_value							
Value after reset	X	X	X	X	X	X	0	0	0	0	0	0	0	0	0	0

Table 7.41 INT_STAT Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b10	Reserved		R
b9, b8	seq_id	The status operation identification field 2’b00: No pending status value 2’b01: The read status operation after the program/erase sequence. It is enabled in the control register.	R/W
b7 to b0	stat_value	The status operation value	R/W

7.4.41 ECC_CNT — ECC Error Counter

This register stores number of value detected during last page read operation. This register content isn't automatically cleared it must be done by software. The new page read operation doesn't overwrite previous register value. Register contain value of the largest error level detected in processed ECC blocks.

Address: 4010 214Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	err_lvl3						—	—	err_lvl2					
Value after reset	X	X	0	0	0	0	0	0	X	X	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	err_lvl1						—	—	err_lvl0					
Value after reset	X	X	0	0	0	0	0	0	X	X	0	0	0	0	0	0

Table 7.42 ECC_CNT Register Contents

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved		R
b29 to b24	err_lvl3	The Memory device 3 error level	R/W
b23, b22	Reserved		R
b21 to b16	err_lvl2	The Memory device 2 error level	R/W
b15, b14	Reserved		R
b13 to b8	err_lvl1	The Memory device 1 error level	R/W
b7, b6	Reserved		R
b5 to b0	err_lvl0	The Memory device 0 error level	R/W

7.4.42 PARAM_REG — PARAMETER Register

This register contains fields that describes the configuration of the NAND Flash Controller core. It is read only register.

Address: 4010 2150h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	asyn_reset_implement	—	—	—	—	dma_implement	small_block_implement	gen_seq_implement	—	ss_implement	noecc_implement	—	—	—
Value after reset	X	X	1	X	X	X	X	1	1	1	X	0	0	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	bch32_conf_implement	big_endian_implement	clearand_implement	toggle_mode_implement	syn_mode_implement	protect_implement	bbm_int_implement	bbm_implement	boot_implement	device_per_bank		bank_num	
Value after reset	X	X	X	1	0	0	0	0	1	1	1	1	1	0	0	0

Table 7.43 PARAM_REG Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved		R
b29	asyn_reset_implement	Flag of the asynchronous reset feature presence This field describes the type of reset implemented in the controller 0: Synchronous reset is implemented 1: Asynchronous reset is implemented	R
b28 to b25	Reserved		R
b24	dma_implement	Flag of the DMA master feature presence This field indicates if the DMA master module is implemented in the controller 0: DMA master is not implemented 1: DMA master is implemented	R
b23	small_block_implement	Flag of the small block devices support feature presence This field specifies if the small block support is implemented in the controller 0: Small block support is not implemented 1: Small block support is implemented	R
b22	gen_seq_implement	Flag of the generic sequence feature presence This field specifies if the generic sequence feature is implemented in the controller 0: Generic sequence is not implemented 1: Generic sequence is implemented	R
b21	Reserved		R
b20	ss_implement	Flag of the super sequence feature presence This field specifies if the super sequence feature is implemented in the controller 0: Super Sequence is not implemented 1: Super Sequence is implemented	R
b19	noecc_implement	Flag of the ECC feature lack This field indicates if the controller does not contain the correction module 0: Correction module is implemented 1: Correction module is not implemented	R
b18 to b13	Reserved		R
b12	bch32_conf_implement	Flag of the BCH32_CONF feature presence This field indicates if the BCH32 conf correction module is implemented in the controller 0: BCH32 conf is not implemented 1: BCH32 conf is implemented	R

Table 7.43 PARAM_REG Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b11	big_endian_implement	Flag that mark selected endianness type This field indicates endianness of the controller 0: Little endian is implemented 1: Big endian is implemented	R
b10	clearnand_implement	Flag of the ClearNAND™ support presence This field indicates if the ClearNAND support is implemented in the controller 0: ClearNAND support is not implemented 1: ClearNAND support is implemented	R
b9	toggle_mode_implement	Flag of the toggle mode feature presence This field indicates if the Toggle Mode support is implemented in the controller 0: Toggle Mode support is not implemented 1: Toggle Mode support is implemented	R
b8	syn_mode_implement	Flag of the synchronous work mode feature presence This field indicates if the synchronous work mode support is implemented in the controller 0: Synchronous mode support is not implemented 1: Synchronous mode support is implemented	R
b7	protect_implement	Flag of the protect feature presence This field specify if the write protection feature is implemented in the controller 0: Write protection is not implemented 1: Write protection is implemented	R
b6	bbm_int_implement	Memory location of the bad block management feature This field indicates the memory location for the BBM mechanism 0: External memory access by the master interface 1: Internal memory build in the controller	R
b5	bbm_implement	Flag of the bad block management feature presence This field specify if the BBM feature is implemented in the controller 0: BBM is not implemented 1: BBM is implemented	R
b4	boot_implement	Flag of the boot feature presence This field specify if the boot feature is implemented in the controller 0: BOOT is not implemented 1: BOOT is implemented	R
b3, b2	device_per_bank	Number of devices per bank This field indicates the number of devices per bank that can be handled by the controller 2’b00: 1 device per bank 2’b01: 2 devices per bank 2’b10: 4 devices per bank 2’b11: 8 devices per bank	R
b1, b0	bank_num	Number of banks This field indicates the number of banks that can be handled by the controller 2’b00: 1 bank 2’b01: 2 banks 2’b10: 4 banks 2’b11: 8 banks	R

7.5 Operation

7.5.1 Programming the NAND Flash Controller

To be able to support future NAND Flash devices, the NAND Flash Controller provides the configurable instruction set that can be extended as new commands in new devices appear.

7.5.2 Command Generation

The NAND Flash devices are constantly extended and evaluated to the larger capacities and higher data throughput. Frequently during this process, new commands appear in the next generation of devices. To allow use of the NAND Flash Controller with future generations of devices, the parameterizable command sequences have been added.

The parameterization allows for a defining set of parameters for each supported command sequence.

7.5.2.1 Instruction Encoding

The controller instruction field is constant and has 32 bits. The instruction field contains the command sequence code and optional parameters. Those parameters are:

- The command codes present in the instruction sequence.
- The flag used to select data destination; possible options are DATA register and FIFO module.
- The flag used to select data source/sink for the command sequence. The possible choices are the AHBS module or the DMA module.
- The command sequence code.

If the given command sequence does not use all parameter fields those unused are ignored. The instruction encoding scheme is presented in the table below:

Table 7.44 Instruction Encoding

Field Name	Bits	Description
CMD_2	[31:24]	Code of the third command in a sequence.
CMD_1/CMD_3*1	[23:16]	Code of the second or fourth command in a sequence.
CMD_0	[15:8]	Code of the first command in a sequence
DATA_SEL	[7]	Data or FIFO selection flag: 0: The FIFO module selected 1: The DATA register selected
INPUT_SEL	[6]	Input module select flag: 0: Select the AHBS module as input 1: Select the DMA module as input
CMD_SEQ	[5:0]	Command code.

Note 1. Depending on the selected command sequence, this field will store the CMD1 or CMD3 code. Both commands are never used in a single sequence.

7.5.2.2 Command Sequence Encoding

The NAND Flash devices use the same set of signals independently of the memory capacity. This allows upgrading of obsolete devices with newer ones without PCB redesign. The disadvantage of this is a more complicated access protocol for the device.

The NAND Flash devices use common IO bus to transfer commands, addresses and data. The predefined set of command sequences is used for the read and write operations on those devices. The set of supported command sequences is not constant for all the NAND Flash device producers and evolves as the devices are more capable.

The NAND Flash Controller must be able to support the new NAND Flash device features as they appear with a minimum effort from the designer side. This objective can be achieved in many cases because most of the new instructions use the previously defined sequence of commands and addresses, along with new command codes, data page size, data spare area size, etc.

The NAND Flash Controller defines the set of commands, addresses and data sequences that allows implementation of all present and many future instructions. The following description of those sequences is adequate to define most of the future NAND Flash device instructions.

The table below contains the command sequence encoding details. Each sequence is encoded according to the fields defined in the GEN_SEQ_CTRL register.

Table 7.45 Command Sequence Encoding

Sequence symbol	Sequence encoding	CMD0	CMD1	CMD2	CMD3	COL_A0*2	COL_A1	ROW_A0	ROW_A1	DATA_EN	DELAY_EN	IMD_SEQ*3
SEQ_0	000000	✓	—	—	—	—	—	—	—	—	DELAY1	—
SEQ_1	100001	✓	—	—	—	1	—	—	—	✓	—	—
SEQ_2	100010	✓	—	—	—	1	—	—	—	✓	DELAY0	—
SEQ_3	000011	✓	—	—	—	1	—	—	—	✓	DELAY1	—
SEQ_4	100100	✓	—	—	—	—	—	—	—	✓	—	✓
SEQ_5	100101	✓	—	—	—	—	—	3	—	✓	—	✓
SEQ_6	100110	✓	—	✓	—	2(1)	—	—	—	✓	—	—
SEQ_7	100111	✓	—	✓	—	2(1)	—	3	—	✓	DELAY0	—
SEQ_8	001000	✓	—	—	—	2(1)	—	—	—	✓	—	—
SEQ_9	101001	✓	✓	—	—	2(1)	—	3	—	—	DELAY1	—
SEQ_10	101010	✓	—	✓	—	2(1)	—	3	—	✓	DELAY0	—
SEQ_11	101011	✓	—	—	—	—	—	—	—	✓	DELAY0	—
SEQ_12	001100	✓	✓	—	—	2(1)	—	3	—	✓	DELAY1	—
SEQ_13	001101	✓	—	—	—	2(1)	—	3	—	✓	DELAY1	—
SEQ_14	001110	✓	✓	—	—	—	—	3	—	—	DELAY1	—
SEQ_15	101111	✓	—	✓	✓	2	2	3	3	✓	DELAY0	—
SEQ_17	110001	✓	—	—	—	2(1)	—	3	—	✓	DELAY1	—
SEQ_18	110010	*1	*1	*1	*1	*1	*1	*1	*1	*1	*1	*1
SEQ_19	010011	*1	*1	*1	*1	*1	*1	*1	*1	*1	*1	*1
SEQ_20	010100	✓	—	—	—	—	—	3	—	—	DELAY1	—
SEQ_21	010101	✓	—	—	—	1	—	—	—	—	—	—
SEQ_22	110110	✓	—	✓	—	2(1)	2	—	3	✓	DELAY0	—
SEQ_23	010111	✓	✓	—	—	—	—	3	—	✓	DELAY1	—
SEQ_24	011000	✓	—	✓	✓	—	—	3	3	—	DELAY0	—
SEQ_25	111001	✓	—	✓	✓	2(1)	2	3	—	✓	—	—

Note: Gray rows – read from NAND Flash memory; White rows – write to NAND Flash memory; Blue rows – Non-directional commands

Note 1. SEQ_18 and SEQ_19 are the parameterized Generic Sequences. The GEN_SEQ_CTRL register defines which parts of sequences are executed.

Note 2. The value given in the brackets relates to the small block mode. In this mode, the controller sends only a single byte as the column address.

Note 3. IMD_SEQ – The command will be sent immediately.

7.5.2.3 Sequence SEQ_0

This non-directional sequence is composed from only one command.

After the command is written to the NAND Flash device, the controller waits until the device goes into the busy state and drives the RnB line low, or sends the READ STATUS command. When delay time (tWB) passes or the device is ready, the sequence ends.

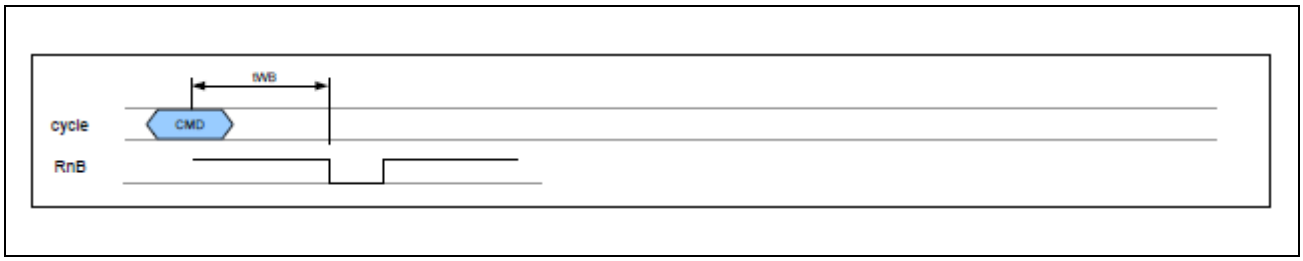


Figure 7.2 SEQ-0 Sequence

7.5.2.4 Sequence SEQ_1

This is a read-sequence that is composed from a single command cycle, single address cycle and the single data cycle with a programmable number of read sequences.

After the address sequence is finished, the controller measures the standard delay of first data read after the last write (tWHR). Next, the read data words are written to the FIFO module. The input module is selected by the INPUT_SEL field of the COMMAND register, the source for the address is placed in the ADDR0_COL register, and the commend code is stored in the CMD_0 field.

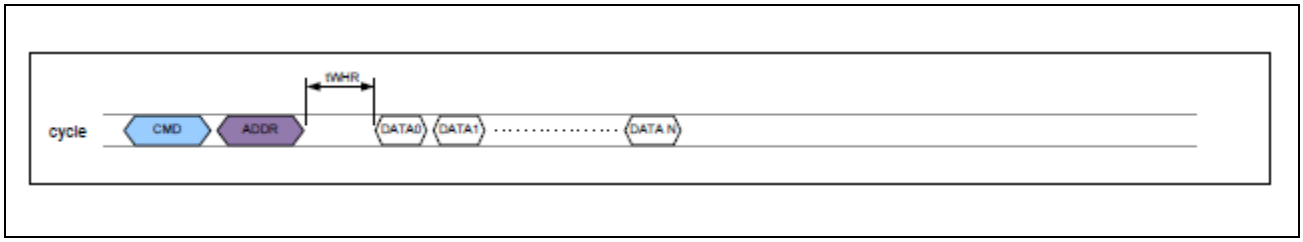


Figure 7.3 SEQ-1 Sequence

7.5.2.5 Sequence SEQ_2

This is a read-sequence and it is similar to the SEQ_1 sequence except that after the address cycle the controller expects that device goes to the busy state.

The controller sequentially checks state of the RnB line or send READ STATUS command to obtain the NAND Flash device status.

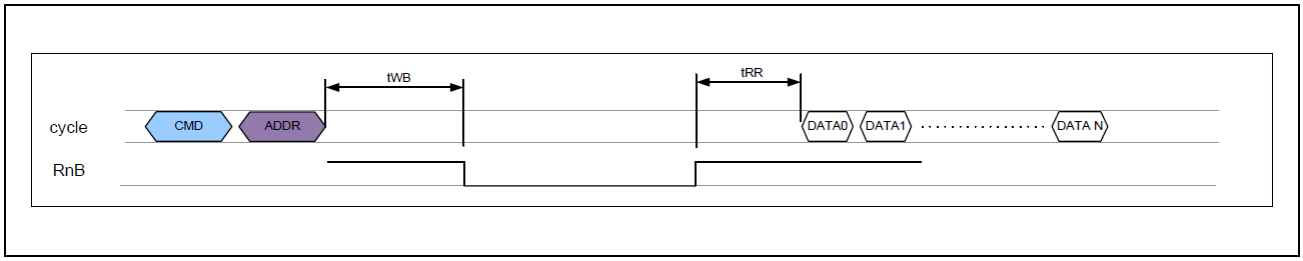


Figure 7.4 SEQ-2 Sequence

7.5.2.6 Sequence SEQ_3

This is a write-sequence that is composed from a single command cycle, single address cycle and single data cycle with a programmable number of write sequences.

After the address sequence is finished, the controller measures the standard delay of first data write after the last address cycle (tADL). The written words are read from the FIFO module. The controller sequentially checks the state of the RnB line or sends the READ STATUS command to obtain the NAND Flash device status. The input module is selected by the INPUT_SEL field of the COMMAND register; the source for the address is placed in the ADDR0_COL register, and the commend code is stored in the CMD_0 field.

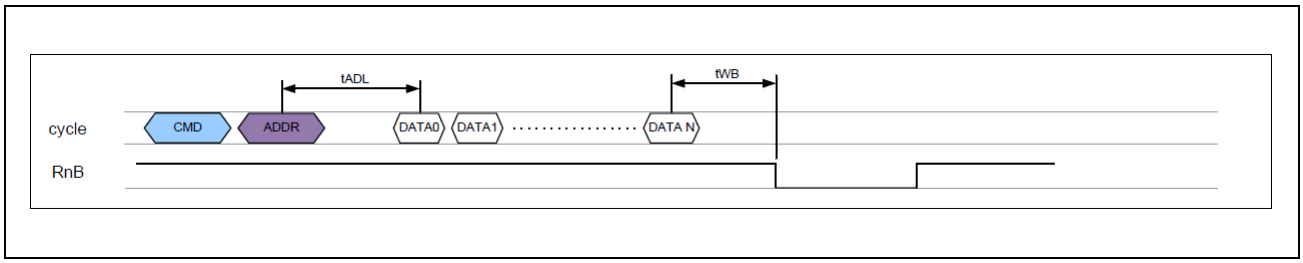


Figure 7.5 SEQ-3 Sequence

7.5.2.7 Sequence SEQ_4

This is a special read-sequence that is used to implement the read status command sequences.

The command is sent immediately. The sequence is composed of a single command cycle and a single data cycle. Between those cycles, the delay is counted (tWHR). The command code is read from the CMD_0 field.

When the DATA register is selected in the COMMAND register, the data is stored in the DATA register. The user defines the number of data in the DATA_REG_SIZE register. The registered mode is allowed only for the read direction.

When the FIFO register is selected in the COMMAND register, the data is stored in the FIFO. Because user has to change DATA_SIZE register, the command will be sent when all memories are ready and the Controller is in the idle state.

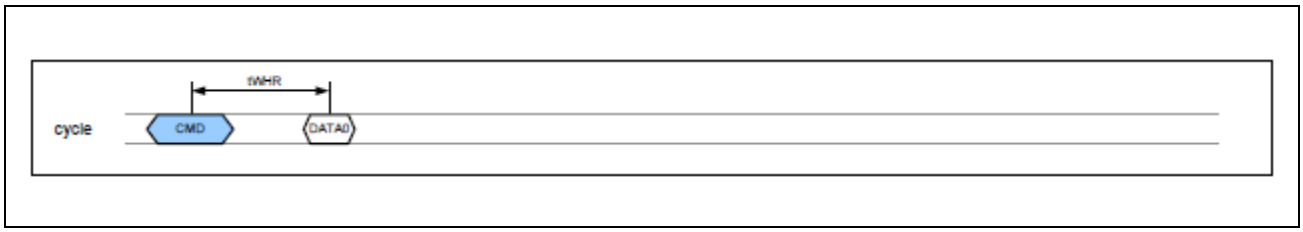


Figure 7.6 SEQ-4 Sequence

7.5.2.8 Sequence SEQ_5

This is a read-sequence and it is similar to the SEQ_4 sequence.

The command is sent immediately. The only difference is that after the command cycle, an additional address cycle is performed. The ADDR0_ROW register is used in this sequence.

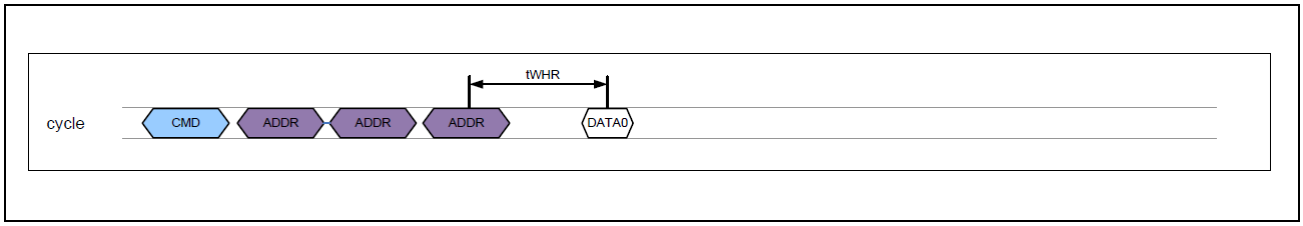


Figure 7.7 SEQ-5 Sequence

7.5.2.9 Sequence SEQ_6

This is a read-sequence.

The sequence of command cycle, address cycle, command cycle is executed. After that, the delay from the change column to the next operation (tCCS) is measured. Finally, the read data cycle is executed. The first command code is encoded in the CMD_0 instruction field; the second command code is encoded in the CMD_2 instruction field; the ADDR0_COL register is used in this sequence; the input module is selected by the INPUT_SEL field.

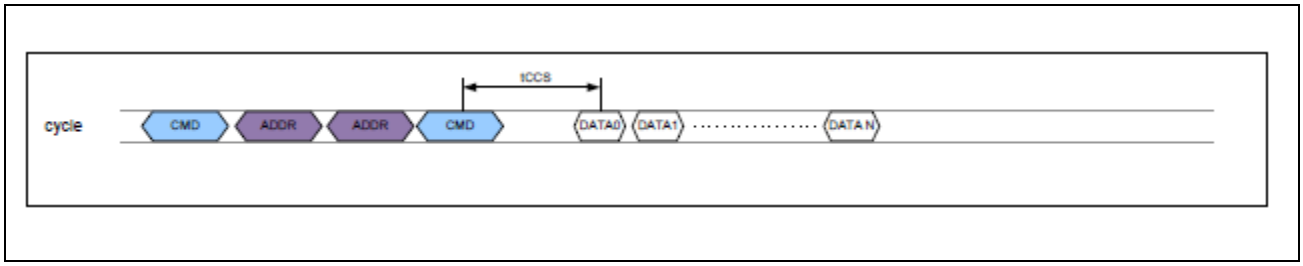


Figure 7.8 SEQ-6 Sequence

7.5.2.10 Sequence SEQ_7

This read-sequence is similar to the SEQ_6 sequence, differing only because the address cycle in this sequence is composed of five bytes (ADDR0_COL and ADDR0_ROW) instead of three bytes.

All else is the same as in the SEQ_6 sequence.

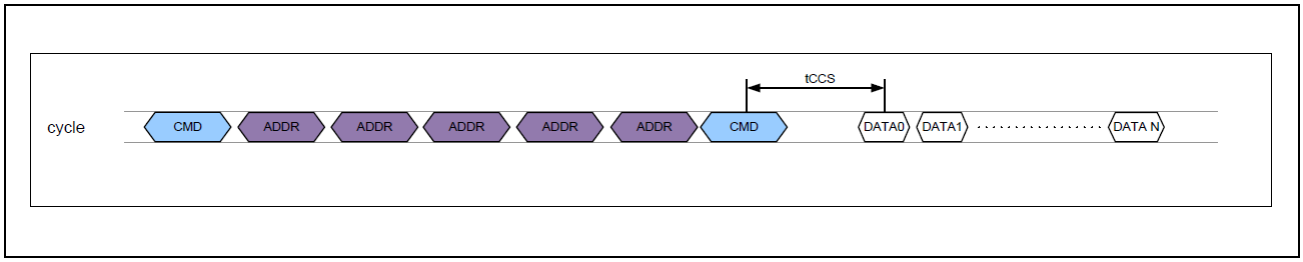


Figure 7.9 SEQ-7 Sequence

7.5.2.11 Sequence SEQ_8

This is a write-sequence.

First, the sequence of command cycle and two bytes address cycle is executed. Next, the delay after the column address changes (tCCS) is measured. Finally, the single data cycle with programmable number of write sequences is executed.

The first command code is encoded in the CMD_0 instruction field; the ADDR0_COL register is used in this sequence; the input module is selected by the INPUT_SEL field.

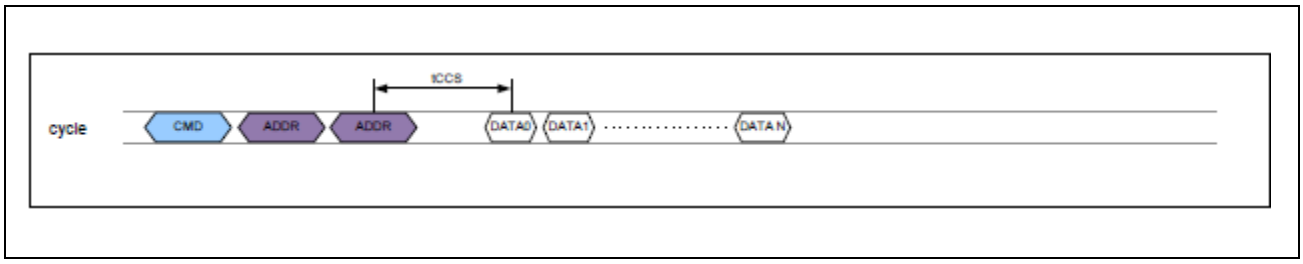


Figure 7.10 SEQ-8 Sequence

7.5.2.12 Sequence SEQ_9

This is a non-directional sequence.

The first step is to execute the five bytes address command cycle. The controller sequentially checks the state of the RnB line or sends the READ STATUS command to obtain the NAND Flash device status.

The first command code is encoded in the CMD_0 instruction field; the second command code is encoded in the CMD_1 instruction field; the ADDR0_COL and ADDR0_ROW registers are used in this sequence.

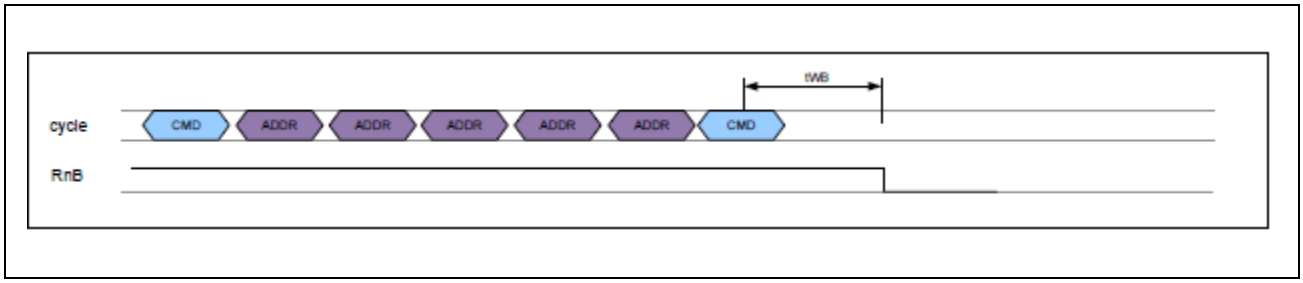


Figure 7.11 SEQ-9 Sequence

7.5.2.13 Sequence SEQ_10

This is a read-sequence.

The SEQ_10 sequence is similar to the SEQ_9, except this sequence is extended by the data read cycle. All else is the same.

The input module is selected by the INPUT_SEL field.

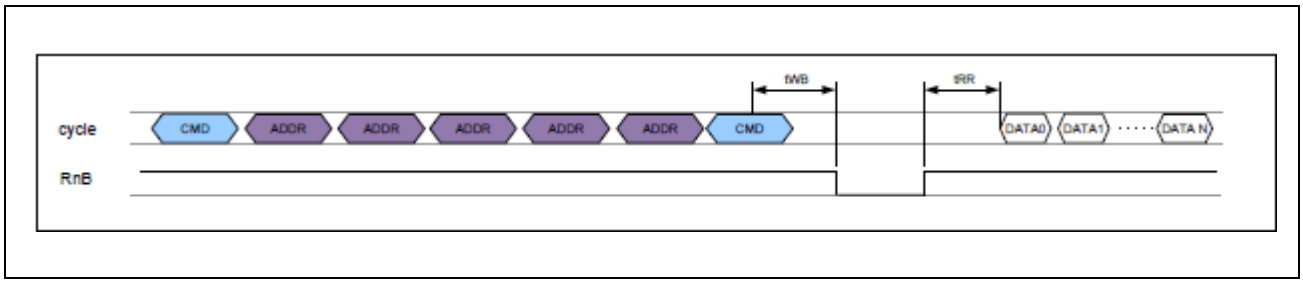


Figure 7.12 SEQ-10 Sequence

7.5.2.14 Sequence SEQ_11

This is the read-sequence.

The first step is to execute the command cycle. Next, the device goes to the busy state. The controller sequentially checks the state of the RnB line or sends the READ STATUS command to obtain the NAND Flash device status. As soon as the device reaches the ready state, the write data cycle with configurable read sequences is executed.

The command code is encoded in the CMD_0 instruction field; the input module is selected by the INPUT_SEL field. The number of transferred bytes are configured using the DATA_SIZE register.

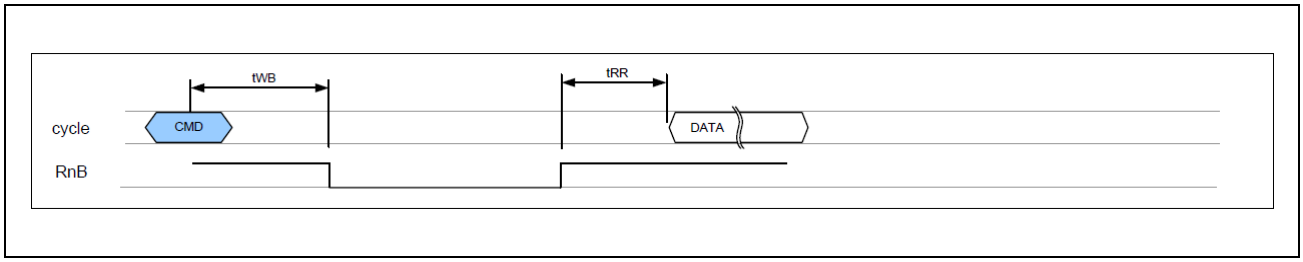


Figure 7.13 SEQ-11 Sequence

7.5.2.15 Sequence SEQ_12

This is a write-sequence.

The SEQ_12 sequence is a series of command cycle, address cycle, and data cycle with a configurable number of write operations and another command cycle. Between the last address cycle and first data cycle, a delay is measured (tADL) and, after the second command cycle, another delay is measured (tWB).

The first command code is encoded in the CMD_0 instruction field; the second command code is encoded in the CMD_1 instruction field; the ADDR0_COL and ADDR0_ROW registers are used in this sequence; the input module is selected by the INPUT_SEL field.

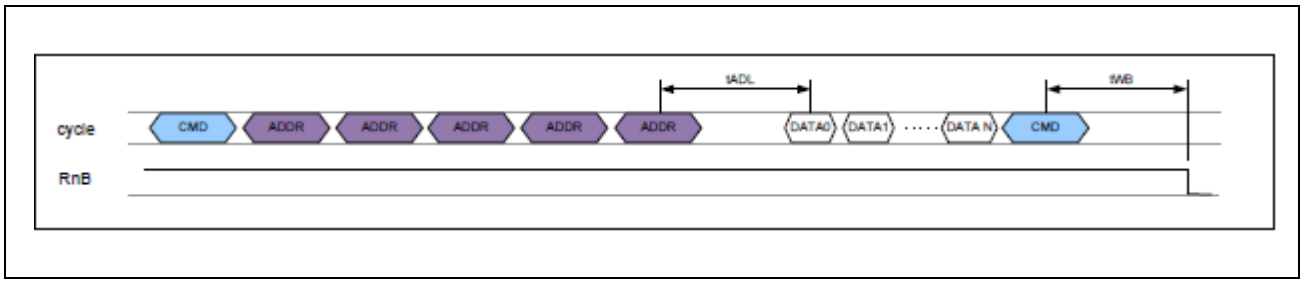


Figure 7.14 SEQ-12 Sequence

7.5.2.16 Sequence SEQ_13

This is a write-sequence.

The SEQ_13 sequence is a series of command cycle and address cycles, data cycle with a configurable number of write operations. Between the last address cycle and first data cycle, a delay is measured (tADL).

The command code is encoded in the CMD_0 instruction field; the ADDR0_COL and ADDR0_ROW registers are used in this sequence; the input module is selected by the INPUT_SEL field.

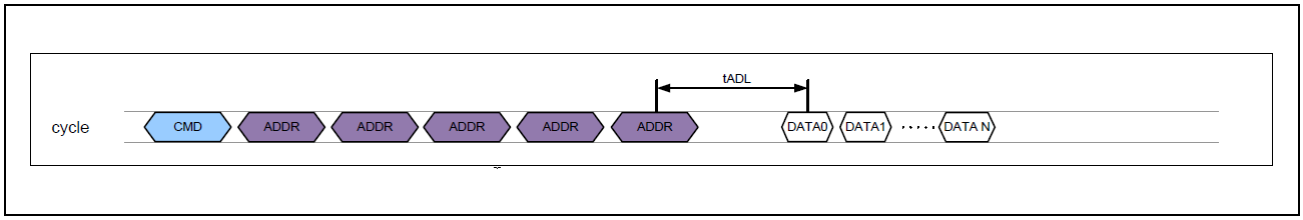


Figure 7.15 SEQ-13 Sequence

7.5.2.17 Sequence SEQ_14

This is a non-directional sequence.

First, the series of command cycle, address cycle, command cycle is executed. The controller sequentially checks the state of the RnB line or sends the READ STATUS command to obtain the NAND Flash device status.

The first command code is encoded in the CMD_0 instruction field; the second command code is encoded in the CMD_1 instruction field; the ADDR0_ROW and ADDR0_COL registers are used in this sequence.

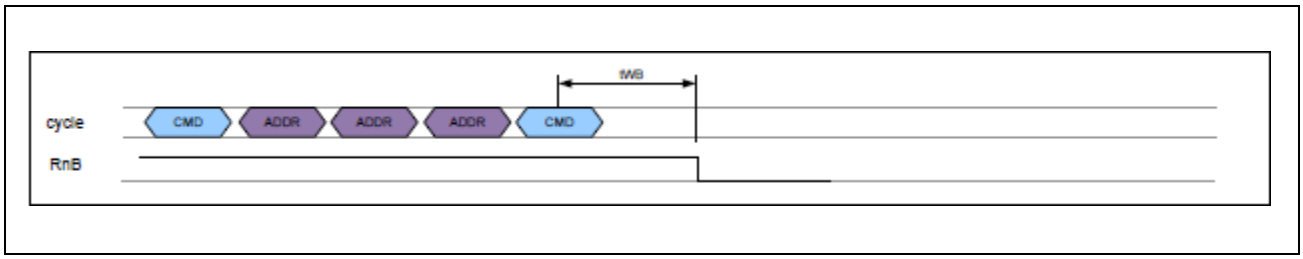


Figure 7.16 SEQ-14 Sequence

7.5.2.18 Sequence SEQ_15

This is a read-sequence.

First, the series of command cycle, address cycle, second command cycle, second address cycle, third command cycle is executed. The controller sequentially checks the state of the RnB line or sends the READ STATUS command to obtain the NAND Flash device status. After the NAND Flash device returns to the ready state, the data sequence, with a configurable number of read operations, is executed.

The first command code is encoded in the CMD_0 instruction field; the second command code is encoded in the CMD_2 instruction field; the third command code is encoded in the CMD_3 instruction field.

In this sequence, all address registers are used. The ADDR0_ROW and ADDR0_COL register content is sent after the first command in the sequence, the ADDR1_ROW and ADDR1_COL register content is sent after the second command in the sequence.

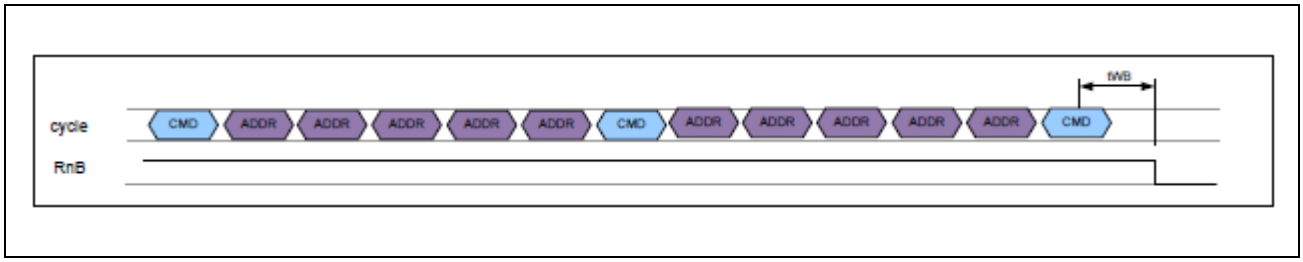


Figure 7.17 SEQ-15 Sequence

7.5.2.19 Sequence SEQ_17

This is a read-sequence.

This sequence is similar to the SEQ_10 sequence, except that the second command cycle is omitted.

This sequence is implemented to use small block memories. The controller sends only four bytes of the address when the small block mode is enabled.

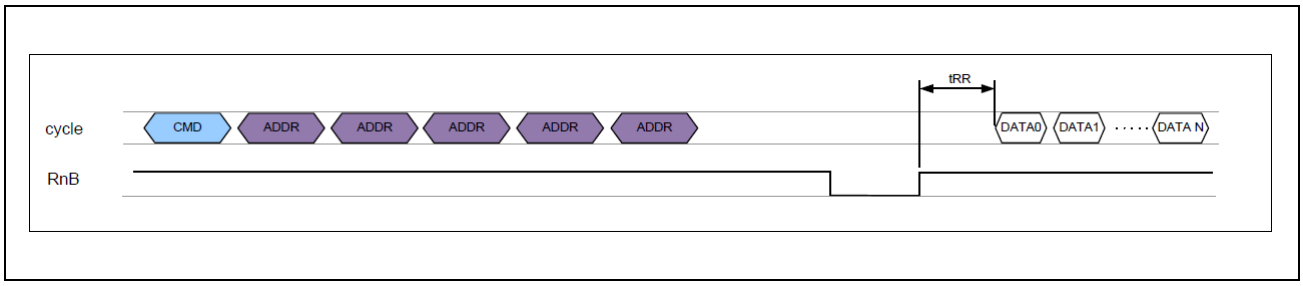


Figure 7.18 SEQ-17 Sequence

7.5.2.20 Sequence SEQ_18

This generic read sequence is described in detail in the following **Section 7.5.3, Generic Command Sequence**.

7.5.2.21 Sequence SEQ_19

This generic write sequence is described in detail in the following **Section 7.5.3, Generic Command Sequence**.

7.5.2.22 Sequence SEQ_20

This non-directional sequence is composed from one command and three addresses bytes.

After the command and addresses are written to the NAND Flash device, the controller waits until the device goes into the busy state and drives the RnB line low, or sends the READ STATUS command. When delay time (tWB) passes or the device is ready, the sequence ends.

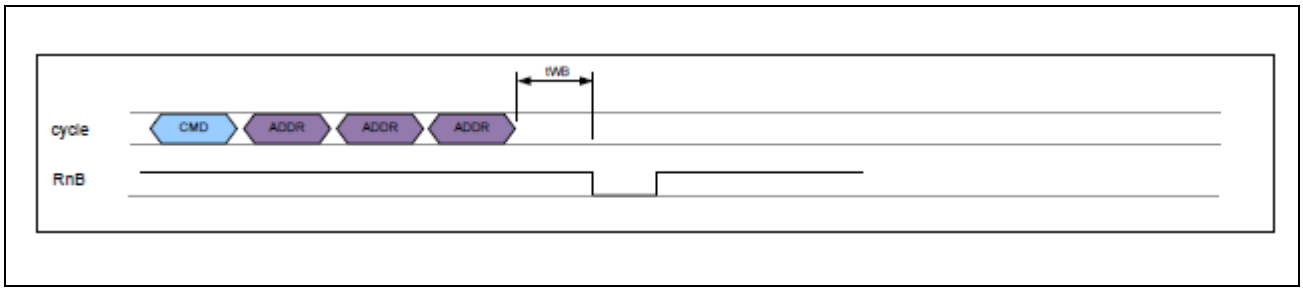


Figure 7.19 SEQ-20 Sequence

7.5.2.23 Sequence SEQ_21

This non-directional sequence is composed from one command and one addresses byte.

After the command and address are written to the NAND Flash device, the sequence ends.

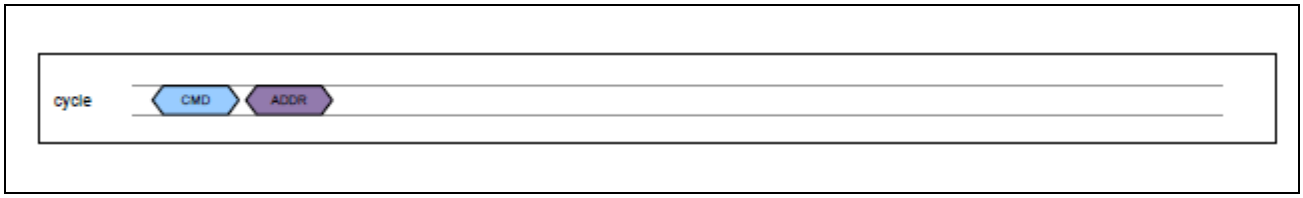


Figure 7.20 SEQ-21 Sequence

7.5.2.24 Sequence SEQ_22

This is a read-sequence.

The first step is to execute the address command cycle. The controller sequentially checks the state of the RnB line or sends the READ STATUS command to obtain the NAND Flash device status.

The first command code is encoded in the CMD_0 instruction field; the second command code is encoded in the CMD_2 instruction field; the ADDR0_COL, ADDR1_COL and ADDR1_ROW registers are used in this sequence.

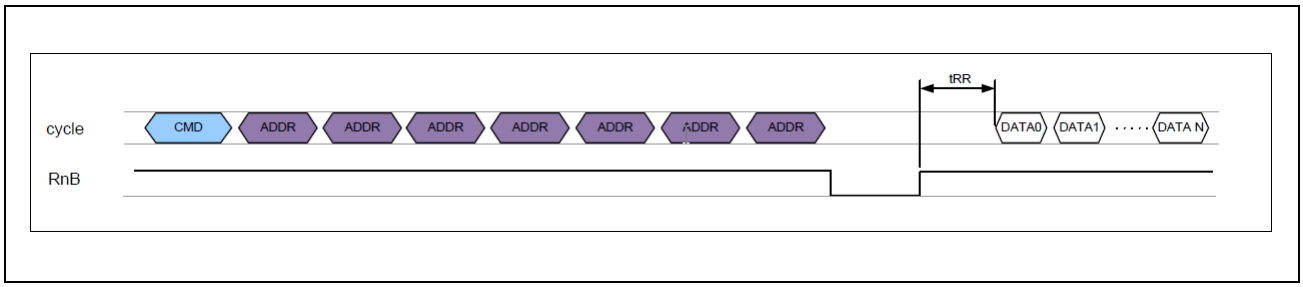


Figure 7.21 SEQ-22 Sequence

7.5.2.25 Sequence SEQ_23

This is a write-sequence.

The SEQ_23 sequence is a series of command cycle, address cycle, and data cycle with a configurable number of write operations and another command cycle. Between the last address cycle and first data cycle, a delay is measured (tADL) and, after the second command cycle, another delay is measured (tWB).

The first command code is encoded in the CMD_0 instruction field; the second command code is encoded in the CMD_1 instruction field; the ADDR0_ROW registers is used in this sequence; the input module is selected by the INPUT_SEL field.

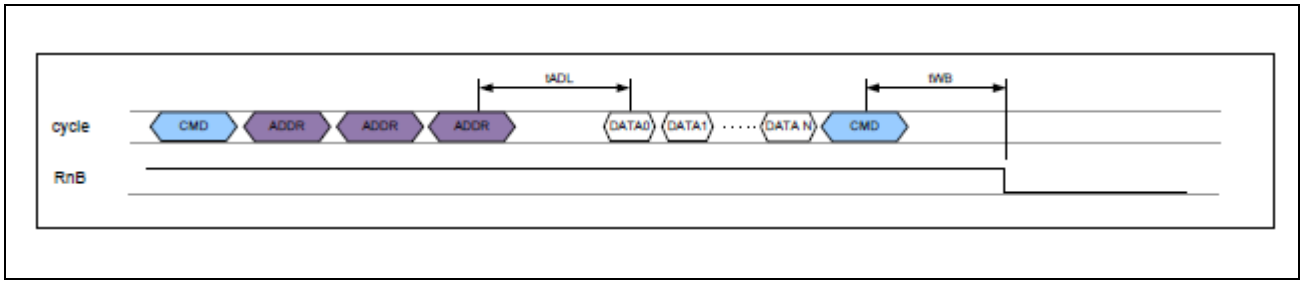


Figure 7.22 SEQ-23 Sequence

7.5.2.26 Sequence SEQ_24

This is a write-sequence.

It is composed from the three commands cycles and two addresses cycles. Both addresses cycle contain the row address part. After the last command cycle the tWB delay is measured.

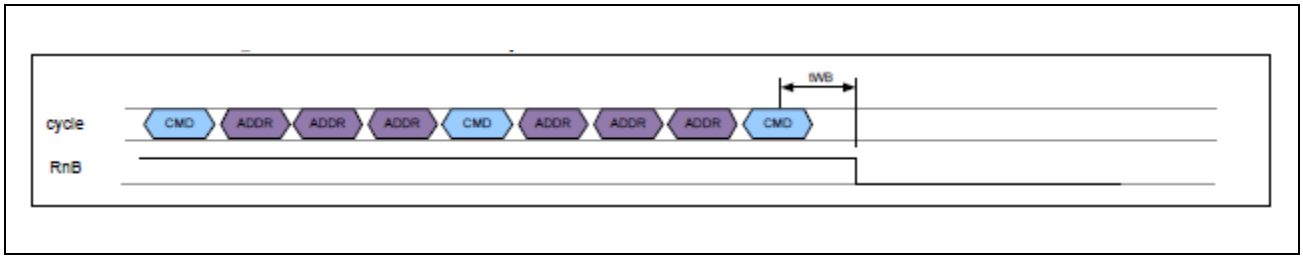


Figure 7.23 SEQ-24 Sequence

7.5.2.27 Sequence SEQ_25

This is a read-sequence.

It is composed from the three commands cycles and two addresses cycles. The first addresses cycle contain the column and row address part. The second address cycle contain only the column address part. After the last command cycle the tWHR delay is measured.

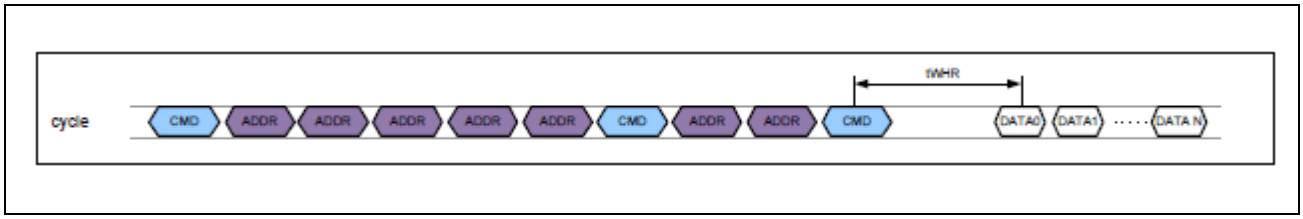


Figure 7.24 SEQ-25 Sequence

7.5.3 Generic Command Sequence

There will be cases where the set of predefined sequences above will not be sufficient to handle a new command sequence. If that occurs, the generic command sequence feature of the NAND Flash Controller can be used.

This sequence is designed to mimic almost every available command supported by the NAND Flash devices; however, additional effort required to trigger such commands.

Generic command sequence is executed in the following steps:

- CMD0

  The first command in the sequence. The value of this command is stored in the CMD_0 field of the COMMAND register.

- ADDR0

  The first address sequence. This is enabled if the COL_A0 and ROW_A0 fields in the GEN_SEQ_CTRL register have values other than zero. In this phase, the address is sent to the NAND Flash device and is read from the ADDR0_COL and ADDR0_ROW registers. The number of the bytes in the address cycle is configured by the COL_A0 and ROW_A0 fields of the GEN_SEQ_CTRL register.

- CMD1

  The fourth command in the sequence. The value of this command is stored in the CMD_1 field of the COMMAND register. This is enabled by the CMD1_EN field in the GEN_SEQ_CTRL register.

- ADDR1

  The second address in the sequence. This is enabled if the COL_A1 and ROW_A1 fields of the GEN_SEQ_CTRL register have a value other than zero. In this phase, the address is sent to the NAND Flash device from the ADDR1_COL and ADDR1_ROW registers. The number of bytes in the address cycle is configured by the COL_A1 and ROW_A1 fields of the GEN_SEQ_CTRL register.

- CMD2

  The third command in the sequence. The value of this command is stored in the CMD_2 field of the COMMAND register. This is enabled by the CMD2_EN field in the GEN_SEQ_CTRL register.

- DELAY0

  Waiting for the device to return to the ready state and continue the sequence. This is enabled by the DELAY_EN field in the GEN_SEQ_CTRL register. Only one delay phase can be present in the generic sequence.

- DATA

  The data phase of the sequence. This is enabled by the DATA_EN field in the GEN_SEQ_CTRL register. Additionally, the transfer direction must be selected by the sequence number. Sequence number 18 reads data from the NAND Flash memory, sequence number 19 writes data to the NAND Flash memory. The size of the transferred data block is configured by the DATA_SIZE register value.

- CMD3

  The second command in the sequence. This is enabled by the CMD3_EN field in the GEN_SEQ_CTRL register. The value of this command is stored in the CMD_3 field of the GEN_SEQ_CTRL register.

- DELAY1

  Waiting for the device to return to the ready state and finish the sequence. This is enabled by the DELAY_EN field in the GEN_SEQ_CTRL register. The controller waits for the device to return to the ready state and finish the sequence. Only one delay phase can be present in generic sequence.

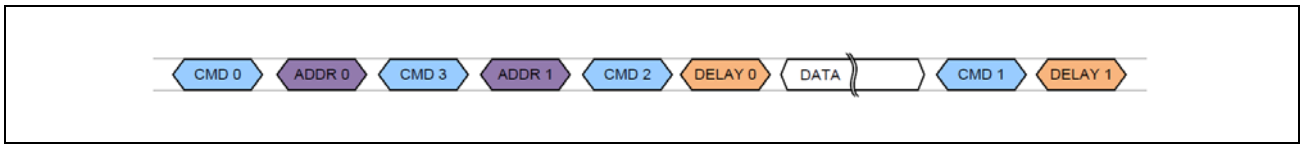


Figure 7.25 Generic Sequence

There are few constraints on the generic sequence usage:

- The DELAY0 and DELAY1 delay phases cannot both be enabled in a single command sequence.
- The TIMINGS_ASYN registers must be set even when the generic sequence is used.

It is possible to force an immediate command sequence execution by enabling the IMD_SEQ bit in the GEN_SEQ_CTRL register.

In this case, the triggered command will be executed even if the previously sent command for a selected device is not completed. If both the IMD_SEQ and DATA_EN flags are enabled in a single sequence, then the register must be selected as data source/sink. The IMD_SEQ is valid only for the read direction. This feature is intended to implement all state read operations.

After each step of the generic sequence, the programmable delay time is measured. These delays are configured using the TIME_GEN_SEQ[0-2] registers. Refer to the TIME_GEN_SEQ[0-2] registers description for further information and see figures below.

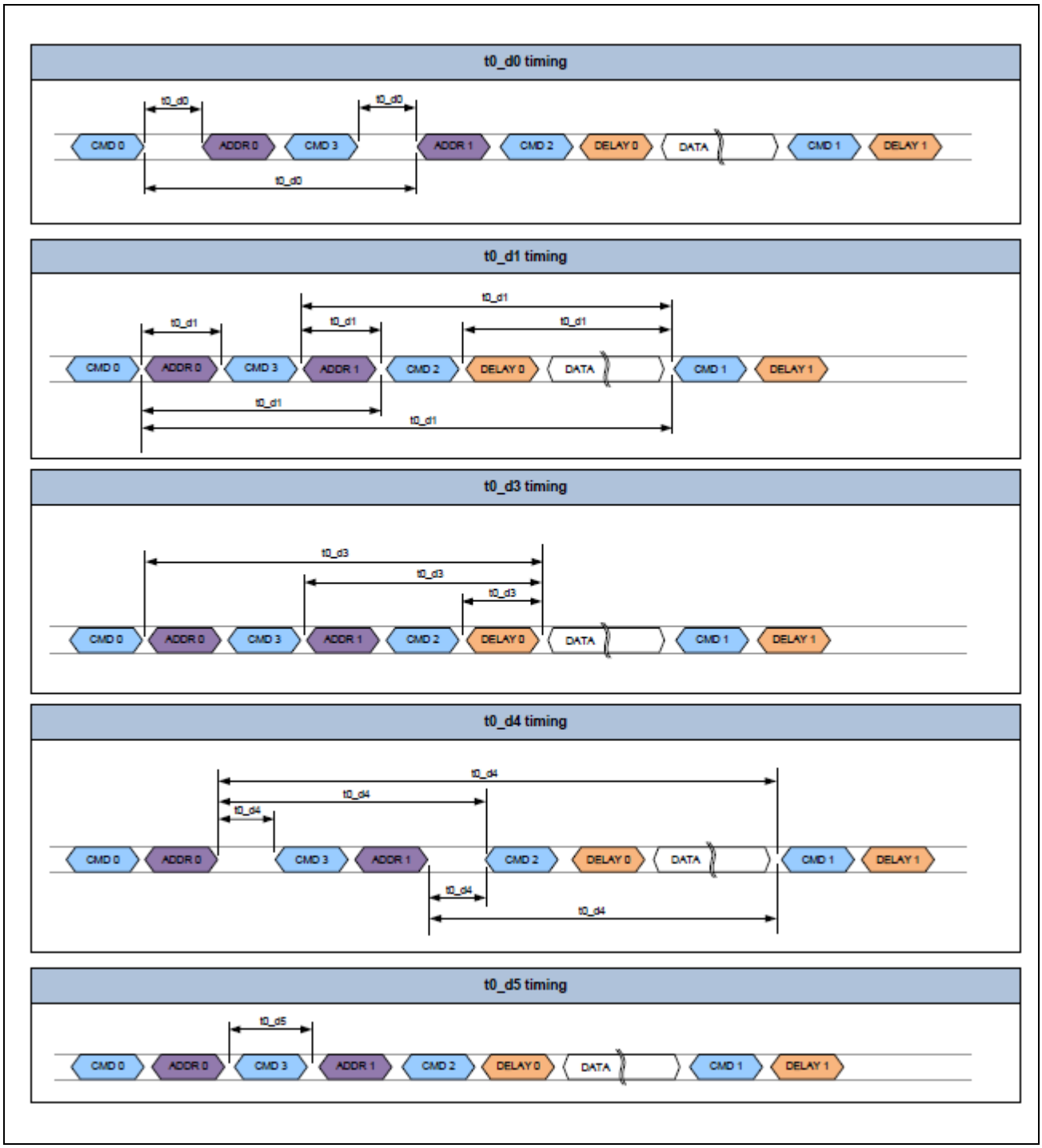


Figure 7.26 Generic Sequence Timing Parameters 1

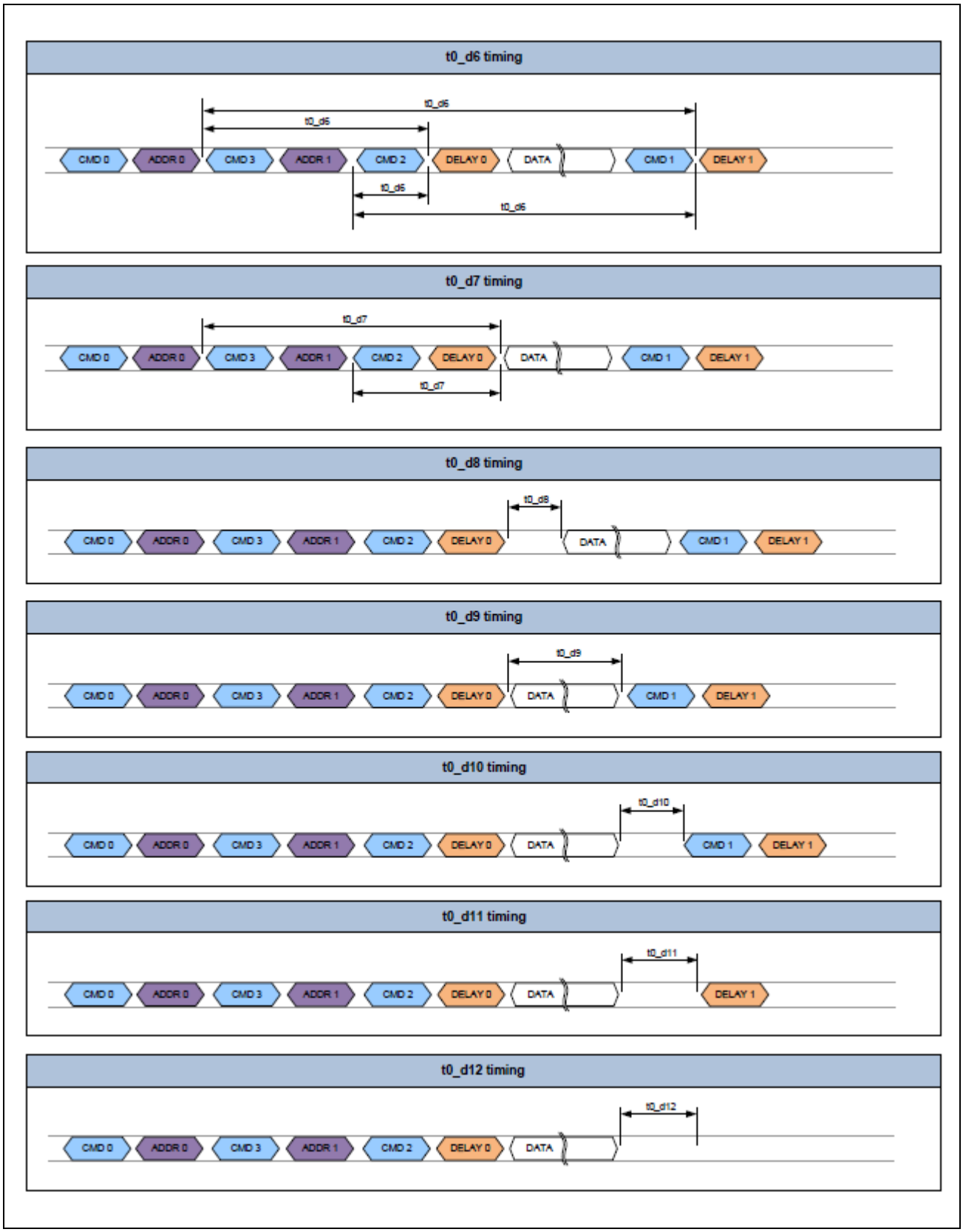


Figure 7.27 Generic Sequence Timing Parameters 2

7.5.4 Instructions

The implementation of the instruction set presented at this point is an example of how to use the instruction encoding scheme presented in **Section 7.5.2.1, Instruction Encoding**. The command sequences presented previously given sufficient information to implement new commands and command sequences for future NAND Flash devices.

7.5.4.1 Instruction Set

The table below contains the basic instruction defined to implement the all command sequences accessible in the ONFI 1.x and 2.2 standards.

Table 7.46 Instruction Set (1/2)

Instruction	CMD_0	CMD_1 / CMD_3	CMD_2	CMD_SEQ	Send when memory is busy
RESET	0xFF	—	—	SEQ_0	No
READ ID	0x90	—	—	SEQ_1	No
READ PARAMETER PAGE	0xEC	—	—	SEQ_2	No
READ UNIQUE ID	0xED	—	—	SEQ_2	No
GET FEATURES	0xEE	—	—	SEQ_2	No
SET FEATURES	0xEF	—	—	SEQ_3	No
READ STATUS	0x70	—	—	SEQ_4	Yes
SELECT LUN WITH STATUS	0x78	—	—	SEQ_5	Yes
LUN STATUS	0x71	—	—	SEQ_5	Yes
DEVICE STATUS	0x72	—	—	SEQ_4	Yes
VOLUME SELECT	0xE1	—	—	SEQ_21	Yes
CHANGE READ COLUMN	0x05	—	0xE0	SEQ_6	No
SELECT CACHE REGISTER	0x06	—	0xE0	SEQ_7	No
CHANGE WRITE COLUMN	0x85	—	—	SEQ_8	No
CHANGE ROW ADDRESS	0x85	0x11	—	SEQ_12	No
READ PAGE	0x00	—	0x30	SEQ_10	No
READ PAGE CACHE	0x31	—	—	SEQ_11	No
READ PAGE CACHE LAST	0x3F	—	—	SEQ_11	No
READ MULTIPLANE	0x00	0x32	—	SEQ_9	No
TWO PLANE PAGE READ	0x00	0x30	0x00	SEQ_15	No
QUEUE PAGE READ	0x07	—	0x37	SEQ_22	No
PROGRAM PAGE	0x80	0x10	—	SEQ_12	No
PROGRAM PAGE IMMEDIATE	0x80	0x10	—	SEQ_23	No
PROGRAM PAGE DELAYED	0x80	0x13	—	SEQ_23	No
PROGRAM PAGE 1	0x80	—	—	SEQ_13	No
PROGRAM PAGE CACHE	0x80	0x15	—	SEQ_12	No
PROGRAM MULTIPLANE	0x80	0x11	—	SEQ_12	No
WRITE PAGE	0x10	—	—	SEQ_0	No
WRITE PAGE CACHE	0x15	—	—	SEQ_0	No
WRITE MULTIPLANE	0x11	—	—	SEQ_0	No
ERASE BLOCK	0x60	0xD0	—	SEQ_14	No
ERASE MULTIPLANE	0x60	0xD1	—	SEQ_14	No
COPYBACK READ	0x00	—	0x35	SEQ_10	No
COPYBACK PROGRAM	0x85	0x10	—	SEQ_9	No
COPYBACK PROGRAM 1	0x85	—	—	SEQ_13	No

Table 7.46 Instruction Set (2/2)

Instruction	CMD_0	CMD_1 / CMD_3	CMD_2	CMD_SEQ	Send when memory is busy
COPYBACK MULTIPLANE	0x85	0x11	—	SEQ_12	No
PROGRAM OTP	0xA0	0x10	—	SEQ_12	No
DATA PROTECT OTP	0xA5	0x10	—	SEQ_9	No
READ PAGE OTP	0xAF	—	0x30	SEQ_10	No

7.5.4.2 Instruction Execution

The execution of each instruction has well-defined phases. The current controller configuration decides if a given phase is executed or not. The command execution cycle functions as follows:

The instruction code is written to the COMMAND register. This triggers the instruction execution.

The DCU unit decodes the instruction and configures the controller for its execution. At this point, the input module for the data FIFO is selected. The NCU module receives the sequence number to execute and the auxiliary parameter that parameterizes this operation.

The command sequence encoded in the instruction is executed in the NCU module. The details of this process depend on the controller configuration.

If the interrupts are enabled after command execution, the interrupt will occur.

The DATA_SEL and INPUT_SEL fields of the instruction code are configurable for every instruction used. When the instruction does not use the register or FIFO address, those fields are ignored. The ignored field has the logical zero value.

7.5.4.3 RESET Command

a. Command Description

The RESET command is used to put a target into a known condition and to abort command sequences in progress.

b. Command Encoding

The RESET instruction uses the SEQ_0 commands sequence. The command is encoded, as shown in the following table:

Table 7.47 RESET Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0xFF	—	—	SEQ_0_ID

7.5.4.4 READ ID Command

a. Command Description

The READ ID command is used to read identifier codes programmed into the target. This command is accepted by the target only when all LUNs on the target are in the IDLE state.

When the command is followed by an address cycle of 0x00, the target returns a 5-byte identifier code that includes the manufacturer ID, device configuration, and part-specific information.

When the READ ID command is followed by an address cycle of 0x20, the target returns the 4-byte ONFI identifier code.

b. Command Encoding

The READ ID instruction uses the SEQ_1 commands sequence. The command is encoded, as shown in the table below:

Table 7.48 READ ID Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x90	0/1	0/1	SEQ_1_ID

7.5.4.5 READ PARAMETER PAGE Command

a. Command Description

The READ PARAMETER PAGE command is used to read the ONFI parameter page programmed into the target. This command is accepted by the target only when all LUNs on the target are idle.

When the command is followed by an address cycle of 00h, the target goes into busy state. After the read process is completed, the controller enables the data output mode to read the parameter page.

b. Command Encoding

The READ PARAMETER PAGE instruction uses the SEQ_2 commands sequence. The command is encoded, as shown in the following table:

Table 7.49 READ PARAMETER PAGE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0xEC	0/1	0/1	SEQ_2_ID

7.5.4.6 READ UNIQUE ID Command

a. Command Description

The READ UNIQUE ID instruction is used to read a unique identifier programmed into the target. This command is accepted by the target only when all LUNs on the target are idle.

When the address cycle of 00h is written to the target, then the target goes into busy state. After the read process is complete, the controller enables the data output mode to read the unique ID. Sixteen copies of the unique ID data are stored in the device. Each copy is 32 bytes. The first 16 bytes are unique data and the second 16 bytes are the complement of the first 16 bytes. The host will XOR the first 16 bytes with the second 16 bytes. If the result is 16 bytes of 0xFF, then that copy of the unique ID data is correct. In the event that there is a non-0xFF result, the host repeats the XOR operation on a subsequent copy of the unique ID data.

b. Command Encoding

The READ UNIQUE ID instruction uses the SEQ_2 commands sequence. The command is encoded, as shown in the following table:

Table 7.50 READ UNIQUE ID Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0xED	0/1	0/1	SEQ_2_ID

7.5.4.7 GET FEATURES Command

a. Command Description

The GET FEATURES instruction reads the sub-feature parameters (P1-P4) from the specified feature address. This command is accepted by the target only when all LUNs on the target are idle.

When the 0xEE command is followed by a feature address, the target goes into busy state. After the target internal read operation completes, the controller enables the data output mode to read the sub-feature parameters.

b. Command Encoding

The GET FEATURES instruction uses the SEQ_2 commands sequence. The command is encoded, as shown in the following table:

Table 7.51 GET FEATURES Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0xEE	0/1	0/1	SEQ_2_ID

7.5.4.8 SET FEATURES Command

a. Command Description

The SET FEATURES instruction writes the sub-feature parameters (P1-P4) to the specified feature address to enable or disable target-specific features. This command is accepted by the target only when all LUNs on the target are idle.

The 0xEF command is followed by a valid feature address. The possible address value depends on the features set implemented in the target device. The address cycle is followed by the configurable number of data cycles. Values of the address and data encoding scheme allowed are found in the device vendor documentation.

b. Command Encoding

The SET FEATURES instruction uses the SEQ_3 commands sequence. The command is encoded, as shown in the following table:

Table 7.52 SET FEATURES Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0xEF	—	0/1	SEQ_3_ID

7.5.4.9 READ STATUS Command

a. Command Description

Each LUN provides its status independently of other LUNs on the same target through its 8-bit status register. Once the READ STATUS instruction is issued, status register output is enabled. The contents of the status register are returned on DQ[7:0] for each data output request.

The READ STATUS command returns the status of the most recently selected LUN.

b. Command Encoding

The READ STATUS instruction uses the SEQ_4 commands sequence. The command is encoded, as shown in the following table:

Table 7.53 READ STATUS Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x70	0/1	—	SEQ_4_ID

CAUTION

This instruction has special meaning because it can be executed when the target is in the BUSY state. The FIFO cannot be used to access the read data because it can be occupied by the operation under execution. For the sequence, the data field of command must select the DATA_REG register as data destination. The DATA_REG_SIZE must be select single byte.

7.5.4.10 DEVICE STATUS Command

a. Command Description

Each LUN provides its status independently of other LUNs on the same target through its 8-bit status register. Once the DEVICE STATUS instruction is issued, status register output is enabled. The contents of the status register are returned on DQ[7:0] for each data output request.

The DEVICE STATUS command returns the status of the most recently selected LUN.

b. Command Encoding

The DEVICE STATUS instruction uses the SEQ_4 commands sequence. The command is encoded, as shown in the following table:

Table 7.54 DEVICE STATUS Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x72	0/1	—	SEQ_4_ID

CAUTION

This instruction has special meaning because it can be executed when the target is in the BUSY state. The FIFO cannot be used to access the read data because it can be occupied by the operation under execution. For the sequence, the data field of command must select the DATA_REG register as data destination. The DATA_REG_SIZE must be select single byte.

7.5.4.11 VOLUME SELECT Command

a. Command Description

The VOLUME SELECT command is used to select a particular volume based on the address specified.

This command is accepted by all initialized devices that share a CE pin. The command may be executed with any volume on the target in any state. When the VOLUME SELECT command is issued, all volumes with unselected volume addresses will be deselected to save power.

If the Volume address entered is invalid or does not match any appointed volume address, all volume addresses will be deselected.

If the VOLUME SELECT command is not issued after CE high time then all volumes revert to their previous state.

b. Command Encoding

The VOLUME SELECT instruction uses the SEQ_21 commands sequence. The command is encoded, as shown in the following table:

Table 7.55 VOLUME SELECT Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0xE1	0/1	—	SEQ_21_ID

7.5.4.12 SELECT LUN WITH STATUS Command

a. Command Description

Each LUN provides its status independently of other LUNs on the same target through its 8-bit status register. Once the SELECT LUN WITH STATUS instruction is issued, status register output is enabled. The contents of the status register are returned on DQ[7:0] for each data output request.

The SELECT LUN WITH STATUS command returns the status of the selected LUN.

b. Command Encoding

The SELECT LUN WITH STATUS instruction uses the SEQ_5 commands sequence. The command is encoded, as shown in the following table:

Table 7.56 SELECT LUN WITH STATUS Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x78	0/1	—	SEQ_5_ID

CAUTION

This instruction has special meaning because it can be executed when the target is in the BUSY state. The FIFO cannot be used to access the read data because it can be occupied by the operation under execution. For the sequence, the data field of command must select the DATA_REG register as data destination. The DATA_REG_SIZE must be select single byte.

7.5.4.13 LUN STATUS Command

a. Command Description

Each LUN provides its status independently of other LUNs on the same target through its 8-bit status register. Once the LUN STATUS instruction is issued, status register output is enabled. The contents of the status register are returned on DQ[7:0] for each data output request. The LUN STATUS command returns the status of the selected LUN.

b. Command Encoding

The LUN STATUS instruction uses the SEQ_5 commands sequence. The command is encoded, as shown in the following table:

Table 7.57 LUN STATUS Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x71	0/1	—	SEQ_5_ID

CAUTION

This instruction has special meaning because it can be executed when the target is in the BUSY state. The FIFO cannot be used to access the read data because it can be occupied by the operation under execution. For the sequence, the data field of command must select the DATA_REG register as data destination. The DATA_REG_SIZE must be select single byte.

7.5.4.14 CHANGE READ COLUMN Command

a. Command Description

The CHANGE READ COLUMN command changes the column address of the selected cache register and enables data output of the last selected LUN. This command is accepted by the selected LUN when it is ready.

Writing 0x05 to the target command register, followed by two column address cycles containing the column address, followed by the 0xE0 command puts the selected LUN into data output mode. The selected LUN stays in data output mode until another valid command is issued.

b. Command Encoding

The CHANGE READ COLUMN instruction uses the SEQ_6 commands sequence. The command is encoded, as shown in the following table:

Table 7.58 CHANGE READ COLUMN Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
0xE0	—	0x05	0/1	—	SEQ_6_ID

7.5.4.15 SELECT CACHE REGISTER Command

a. Command Description

The SELECT CACHE REGISTER command enables data output on the addressed LUN and cache register at the specified column address. This command is accepted by a LUN when it is ready.

Writing the 0x06 to the target internal command register, followed by two column address cycles and three row address cycles, followed by 0xE0 enables data output mode on the address LUN and cache register at the specified column address.

b. Command Encoding

The SELECT CACHE REGISTER instruction uses the SEQ_7 commands sequence. The command is encoded, as shown in the following table:

Table 7.59 SELECT CACHE REGISTER Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
0xE0	—	0x06	0/1	—	SEQ_7_ID

7.5.4.16 CHANGE WRITE COLUMN Command

a. Command Description

The CHANGE WRITE COLUMN command changes the column address of the selected cache register and enables data input on the last selected LUN. Writing the 0x85 to the target internal command register, followed by two column address cycles containing the column address puts the selected LUN into data input mode.

b. Command Encoding

The CHANGE WRITE COLUMN instruction uses the SEQ_8 commands sequence. The command is encoded, as shown in the following table:

Table 7.60 CHANGE WRITE COLUMN Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x85	—	—	SEQ_8_ID

7.5.4.17 CHANGE ROW ADDRESS Command

a. Command Description

The CHANGE ROW ADDRESS command changes the row address (block and page) where the cache register contents are to be programmed in the NAND array. It also changes the column address of the selected cache register and enables data input on the specified LUN.

b. Command Encoding

The CHANGE ROW ADDRESS instruction uses the SEQ_12 commands sequence. The command is encoded, as shown in the following table:

Table 7.61 CHANGE ROW ADDRESS Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0x11	0x85	0	—	SEQ_12_ID

7.5.4.18 READ PAGE Command

a. Command Description

The READ PAGE command copies a page from the NAND Flash array to its respective cache register and enables data output. This command is accepted by the LUN when it is ready.

To read a page from the NAND Flash array, the controller writes the 0x00 command to the target internal command register, then writes 5 address cycles to the address registers, and concludes with the 0x30 command.

The selected LUN will go into busy state as the data is transferred. When the LUN is ready, data output is enabled for the cache register linked to the plane addressed in the READ PAGE command.

The controller reads the programmed number of bytes to the FIFO.

b. Command Encoding

The READ PAGE instruction uses the SEQ_10 commands sequence. The command is encoded, as shown in the following table:

Table 7.62 READ PAGE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
0x30	—	0x00	0/1	0/1	SEQ_10_ID

7.5.4.19 READ PAGE CACHE Command

a. Command Description

The READ PAGE CACHE command reads the next sequential page within a block into the DATA register, while the previous page is output from the cache register. To issue this command, the controller writes 0x31 to the target internal command register. After this command is issued, the RnB goes LOW and the LUN goes into busy state. Afterwards, the RnB goes HIGH and the LUN is busy with a cache operation, indicating that the cache register is available and that the specified page is copying from the NAND Flash array to the DATA register. At this point, data is read from the cache register.

b. Command Encoding

The READ PAGE CACHE instruction uses the SEQ_11 commands sequence. The command is encoded, as shown in the following table:

Table 7.63 READ PAGE CACHE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x31	0/1	0/1	SEQ_11_ID

7.5.4.20 READ PAGE CACHE LAST Command

a. Command Description

The READ PAGE CACHE LAST command ends the READ PAGE CACHE sequence and copies a page from the DATA register to the cache register. This command is accepted by the LUN when it is ready.

To issue the READ PAGE CACHE LAST command, the controller writes 0x3F to the target internal command register. After this command is issued, RnB goes LOW and the LUN goes into busy state. Afterwards, the RnB goes HIGH and the LUN is ready. At this point, data from the targets cache register is read into the FIFO.

b. Command Encoding

The READ PAGE CACHE LAST instruction uses the SEQ_11 commands sequence. The command is encoded, as shown in the following table:

Table 7.64 READ PAGE CACHE LAST Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x3F	0/1	0/1	SEQ_11_ID

7.5.4.21 READ MULTIPLANE Command

a. Command Description

The READ MULTIPLANE command queues a plane to transfer data from the NAND array to its cache register. This command can be issued one or more times. Each time a new plane address is specified, that plane is also queued for data transfer. To select the final plane and to begin the read operation for all previously queued planes, issue the READ PAGE command. All queued planes will transfer data from the NAND array to their cache registers.

b. Command Encoding

The READ MULTIPLANE instruction uses the SEQ_9 commands sequence. The command is encoded, as shown in the following table:

Table 7.65 READ MULTIPLANE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0x32	0x00	—	—	SEQ_9_ID

7.5.4.22 QUEUE PAGE READ Command

a. Command Description

The QUEUE PAGE READ operation allows for partial-page reads by using a 7-address cycle. The first two bytes of the address cycle indicate the length of the page to read - those bytes are stored in ADDR0 register, followed by the column and row addresses - those bytes are stored in the ADDR1 register. This can help overall performance should only a portion of the page data be needed because only the codeword containing the requested data will have ECC decoded.

b. Command Encoding

The QUEUE PAGE READ instruction uses the SEQ_22 commands sequence. The command is encoded, as shown in the following table:

Table 7.66 QUEUE PAGE READ Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
0x37	—	0x07	—	—	SEQ_22_ID

7.5.4.23 TWO PLANE PAGE READ Command

a. Command Description

This command was implemented to preserve the compatibility with the ONFI 1.x and some older devices.

The TWO PLANE PAGE READ (00h-00h-30h) operation is similar to the PAGE READ (00h-30h) operation. It transfers two pages of data from the NAND Flash array to the DATA registers. Each page must be from a different plane on the same die. The software is responsible for generating correct addresses for the requested pages. Both the ADDR0 and ADDR1 address registers are used in this case.

b. Command Encoding

The TWO PLANE PAGE READ instruction uses the SEQ_15 commands sequence. The command is encoded, as shown in the table below. In this case, both the address registers are used.

Table 7.67 TWO PLANE PAGE READ Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
0x00	0x30	0x00	0/1	0/1	SEQ_15_ID

7.5.4.24 PROGRAM PAGE Command

a. Command Description

The PROGRAM PAGE command allows the host to input data to a cache register and moves the data from the cache register to the specified block and page address in the array of the selected LUN. This command is accepted by the LUN when it is ready.

To input a page to the cache register and move it to the NAND array at the block and page address specified, the controller writes 0x80 to the target internal command register. Then five address cycles containing the column address and row address are written. Data input cycles follow. Serial data is input, beginning at the column address specified.

When data input is complete, the controller writes 0x10 to the target internal command register. The selected LUN goes into the busy state.

b. Command Encoding

The PROGRAM PAGE instruction uses the SEQ_12 commands sequence. The command is encoded, as shown in the following table:

Table 7.68 PROGRAM PAGE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0x10	0x80	0	0/1	SEQ_12_ID

7.5.4.25 PROGRAM PAGE IMMEDIATE Command

a. Command Description

The PROGRAM PAGE IMMEDIATE command allows the host to input data to a cache register and moves the data from the cache register to the specified block and page address in the array of the selected LUN. This command is accepted by the LUN when it is ready.

To input a page to the cache register and move it to the NAND array at the block and page address specified, the controller writes 0x80 to the target internal command register. Then three address cycles containing the row address are written. Data input cycles follow. Serial data is input, beginning at the column address specified.

When data input is complete, the controller writes 0x10 to the target internal command register. The selected LUN goes into the busy state. This command writes only the row address to the NAND Flash device.

b. Command Encoding

The PROGRAM PAGE IMMEDIATE instruction uses the SEQ_23 commands sequence. The command is encoded, as shown in the following table:

Table 7.69 PROGRAM PAGE IMMEDIATE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0x10	0x80	0	0/1	SEQ_23_ID

7.5.4.26 PROGRAM PAGE DELAYED Command

a. Command Description

The device internal controller can automatically manage multi-plane programming. It does this with PROGRAM PAGE DELAYED command.

When this command issued, the controller will delay issuing the program operation to the array until the address for the subsequent program operation is examined. If that operation allows the previous operation to complete as part of multi-plane operation, the controller will issue a multi-plane program to the array.

Multi-plane operations are only completed if an LUN address from plane 0 is issued prior to a LUN address from plane 1. If the subsequent program operation does not allow the multi-plane operation, the controller will immediately start the previous program. It is presumed that the host may use this command to initiate all the program operations. In this way the host does not have to maintain information regarding multi-plane operation usage.

b. Command Encoding

The PROGRAM PAGE DELAYED instruction uses the SEQ_23 commands sequence. The command is encoded, as shown in the following table:

Table 7.70 PROGRAM PAGE DELAYED Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0x13	0x80	0	0/1	SEQ_23_ID

7.5.4.27 PROGRAM PAGE 1 Command

a. Command Description

The PROGRAM PAGE 1 command allows the host to input data to a cache register and moves the data from the cache register to the specified block and page address in the array of the selected LUN. This command is accepted by the LUN when it is ready.

To input a page to the cache register and move it to the NAND array at the block and page address specified, the controller writes 0x80 to the target internal command register. Then five address cycles containing the column address and row address are written. Data input cycles follow. Serial data is input, beginning at the column address specified.

When data input is complete, the commands sequence ends

b. Command Encoding

The PROGRAM PAGE 1 instruction uses the SEQ_13 commands sequence. The command is encoded, as shown in the following table:

Table 7.71 PROGRAM PAGE 1 Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x80	0	0/1	SEQ_13_ID

7.5.4.28 PROGRAM PAGE CACHE Command

a. Command Description

The PROGRAM PAGE CACHE command allows the controller to input data to a cache register, copies the data from the cache register to the DATA register, and then moves the DATA register contents to the specified block and page address in the array of the selected LUN.

After the data is copied to the DATA register, the cache register is available for additional PROGRAM PAGE CACHE or PROGRAM PAGE commands. The PROGRAM PAGE CACHE command is accepted by the LUN when it is ready.

To input a page to the cache register to move it to the NAND array at the block and page address specified, the controller writes 0x80 to the target internal command register. Then five address cycles containing the column address and row address are written. Data input cycles follow. Serial data is input, beginning at the column address specified.

When data input is complete, 0x15 is written to the command register. The selected LUN goes into busy state to allow the DATA register to become available from a previous program cache operation, to copy data from the cache register to the DATA register, and then to begin moving the DATA register contents to the specified page and block address.

b. Command Encoding

The PROGRAM PAGE CACHE instruction uses the SEQ_12 commands sequence. Command is encoded as shown in the following table:

Table 7.72 PROGRAM PAGE CACHE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0x15	0x80	0	0/1	SEQ_12_ID

7.5.4.29 PROGRAM MULTIPLANE Command

a. Command Description

The PROGRAM MULTIPLANE command allows the controller to input data to the addressed plane cache register and queue the cache register to ultimately be moved to the NAND array. This command can be issued one or more times. Each time a new plane address is specified that plane is also queued for data transfer. This command is accepted by the LUN when it is ready.

The controller writes 0x80 to the target internal command register. Then five address cycles containing the column address and row address are written. Data input cycles follow. Serial data is input beginning at the column address specified.

When data input is complete, the controller writes 0x11 to the target internal command register.

b. Command Encoding

The PROGRAM MULTIPLANE instruction uses the SEQ_12 commands sequence. The command is encoded, as shown in the following table:

Table 7.73 PROGRAM MULTIPLANE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0x11	0x80	0	0/1	SEQ_12_ID

7.5.4.30 WRITE PAGE Command

a. Command Description

The WRITE PAGE command allows the controller to move data from the targets cache register to the NAND array. This command is accepted by the LUN when it is ready.

The controller writes 0x10 to the target internal command register.

b. Command Encoding

The WRITE PAGE instruction uses the SEQ_0 commands sequence. The command is encoded, as shown in the following table:

Table 7.74 WRITE PAGE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x10	—	—	SEQ_0_ID

7.5.4.31 WRITE PAGE CACHE Command

a. Command Description

The WRITE PAGE CACHE command allows the controller to move data from the targets cache register to the targets DATA register. This command is accepted by the LUN when it is ready.

The controller writes 0x15 to the target internal command register.

b. Command Encoding

The WRITE PAGE CACHE instruction uses the SEQ_0 commands sequence. The command is encoded, as shown in the following table:

Table 7.75 WRITE PAGE CACHE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x15	—	—	SEQ_0_ID

7.5.4.32 WRITE MULTIPLANE Command

a. Command Description

The WRITE MULTIPLANE command allows the controller to queue data from the targets cache register to the NAND array. This command is accepted by the LUN when it is ready.

The controller writes 0x11 to the target internal command register.

b. Command Encoding

The WRITE MULTIPLANE instruction uses the SEQ_0 commands sequence. The command is encoded, as shown in the following table:

Table 7.76 WRITE MULTIPLANE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x11	—	—	SEQ_0_ID

7.5.4.33 ERASE BLOCK Command

a. Command Description

The ERASE BLOCK command erases the specified block in the NAND array. This command is accepted by the LUN when it is ready.

To erase a block, the controller writes 0x60 to the target internal command register. Then three address cycles containing the row address are written; the column address is ignored. Finally, 0xD0 is written to the target internal command register.

b. Command Encoding

The ERASE BLOCK instruction uses the SEQ_14 commands sequence. The command is encoded, as shown in the following table:

Table 7.77 ERASE BLOCK Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0xD0	0x60	—	—	SEQ_14_ID

7.5.4.34 ERASE MULTIPLANE Command

a. Command Description

The ERASE MULTIPLANE command queues a block in the specified plane to be erased from the NAND array. This command can be issued one or more times. Each time a new plane address is specified, that plane is also queued for a block to be erased. This command is accepted by the LUN when it is ready.

To queue a block to be erased, the controller writes0x60 to the command register. Then three address cycles containing the row address are written; the column address is ignored. Finally, 0xD1 is written to the command register.

b. Command Encoding

The ERASE MULTIPLANE instruction uses the SEQ_14 commands sequence. The command is encoded, as shown in the following table:

Table 7.78 ERASE MULTIPLANE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0xD1	0x60	—	—	SEQ_14_ID

7.5.4.35 COPYBACK READ Command

a. Command Description

The COPYBACK READ command is functionally identical to the READ PAGE command, except that 0x35 is written to the target internal command register instead of 0x30.

For more detail, see the READ PAGE command description.

b. Command Encoding

The COPYBACK READ instruction uses the SEQ_10 commands sequence. The command is encoded, as shown in the following table:

Table 7.79 COPYBACK READ Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
0x35	—	0x00	0/1	0/1	SEQ_10_ID

7.5.4.36 COPYBACK PROGRAM Command

a. Command Description

The Copyback function reads a page of data from one location and then moves that data to a second location. The COPYBACK PROGRAM command is functionally identical to the PROGRAM PAGE command, except that when 0x85 is written to the target internal command register, the cache register contents are not cleared.

The SEQ_9 command sequence does not have the data phase, so the data from the cache register are written into the second location without modification. If data must be written with modification to the second location, the SEQ_12 command sequence, which includes the data phase, is used.

b. Command Encoding

The COPYBACK PROGRAM instruction uses the SEQ_9 commands sequence. The command is encoded, as shown in the following table:

Table 7.80 COPYBACK PROGRAM Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0x10	0x85	—	—	SEQ_9_ID

7.5.4.37 COPYBACK PROGRAM 1 Command

a. Command Description

The COPYBACK PROGRAM 1 command is functionally identical to the PROGRAM PAGE 1 command, except that when 0x85 is written to the target internal command register, the cache register contents are not cleared.

See **Section 7.5.4.27, PROGRAM PAGE 1 Command** for further details.

b. Command Encoding

The COPYBACK PROGRAM 1 instruction uses the SEQ_13 commands sequence. The command is encoded as shown in the following table:

Table 7.81 COPYBACK PROGRAM 1 Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	—	0x85	0	—	SEQ_13_ID

7.5.4.38 COPYBACK MULTIPLANE Command

a. Command Description

The COPYBACK MULTIPLANE command is functionally identical to the PROGRAM MULTIPLANE command, except that when 0x85 is written to the target internal command register, cache register contents are not cleared.

See **Section 7.5.4.29, PROGRAM MULTIPLANE Command** for further details.

b. Command Encoding

The COPYBACK MULTIPLANE instruction uses the SEQ_12 commands sequence. The command is encoded, as shown in the following table:

Table 7.82 COPYBACK MULTIPLANE Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0x11	0x85	0	0/1	SEQ_12_ID

7.5.4.39 PROGRAM OTP Command

a. Command Description

The PROGRAM OTP command is used to write data to the pages within the OTP area. An entire page can be programmed at one time, or a page can be partially programmed up to four times. There is no ERASE operation for the OTP pages.

To use the PROGRAM OTP command, the controller issues the 0xA0 command. Next, 5 address cycles are issued. The address write is followed by a programmable number of data cycles.

After data input is complete, the controller issues the 0x10 command. The internal control logic automatically executes the proper programming algorithm and controls the necessary timing for programming and verification.

b. Command Encoding

The PROGRAM OTP instruction uses the SEQ_12 commands sequence. The command is encoded as shown in the following table:

Table 7.83 PROGRAM OTP Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0x10	0xA0	0	0/1	SEQ_12_ID

7.5.4.40 DATA PROTECT OTP Command

a. Command Description

The DATA PROTECT OTP command is used to protect all the data in the OTP area. After the data is protected, it cannot be further programmed. When the OTP area is protected, the pages within the area are no longer programmable and cannot be unprotected.

To use the DATA PROTECT OTP command, the controller issues the 0xA5 command. Next, the controller issues the following 5 addresses cycles. Finally, the 0x10 command is issued.

b. Command Encoding

The DATA PROTECT OTP instruction uses the SEQ_9 commands sequence. The command is encoded, as shown in the following table:

Table 7.84 DATA_PROTECT OTP Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
—	0x10	0xA5	—	—	SEQ_9_ID

7.5.4.41 PAGE READ OTP Command

a. Command Description

The PAGE READ OTP command is used to read data from a page within the OTP area. An OTP page within the OTP area is available for reading data whether or not the area is protected.

To use the PAGE READ OTP command, the controller issues the 0xAF command. Next, 5 address cycles are issued. Finally, the 0x30 command is issued. After internal read from the NAND matrix is ended, the data is copied to the FIFO.

b. Command Encoding

The PAGE READ OTP instruction uses the SEQ_10 commands sequence. The command is encoded, as shown in the following table:

Table 7.85 PAGE READ OTP Instruction Encoding

CMD_2	CMD_1 / CMD_3	CMD_0	DATA_SEL	INPUT_SEL	CMD_SEQ
0x30	—	0xAF	0/1	0/1	SEQ_10_ID

7.5.5 Multi LUN Work Mode

The multi LUN work mode is enabled by setting the MLUN_EN bit in the CONTROL register. In this mode, each LUN is handled as separate target. The active LUN number is decoded directly from the address value.

This multi LUN mode is parameterized by the following parameters:

- The MLUN_IDX field in the MLUN register. This parameter provides the bits index for the last address byte where the LUN select bits is present. This parameter must be set according to the NAND Flash device datasheet in use.
- The MLUN_SEL bits in MLUN register is used to configure the number of LUN-s per device.
- The LUN_STATUS_0 holds the LUN-s statuses, each bit corresponds to the single LUN status.
- The STATE_MASK field in the STATUS_MASK register. This parameter is used to mask the part of the LUN status byte that will be ignored during the LUN ready/busy check.

7.5.6 Remapping Mechanism

The remapping mechanism implemented in the core is designed to support the bad blocks management solution in the core application. The hardware remapping mechanism relieves software from the time-consuming operations of finding the physical address for the given linear address in the requested operation. The software initializes only the remapping tables for uses in the application of the NAND Flash device. It sorts those tables in ascending order, and then the whole operation of searching tables and substituting addresses is accomplished automatically.

The remapping solution uses two groups of the control registers:

- The pointer registers DEV0_PTR - DEV3_PTR. These registers store the address in the system memory where record tables used by the BBM mechanism are placed. Each device in the bank uses a separate table. All tables are sorted in ascending order.
- The size registers DEV0_SIZE - DEV3_SIZE. These registers store the number of records in a table. Each device in the bank uses a separate register to store the table size.

The remapping module uses the special CAM memory implementation built around the 32-bit registers. This solution requires use of the record of the eight 32-bit width words as the smallest entity of data in the BBM solution. Each row in the record has two 16-bit width columns. The column that uses the lower part of the word stores the source address that will be replaced by the destination address in the remapping process. The column that uses the higher part of the word stores the destination address which replaces the source address in the remapping process. The unused rows in the record are filled with all ones values.

The following figure shows the record structure:

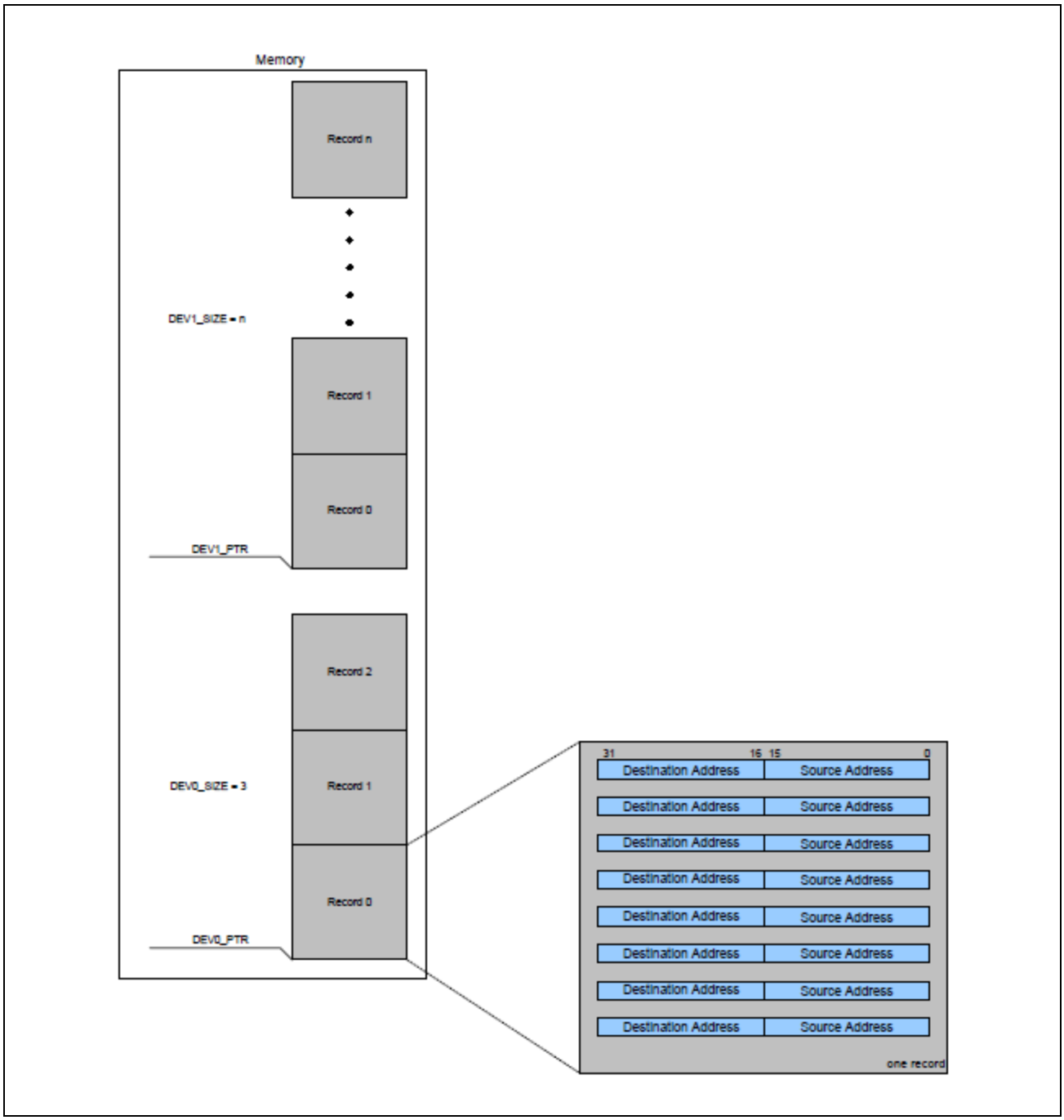


Figure 7.28 Example of BBM Records

The remapping process begins with the search. The source address fields are compared in parallel with the current request block address. If there is a source address equal to the request address, then the remapping module returns the destination address from the same row in the record. If there is no match, then the remapping module returns the request address itself.

7.5.7 Interrupts Mechanism

The NAND Flash Controller interrupt system uses two control registers:

- The interrupt mask register (INT_MASK) - each bit of this register masks a single interrupt. The register is described in more detail in **Section 7.4, Register Description**.
- The interrupt flag register (INT_STATUS) - each bit of this register is an active flag from the single interrupt source. The register is described in more detail in **Section 7.4, Register Description**.

Both registers are aligned in the same way. Corresponding bits of these registers are related to the same interrupt source. On the core, only the top level single interrupt line is present. This line is set when any of the bit pairs in the interrupts flag and mask registers are set.

The available interrupt sources are:

- Write/Erase protects mechanism interrupt.
  This interrupt occurs when the write/erase operation to the protected area is triggered. This interrupt is valid only if the hardware write/erase feature is implemented in the core and is enabled in the CONTROL register.
- Command sequence finished interrupt.
  This interrupt occurs when the previously triggered command sequence is finished and the new one can be started. The command sequence is marked as finished when the full sequence is executed or when the NAND Flash device goes into the busy state.
- The ECC module detects the uncorrectable error in the transmitted data.
- The ECC module detects when the configured errors threshold level is exceeded.
- The memory device is ready.
  This interrupt occurs when the NAND Flash device finishes executing the programmed command sequence and is ready for the new one. Each NAND Flash device has a single interrupt flag. The difference between “command sequence finished” interrupt and “memory device is ready” interrupt is presented in the following figure.
- The error on the slave interface during access to the controller FIFO.
  This interrupt occurs if the access to the FIFO memory has the opposite direction to the current FIFO configuration: the FIFO is read when it is configured for write, or the FIFO is write when it is configured for read.

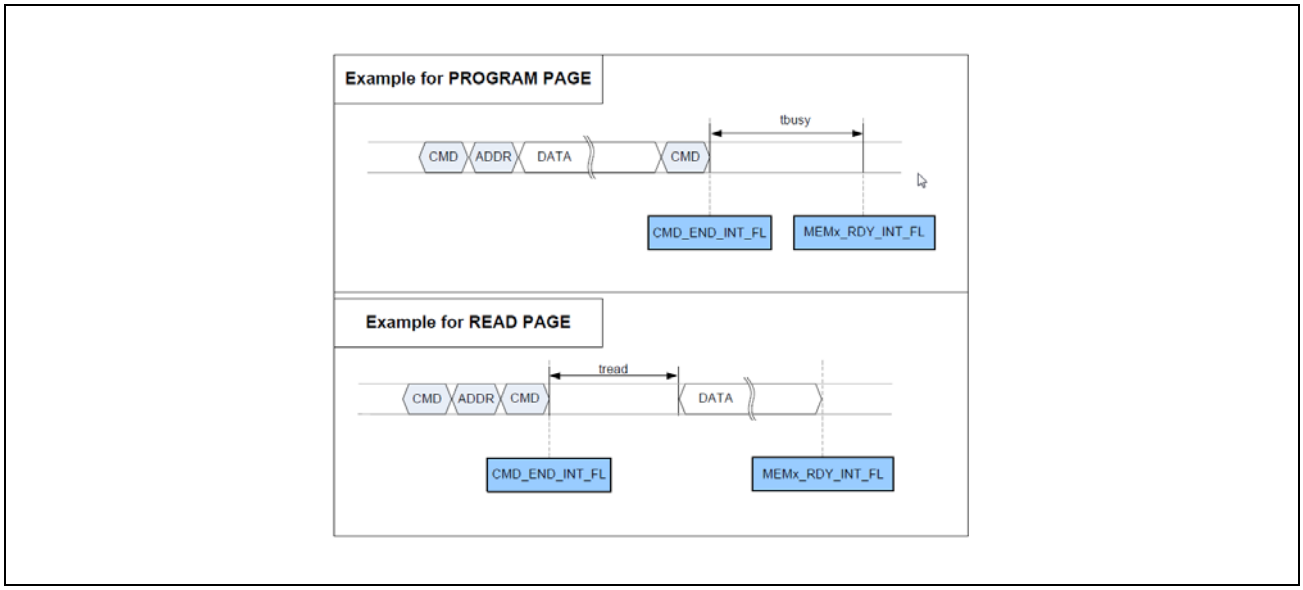


Figure 7.29 Command Sequence End and Memory-Ready Interrupts

7.6 Setup and Configuration

The CONTROL register is the main control register in the NAND Flash Controller.

The following bits configure basic settings of the controller:

- INT_EN – Bit which enables Interrupt.
- ECC_EN – Bit which enables Hardware ECC.
- BLOCK_SIZE – Bits which configure block size.
- IO_WIDTH – Bit which configures width of the I/O bus connected to the NAND Flash memory device.
- BBM_EN – Bit which enables remapping process.
- PROT_EN – Bit which enables Protect mechanism.
- ADDR[n]_AUTO_INCR – Addresses auto increment for row address register 0 or 1.
- SMALL_BLOCK_EN – Bit which enables Small Block Mode.
- MLUN_EN – Bit which enables Multi LUN mode.
- AUTO_READ_STAT_EN – Bit which activates automatic read status after the PROGRAM PAGE and BLOCK ERASE commands.
- READ_STATUS_EN – Bit which chooses whether the controller checks RnB lines or sends READ_STATUS commands.
- ECC_BLOCK_SIZE – Bits which define ECC Block Size.

The registers described below are configured according to the settings of other bits in the CONTROL register:

(1) Basic setting is done with the CONTROL register.

(2) If INT_EN bit is set, the software must write the mask into the INT_MASK register, which masks the selected interrupts source in the NAND Flash Controller.

(3) If ECC_EN bit is set, the software must correctly configure the ECC module by writing the appropriate configuration into the ECC_CTRL. Additionally, the software configures the offset in the ECC_OFFSET register. In small block mode, the value in the ECC_OFFSET is ignored and the correction words are located in the NAND Flash memory device, right behind the data.

(4) The write number of the data which will be transferred by the controller (DATA_SIZE register). When ECC is enabled, there are some restrictions to the DATA_SIZE value.

(5) If the BBM_EN bit is set, the software must initialize the remapping tables (DEV[n]_PTR and DEV[n]_SIZE registers).

(6) If the PROT_EN bit is set, the software can protect the area space which cannot be erased or overwritten. The PROTECT register defines the area that will be protected against any modifications.

(7) Additionally, the software must configure time parameters which can be found in the TIMINGS_ASYN register for asynchronous mode. Additionally, the software must configure the TIME_SEQ_0 and TIME_SEQ_1 registers.

When the NAND Flash Controller uses DMA to transfer data, the software must configure the DMA_ADDR, DMA_CTRL and DMA_CNT registers. The software can modify these registers before any transfer or during the initialization procedure.

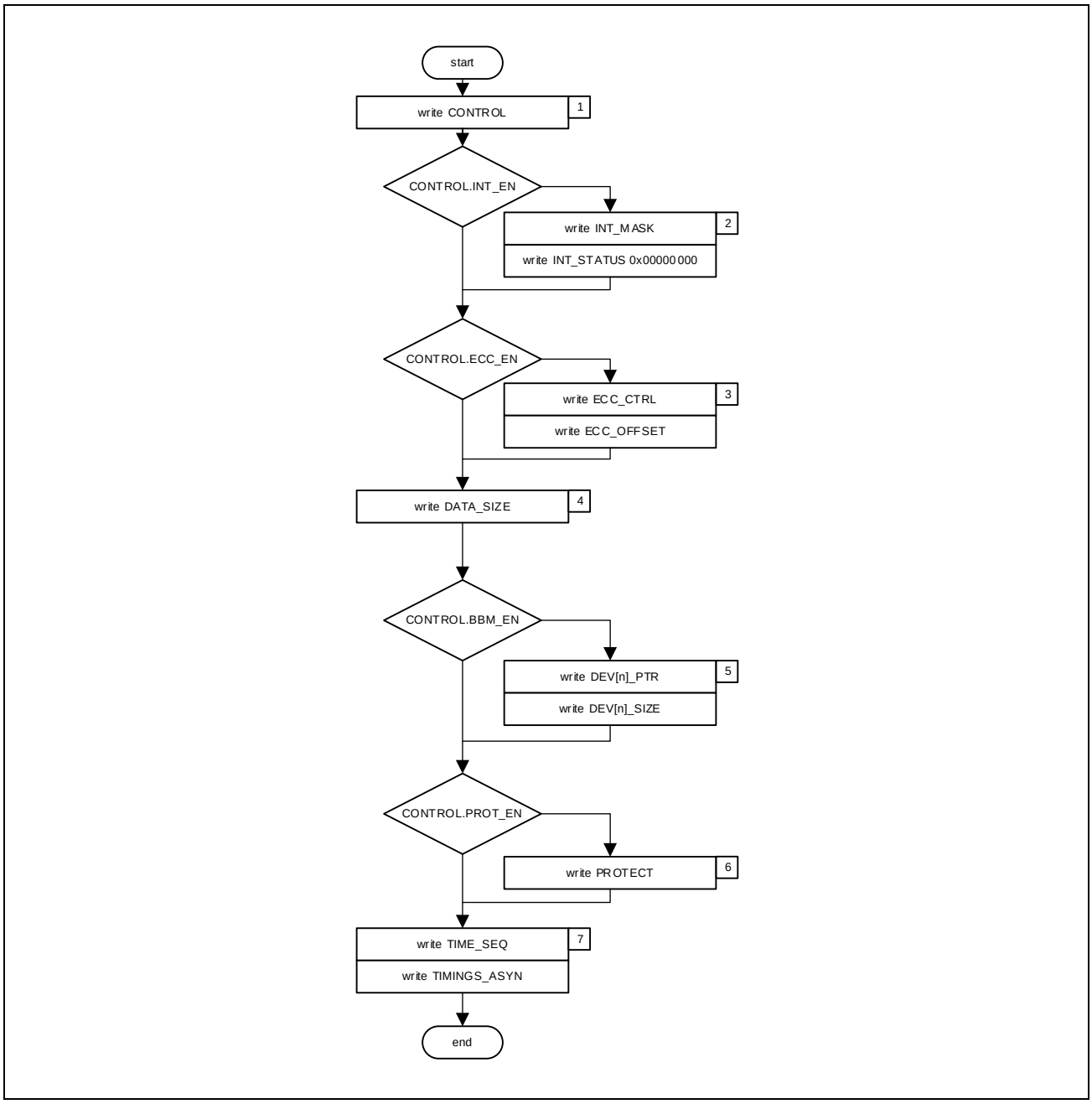


Figure 7.30 Configuration

7.6.1 Send Data to NAND Flash via Slave Interface

(1) The NAND Flash Controller must be correctly configured before sending data to the NAND Flash memory device. The setup process is described in detail in the previous **Section 7.6, Setup and Configuration** and in the **Section 7.4, Register Description**.

(2) Write the address of the data in NAND Flash memory into the address register 0 (ADDR0_COL and ADDR0_ROW registers). Write the number of data which you want to read (DATA_SIZE register). Choose the active memory device in the MEM_CTRL register (MEM_CE bits). Additionally, the software must check that the MEM[n]_WP bit which corresponds to the memory device (write protect must be disabled).

(3) To use the simplest program command, write 0x0010800C to the COMMAND register (PROGRAM PAGE command, FIFO module selected, AHBS module as input).

(4) Write data to the FIFO using the FIFO_DATA register. Data is sent to the NAND Flash memory device.

When the memory is ready for further work, the appropriate bit MEM[n]_ST is set. When the software needs to send the command to another memory, it is not necessary to wait for the MEM[n]_ST bit to be set.

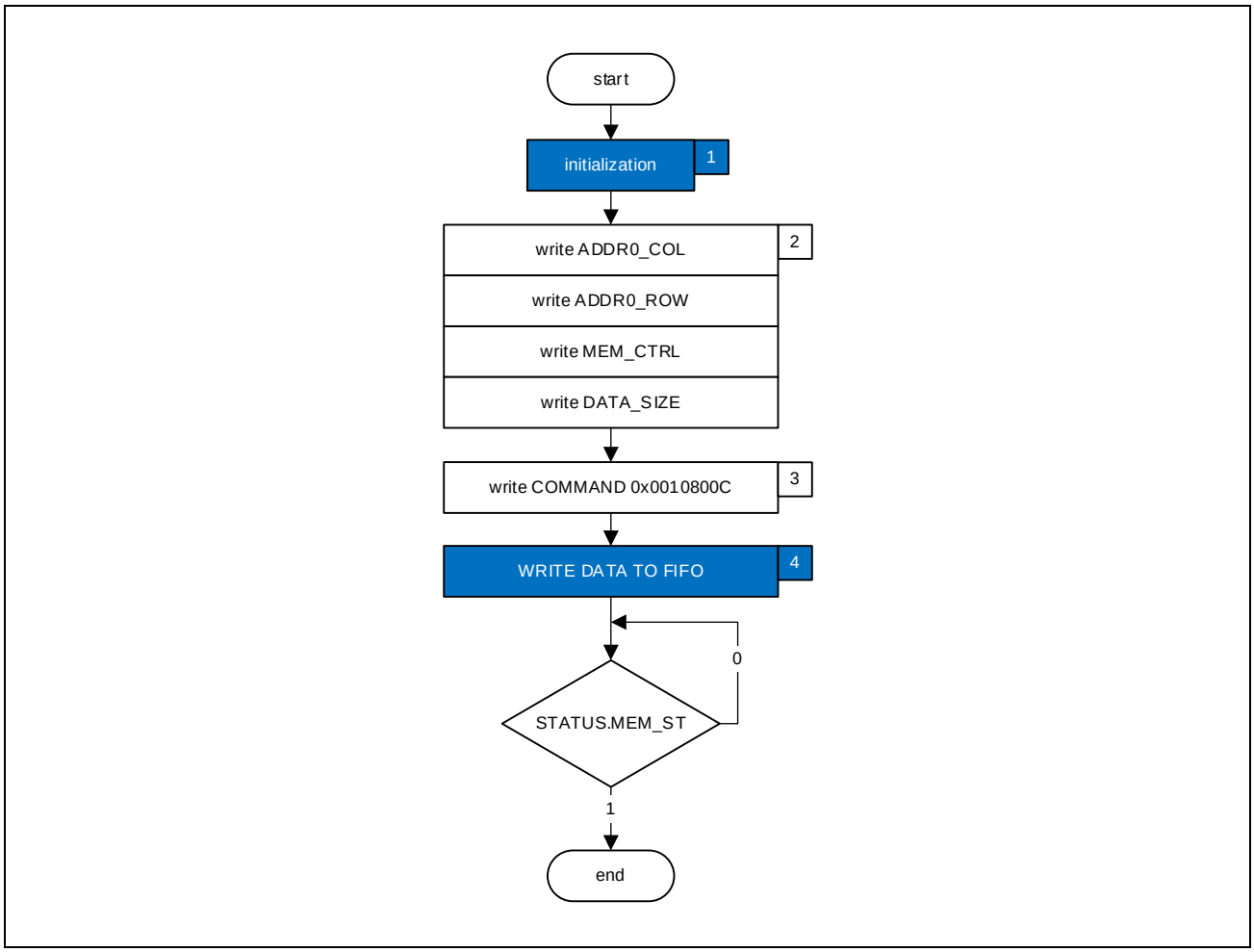


Figure 7.31 Write Data to NAND Flash Memory via Slave Interface

7.6.2 Read Data from NAND Flash via Slave Interface

(1) The NAND Flash Controller must be correctly configured before sending data to the NAND Flash memory device. The setup process is described in detail in the previous **Section 7.6, Setup and Configuration** and in the **Section 7.4, Register Description**.

(2) Write the address of the data in NAND Flash memory into the address register 0 (ADDR0_COL and ADDR0_ROW registers). Write the number of data which you want to read (DATA_SIZE register). Choose the active memory device in the MEM_CTRL register (MEM_CE bits).

(3) To use the simplest read command, write 0x3000002A to the COMMAND register (READ PAGE command, FIFO module selected, AHBS module as input).

(4) It is recommended to read FIFO_STATE register and wait when CF_EMPTY bit is set. After that wait for the DF_R_EMPTY bit in the FIFO_STATE register to be clear.

(5) Read data from the FIFO using the FIFO_DATA register.

When the memory is ready for further work, the appropriate bit MEM[n]_ST is set. When the software needs to send the command to another memory, it is not necessary to wait for the MEM[n]_ST bit to be set.

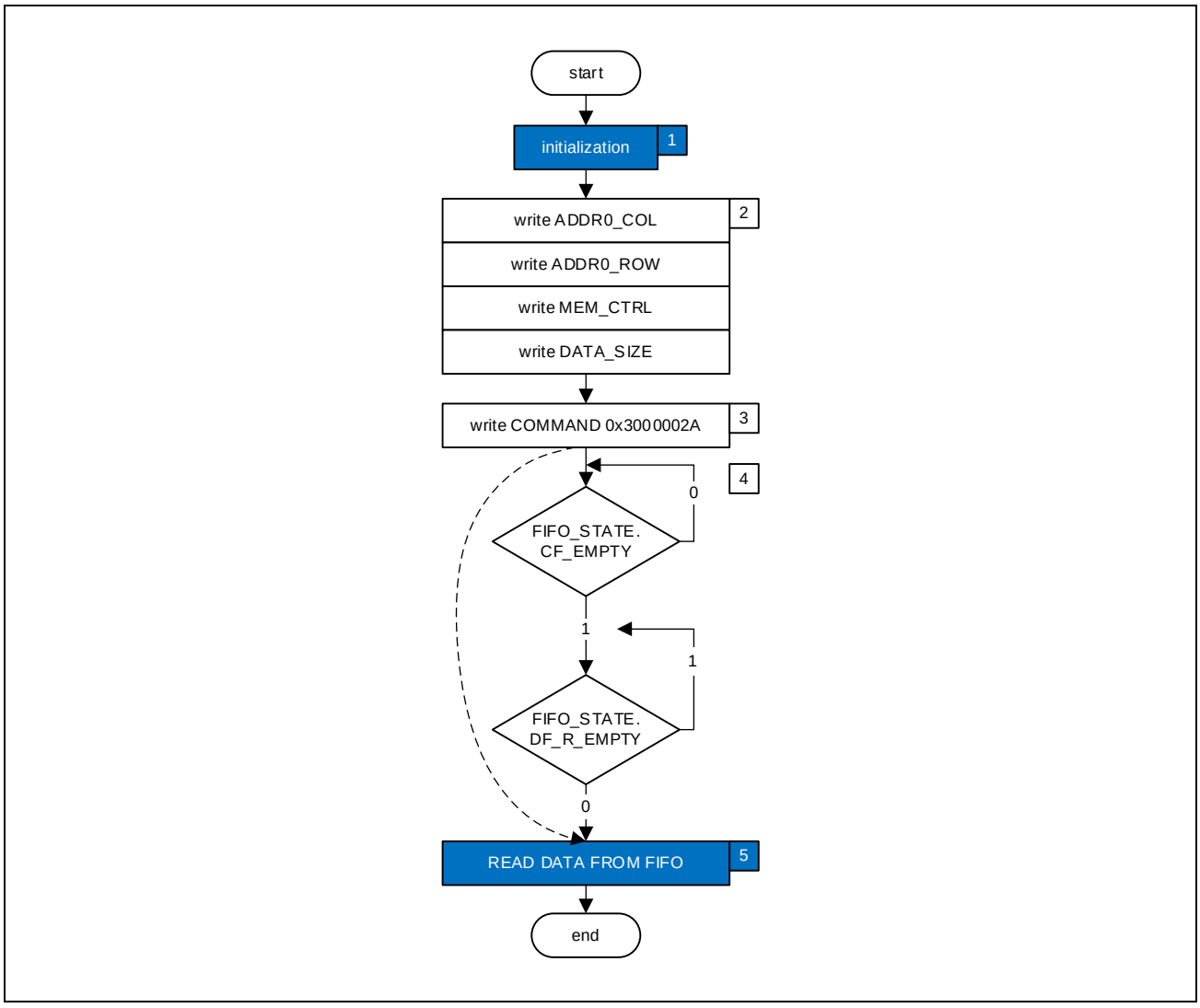


Figure 7.32 Read Data from NAND Flash Memory via Slave Interface

7.6.3 Send Data to NAND Flash via Master Interface (Using DMA)

(1) The NAND Flash Controller must be correctly configured before sending data to the NAND Flash memory device. The setup process is described in detail in the previous **Section 7.6, Setup and Configuration** and in the **Section 7.4, Register Description**.

(2) Set the INT_EN bit in the CONTROL register to enable interrupt.

(3) Select active interrupt in the INT_MASK register and write 0x00000000 into the INT_STATUS register to clear all interrupts.

(4) Select the DMA work mode to configure the DMA module correctly: In register's-managed mode, the address of the data is written in the system memory (DMA_ADDR register). Write the number of the transferred data into the DMA_CNT register. Set bit DMA_START to start DMA when the command sequence is sent to the NAND Flash memory. Bits ERR_FLAG and DMA_READY are read-only. The first indicates the error on the internal system bus while DMA is transferring data; the second indicates when DMA is ready (transfer is completed).

(5) Write the address of the data in the NAND Flash memory device into the address register 0. Choose the active memory device in the MEM_CTRL register (MEM_CE bits). Additionally, the software must confirm that the MEM[n]_WP bit which corresponds to the memory device (Write protect must be disabled).

(6) To use the simplest program command, write 0x0010804C to the COMMAND register (PROGRAM PAGE command, FIFO module selected, DMA module as input).

(7) When the memory is ready for further work, the appropriate bit MEM[n]_ST is set and interrupt is active.

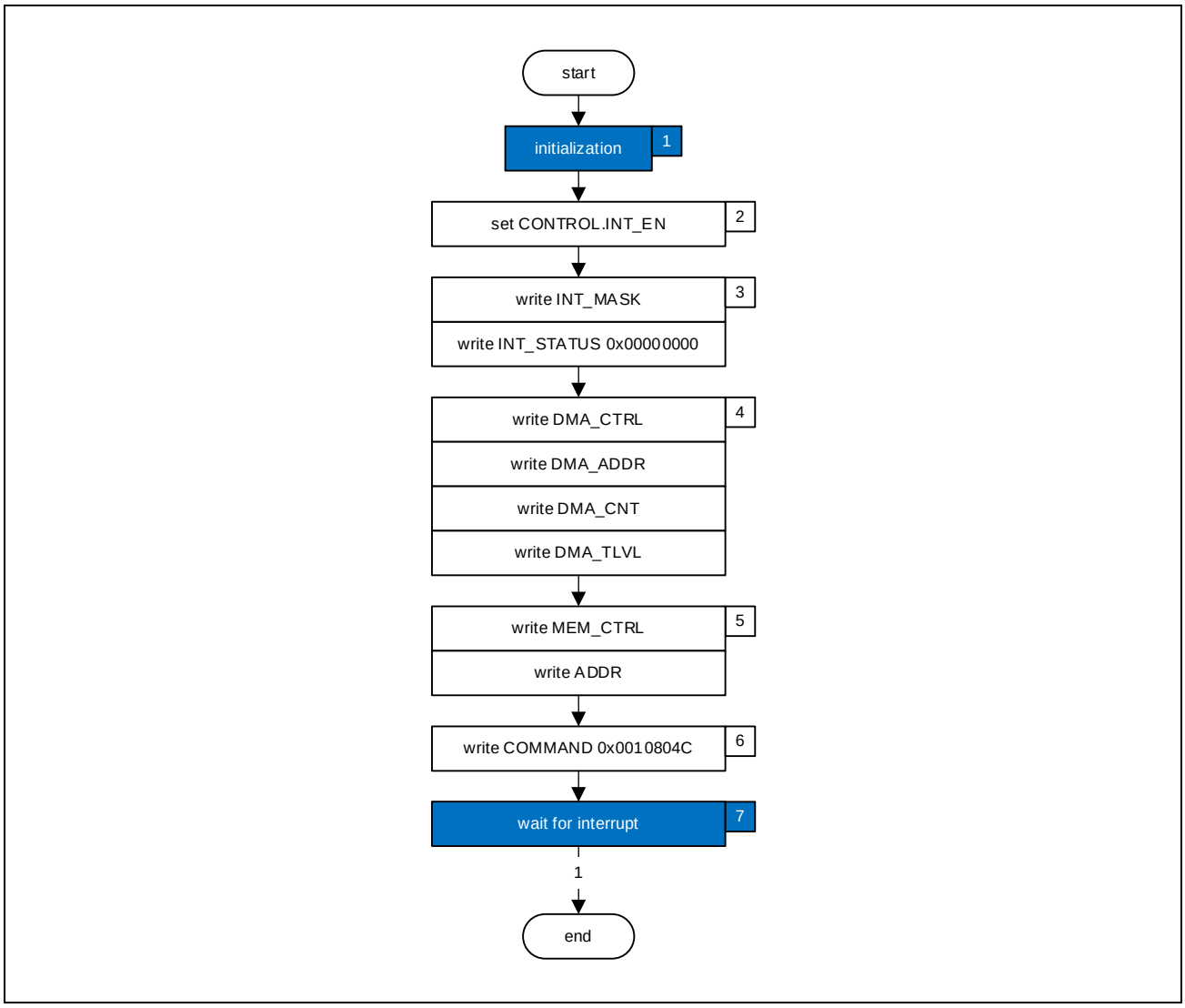


Figure 7.33 Send Data to NAND Flash Using DMA Interrupt Enable

7.6.4 Fast Writing and Reading of Several Pages from the Memory Using DMA

(1) The NAND Flash Controller must be correctly configured before sending data to the NAND Flash memory device. The setup process is described in detail in the previous **Section 7.6, Setup and Configuration** and in the **Section 7.4, Register Description**.

(2) Set INT_EN bit in CONTROL register to enable interrupt.

(3) Select the active interrupt (CMD_END_INT_EN) in the INT_MASK register and write 0x00000000 into the INT_STATUS register to clear all interrupts.

(4) In Scatter-Gather mode, it is necessary to write the descriptors into the system memory. Select the DMA work mode to correctly configure the DMA module. In Scatter-Gather mode, it is not necessary to configure the DMA_CNT register.

(5) Set DMA_START bit to start the DMA when the command sequence is sent to the NAND Flash memory. Bits ERR_FLAG and DMA_READY are read-only. The former indicates the error on the system bus while DMA is transferring data, the latter indicates when the DMA is ready (transfer is completed).

(6) Choose the active memory device in the MEM_CTRL register (MEM_CE bits). Additionally, the software must confirm if the MEM[n]_WP bit which corresponds to the memory device (Write protection must be disable). Set the ADDR0_AUTO_INCR bit to auto increment row address 0 register after each command. Write address of the data in the NAND Flash memory device into the address register 0.

(7) Write 0x00000000 into the INT_STATUS register to clear all interrupts.

(8) Write address of the first descriptor into the DMA_ADDR register.

(9) Write PROGRAM PAGE CACHE command to the NAND Flash memory device by writing 0x0015804C into the COMMAND register (DMA module as input, FIFO module selected). When the controller is ready for further work, the appropriate CMD_END_INT_FL bit is set and the interrupt is activated.

(10) When the number of the pages to transfer does not equal one, go to section 7.

(11) The last command in the sequence of sending data is the PROGRAM PAGE (write 0x0010804C) into the COMMAND register (DMA module as input, FIFO module selected).

(12) Write the address of the read data to the NAND Flash memory device into address register 0.

(13) Write the new descriptors to the system memory.

(14) If DMA should work in the same work mode and burst type do not modify DMA_BURST and DMA_MODE bits. Set DMA_START bit to start DMA when the command sequence is sent to the NAND Flash memory.

(15) Write the READ PAGE command into the NAND Flash memory device by writing 0x30000069 into the COMMAND register (FIFO module selected, DMA module as input).

(16) Write the address of the first descriptor to the DMA_ADDR register.

(17) Write 0x00000000 into the INT_STATUS register to clear all interrupts.

(18) Write READ PAGE CACHE command to the NAND Flash memory device by writing 0x0000316B into the COMMAND register (FIFO module selected, DMA module as input).

(19) When the controller is ready for further work, the appropriate CMD_END_INT_FL bit is set and the interrupt is activated. When the number of the pages to transfer does not equal zero, go to section 16.

(20) The last command in the sequence of reading data is READ PAGE CACHE LAST (write 0x00003F6B into the COMMAND register).

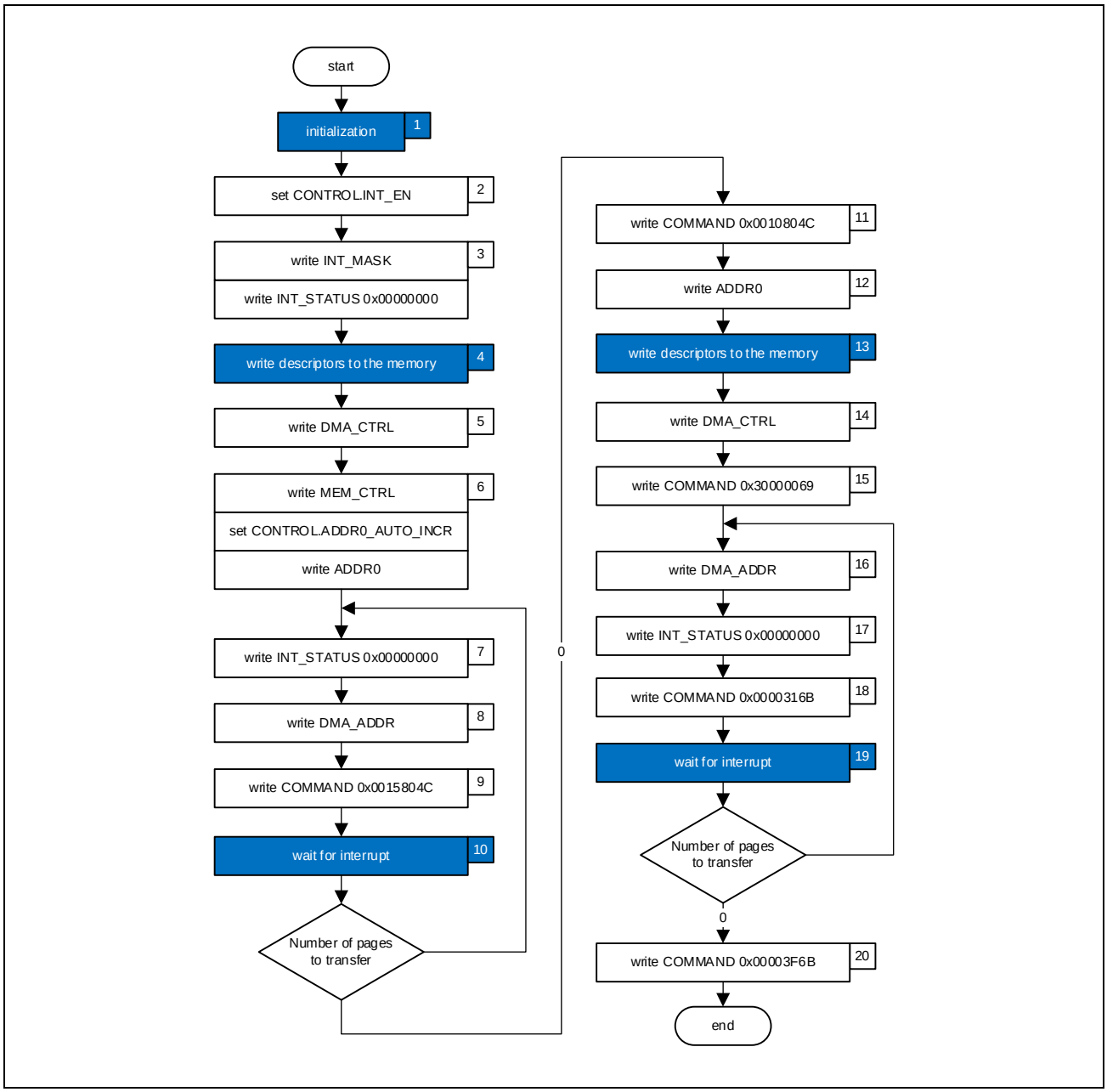


Figure 7.34 Fast Writing and Reading of Several Pages from the Memory Using DMA

7.6.5 Writing of Data into Two NAND Flash Memory Devices

Here is an example of how to write data into two NAND Flash memory devices:

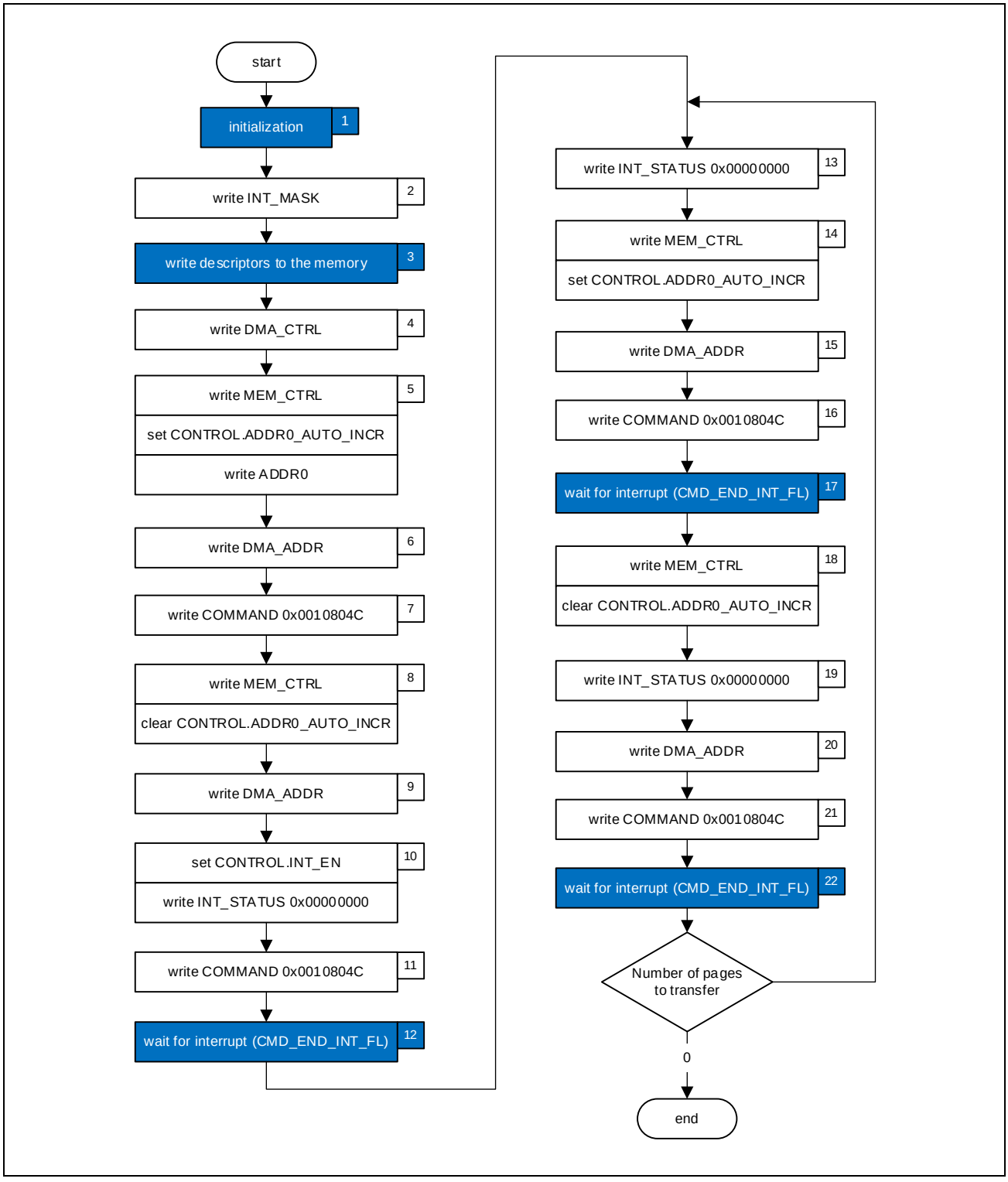


Figure 7.35 Writing Data into Two NAND Flash Memory Devices 1

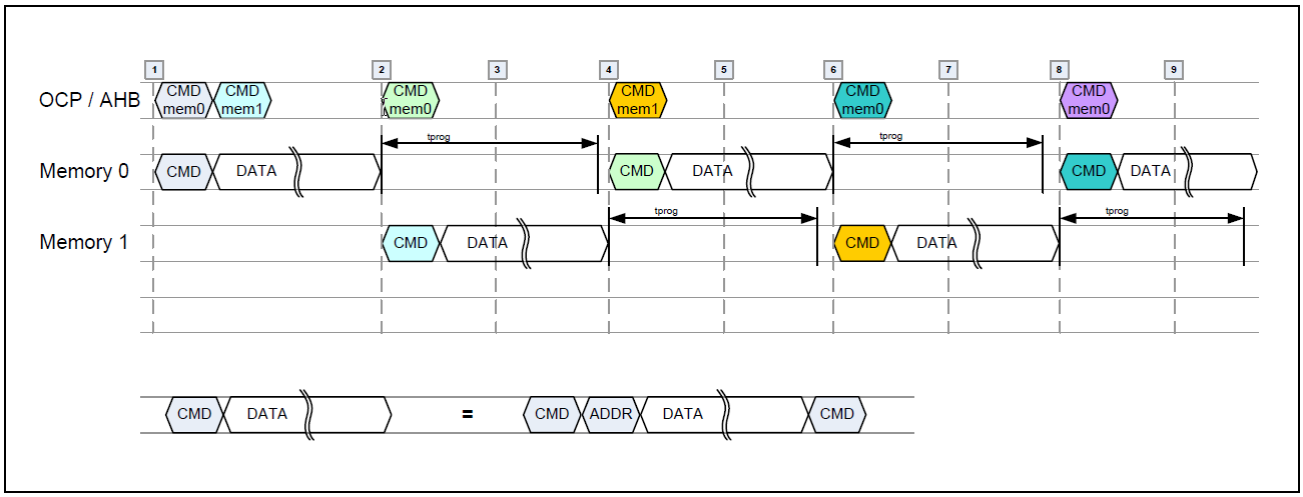


Figure 7.36 Writing Data into Two NAND Flash Memory Devices 2

7.6.6 Reading Data from Two NAND Flash Memory Devices

Here is an example of how to read data from two NAND Flash memory devices:

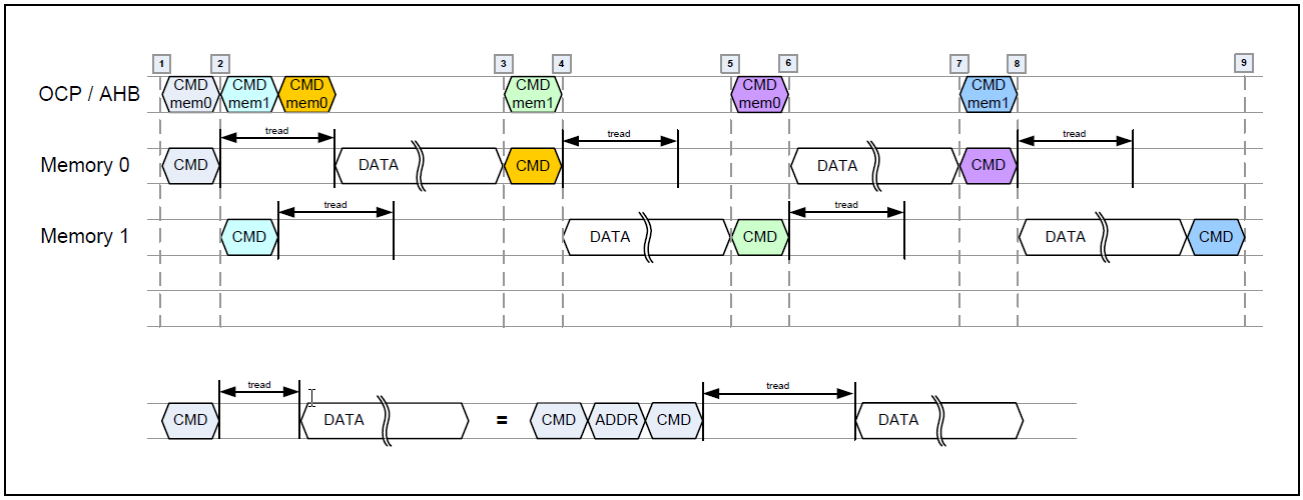


Figure 7.37 Reading Data from Two NAND Flash Memory Devices

7.6.7 Writing Data into Four NAND Flash Memory Devices

Here is an example of how to write data into four NAND Flash memory devices:

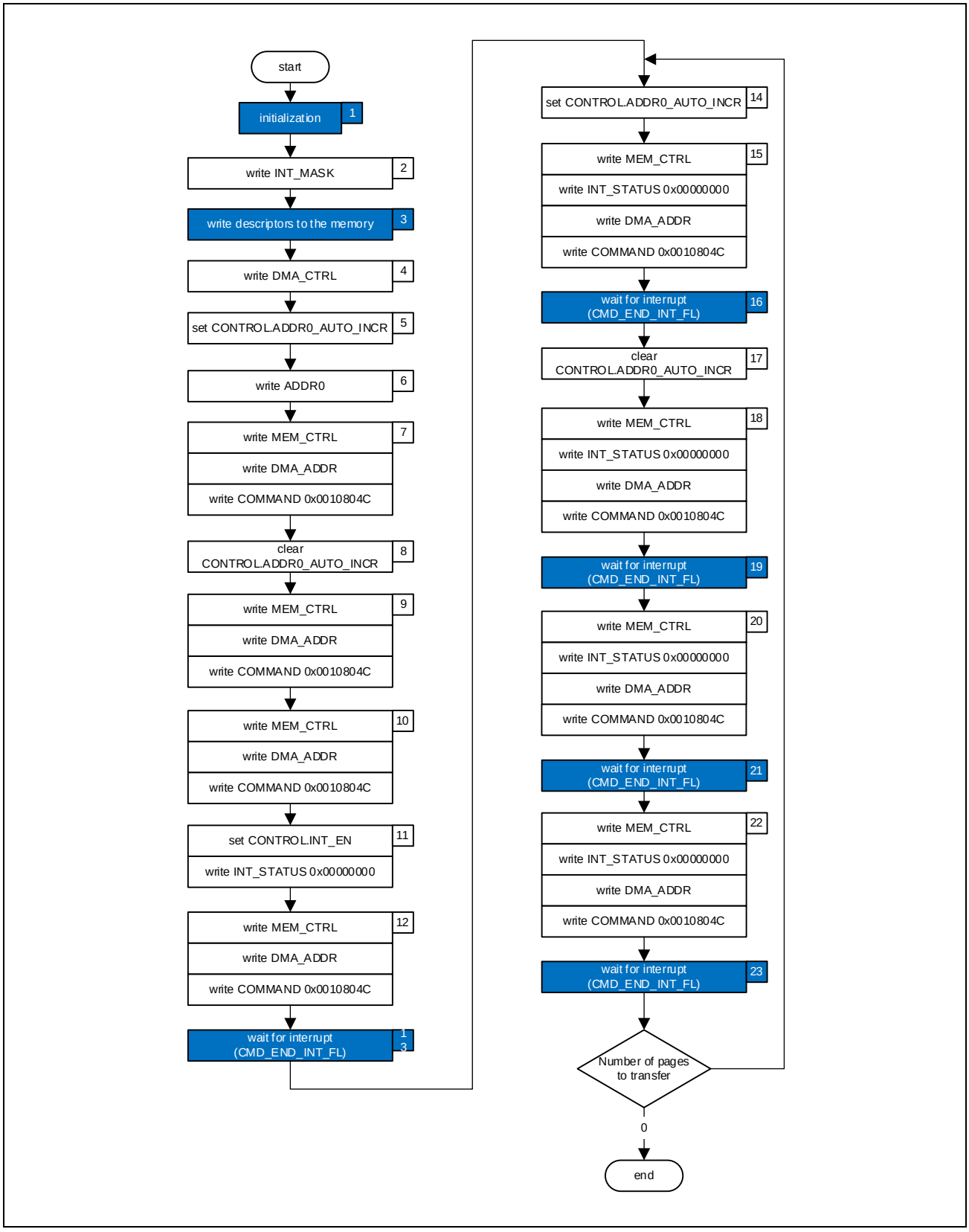


Figure 7.38 Writing Data into Four NAND Flash Memory Devices 1

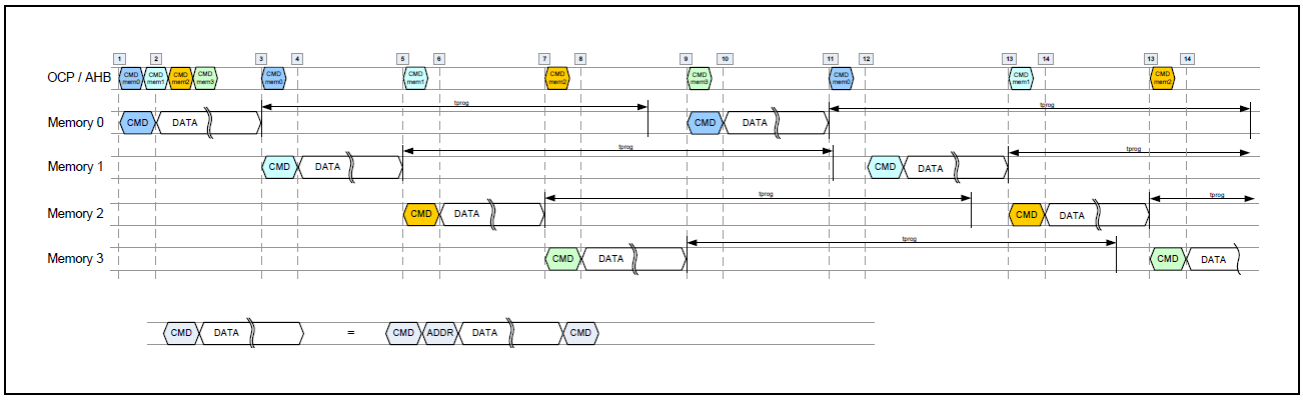


Figure 7.39 Writing Data into Four NAND Flash Memory Devices 2

7.6.8 Reading Data from Four NAND Flash Memory Devices

Here is an example of how to read data from four NAND Flash memory devices:

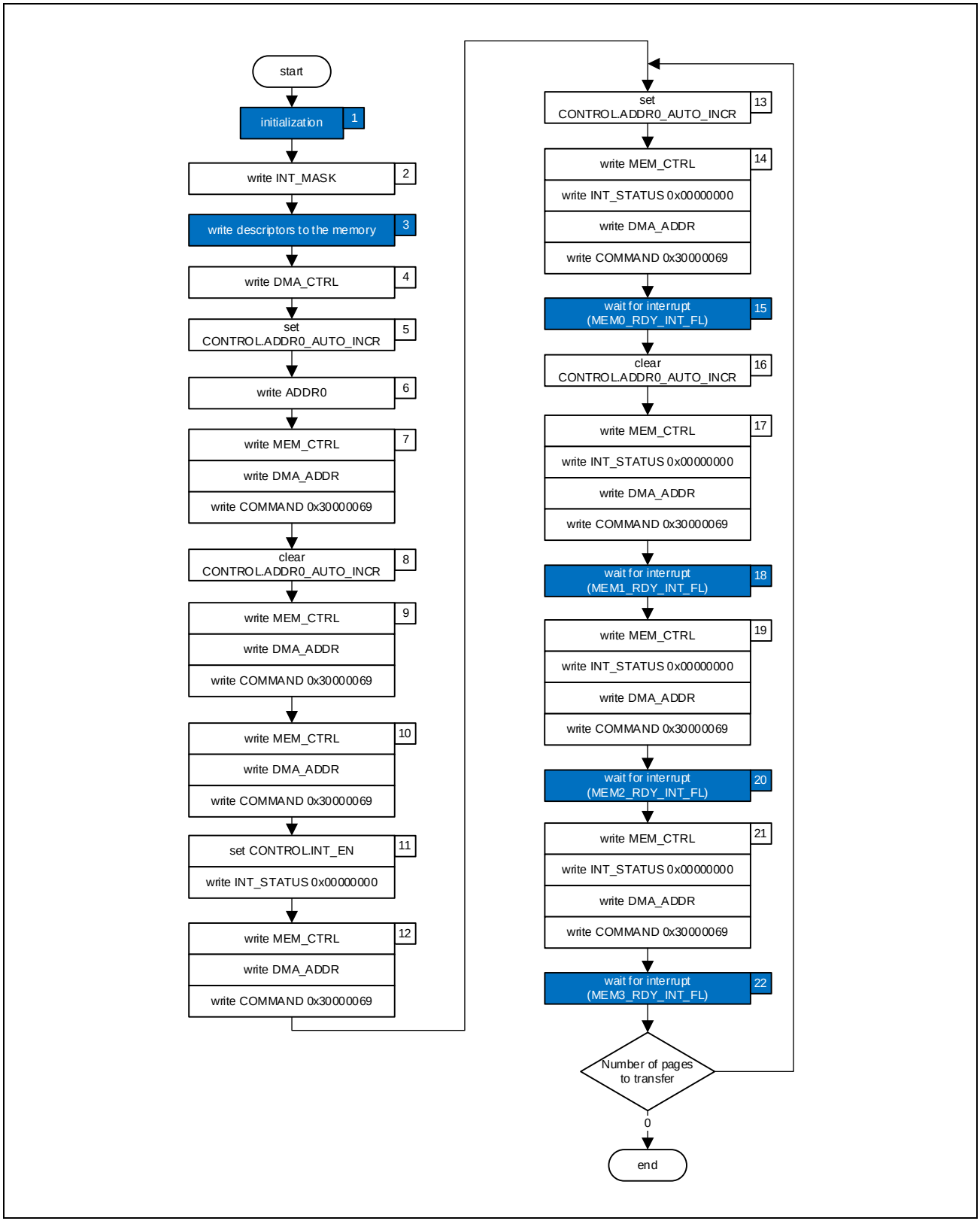


Figure 7.40 Reading Data from Four NAND Flash Memory Devices 1

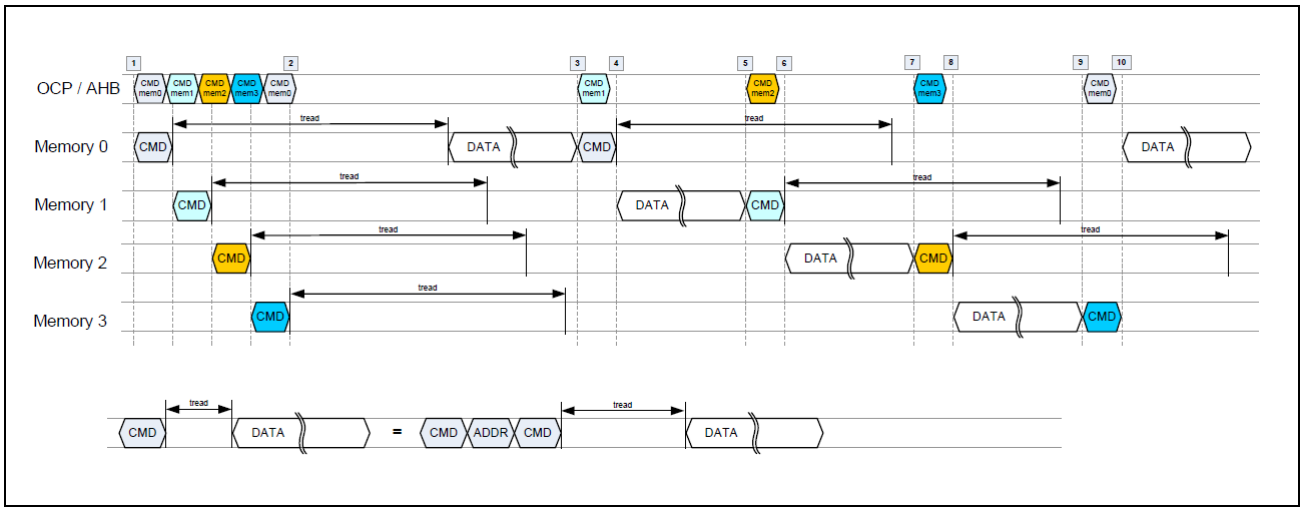


Figure 7.41 Reading Data from Four NAND Flash Memory Devices 2

7.6.9 Writing Partial Pages

The following procedure need to be used to write partial pages with ECC engine enabled:

(1) The NAND Flash Controller must be correctly configured before sending data to the NAND Flash memory device. The setup process is described in detail in the previous **Section 7.6, Setup and Configuration** and in the **Section 7.4, Register Description**.

(2) Write the address of the data in NAND Flash memory into the address register 0 (ADDR0_COL and ADDR0_ROW registers). You need to write partial page offset into the ADDR0_COL register. Write the number of data which you want to read (DATA_SIZE register) - in this case you need to set partial sector size. Choose the active memory device in the MEM_CTRL register (MEM_CE bits). Write the offset value of the ECC data into the ECC_OFFSET register. Additionally, the software must check that the MEM[n]_WP bit which corresponds to the memory device (write protect must be disabled).

(3) To use the simplest program command, write 0x0010800C to the COMMAND register (PROGRAM PAGE command, FIFO module selected, AHBS module as input).

(4) Write data to the FIFO using the FIFO_DATA register. Data is sent to the NAND Flash memory device.

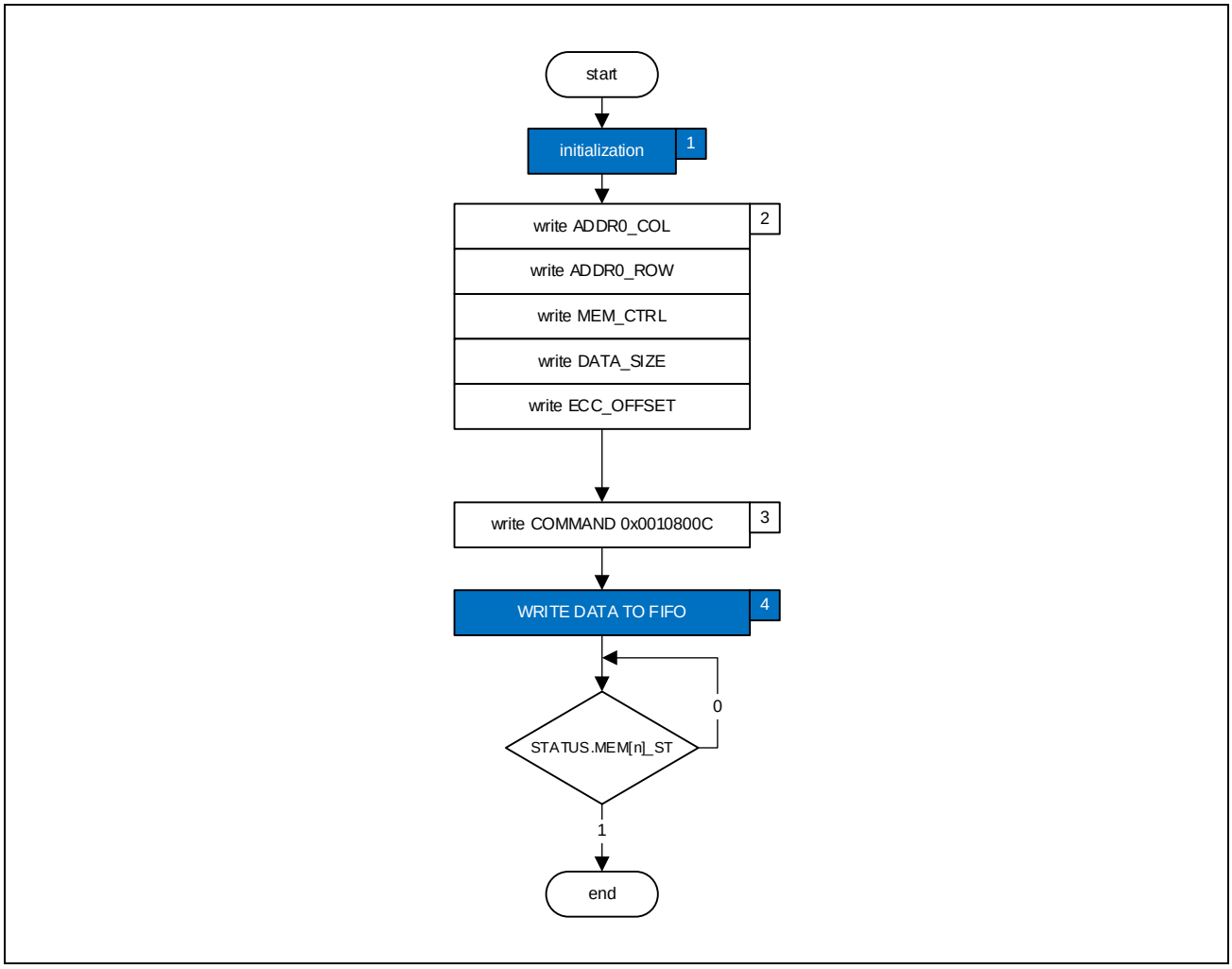


Figure 7.42 Writing Partial Pages

7.6.10 Reading Partial Pages

(1) The NAND Flash Controller must be correctly configured before sending data to the NAND Flash memory device. The setup process is described in detail in the previous **Section 7.6, Setup and Configuration** and in the **Section 7.4, Register Description**.

(2) Write the address of the data in NAND Flash memory into the address register 0 (ADDR0_COL and ADDR0_ROW registers). You need to write partial page offset into the ADDR0_COL register. Write the number of data which you want to read (DATA_SIZE register) - in this case you need to set partial page size. Choose the active memory device in the MEM_CTRL register (MEM_CE bits).

(3) To use the simplest read command, write 0x3000002A to the COMMAND register (READ PAGE command, FIFO module selected, AHBS module as input).

(4) It is recommended to read FIFO_STATE register and wait when CF_EMPTY bit is set. After that wait for the DF_R_EMPTY bit in the FIFO_STATE register to be clear.

(5) Read data from the FIFO using the FIFO_DATA register.

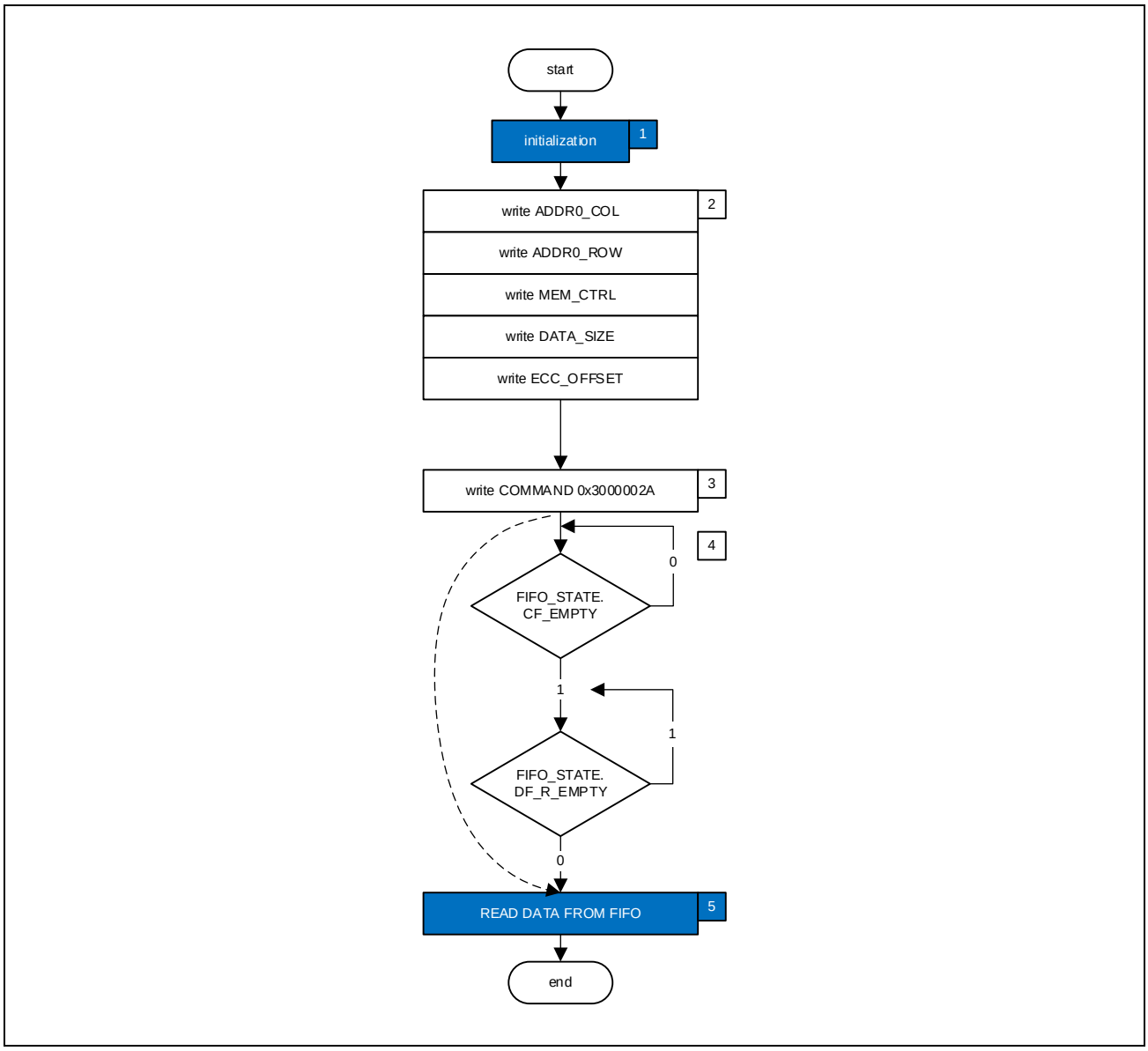


Figure 7.43 Reading Partial Pages

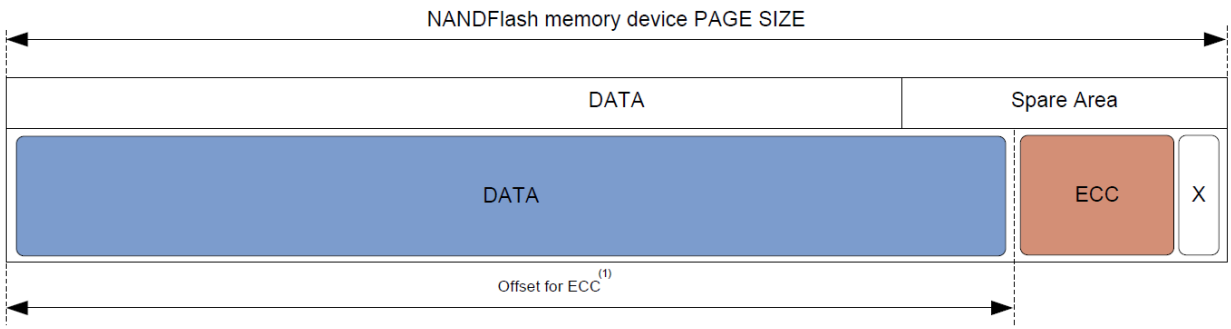
7.7 ECC Module

The ECC module is based on one of the BCH algorithms and allows correction of multiple bit errors. The ECC engine integrated into the NAND Flash Controller has the following properties:

- The encoder and decoder works on the 256, 512, 1024 bytes data blocks.
- Programmable correction capability: 2, 4, 8, 16, 24 or 32 errors.
- The corrected data words are aligned to the 32 bits.
- The correction words are aligned to the 32 bits.
- Correction words are placed after the data.
- The NAND Flash control unit writes to the target NAND Flash device only the valid correction word bytes.

7.7.1 ECC and Data Location within the Page

There is one method of organizing information within the page. Generally, the physical data organization within the page is transparent to the software. ECC and extended information is stored at the top of the page, above all subpages of user data – see figure below.



Note 1. The ECC_OFFSET register stores the offset value from the beginning of the page to the place where correction words are stored. In the small block mode, the value in the ECC_OFFSET is ignored and the correction words are located in the NAND Flash memory device, right behind the data.

Figure 7.44 Data in Data Area, ECC Stored in the Spare Area

7.7.2 BCH Algorithm Implementation

- Data block length: 256 (small block memory), 512 bytes or 1024 bytes.
- Programmable memory page length, multiple of the block length by any power of 2.
- Programmable correction capability: 2, 4, 8, 16, 24 or 32 errors.
- Separate encoder and decoder modules.
- Calculation of correction data performed during write to memory.
- Error detection performed during read data from memory.
- Internal pipeline allows the correction of errors in one data block simultaneously with detection of errors in the following data block.

Table 7.86 Size of Correction Bytes

Correction Capability	ECC Block Size	Size of Correction Bytes per one ECC Block (IO_WIDTH = 8)
2	256 / 512 / 1024 B	4
4	256 / 512 / 1024 B	7
8	256 / 512 / 1024 B	14
16	256 / 512 / 1024 B	28
24	256 / 512 / 1024 B	42
32	256 / 512 / 1024 B	56

7.8 Usage Notes

7.8.1 ADDR[n]_COL and ADDR[n]_ROW Registers

The address registers store the packaged version of the address that will be used by the next command sequence during access to the NAND Flash device.

Address	Register Symbol	Register Name
4010 2024h	ADDR0_COL	Column Address 0 Register
4010 2028h	ADDR0_ROW	Row Address 0 Register
4010 202Ch	ADDR1_COL	Column Address 1 Register
4010 2030h	ADDR1_ROW	Row Address 1 Register

ADDR[n]_COL – [15:0] – Column address. A15-A0 address bits.

ADDR[n]_ROW – [23:0] – Row address. A39-A16 address bits (Page address, Block address and LUN address in the ONFI case).

CAUTION

There is no register that defines the total memory size of the NAND Flash memory chip, thus the controller is not able to determine which address bits in ADDR[n]_COL and ADDR[n]_ROW registers are important and which must be zero. For this reason, the software must take special care with the values written to these registers. Incorrect values of unused address bits (none ‘0’ values) can cause errors in memory access.

A relation between address register’s and memory device address width is configured by the command sequence field of the COMMAND register. This field determines which command sequence has to be used and how many number of address bytes are used when addressing a NAND Flash memory device. (Example: In order to erase blocks, the three address cycles containing the row address are written into the NAND Flash memory device. The NAND Flash Controller automatically writes bits A39-A16 to the NAND Flash device). Refer to **Section 7.5.2.2, Command Sequence Encoding** in order to see how many address cycles are written into the NAND Flash memory device by each Command Sequence.

The address written to the address register must be aligned according to the NAND Flash device. Unused bits must be padded with zeros.

CAUTION

When the auto increment for the row address register is enabled, the proper value of this register can be read only when the bit CTRL_STAT in STATUS register is clear.

Table 7.87 Address Cycles

Address Cycle	FNAND_IO [0]	FNAND_IO [1]	FNAND_IO [2]	FNAND_IO [3]	FNAND_IO [4]	FNAND_IO [5]	FNAND_IO [6]	FNAND_IO [7]
1st cycle	A0	A1	A2	A3	A4	A5	A6	A7
2nd cycle	A8	A9	A10	A11	A12	A13	A14	A15
3rd cycle	A16	A17	A18	A19	A20	A21	A22	A23
4th cycle	A24	A25	A26	A27	A28	A29	A30	A31
5th cycle	A32	A33	A34	A35	A36	A37	A38	A39

7.8.2 Protect Register (PROTECT)

The NAND Flash Controller allows the defining of the area that will be protected against any modifications.

Address	Register Symbol	Register Name
4010 2034h	PROTECT	Protect Register

The protected area is a space which cannot be erased or overwritten. Any attempt to erase/overwrite this space always ends with an error. Since write and erase processes have constraints (only Page can be written and only Block can be erased), the protected area can be defined with Block-size precision.

The PROTECT register lower bits [15:0] define the beginning address of the protected area and are related to the NAND Flash memory block address bits of ADDR0_ROW and ADDR1_ROW register's.

The PROTECT register higher bits [31:16] define the ending address of the protected area and are related to the NAND Flash memory block address bits of ADDR0_ROW and ADDR1_ROW register's.

Figure below shows how the PROTECT register fields are used to define the protected area.

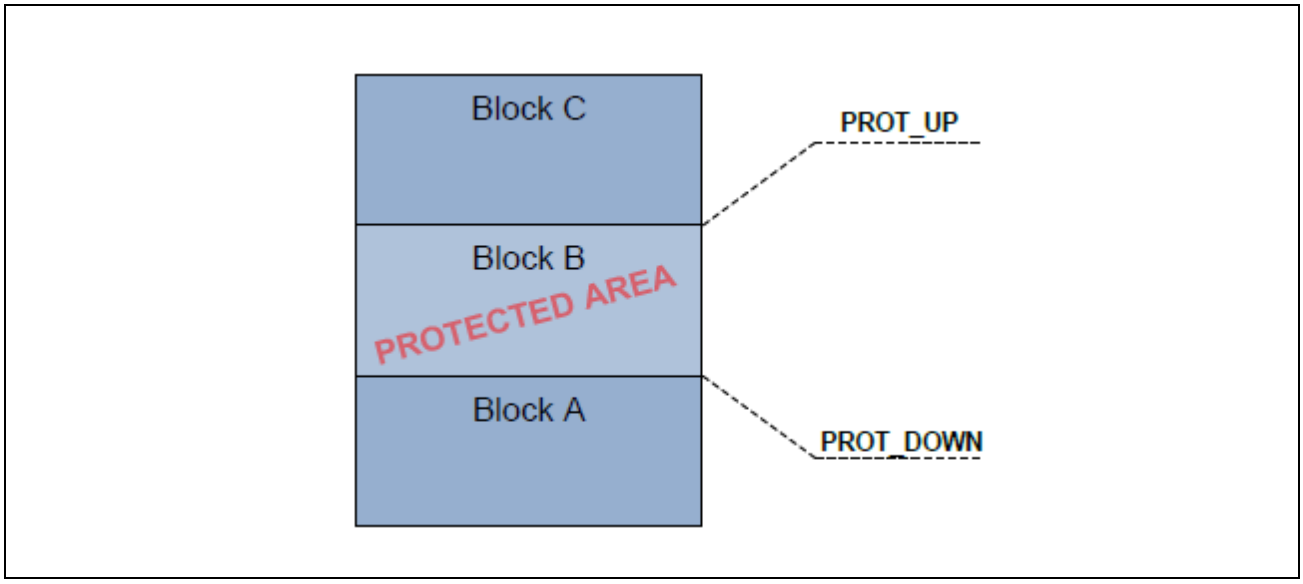


Figure 7.45 Protected Area

7.8.3 Asynchronous Mode Timings Register (TIMINGS_ASYN)

The NAND Flash Controller is intended for use with a wide range of host clock rates. To maximize flexibility, some timing parameters are configurable. Two waveform configuration parameters are defined in the TIMINGS_ASYN register.

Address	Register Symbol	Register Name
4010 2088h	TIMINGS_ASYN	Asynchronous Mode Timings Register

The value generated by the controller equals the minimum value written into the register, increased by 1. All the timings are generated using the NAND_ECLK clock signal.

The figure below shows how timings parameters are mapped to the NAND Flash interface. The upper part of figure shows read transfer, lower part shows write transfer.

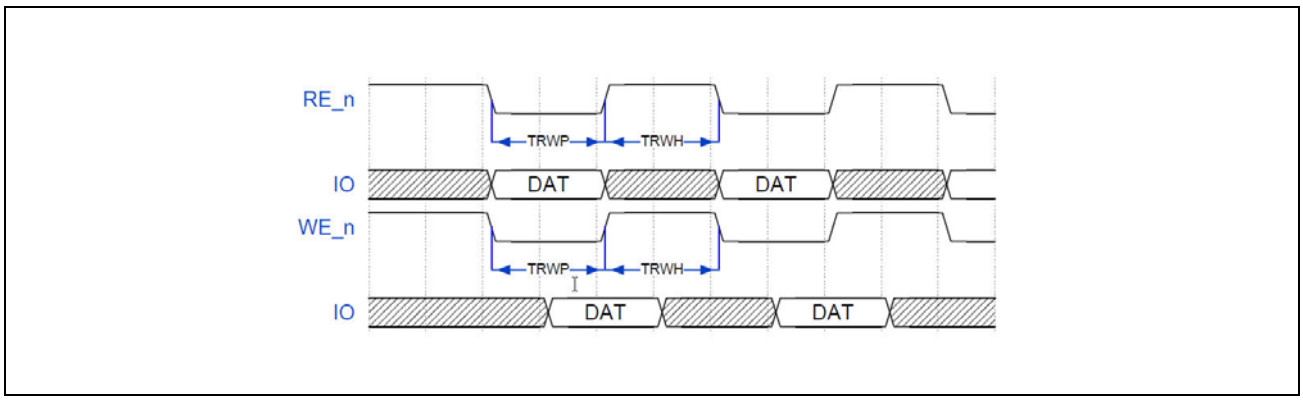


Figure 7.46 Timing Parameters (ASYNC)

7.8.4 Command Sequence Timing Register 1 (TIME_SEQ_1)

The NAND Flash Controller is intended for use with a wide range of host clock rates. To maximize flexibility, some timing parameters are configurable. Some waveform configuration parameters are defined in the TIME_SEQ_1 register.

Address	Register Symbol	Register Name
4010 2094h	TIME_SEQ_1	Command sequence timing register 1

The value generated by the controller equals the minimum value written into the register, increased by 1. All the timings are generated using the NAND_ECLK clock signal.

The figure below shows how the t_{WW} timing parameter is mapped to the NAND flash interface.

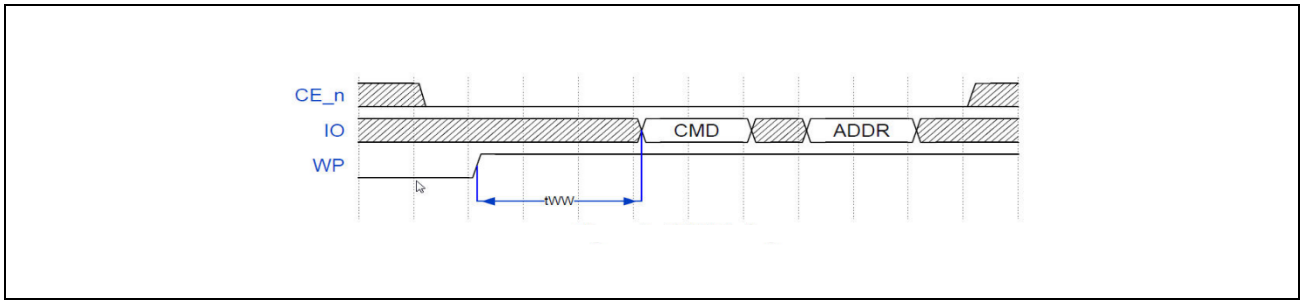


Figure 7.47 Timing Parameters (t_{WW})

Section 8 Quad IO SPI

8.1 Overview

- Up to 2 units
- Single, dual or quad I/O instructions supported
- Memory mapped ‘direct’ access to FLASH data supported
- Supported read performance enhanced mode (NoCMD mode) for most QSPI flash devices
- Remap address direct access
- Programmable device sizes
- Up to 4 chip selects
- Support for 1/2/3/4 byte addressing
- Support for programmable page size default 256 bytes
- Support for programmable number of bytes per device block
- Programmable write protected regions
- Transmit and receive FIFOs are 16 bytes
- Legacy mode allowing software direct access to low level transmit and receive FIFOs
- Set of control registers to perform any FLASH command
- Support for write burst in direct access
- Not support for Burst Read with wrap

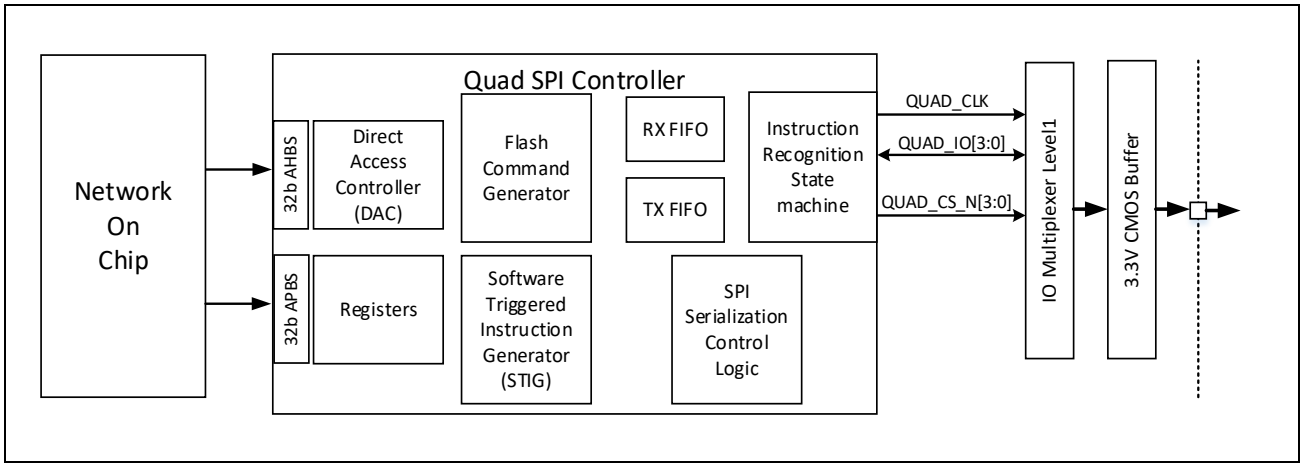


Figure 8.1 Quad SPI Controller Interfaces and Connections

8.2 Signal interfaces

Signal Name	Input Output	Description
Clock		
QSPI[m]_HCLK	Input	Internal bus clock (AHB)
QSPI[m]_PCLK	Input	Internal bus clock (APB)
QSPI[m]_REFCLK	Input	Reference clock for external signal
Interrupt		
QSPI[m]_Int	Output	Level sensitive interrupt, Active High
External Signal		
QUAD[m]_CLK	Output	Master clock output
QUAD[m]_IO[3:0]	Input/Output	Data input/output
QUAD[m]_CS_N[3:0]	Output	Chip select

Note: m = 1 or 2
Index removed style is used in this chapter. Ex) QUAD_CLK

CAUTION

QUAD[m]_IO[3]: Not support for HOLD signal to the flash device when not in quad I/O mode

8.3 Register Map

Table 8.1 Register Map of QSPI1

Address	Register Symbol	Register Name
4000 5000h	config_reg	QSPI Configuration Register
4000 5004h	dev_instr_rd_config_reg	Device Read Instruction Configuration Register
4000 5008h	dev_instr_wr_config_reg	Device Write Instruction Configuration Register
4000 500Ch	dev_delay_reg	QSPI Device Delay Register
4000 5010h	rd_data_capture_reg	Read Data Capture Register
4000 5014h	dev_size_config_reg	Device Size Configuration Register
4000 5024h	remap_addr_reg	Remap Address Register
4000 5028h	mode_bit_config_reg	Mode Bit Configuration Register
4000 5030h	tx_thresh_reg	TX Threshold Register
4000 5034h	rx_thresh_reg	RX Threshold Register
4000 5038h	write_completion_ctrl_reg	Write Completion Control Register
4000 503Ch	no_of_polls_bef_exp_reg	Polling Expiration Register
4000 5040h	irq_status_reg	Interrupt Status Register
4000 5044h	irq_mask_reg	Interrupt Mask Register
4000 5050h	lower_wr_prot_reg	Lower Write Protection Register
4000 5054h	upper_wr_prot_reg	Upper Write Protection Register
4000 5058h	wr_prot_ctrl_reg	Write Protection Control Register
4000 5090h	flash_cmd_ctrl_reg	Flash Command Control Register
4000 5094h	flash_cmd_addr_reg	Flash Command Address Register
4000 50A0h	flash_rd_data_lower_reg	Flash Command Read Data Register (Lower)
4000 50A4h	flash_rd_data_upper_reg	Flash Command Read Data Register (Upper)
4000 50A8h	flash_wr_data_lower_reg	Flash Command Write Data Register (Lower)
4000 50ACh	flash_wr_data_upper_reg	Flash Command Write Data Register (Upper)
4000 50B0h	polling_flash_status_reg	Polling Flash Status Register
4000 50FCh	module_id_reg	Module ID Register

Table 8.2 Register Map of QSPI2 (RZ/N1S only)

Address	Register Symbol	Register Name
4000 E000h	config_reg	QSPI Configuration Register
4000 E004h	dev_instr_rd_config_reg	Device Read Instruction Configuration Register
4000 E008h	dev_instr_wr_config_reg	Device Write Instruction Configuration Register
4000 E00Ch	dev_delay_reg	QSPI Device Delay Register
4000 E010h	rd_data_capture_reg	Read Data Capture Register
4000 E014h	dev_size_config_reg	Device Size Configuration Register
4000 E024h	remap_addr_reg	Remap Address Register
4000 E028h	mode_bit_config_reg	Mode Bit Configuration Register
4000 E030h	tx_thresh_reg	TX Threshold Register
4000 E034h	rx_thresh_reg	RX Threshold Register
4000 E038h	write_completion_ctrl_reg	Write Completion Control Register
4000 E03Ch	no_of_polls_bef_exp_reg	Polling Expiration Register
4000 E040h	irq_status_reg	Interrupt Status Register
4000 E044h	irq_mask_reg	Interrupt Mask Register
4000 E050h	lower_wr_prot_reg	Lower Write Protection Register
4000 E054h	upper_wr_prot_reg	Upper Write Protection Register
4000 E058h	wr_prot_ctrl_reg	Write Protection Control Register
4000 E090h	flash_cmd_ctrl_reg	Flash Command Control Register
4000 E094h	flash_cmd_addr_reg	Flash Command Address Register
4000 E0A0h	flash_rd_data_lower_reg	Flash Command Read Data Register (Lower)
4000 E0A4h	flash_rd_data_upper_reg	Flash Command Read Data Register (Upper)
4000 E0A8h	flash_wr_data_lower_reg	Flash Command Write Data Register (Lower)
4000 E0ACh	flash_wr_data_upper_reg	Flash Command Write Data Register (Upper)
4000 E0B0h	polling_flash_status_reg	Polling Flash Status Register
4000 E0FCh	module_id_reg	Module ID Register

8.4 Register Description

8.4.1 config_reg — QSPI Configuration Register

Address: 4000 5000h (QSPI1)
4000 E000h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	qspi_idle_fld	—	—	—	—	—	—	—	enable_ahb_decoder_fld	mstr_baud_div_fld				enter_xip_mode_imm_fld	enter_xip_mode_fld	enb_ahb_addr_remap_fld
Value after reset	1	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	wr_prot_flash_fld	periph_cs_lines_fld				periph_sel_dec_fld	enb_legacy_ip_mode_fld	enb_dir_acc_ctlr_fld	—	—	—	—	sel_clk_phase_fld	sel_clk_pol_fld	enb_qspi_fld
Value after reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1

Table 8.3 config_reg Register Contents (1/3)

Bit Position	Bit Name	Function	R/W
b31	qspi_idle_fld	Serial interface and QSPI pipeline are IDLE This is a STATUS read-only bit. Note this is a retimed signal, so there will be some inherent delay on the generation of this status signal. 0: Non-Idle mode 1: Idle mode	R
b30 to b24	Reserved		R
b23	enable_ahb_decoder_fld	Enable Address Decoder (Direct Access Mode Only) Not support for the address decoder. Keep the initial value.	R/W
b22 to b19	mstr_baud_div_fld	Baud Rate Divisor (4 to 32) SPI baud rate = (QSPI_REFCLK) / ((mstr_baud_div_fld+1) × 2) mstr_baud_div_fld — Actual Divisor 0000b — Prohibit 0001b — 4 ... — ... 1110b — 30 1111b — 32 Set these bits up before enabling the Quad SPI Controller.	R/W
b18	enter_xip_mode_imm_fld	Enter NoCMD Mode immediately For detailed description refer to **Section 8.6.5, Entering and Exiting NoCMD mode**. 0: If NoCMD mode is enabled, then setting to 0 will cause the controller to exit NoCMD mode after the next READ instruction has completed. 1: Access to the device in NoCMD mode immediately Use this register when the external device is in NoCMD mode. The controller will assume the next READ instruction will be passed to the device as an NoCMD mode instruction, and therefore will not require the READ opcode to be transferred. ***Note*)** To exit NoCMD mode on Quad SPI Controller, this bit should be set to 0. This will take effect in the attached device only after the next READ operation is executed. Software therefore should ensure that at least one READ operation is requested after resetting this bit in order to be sure that NoCMD mode is exited. This bit is synchronized in hardware and can be set/unset while the controller is operational. Don't write 1 to enter_xip_mode_fld at the same time.	R/W

Table 8.3 config_reg Register Contents (2/3)

Bit Position	Bit Name	Function	R/W
b17	enter_xip_mode_fld	Enter NoCMD Mode on next READ For detailed description refer to **Section 8.6.5, Entering and Exiting NoCMD mode**. 0: If NoCMD mode is enabled, then setting to 0 will cause the controller to exit NoCMD mode after the next READ instruction has completed. 1: If NoCMD mode is disabled, then setting to "1" will inform the controller that the device is ready to enter NoCMD mode on the next READ instruction. The controller will therefore send the appropriate command sequence, including mode bits to cause the device to enter NoCMD mode. Use this register after the controller has ensured the FLASH device has been configured to be ready to enter NoCMD mode. ***Note*)** To exit NoCMD mode on Quad SPI Controller, this bit should be set to 0. This will take effect in the attached device only after the next READ operation is executed. Software therefore should ensure that at least one READ operation is requested after resetting this bit in order to be sure that NoCMD mode is exited. This bit is synchronized in hardware and can be set/unset while the controller is operational. Don't write 1 to enter_xip_mode_imm_fld at the same time.	R/W
b16	enb_ahb_addr_remap_fld	Enable AHB Address Re-mapping (Direct Access Mode Only) When set to 1, the incoming Memory Map address will be adapted and sent to the FLASH device as (address + N), where N is the value stored in the Remap Address Register. This bit is synchronized in hardware and can be set/unset while the controller is operational.	R/W
b15	Reserved		R
b14	wr_prot_flash_fld	Set to drive the Write Protect pin of the FLASH device. Note that the WP pin is only valid in SINGLE or DUAL transfer modes. During QUAD transfers, the WP pin is used for transferring data and therefore any setting of this register bit will be ignored. This bit is synchronized in hardware and can be set/unset while the controller is operational. 0: Disable write protect 1: Enable write protect	R/W
b13 to b10	periph_cs_lines_fld	Peripheral Chip Select Lines If periph_sel_dec_fld = 0, xxx0b: QUAD_CS_N[3:0] = 1110b xx01b: QUAD_CS_N[3:0] = 1101b x011b: QUAD_CS_N[3:0] = 1011b 0111b: QUAD_CS_N[3:0] = 0111b 1111b: QUAD_CS_N[3:0] = 1111b (no peripheral selected) If periph_sel_dec_fld = 1, QUAD_CS_N[3:0] = periph_cs_lines_fld	R/W
b9	periph_sel_dec_fld	Peripheral select decode 0: Only 1 of 4 selects QUAD_CS_N[3:0] is active 1: QUAD_CS_N[3:0] = periph_cs_lines_fld	R/W
b8	enb_legacy_ip_mode_fld	Legacy IP Mode Enable 0: Use Direct Access Controller or STIG interface for data transfers 1: Legacy Mode is enabled. The controller via the AHB interface accesses the FLASH device in legacy mode (For detailed description refer to **Section 8.6.4, Using SPI Legacy Mode**). This bit is synchronized in hardware and can be set/unset while the controller is operational.	R/W

Table 8.3 config_reg Register Contents (3/3)

Bit Position	Bit Name	Function	R/W
b7	enb_dir_acc_ctlr_fld	Enable Direct Access Controller 0: Disable the Direct Access Controller once current transfer of the data word is complete. 1: Enable the Direct Access Controller When the Direct Access Controller is disabled, all AHB requested are completed with an error response. This bit is synchronized in hardware and can be set/unset while the controller is operational.	R/W
b6 to b3	Reserved		R
b2	sel_clk_phase_fld	CPHA 0: Data is sampled on the leading edge of the clock 1: Data is sampled on the trailing edge of the clock	R/W
b1	sel_clk_pol_fld	CPOL 0: The leading edge of the clock is rising and the trailing edge is falling 1: The leading edge of the clock is falling and the trailing edge is rising	R/W
b0	enb_qspi_fld	QSPI Enable 0: Disable the QSPI once current transfer of the data word is complete. 1: Enable the QSPI When QSPI Enable = 0, all output enables are inactive and all pins are set to input mode.	R/W

8.4.2 dev_instr_rd_config_reg — Device Read Instruction Configuration Register

Address: 4000 5004h (QSPI1)
4000 E004h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	dummy_rd_clk_cycles_fld					—	—	—	mode_bit_enable_fld	—	—	data_xfer_type_ext_mode_fld	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	addr_xfer_type_std_mode_fld		—	—	instr_type_fld		rd_opcode_non_xip_fld							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table 8.4 dev_instr_rd_config_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b29	Reserved		R
b28 to b24	dummy_rd_clk_cycles_fld	Dummy Read Clock Cycles Number of dummy clock cycles required by device for read instruction.	R/W
b23 to b21	Reserved		R
b20	mode_bit_enable_fld	Mode Bit Enable Set this field to 1 to ensure that the mode bits as defined in the Mode Bit Configuration Register are sent following the address bytes.	R/W
b19, b18	Reserved		R
b17, b16	data_xfer_type_ext_mode_fld	Data Transfer Type, when instr_type_fld is 0. 0: SIO mode - data is shifted to the device on QUAD_IO0 only and from the device on QUAD_IO1 only 1: Used for Dual Input/Output instructions. For data transfers, QUAD_IO[1:0] are used as both inputs and outputs. 2: Used for Quad Input/Output instructions. For data transfers, QUAD_IO[3:0] are used as both inputs and outputs. 3: Reserved (not used) When instr_type_fld is not 0, this bit is ignored.	R/W
b15, b14	Reserved		R
b13, b12	addr_xfer_type_std_mode_fld	Address Transfer Type, when instr_type_fld is 0 0: Addresses transferred on QUAD_IO0 only 1: Addresses transferred on QUAD_IO[1:0] only 2: Addresses transferred on QUAD_IO[3:0] 3: Reserved (not used) When instr_type_fld is not 0, this bit is ignored.	R/W
b11, b10	Reserved		R
b9, b8	instr_type_fld	Instruction Type 0: Use Standard SPI mode (Instruction always sent on QUAD_IO0 only) 1: Use DIO-SPI mode (Instructions, Address and Data always sent on QUAD_IO[1:0]) 2: Use QIO-SPI mode (Instructions, Address and Data always sent on QUAD_IO[3:0]) 3: Reserved (not used) ***Note*)** These bits are relevant not just to READ transfers. They are global settings and will affect READ's, WRITES.	R/W
b7 to b0	rd_opcode_non_xip_fld	Read Opcode in non-NoCMD mode Read Opcode to use when not in NoCMD mode	R/W

8.4.3 dev_instr_wr_config_reg — Device Write Instruction Configuration Register

Address: 4000 5008h (QSPI1)
4000 E008h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	dummy_wr_clk_cycles_fld					—	—	—	—	—	—	data_xfer_type_ext_mode_fld	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	addr_xfer_type_std_mode_fld		—	—	—	wel_dis_fld	wr_opcode_fld							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table 8.5 dev_instr_wr_config_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b29	Reserved		R
b28 to b24	dummy_wr_clk_cycles_fld	Dummy Write Clock Cycles Number of dummy clock cycles required by device for write instruction.	R/W
b23 to b18	Reserved		R
b17, b16	data_xfer_type_ext_mode_fld	Data Transfer Type, when instr_type_fld is 0. 0: SIO mode data is shifted to the device on QUAD_IO0 only and from the device on QUAD_IO1 only 1: Used for Dual Input/Output instructions. For data transfers, QUAD_IO[1:0] are used as both inputs and outputs. 2: Used for Quad Input/Output instructions. For data transfers, QUAD_IO[3:0] are used as both inputs and outputs. 3: Reserved (not used) When instr_type_fld is not 0, this bit is ignored.	R/W
b15, b14	Reserved		R
b13, b12	addr_xfer_type_std_mode_fld	Address Transfer Type, when instr_type_fld is 0. 0: Addresses transferred on QUAD_IO0 only 1: Addresses transferred on QUAD_IO[1:0] only 2: Addresses transferred on QUAD_IO[3:0] 3: Reserved (not used) When instr_type_fld is not 0, this bit is ignored.	R/W
b11 to b9	Reserved		R
b8	wel_dis_fld	WEL Disable This is to turn off automatic issuing of WriteEnableLatch Command before write operation for Direct Access Controller 0: Enable automatic issuing of WEL 1: Disable automatic issuing of WEL	R/W
b7 to b0	wr_opcode_fld	Write Opcode	R/W

8.4.4 dev_delay_reg — QSPI Device Delay Register

This register is used to introduce relative delays into the generation of the master output signals. All timings are defined in cycles of the QSPI_REFCLK.

Address: 4000 500Ch (QSPI1)
4000 E00Ch (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	d_nss_fld								d_btwn_fld							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	d_after_fld								d_init_fld							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.6 dev_delay_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	d_nss_fld	Chip Select De-Assert Added delay in QSPI_REFCLK for the length that the chip select outputs are de-asserted between transactions. The minimum delay for chip select to be de-asserted (0x0) is; 1 QUAD_CLK + 1 QSPI_REFCLK If value=X, then the chip select de-assert time will be; 1 QUAD_CLK + 1 QSPI_REFCLK + X QSPI_REFCLKs.	R/W
b23 to b16	d_btwn_fld	Chip Select De-Assert Different Slaves Delay in QSPI_REFCLK between one chip select being de-activated and the activation of another. This is used to ensure a quiet period between the selection of two different slaves. The minimum delay (0x0) is; 1 QUAD_CLK + 3 QSPI_REFCLKs. If the value=X, then the delay will be; 1 QUAD_CLK + 3 QSPI_REFCLKs + X QSPI_REFCLKs	R/W
b15 to b8	d_after_fld	Chip Select End of Transfer Delay in QSPI_REFCLK between last bit of current transaction and de-asserting the device chip select. By default, the chip select will be de-asserted on the cycle following the completion of the current transaction. If the value=X, then the chip select will be de-asserted X QSPI_REFCLKs after the last falling edge of QUAD_CLK.	R/W
b7 to b0	d_init_fld	Chip Select Start of Transfer Delay in QSPI_REFCLK between setting chip select low and first bit transfer. By default (value=0), chip select will be asserted half a QUAD_CLK period before the first rising edge of QUAD_CLK. If the value=X, the chip select will be asserted half a QUAD_CLK period before the first rising edge of QUAD_CLK + X QSPI_REFCLKs.	R/W

8.4.5 rd_data_capture_reg — Read Data Capture Register

Address: 4000 5010h (QSPI1)
4000 E010h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	sample_edge_sel_fld	delay_fld				—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 8.7 rd_data_capture_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b6	Reserved		R
b5	sample_edge_sel_fld	Sample edge selection Choose edge on which data from flash memory will be sampled 0: Data outputs from flash memory are sampled on falling edge of the QSPI_REFCLK 1: Data outputs from flash memory are sampled on rising edge of the QSPI_REFCLK	R/W
b4 to b1	delay_fld	Read Delay Delay the read data capturing logic by the programmed number of QSPI_REFCLK cycles	R/W
b0	Reserved	Keep the reset value	R/W

8.4.6 dev_size_config_reg — Device Size Configuration Register

Address: 4000 5014h (QSPI1)
4000 E014h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	dev_size_resv_fld			mem_size_on_cs3_fld		mem_size_on_cs2_fld		mem_size_on_cs1_fld		mem_size_on_cs0_fld		bytes_per_subsector_fld				
Value after reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	bytes_per_device_page_fld												num_addr_bytes_fld			
Value after reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0

Table 8.8 dev_size_config_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b29	dev_size_resv_fld		R
b28, b27	mem_size_on_cs3_fld	Size of Flash Device connected to QUAD_CS_N[3] pin This field is only valid when Address Decoder is enabled (enable_ahb_decoder_fld = 1). 00b: Size of 512 Mb. 01b: Size of 1 Gb. 10b: Size of 2 Gb. 11b: Size of 4 Gb. ***Caution***) This bit is ignored because decoding the Address Decoder is not supported.	R/W
b26, b25	mem_size_on_cs2_fld	Size of Flash Device connected to QUAD_CS_N[2] pin Same as mem_size_on_cs3_fld	R/W
b24, b23	mem_size_on_cs1_fld	Size of Flash Device connected to QUAD_CS_N[1] pin Same as mem_size_on_cs3_fld	R/W
b22, b21	mem_size_on_cs0_fld	Size of Flash Device connected to QUAD_CS_N[0] pin Same as mem_size_on_cs3_fld	R/W
b20 to b16	bytes_per_subsector_fld	Number of bytes per Block This is required by the controller for performing the write protection logic. The number of bytes per block must be a power of 2 number. 0 = 1 byte 1 = 2 bytes .. 16 = 65536 bytes	R/W
b15 to b4	bytes_per_device_page_fld	Number of bytes per device page This is required by the controller for performing FLASH writes up to and across page boundaries.	R/W
b3 to b0	num_addr_bytes_fld	Number of address bytes (= num_addr_bytes_fld + 1) This field should be set the address byte number of the device. A value of 0 = 1 byte.	R/W

8.4.7 remap_addr_reg — Remap Address Register

Address: 4000 5024h (QSPI1)
4000 E024h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	value_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	value_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.9 remap_addr_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	value_fld	This register is used to remap an incoming direct access address to a different address used by the FLASH device.	R/W

8.4.8 mode_bit_config_reg — Mode Bit Configuration Register

Address: 4000 5028h (QSPI1)
4000 E028h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	mode_fld							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.10 mode_bit_config_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7 to b0	mode_fld	Mode bits These are the 8 mode bits that are sent to the device following the address bytes.	R/W

8.4.9 tx_thresh_reg — TX Threshold Register

Address: 4000 5030h (QSPI1)
4000 E030h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	tx_thresh_resv_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	tx_thresh_resv_fld												level_fld			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 8.11 tx_thresh_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	tx_thresh_resv_fld	Reserved	R
b3 to b0	level_fld	Defines the level at which the TX FIFO not full interrupt is generated (Legacy mode only)	R/W

8.4.10 rx_thresh_reg — RX Threshold Register

Address: 4000 5034h (QSPI1)
4000 E034h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	level_fld			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 8.12 rx_thresh_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	Reserved		R
b3 to b0	level_fld	Defines the level at which the RX FIFO not empty interrupt is generated (Legacy mode only)	R/W

8.4.11 write_completion_ctrl_reg — Write Completion Control Register

This register defines how the controller will poll the device following a write transfer.

Address: 4000 5038h (QSPI1)
4000 E038h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	poll_rep_delay_fld								poll_count_fld							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	disable_polling_fld	polling_polarity_fld	—	—	polling_bit_index_fld			opcode_fld							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table 8.13 write_completion_ctrl_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	poll_rep_delay_fld	Polling repetition delay The polling interval is extended by setting this bit.	R/W
b23 to b16	poll_count_fld	Poll count Defines the number of times the controller should expect to see a true result from the polling in successive reads of the device register.	R/W
b15	Reserved		R
b14	disable_polling_fld	Disable polling This switches off the automatic polling function. 0: Enable automatic polling 1: Disable automatic polling	R/W
b13	polling_polarity_fld	Polling polarity Defines the polling polarity. 1: The write transfer to the device will be completed if the polled bit is equal to “1”. 0: The write transfer to the device will be completed if the polled bit is equal to “0”.	R/W
b12, b11	Reserved		R
b10 to b8	polling_bit_index_fld	Polling bit index Defines the bit index that should be polled. A value of 010b means that bit 2 of the returned data will be polled for. A value of 111b means that bit 7 of the returned data will be polled for.	R/W
b7 to b0	opcode_fld	Opcode Defines the opcode that should be issued by the controller when it is automatically polling for device program completion. This command is issued following all device write operations. By default, this will poll the standard device STATUS register using opcode 0x05	R/W

8.4.12 no_of_polls_bef_exp_reg — Polling Expiration Register

Address: 4000 503Ch (QSPI1)
4000 E03Ch (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	no_of_polls_bef_exp_fld															
Value after reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	no_of_polls_bef_exp_fld															
Value after reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 8.14 no_of_polls_bef_exp_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	no_of_polls_bef_exp_fld	Defines the maximum number of polls that will stop polling and generate an interrupt if the write is not completed. This field should be greater than 32769.	R/W

8.4.13 irq_status_reg — Interrupt Status Register

The status fields in this register are set when the described event occurs and the interrupt is enabled in the Interrupt Mask Register. When any of these bit fields are set, the interrupt output is asserted high. The fields are cleared by reading this register. Note that bit fields 7 through 11 are only valid when legacy SPI mode is active.

Address: 4000 5040h (QSPI1)
4000 E040h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	poll_exp_int_fld	—	rx_fifo_full_fld	rx_fifo_not_empty_fld	tx_fifo_full_fld	tx_fifo_not_full_fld	recv_overflow_fld	—	illegal_access_det_fld	prot_wr_attempt_fld	—	—	underflow_det_fld	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.15 irq_status_reg Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b14	Reserved		R
b13	poll_exp_int_fld	The maximum number of programmed polling cycles is expired	R
b12	Reserved		R
b11	rx_fifo_full_fld	RX FIFO full (Current FIFO status) This can be ignored in non-SPI legacy mode. 0: FIFO is not full 1: FIFO is full	R
b10	rx_fifo_not_empty_fld	RX FIFO not empty (Current FIFO status) This can be ignored in non-SPI legacy mode. 0: FIFO has < RX THRESHOLD entries 1: FIFO has ≥ RX THRESHOLD entries	R
b9	tx_fifo_full_fld	TX FIFO full (Current FIFO status) This can be ignored in non-SPI legacy mode. 0: FIFO is not full 1: FIFO is full	R
b8	tx_fifo_not_full_fld	TX FIFO not full (Current FIFO status) This can be ignored in non-SPI legacy mode. 0: FIFO has > TX THRESHOLD entries 1: FIFO has ≤ TX THRESHOLD entries	R
b7	recv_overflow_fld	Receive Overflow This should only occur in Legacy SPI mode. Set if an attempt is made to push the RX FIFO when it is full. This bit is reset only by a module reset and cleared only when this register is read. If a new push to the RX FIFO occurs coincident with a register read this flag will remain set. 0: No overflow has been detected. 1: An overflow has occurred.	R
b6	Reserved		R

Table 8.15 irq_status_reg Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b5	illegal_access_det_fld	Illegal AHB access has been detected. This interrupt is also triggered, if AHB access is performed when the DirectAccessController is disabled. 0: Illegal AHB access is not detected 1: Illegal AHB access is detected	R
b4	prot_wr_attempt_fld	Write to protected area was attempted and rejected. 0: Write to protected area was not attempted 1: Write to protected area was attempted	R
b3, b2	Reserved		R
b1	underflow_det_fld	Underflow Detected 0: No underflow has been detected 1: Underflow is detected and an attempt to transfer data is made when the TX FIFO is empty. This may occur when AHB write data is being supplied too slowly to keep up with the requested write operation This bit is reset only by a module reset and cleared only when this register is read.	R
b0	Reserved		R

8.4.14 irq_mask_reg — Interrupt Mask Register

0: The interrupt for the corresponding Interrupt Status Register bit is disabled.
1: The interrupt for the corresponding Interrupt Status Register bit is enabled.

Address: 4000 5044h (QSPI1)
4000 E044h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	poll_exp_int_mask_fld	—	rx_fifo_full_mask_fld	rx_fifo_not_empty_mask_fld	tx_fifo_full_mask_fld	tx_fifo_not_full_mask_fld	recv_overflow_mask_fld	—	illegal_access_det_mask_fld	prot_wr_attempt_mask_fld	—	—	underflow_det_mask_fld	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.16 irq_mask_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b14	Reserved		R
b13	poll_exp_int_mask_fld	Polling expiration detected mask	R/W
b12	Reserved	Keep the reset value	R/W
b11	rx_fifo_full_mask_fld	RX FIFO full mask	R/W
b10	rx_fifo_not_empty_mask_fld	RX FIFO not empty mask	R/W
b9	tx_fifo_full_mask_fld	TX FIFO full mask	R/W
b8	tx_fifo_not_full_mask_fld	TX FIFO not full mask	R/W
b7	recv_overflow_mask_fld	Receive Overflow mask	R/W
b6	Reserved	Keep the reset value	R/W
b5	illegal_access_det_mask_fld	Illegal Access Detected mask	R/W
b4	prot_wr_attempt_mask_fld	Protected Area Write Attempt mask	R/W
b3, b2	Reserved	Keep the reset value	R/W
b1	underflow_det_mask_fld	Underflow Detected mask	R/W
b0	Reserved	Keep the reset value	R/W

8.4.15 lower_wr_prot_reg — Lower Write Protection Register

Address: 4000 5050h (QSPI1)
4000 E050h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	subsector_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	subsector_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.17 lower_wr_prot_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	subsector_fld	The block number that defines the lower block in the range of blocks that is to be locked from writing. The definition of a block in terms of number of bytes is programmable via the Device Size Configuration Register. Modify this bit only when write protection feature is low.	R/W

8.4.16 upper_wr_prot_reg — Upper Write Protection Register

Address: 4000 5054h (QSPI1)
4000 E054h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	subsector_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	subsector_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.18 upper_wr_prot_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	subsector_fld	The block number that defines the upper block in the range of blocks that is to be locked from writing. The definition of a block in terms of number of bytes is programmable via the Device Size Configuration Register. Modify this bit only when write protection feature is off.	R/W

8.4.17 wr_prot_ctrl_reg — Write Protection Control Register

Address: 4000 5058h (QSPI1)
4000 E058h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	enb_fld	inv_fld
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.19 wr_prot_ctrl_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b2	Reserved		R
b1	enb_fld	Write Protection Enable Bit 1: Any AHB write access with an address within the protection region defined in the Lower and Upper Write Protection Registers is rejected. An AHB error response is generated and an interrupt source triggered. 0: The protection region is disabled.	R/W
b0	inv_fld	Write Protection Inversion Bit 1: The protection region defined in the Lower and Upper Write Protection Registers is inverted meaning it is the region that the system is permitted to write to. 0: The protection region defined in the Lower and Upper Write Protection Registers is the region that the system is not permitted to write to. Modify this bit only when write protection feature is low.	R/W

8.4.18 flash_cmd_ctrl_reg — Flash Command Control Register

This register is used for STIG. Refer to **Section 8.5.3, Software Triggered Instruction Generator (STIG)**.

Address: 4000 5090h (QSPI1)
4000 E090h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	cmd_opcode_fld								enb_read_data_fld	num_rd_data_bytes_fld			enb_comd_addr_fld	enb_mode_bit_fld	num_addr_bytes_fld	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	enb_write_data_fld	num_wr_data_bytes_fld			num_dummy_bytes_fld					—	—	—	—	—	cmd_exec_status_fld	cmd_exec_fld
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.20 flash_cmd_ctrl_reg Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b24	cmd_opcode_fld	Command Opcode The command opcode field should be setup before triggering the command. For example, 0x20 maps to SubSector Erase. Writing to the execute field (bit 0) of this register launches the command. ***Note*)** Using this approach to issue commands to the device will make use of the instruction type of the Device Read Instruction Configuration Register. 00b: The command opcode data will be transferred in a serial fashion. 01b: The command opcode, command address, command dummy bytes and command data will all be transferred in parallel using QUAD_IO[1:0] pins. 10b: The command opcode, command address, command dummy bytes and command data will all be transferred in parallel using QUAD_IO[3:0] pins.	R/W
b23	enb_read_data_fld	Read Data Enable Set to 1 if the command specified in the command opcode field (bits[31:24]) requires read data bytes to be received from the device.	R/W
b22 to b20	num_rd_data_bytes_fld	Number of Read Data Bytes Up to 8 data bytes may be read using this command. Set to 0 for 1 byte and 7 for 8 bytes.	R/W
b19	enb_comd_addr_fld	Command Address Enable Set to 1 if the command specified in bits[31:24] requires an address. This should be setup before triggering the command via writing a 1 to the execute field.	R/W
b18	enb_mode_bit_fld	Mode Bit Enable Set to 1 to ensure the mode bits as defined in the Mode Bit Configuration Register are sent following the address bytes.	R/W
b17, b16	num_addr_bytes_fld	Number of Address Bytes Set to the number of address bytes required (the address itself is programmed in the Flash Command Address Register). This should be setup before triggering the command via bit 0 of this register. 00b: 1 address byte 01b: 2 address bytes 10b: 3 address bytes 11b: 4 address bytes	R/W
b15	enb_write_data_fld	Write Data Enable Set to 1 if the command specified in the command opcode field requires write data bytes to be sent to the device.	R/W
b14 to b12	num_wr_data_bytes_fld	Number of Write Data Bytes Up to 8 Data bytes may be written using this command Set to 0 for 1 byte, 7 for 8 bytes.	R/W

Table 8.20 flash_cmd_ctrl_reg Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b11 to b7	num_dummy_bytes_fld	Number of Dummy Bytes Set to the number of dummy bytes required This should be setup before triggering the command via the execute field of this register. ***Note*)** This field must not be greater than 5'h03. Setting more dummy bytes can cause TX FIFO full condition to be triggered or overlooking some data for read commands and the behavior of the module is unpredictable.	R/W
b6 to b2	Reserved		R
b1	cmd_exec_status_fld	Command execution status When the command is in progress, this bit is set.	R
b0	cmd_exec_fld	Execute the command The command is executed by writing 1b to this bit.	W

8.4.19 flash_cmd_addr_reg — Flash Command Address Register

Address: 4000 5094h (QSPI1)
4000 E094h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	addr_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	addr_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.21 flash_cmd_addr_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	addr_fld	Command Address This field should be setup before triggering the command in bit 0 of the Flash Command Control Register. It is the address used by the command specified in the cmd_opcode_fld (bits[31:24]) of the Flash Command Control Register.	R/W

8.4.20 flash_rd_data_lower_reg — Flash Command Read Data Register (Lower)

Address: 4000 50A0h (QSPI1)
4000 E0A0h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	data_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	data_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.22 flash_rd_data_lower_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	data_fld	Command Read Data (Lower byte) This is the data that is returned by the flash device for any status or configuration read operation carried out by triggering the event in the Flash Command Control Register. The register will be valid when bit 1 in the Flash Command Control Register is 0.	R

8.4.21 flash_rd_data_upper_reg — Flash Command Read Data Register (Upper)

Address: 4000 50A4h (QSPI1)
4000 E0A4h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	data_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	data_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.23 flash_rd_data_upper_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	data_fld	Command Read Data (Upper byte) This is the data that is returned by the FLASH device for any status or configuration read operation carried out by triggering the event in the Flash Command Control Register. The register will be valid when the polling bit in the Flash Command Control Register is low.	R

8.4.22 flash_wr_data_lower_reg — Flash Command Write Data Register (Lower)

Address: 4000 50A8h (QSPI1)
4000 E0A8h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	data_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	data_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.24 flash_wr_data_lower_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	data_fld	Command Write Data Lower Byte This is the command write data lower byte. This should be setup before triggering the command in bit 0 of the Flash Command Control Register. It is the data that is to be written to the flash for any status or configuration write operation carried out by triggering the event in the Flash Command Control Register.	R/W

8.4.23 flash_wr_data_upper_reg — Flash Command Write Data Register (Upper)

Address: 4000 50ACh (QSPI1)
4000 E0ACh (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	data_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	data_fld															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.25 flash_wr_data_upper_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	data_fld	Command Write Data Upper Byte This is the command write data upper byte. This should be setup before triggering the command in bit 0 of the Flash Command Control Register. It is the data that is to be written to the flash for any status or configuration write operation carried out by triggering the event in the Flash Command Control Register.	R/W

8.4.24 polling_flash_status_reg — Polling Flash Status Register

Address: 4000 50B0h (QSPI1)
4000 E0B0h (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	device_status_valid_fld	device_status_fld							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8.26 polling_flash_status_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b9	Reserved		R
b8	device_status_valid_fld	Polling Status Valid This bit is set when the value in bits from 7 to 0 is valid.	R
b7 to b0	device_status_fld	Flash Status Actual status register of the device	R

8.4.25 module_id_reg — Module ID Register

Address: 4000 50FCh (QSPI1)
4000 E0FCh (QSPI2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	value_fld							
Value after reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	value_fld															
Value after reset	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0

Table 8.27 module_id_reg Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved		R
b23 to b0	value_fld	Module/Revision ID number	R

8.5 Operation

8.5.1 AHB Control Interface

The AHB slave controller performs the following operation.

- Validates incoming AHB accesses
- Responds to invalid requests
- Performs any required byte and half-word reordering
- Blocks write that violate the programmed write protection rules
- Forwards the transfer request to the direct access controller

8.5.1.1 Remapping the Memory Map address

The Quad SPI Controller does not implement any specific address decoding to error incoming addresses that may lie outside the connected FLASH memory space. The incoming direct access address, by default, maps directly to the address sent serially to the FLASH device.

If the FLASH device has a 24-bit address, then the 24 LSBs of the Memory Map address will be forwarded. A remap feature is available to remap all incoming Memory Map addresses to ADDRESS+ N, where N is the value store in the Remap Address Register. It is enabled via the enb_ahb_addr_remap_fld bit of QSPI Configuration Register. Disable the Quad SPI Controller before configuring the remap setting. This feature could be used when software needs to move boot code to another FLASH region.

8.5.1.2 Write Protection

In order to protect the FLASH device, a write protection feature is implemented in hardware and controlled from software. Any write detected, pointing to protected FLASH area, will generate a bus error.

A programmable region of the FLASH device, defined as a number of FLASH “blocks” starting from a particular block number can be protected. Three programmable protection registers (Lower Write Protection Register/Upper Write Protection Register/Write Protection Control Register) are provided.

The first register defines the FLASH block that is located at the bottom of the region to be protected.

The second register defines the FLASH block that is located at the top of the region to be protected.

The third register is a control register consisting of 2 bits. Bit 0 allows software to invert the region that is being protected, causing the programmed region to become the only areas of FLASH memory that is not protected from writes. Bit 1 is the write protection enable bit. When low, the FLASH device is unprotected.

A block can be between 1 and 64 Kbytes, programmed via the Device Size Configuration Register.

8.5.2 Direct Access Controller (DAC)

Direct access refers to the operation where the Memory Map address access directly trigger a read or write to FLASH memory. It can be used to both access and directly execute code from external FLASH memory.

FLASH erase operations, which may be required before a page write, are triggered by software using the documented programming interface.

Once a page program cycle has been started, the Quad SPI Controller will automatically poll for the write cycle to complete before the next access. This is achieved by holding any subsequent direct accesses in the wait state. The wait time depends to the device.

8.5.3 Software Triggered Instruction Generator (STIG)

The direct access controller is used to transfer data. In order to access the volatile and non-volatile configuration registers, the SPI Status register, other status/protection registers as well as to perform ERASE functions, a separate software controller is required.

The software triggered instruction generator, or STIG, is controlled using the Flash Command Control Register by setting up the command to issue to the FLASH device. This is a generic controller and can be used to perform any instruction that the FLASH device supports from the extended SPI protocol.

Bit 0 is used to trigger the command, bit 1 used by software to poll the status of the command execution.

For reads, when the command has been serviced (bit 1 toggles from “1” to “0”), up to 8 bytes of read data will be placed in the Flash Command Read Data Register.

For writes, the write data should be placed in the Flash Command Write Data Register.

8.5.4 Servicing a STIG request

A STIG request will cause the Quad SPI Controller to interrogate the Flash Command Control Register to determine what and how many bytes it should send to the FLASH device.

If there is an address to send, then the address (the size of which is also programmed in the same register) is sent next. The address itself is stored in the Flash Command Address Register.

If there are any dummy bytes to send (the size of which is also programmed in the same register) then those are sent next.

If there is data to write or read (the size of which is also programmed in the same register) then for the case of writes, up to 8 bytes can be sent (as stored in the Flash Command Write Data Registers) next.

In the read case, when the read data has been collected from the FLASH device, the Quad SPI Controller stores that in the Flash Command Read Data Registers.

When the Quad SPI Controller starts to service a STIG request, it sets bit 1 of the Flash Command Control Register to indicate a command execution is in progress.

8.5.5 Arbitration between Direct Access Controller and STIG

When multiple controllers are active simultaneously, a simple fixed-priority arbitration scheme is used to arbitrate between each interface and access the external FLASH. The fixed priority is defined as follows, highest priority first.

(1) The Direct Access Write

(2) The STIG

(3) The Direct Access Read

Each controller is back pressured while waiting to be serviced.

8.5.6 SPI Command Translation

Requests issued by the direct access controller or the STIG will be translated into a sequence of byte transfers to send downstream (before serialization to the FLASH device). These sequences depend on the requested transfer but an example of a typical 1-byte non sequential READ is shown below:

INSTRUCTION OPCODE → ADDRESS → Mode Byte → Dummy Bytes → 1 byte of don't care

For sequential accesses, an extra byte of data per read is pushed to the FLASH device on the back of the above sequence assuming it can be done so with no gap between each transferred byte.

The actual sequence sent to the FLASH device depends on the requested transfers, whether the transfer is non-sequential or sequential, whether the device has been configured in NoCMD mode and the state of the Device Read/Write Instruction Configuration Registers.

For writes, the write enable latch (WEL) within the FLASH device itself must be high before a write sequence can be issued. The Quad SPI Controller will automatically issue the write enable latch command before triggering a write command via the direct access controller - i.e. the user does not need to perform this operation. The op-code for this operation is typically 06h and is common between devices.

When write requests from the direct access controller are no longer being received and all outstanding requests have been sent, the FLASH device will automatically start the page program write cycle. Any incoming request at this time will be held in wait states until the cycle has completed. The Quad SPI Controller will automatically poll the FLASH device SPI status register to identify when the write cycle has completed. This is achieved by sending the command to read status register to the FLASH device and waiting until the device itself has indicated the write cycle has completed (until the Write in Progress bit [0] has cleared to zero and the write enable latch bit has also cleared to zero).

The write enable latch command and the command to read status register are the ones that are sent by the controller automatically. For any other specific instruction that the user determines should be sent to the device (for example if the device needs to be unprotected before a write command is issued), these should be handled separately by issuing FLASH commands via the STIG.

8.5.7 Selecting the Flash Instruction Type

In order to send the correct READ and WRITE opcodes, software should initialize the Device READ Instruction Configuration Register and the Device WRITE Instruction Configuration Register. These registers include fields to setup the required instruction op-codes that is intended to be used to access the FLASH (default is basic READ and basic page program) as well as the instruction type and whether the instruction uses single, dual or quad pins for address and data transfer.

Despite being applicable for both READs and WRITEs, the Instruction Type field (instr_type_fld) in the Device READ Instruction Configuration Register only appears once - it is not included in the Device WRITE Instruction Configuration Register.

If software sets the Instruction Type field to anything other than “0”, then the address xfer type and the data xfer type bits of both the registers become don’t care. It is made available to allow software to support the less common FLASH instructions where the opcode, address and data are sent on 2 or 4 lanes (the opcode from most instructions are sent serially to the FLASH device, even for dual/quad instructions).

Also note that for devices that support instructions that can send the OPCODE over 2 or 4 lanes, the name for these instructions are not consistently named in the FLASH datasheets. One of the devices that support these instructions is the Numonyx (Micron) N25Q128. The extra READ instructions are called DCFR and QCFR. The WRITE instructions are DCPP and QCPP. For reference, below is a table illustrating how software should configure the Quad SPI Controller for each specific READ and WRITE instruction supported by the N25Q128 device.

Table 8.28 Number of Lanes for READ Instructions Supported by the N25Q128 Device

Instruction	Lanes Used by OPCODE	Lanes Used to Send Address	Lanes Used to Send Data	Device Read Instruction Config Register		
				instr_type_fld	addr_xfer_type_std_mode_fld	data_xfer_type_ext_mode_fld
READ	1	1	1	0	0	0
FAST_READ	1	1	1	0	0	0
DOFR (Dual Output Fast Read)	1	1	2	0	0	1
DIOFR (Dual I/O Fast Read)	1	2	2	0	1	1
QOFR (Quad Output Fast Read)	1	1	4	0	0	2
QIOFR (Quad I/O Fast Read)	1	4	4	0	2	2
DCFR (Dual Command Fast Read)	2	2	2	1	Don’t care	Don’t care
QCFR (Quad Command Fast Read)	4	4	4	2	Don’t care	Don’t care

Table 8.29 Number of Lanes for WRITE Instructions Supported by the N25Q128 Device

Instruction	Lanes Used by OPCODE	Lanes Used to Send Address	Lanes Used to Send Data	Device Write Instruction Config Register		
				instr_type_fld *1	addr_xfer_type_std_mode_fld	data_xfer_type_ext_mode_fld
PP (Page Program)	1	1	1	0	0	0
DIFP (Dual Input Fast Program)	1	1	2	0	0	1
DIEFP (Dual Input Extended Fast Program)	1	2	2	0	1	1
QIFP (Quad Input Fast Program)	1	1	4	0	0	2
QIEFP (Quad Input Extended Fast Program)	1	4	4	0	2	2
DCPP (Dual Command Page Program)	2	2	2	1	Don't care	Don't care
QCPP (Quad Command Page Program)	4	4	4	2	Don't care	Don't care

Note 1. Device Read Instruction Configuration Register

8.5.8 APB Interface and Register Module

The APB interface is used to configure the controller and perform software controlled FLASH accesses using the Flash Command Control Register (refer to **Section 8.5.3, Software Triggered Instruction Generator (STIG)** and **Section 8.6.3, Using the Flash Command Control Register (STIG Operation)** for detailed description).

8.5.9 Read Data Capturing

There are programmable features allowing software to tune the logic that captures the read data from the device.

After reset, the QSPI_REFCLK delay is in a disabled state. This should be valid for most applications, and can be used for device enumeration, where the device clock will be configured variously. The Read Data Capture Register provides the control for these features.

Bit[4:1] controls the additional number of read data capture cycles (this is the fast QSPI_REFCLK, running at least x4 of the device clock) that should be applied to the internal read data capture circuit. The large clock-to-out delay of the flash memory together with trace delays as well as other device delays may impose a maximum flash clock frequency which is less than the flash memory device itself can operate at. To compensate, software should set this register to a value that guarantees robust data captures.

8.5.9.1 Example of an 8 byte Read Transfer

To help understand the high level behavior of the Quad SPI interface, the following diagram shows the primary stages in a read transfer. Note this is just one type of read instruction so there are many variations on this.

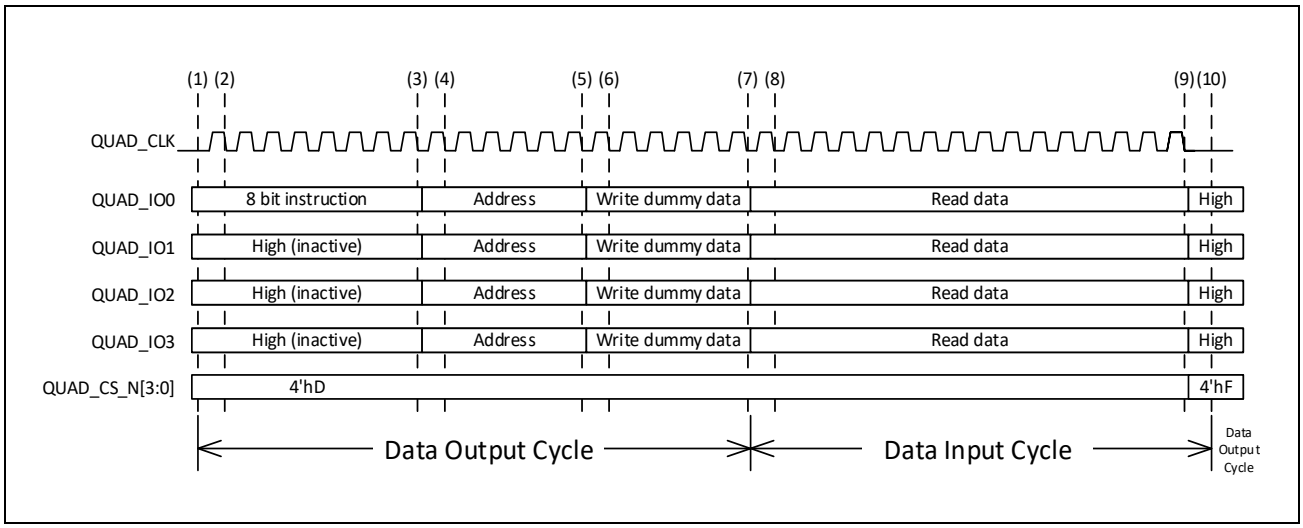


Figure 8.2 Example of an 8 Byte Read Transfer

(1) Start of transaction, chip select (QUAD_CS_N) activates slave 1 (transitions from 4'hF to 4'hD).

(2) Start of instruction phase. The first of 8 bits are output only on QUAD_IO0. Other QUAD_IO pins are unused at this stage.

(3) End of instruction phase. The last of the 8 instruction bits is out.

(4) Start of address phase. The example uses a 3 byte address output across all 4 QUAD_IO pins. One address nibble is output per clock cycle so 3 bytes takes 6 cycles.

(5) End of address phase. The last address nibble is output using all 4 QUAD_IO pins.

(6) The example read transaction requires 3 bytes of dummy write data. The edge is where the first of the 6 nibbles are output across all 4 QUAD_IO pins.

(7) Last nibble of dummy write data. This is the last write phase of the read transfer therefore QUAD_IO pins change "Data Input Cycle".

(8) First cycle of read data on QUAD_IO pins. Example is reading 8 bytes which will take 16 cycles using all QUAD_IO pins.

(9) Last cycle of read data input. QUAD_IO pins switch back to being floating. Chip enable is de-asserted for slave 1. Transaction complete.

(10) QUAD_IO pins change “DATA Output Cycle” for next transaction.

8.6 Programming Quad SPI Controller

Software is responsible for configuring the Quad SPI Controller before it can communicate with the FLASH device.

It is assumed that the controller's static configuration bits are setup before the Quad SPI Controller is "enabled" via bit[0] of the "QSPI Configuration Register". This is necessary in order to avoid any clock boundary issues. If the user wishes to change the controller's configuration, the QSPI Enable bit should be set to "disabled" before reconfiguring.

8.6.1 Configuring the Quad SPI Controller for use after reset

The Quad SPI Controller has been designed to wake up in a state that is suitable for performing basic reads and writes using the direct access controller. The BASIC read (opcode 03h) and BASIC write (opcode 02h) instructions are operations supported by all target devices.

The Controller also wakes up with a baud rate divider setting of divide-by-32. Assuming the QSPI_REFCLK is operating at 166.67 MHz after reset, then this means the effective QUAD_CLK is just 5.2 MHz. This should be slow enough to meet all timing requirements of all target devices without any further device programming.

If the target device does not use 3 address bytes, the Device Size Configuration Register must be modified to the appropriate size.

If software plans to write to the device, and the number of bytes per device page is NOT equal to 256, then the Device Size Configuration Register must also be modified.

While not a requirement, it is prudent for software to enable the write protect feature prior to enabling the Quad SPI Controller. This will block any direct writes from taking effect. To do so, the protection registers (Offset 50h, 54h and 58h) should be setup and the number of bytes per device block in the Device Size Configuration Register should also be setup.

8.6.2 Configuring the Quad SPI Controller for optimal use

To access the flash optimally, software must configure the controller accurately.

(1) Wait until any pending STIG operation has completed.

(2) Disable the DAC (Direct Access Controller) — bit 7 of the QSPI Configuration Register. It is permitted, but not necessary to also disable the Quad SPI Controller completely here via bit 0 of the same register.

(3) Update the Device Write/Read Instruction Configuration Register for the instruction type you wish to use for direct writes and reads.

(4) Update Mode Bit Configuration Register, if mode bits have been enabled in the Device Read Instruction Configuration Register.

(5) Update the Device Size Configuration Register as needed. The number of address bytes is a key configuration setting required for performing reads and writes. The number of bytes per page is required for performing any write. The number of bytes per device block is only required if the write protect feature is used.

(6) Update QSPI Device Delay Register. This register allows the user to tweak how the chip select is driven after each FLASH access. This is required as each device may have different timing requirements. As the serial clock frequency is increased, these timing requirements become more important. Note the numbers programmed in this register are based on the period of QSPI_REFCLK. Example: A device needs 50 ns minimum time before CS can be reasserted after it has been de-asserted. By default, the controller will only provide a minimum of 1 QUAD_CLK period. When the device is operating at 62.5 MHz, the QUAD_CLK period is only 16 ns, so 34 ns extra is required. Since the register defines the number of QSPI_REFCLK cycles to add, and QSPI_REFCLK is

running at 250 MHz (4 ns period), then the user should program a value of at least 9 to the d_nss_fld of this register.

(7) Update Remap Address Register, if required.

(8) Setup and enable Write Protection Registers, if they are required and if they have not already been setup from post initialization.

(9) Enable required interrupts via the Interrupt Mask Register.

(10) Setup the baud rate divisor in the QSPI Configuration Register to define the required clock frequency of the target device.

(11) Update Read Data Capture Register. This register will delay when the read data is captured and can help when the read data path from the device to the controller is long and the device clock frequency is high. An update to this register may not be necessary.

(12) Enable the Quad SPI Controller and the DAC via the QSPI Configuration Register.

8.6.3 Using the Flash Command Control Register (STIG Operation)

The Flash Command Control Register provides software a means to access the FLASH device is a flexible and programmable manner. This is a STIG operation (Software Triggered Instruction Generator).

The instruction opcode, number of address bytes (if any), the address itself, number of dummy bytes (if any), number of write data bytes(if any), the write data itself and the number of read data bytes(if any) can be programmed. Once these have been programmed, software can trigger the command via bit 0 and wait for its completion by polling bit 1.

This method of accessing the FLASH is the typical mechanism that software would use to access the FLASH device's registers, as well as for performing ERASE operations. It can also be used to access the FLASH array itself, although it only has the ability to drive a single I/O pin, so dual and quad instructions would not be possible. In addition, a maximum of 8 data bytes may be read or written at any one time, defined in the Flash Command Write/Read Data Registers (Offset A0h, A4h, A8h and ACh).

Commands issued using this interface have a higher priority than all other direct READ accesses, and will therefore interrupt any READ commands being requested by the direct access controller. It will, however, not interrupt a write sequence that may have been issued via the direct access controller. In these cases, software may find it takes a long time for bit 1 to indicate the operation has completed.

8.6.4 Using SPI Legacy Mode

SPI legacy mode allows software to access the internal TX-FIFO and RX-FIFO directly, thus bypassing the direct and STIG controllers. These controllers should therefore be disabled before legacy mode is entered.

Legacy mode allows the user to issue any FLASH instruction to the device, but does place a heavy software overhead in order to manage the fill levels of the FIFO's effectively. This is because the legacy SPI core is bi-directional in nature, with data continuously being transferred while the chip select is enabled.

Even if the driver only wishes to read data from the FLASH device, dummy data must be written out to ensure the chip select stays active, and vice versa for write transactions. This means that to perform a basic READ of 4 bytes to a device that has 3 address bytes, software would have to write a total of 8 bytes to the TX FIFO. The first byte would be the instruction opcode, the next 3 bytes are the address, and the final 4 bytes would be dummy data to ensure the chip select stays active while the read data is returned.

Similarly, since 8 bytes were written to the TX-FIFO, software should expect 8 bytes to be returned in the RX-FIFO. The first 4 bytes of this would be the unnecessary READ data, leaving the final 4 bytes holding the required READ data. Since the TX-FIFO and RX-FIFO are of limited depth (8 bytes) in legacy mode, software has a responsibility to maintain the FIFO levels to ensure the TX-FIFO does not become exhausted during the instruction execution and the RX-FIFO doesn't overflow. This can place a lot of overhead on software.

Interrupts are provided to indicate when the fill levels pass programmable watermarks, which are TX/RX Threshold Registers. Software accesses the TX-FIFO by writing any value to any address via the QSPI[m] ROM area to the Quad SPI Controller while legacy mode is enabled. Software accesses the RX-FIFO by reading any address via the QSPI[m] ROM area to the Quad SPI Controller while legacy mode is enabled. (m = 1 or 2)

8.6.5 Entering and Exiting NoCMD mode

The read instruction can be reduced by NoCMD mode, and the performance can be improved especially in the case of Execute in Place (XIP). The name of this mode is vary depending on the FLASH device and manufacturer.

To enter or exit the NoCMD mode, set the mode bit according to the device. For each Mode Bit setting, refer to the device datasheet.

The below procedures are common entering and exiting procedures for most flash devices, Refer to the data sheet of each device for details.

8.6.5.1 Entering NoCMD mode

To enter NoCMD mode, perform the following steps:

(1) Disable the Direct Access Controller to ensure no new read or write accesses will be sent to the FLASH device.

(2) Configure the mode bits to the values of the device, depending on the FLASH device and manufacturer. Set Mode Bit Enable to 1.

(3) When the device is not in NoCMD mode:
Set bit 17 (enter_xip_mode_fld) of the QSPI Configuration Register to 1.

When the device is in NoCMD mode:
Set bit 18 (enter_xip_mode_imm_fld) of the QSPI Configuration Register to 1.

8.6.5.2 Exiting NoCMD mode

To exit NoCMD mode, perform the following steps:

(1) Disable the Direct Access Controller to ensure no new read or write accesses will be sent to the FLASH device.

(2) Restore the setting related to mode bits to the values before entering NoCMD mode, depending on the FLASH device and manufacturer.

(3) Set bit 17 (enter_xip_mode_fld) and bit 18 (enter_xip_mode_imm_fld) of the QSPI Configuration Register to 0.

A READ operation with disabling NoCMD mode is required to exit the mode in the FLASH device, so NoCMD mode will internally stay active in the device until the next READ operation. The method for disabling NoCMD mode in the flash device depends on the device.

8.6.6 Using the Write Protection Registers

After reset, the AHB protection mechanism wakes in a disabled state. Software can use the protection registers to block QSPI Memory Map area writes.

Refer to **Section 8.5.1.2, Write Protection** for details. It is recommended that the Quad SPI Controller is disabled before configuring the protection registers.

8.7 Usage Notes

8.7.1 4-byte Address Output

The incoming direct access address maps directly to the address sent serially to the FLASH device. When the 4-byte address is output (dev_size_config_reg. num_addr_bytes_fld = 3), the access to address 0x10000000 in the QSPI1 ROM area will sent the same address to the FLASH device.

The address remapping function makes it possible to make the address appropriate for the FLASH device. Set the offset value to Remap Address Register (remap_addr_reg) and set enb_ahb_addr_remap_fld bit of QSPI Configuration Register (config_reg) to 1 to enable the address remapping function.

Ex.)

To send 4-byte address 0x00000000 to the FLASH device by accessing to address 0x10000000 in the QSPI1 ROM area, set remap_addr_reg = 0xF0000000 and enable the address remapping function.

8.7.2 Write Protection Area

Any writing to the protected area set in Write Protection Register will generate a bus error. The block sets in Write Protection Register is based on the Memory Map address.

The address remapping function makes it possible to make the starting block appropriate for the FLASH device. Set the offset value to Remap Address Register (remap_addr_reg) and set enb_ahb_addr_remap_fld bit of QSPI Configuration Register (config_reg) to 1 to enable the address remapping function.

Ex.)

To set 0x10000000 in QSPI1 ROM area as 0 block base, set remap_addr_reg = 0xF0000000 and enable the address remapping function.

8.7.3 Automatically Polling for Write Complete

In direct access, the controller will automatically poll for the write cycle complete before the next access. The next access may be delayed for this reason. The wait time depends to the page program period of the FLASH device.

The automatic polling function is controlled by disable_polling_fld bit of Write Completion Control Register (write_completion_ctrl_reg). If disabling automatically polling, the user must confirm that the FLASH device write cycle is complete.

8.7.4 Chip Select Output

Because the function of enable_ahb_decoder_fld bit of QSPI Configuration Register (config_reg) cannot be used, the chip select cannot be automatically selected by the address decoder. Use periph_sel_dec_fld and periph_cs_lines_fld bits of QSPI Configuration Register (config_reg) by software setting.

Section 9 SDIO/SD/eMMC Controller

9.1 Overview

There are two instances of SDIO/SD/eMMC Controllers, each instance supports SD, SDIO, and eMMC. This controller supports connection to a single slot and performs multi-block writes and erases that lower access overhead.

- SD/SDIO Card interface
  - Transfers data in 1 bit or 4 bits mode
  - Speeds:
    Default Speed Mode: 3.3 V signaling, Frequency up to 25 MHz, up to 12.5 MB/sec
    High Speed Mode: 3.3 V signaling, Frequency up to 50 MHz, up to 25 MB/sec
- eMMC interface
  - Transfers data in 1 bit, 4 bits or 8 bits mode
  - Speeds:
    Backwards Compatibility Mode: 3.3 V signaling, Frequency up to 25 MHz, up to 25 MB/sec
    High Speed SDR Mode: 3.3 V signaling, Frequency up to 50 MHz, up to 50 MB/sec
- Miscellaneous
  - Card Detection (Insertion/Removal)
  - Support for SPI mode
  - Support for PIO/SDMA/ADMA2 transfer
  - Support for Read wait Control, Suspend/Resume operation
  - Support for FIFO Overrun and Underrun condition by stopping SD clock
  - Support for Multi Media Card Interrupt mode
  - Wakeup capability

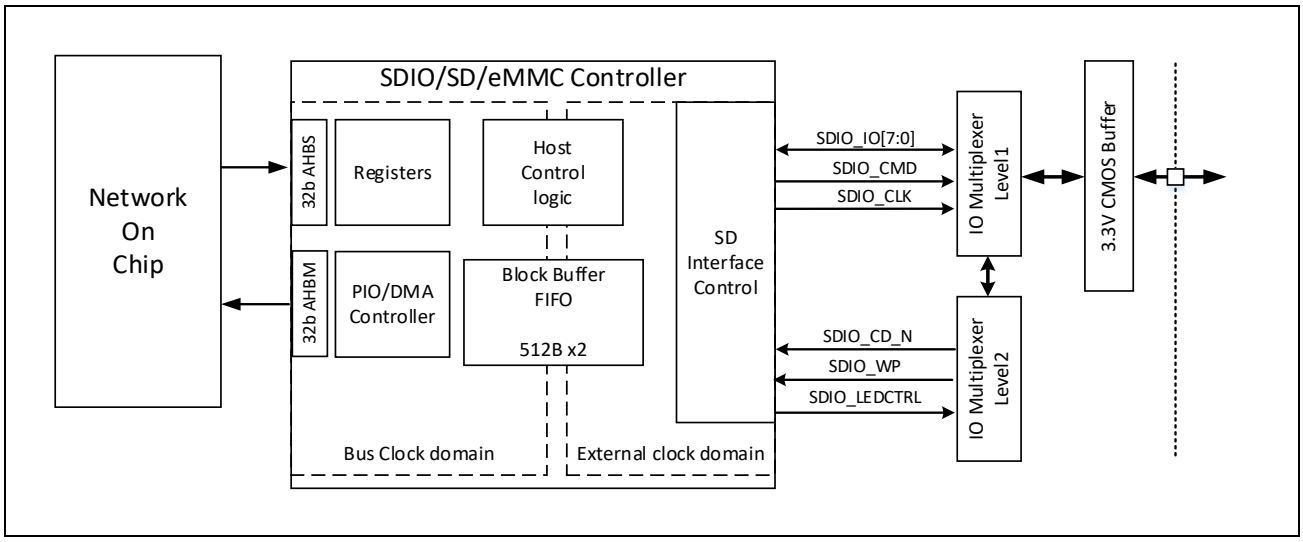


Figure 9.1 SDIO/SD/eMMC1 & 2 Controller Interfaces and Connections

9.2 Signal Interfaces

Signal Name	Input Output	Description
Clock		
SDIO[m]_HCLK	Input	Internal bus clock (AHB)
SDIO[m]_ECLK	Input	External Interface clock
Interrupt		
SDIF[m]_Int	Output	SD/SDIO/eMMC interrupt, level sensitive, Active High
SDIF[m]_wkup_Int	Output	Wakeup interrupt, level sensitive, Active High
External Signal		
SDIO[m]_CLK	Output	Clock
SDIO[m]_CMD	I/O	Command/Response
SDIO[m]_IO[7:0]	I/O	DAT line
SDIO[m]_CD_N	Input	Card Detection, Active Low
SDIO[m]_WP	Input	Write Protect, Active High
SDIO[m]_LEDCTRL	Output	Card access status

Note: m=1 or 2
Index removed style is used in this chapter.
Ex) SDIO_CLK

9.3 Register Map

9.3.1 Register Map (SDIO1)

Table 9.1 SDIO Controller 1 Register Map

Address	Register Symbol	Register Name
4010 0000h	reg_sdmasysaddrlo	SDMA System Address Register Low
4010 0002h	reg_sdmasysaddrhi	SDMA System Address Register High
4010 0004h	reg_blocksize	Block Size Register
4010 0006h	reg_blockcount	Block Count Register
4010 0008h	reg_argument1lo	Argument1 Register Low
4010 000Ah	reg_argument1hi	Argument1 Register High
4010 000Ch	reg_transfermode	Transfer Mode Register
4010 000Eh	reg_command	Command Register
4010 0010h + 2h × n	reg_response[n] (n = 0..7)	Response Register [n]
4010 0020h	reg_dataport	Buffer Data Port Register
4010 0024h	reg_presentstate	Present State Register
4010 0028h	reg_hostcontrol1	Host Control 1 Register
4010 0029h	reg_powercontrol	Power Control Register
4010 002Ah	reg_blockgapcontrol	Block Gap Control Register
4010 002Bh	reg_wakeupcontrol	Wake-up Control Register
4010 002Ch	reg_clockcontrol	Clock Control Register
4010 002Eh	reg_timeoutcontrol	Timeout Control Register
4010 002Fh	reg_softwarereset	Software Reset Register
4010 0030h	reg_normalintrsts	Normal Interrupt Status Register
4010 0032h	reg_errorintrsts	Error Interrupt Status Register
4010 0034h	reg_normalintrstsena	Normal Interrupt Status Enable Register
4010 0036h	reg_errorintrstsena	Error Interrupt Status Enable Register
4010 0038h	reg_normalintrsigena	Normal Interrupt Signal Enable Register
4010 003Ah	reg_errorintrsigena	Error Interrupt Signal Enable Register
4010 003Ch	reg_autocmderrsts	Auto CMD Error Status Register
4010 003Eh	reg_hostcontrol2	Host Control 2 Register
4010 0040h	reg_capabilities	Capabilities Register
4010 0044h	reg_capabilities_cont	Capabilities Register (Continue)
4010 0048h	reg_maxcurrentcap	Maximum Current Capabilities Register
4010 0050h	reg_ForceEventforAUTOCMDErrorStatus	Force Event for Auto CMD Error Status Register
4010 0052h	reg_forceeventforerrintsts	Force Event for Error Interrupt Status Register
4010 0054h	reg_admaerrsts	ADMA Error Status Register
4010 0058h	reg_admasysaddr0	ADMA System Address Register Low
4010 005Ah	reg_admasysaddr1	ADMA System Address Register High
4010 0060h	reg_presetvalue0	Preset Value Register for Initialization
4010 0062h	reg_presetvalue1	Preset Value Register for Default Speed
4010 0064h	reg_presetvalue2	Preset Value Register for High Speed
4010 00FCh	reg_slotintrsts	Slot Interrupt Status Register
4010 00FEh	reg_hostcontrollerver	Host Controller Version Register

9.3.2 Register Map (SDIO2)

Table 9.2 SDIO Controller 2 Register Map

Address	Register Symbol	Register Name
4010 1000h	reg_sdmasysaddrlo	SDMA System Address Register Low
4010 1002h	reg_sdmasysaddrhi	SDMA System Address Register High
4010 1004h	reg_blocksize	Block Size Register
4010 1006h	reg_blockcount	Block Count Register
4010 1008h	reg_argument1lo	Argument1 Register Low
4010 100Ah	reg_argument1hi	Argument1 Register High
4010 100Ch	reg_transfermode	Transfer Mode Register
4010 100Eh	reg_command	Command Register
4010 1010h + 2h × n	reg_response[n] (n = 0..7)	Response Register [n]
4010 1020h	reg_dataport	Buffer Data Port Register
4010 1024h	reg_presentstate	Present State Register
4010 1028h	reg_hostcontrol1	Host Control 1 Register
4010 1029h	reg_powercontrol	Power Control Register
4010 102Ah	reg_blockgapcontrol	Block Gap Control Register
4010 102Bh	reg_wakeupcontrol	Wake-up Control Register
4010 102Ch	reg_clockcontrol	Clock Control Register
4010 102Eh	reg_timeoutcontrol	Timeout Control Register
4010 102Fh	reg_softwarereset	Software Reset Register
4010 1030h	reg_normalintrsts	Normal Interrupt Status Register
4010 1032h	reg_errorintrsts	Error Interrupt Status Register
4010 1034h	reg_normalintrstsena	Normal Interrupt Status Enable Register
4010 1036h	reg_errorintrstsena	Error Interrupt Status Enable Register
4010 1038h	reg_normalintrsigena	Normal Interrupt Signal Enable Register
4010 103Ah	reg_errorintrsigena	Error Interrupt Signal Enable Register
4010 103Ch	reg_autocmderrsts	Auto CMD Error Status Register
4010 103Eh	reg_hostcontrol2	Host Control 2 Register
4010 1040h	reg_capabilities	Capabilities Register
4010 1044h	reg_capabilities_cont	Capabilities Register (Continue)
4010 1048h	reg_maxcurrentcap	Maximum Current Capabilities Register
4010 1050h	reg_ForceEventforAUTOCMDErrorStatus	Force Event for Auto CMD Error Status Register
4010 1052h	reg_forceeventforerrintsts	Force Event for Error Interrupt Status Register
4010 1054h	reg_admaerrsts	ADMA Error Status Register
4010 1058h	reg_admasysaddr0	ADMA System Address Register Low
4010 105Ah	reg_admasysaddr1	ADMA System Address Register High
4010 1060h	reg_presetvalue0	Preset Value Register for Initialization
4010 1062h	reg_presetvalue1	Preset Value Register for Default Speed
4010 1064h	reg_presetvalue2	Preset Value Register for High Speed
4010 10FCh	reg_slotintrsts	Slot Interrupt Status Register
4010 10FEh	reg_hostcontrollerver	Host Controller Version Register

9.4 Register Description

9.4.1 reg_sdmasysaddrlo — SDMA System Address Register Low

This register contains the Lower 16-bit of physical system memory address used for DMA transfers or the second argument for the Auto CMD23.

Address: 4010 0000h (SDIO1)
4010 1000h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	sdma_sysaddress															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.3 reg_sdmasysaddrlo Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b0	sdma_sysaddress	This field contains the Lower 16-bit of physical system memory address used for DMA transfers or the second argument for the Auto CMD23. **(1) SDMA System Address** This register contains the system memory address for a SDMA transfer. When the Host Controller (HC) stops a SDMA transfer, this register shall point to the system address of the next contiguous data position. It can be accessed only if no transaction is executing (i.e., after a transaction has stopped). Read operations during transfers may return an invalid value. The Host Driver (HD) shall initialize this register before starting a SDMA transaction. After SDMA has stopped, the next system address of the next contiguous data position can be read from this register. The SDMA transfer waits at the every boundary specified by the Host SDMA Buffer Boundary in the Block Size Register. The HC generates DMA Interrupt to request the HD to update this register. The HD sets the next system address of the next data position to this register. When the most upper byte of this register (003h) is written, the HC restarts the SDMA transfer. When restarting SDMA by the Resume command or by setting Continue Request in the Block Gap Control Register, the HC shall start at the next contiguous address stored here in the SDMA System Address Register. ADMA does not use this register. **(2) Argument 2** This register is used with the Auto CMD23 to set a 32-bit block count value to the argument of the CMD23 while executing Auto CMD23. If Auto CMD23 is used with ADMA, the full 32-bit block count value can be used. If Auto CMD23 is used without ADMA, the available block count value is limited by the Block Count Register. 65535 blocks are the maximum value in this case.	R/W

9.4.2 reg_sdmasysaddrhi — SDMA System Address Register High

This register contains the Higher 16-bit of physical system memory address used for DMA transfers or the second argument for the Auto CMD23.

Address: 4010 0002h (SDIO1)
4010 1002h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	sdma_sysaddress															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.4 reg_sdmasysaddrhi Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b0	sdma_sysaddress	This Field contains the Higher 16 bits of Physical system memory address used for DMA transfers. Details of this field is described in reg_admasysaddr0.	R/W

9.4.3 reg_blocksize — Block Size Register

This register is used to configure the number of bytes in a data block.

Address: 4010 0004h (SDIO1)
4010 1004h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	sdma_bufboundary			xfer_blocksize											
Value after reset	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.5 reg_blocksize Register Contents

Bit Position	Bit Name	Function	R/W
b15	Reserved		R
b14 to b12	sdma_bufboundary	Host SDMA Buffer Boundary To perform long DMA transfer, SDMA System Address Register shall be updated at every system boundary during DMA transfer. These bits specify the size of contiguous buffer in the system memory. The DMA transfer shall wait at the every boundary specified by these fields and the HC generates the DMA Interrupt to request the HD to update the SDMA System Address Register. These bits shall support when the DMA Support in the Capabilities Register is set to 1 and this function is active when the DMA Enable in the Transfer Mode Register is set to 1. 000b: 4 KB (Detects A11 Carry out) 001b: 8 KB (Detects A12 Carry out) 010b: 16 KB (Detects A13 Carry out) 011b: 32 KB (Detects A14 Carry out) 100b: 64 KB (Detects A15 Carry out) 101b: 128 KB (Detects A16 Carry out) 110b: 256 KB (Detects A17 Carry out) 111b: 512 KB (Detects A18 Carry out)	R/W
b11 to b0	xfer_blocksize	Transfer Block Size This field specifies the block size for block data transfers for CMD17, CMD18, CMD24, CMD25 and CMD53. It can be accessed only if no transaction is executing (i.e after a transaction has stopped). Read operations during transfer return an invalid value and write operations shall be ignored. 0000h: No Data Transfer 0001h: 1 Byte 0002h: 2 Bytes 0003h: 3 Bytes 0004h: 4 Bytes ... ... 01FFh: 511 Bytes 0200h: 512 Bytes ... ... 0800h: 2048 Bytes	R/W

9.4.4 reg_blockcount — Block Count Register

This register is used to configure the number of data blocks.

Address: 4010 0006h (SDIO1)
4010 1006h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	xfer_blockcount															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.6 reg_blockcount Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b0	xfer_blockcount	Blocks Count for Current Transfer This register is enabled when Block Count Enable in the Transfer Mode Register is set to 1 and is valid only for multiple block transfers. The HC decrements the block count after each block transfer and stops when the count reaches zero. It can be accessed only if no transaction is executing (i.e. after a transaction has stopped). Read operations during transfer return an invalid value and write operations shall be ignored. When saving transfer context as a result of Suspend command, the number of blocks yet to be transferred can be determined by reading this register. When restoring transfer context prior to issuing a Resume command, the HD shall restore the previously save block count. 0000h: Stop Count 0001h: 1 block 0002h: 2 blocks ... ... FFFFh: 65535 blocks	R/W

9.4.5 reg_argument1lo — Argument1 Register Low

This register contains Lower bits of SD Command Argument.

Address: 4010 0008h (SDIO1)
4010 1008h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	command_argument1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.7 reg_argument1lo Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b0	command_argument1	The SD Command Argument is specified as bit23-8 of Command-Format.	R/W

9.4.6 reg_argument1hi — Argument1 Register High

This register contains higher bits of SD Command Argument.

Address: 4010 000Ah (SDIO1)
4010 100Ah (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	command_argument1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.8 reg_argument1hi Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b0	command_argument1	The SD Command Argument is specified as bit39-24 of Command-Format.	R/W

9.4.7 reg_transfermode — Transfer Mode Register

This register is used to control the operations of data transfers.

Address: 4010 000Ch (SDIO1)
4010 100Ch (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	xfermode_multiblksel	xfermode_dataxferdir	xfermode_autocmdena		xfermode_blkcntena	xfermode_dmaenable
Value after reset	X	X	X	X	X	X	X	X	X	X	0	0	0	0	0	0

Table 9.9 reg_transfermode Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b15 to b6	Reserved		R
b5	xfermode_multiblksel	Multi/Single Block Select This bit enables multiple block data transfers. 0: Single Block 1: Multiple Block	R/W
b4	xfermode_dataxferdir	Data Transfer Direction Select This bit defines the direction of data transfers. 0: Write from Host to Card 1: Read from Card to Host	R/W
b3, b2	xfermode_autocmdena	Auto CMD Enable This field determines use of auto command functions. 00b: Auto Command Disabled 01b: Auto CMD12 Enabled 10b: Auto CMD23 Enabled 11b: Reserved There are two methods to stop Multiple-block read and write operation. **(1) Auto CMD12 Enable** Multiple-block read and write commands for memory require CMD12 to stop the operation. When this field is set to 01b, the HC issues CMD12 automatically when last block transfer is completed. Auto CMD12 error is indicated to the Auto CMD Error Status Register. The HD shall not set this bit if the command does not require CMD12. **(2) Auto CMD23 Enable** When this bit field is set to 10b, the HC issues a CMD23 automatically before issuing a command specified in the Command Register. The following conditions are required to use the Auto CMD23. • A memory card that supports CMD23 (SCR[33]=1). • If DMA is used, it shall be ADMA. • Only when CMD18 or CMD25 is issued. By writing the Command Register, the HC issues a CMD23 first and then issues a command specified by the Command Index in Command Register 32-bit block count value for CMD23 is set to SDMA System Address Register.	R/W
b1	xfermode_blkcntena	Block Count Enable This bit is used to enable the Block Count Register, which is only relevant for multiple block transfers. When this bit is 0, the Block Count Register is disabled, which is useful in executing an infinite transfer. 0: To disable Block count 1: To enable Block Count	R/W

Table 9.9 reg_transfermode Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b0	xfermode_dmaenable	DMA Enable DMA can be enabled only if DMA Support bit in the Capabilities Register is set. If this bit is set to 1, a DMA operation shall begin when the HD writes to the upper byte of Command Register (00Fh). 0: To disable DMA 1: To enable DMA	R/W

9.4.8 reg_command — Command Register

This register is used to program the Command for host controller.

Address: 4010 000Eh (SDIO1)
4010 100Eh (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	command_cmdindex						command_cmdtype		command_datapresent	command_indexchkena	command_crcchkena	—	command_responsetype	
Value after reset	X	X	0	0	0	0	0	0	0	0	0	0	0	X	0	0

Table 9.10 reg_command Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b15, b14	Reserved		R
b13 to b8	command_cmdindex	Command Index This bit shall be set to the command number (CMD0-63, ACMD0-63).	R/W
b7, b6	command_cmdtype	Command Type This bit is used for select different command types. There are three types of special commands. Suspend, Resume and Abort. These bits shall be set to 00b for all other commands. • Suspend Command: If the Suspend command succeeds, the HC shall assume the SD Bus has been released and that it is possible to issue the next command which uses the DAT line. The HC shall de-assert Read Wait for read transactions and stop checking busy for write transactions. The Interrupt cycle shall start, in 4-bit mode. If the Suspend command fails, the HC shall maintain its current state and the HD shall restart the transfer by setting Continue Request in the Block Gap Control Register. • Resume Command: The HD re-starts the data transfer by restoring the registers in the range of 000-00Dh. The HC shall check for busy before starting write transfers. • Abort Command: If this command is set when executing a read transfer, the HC shall stop reads to the buffer. If this command is set when executing a write transfer, the HC shall stop driving the DAT line. After issuing the Abort command, the HD should issue a software reset. 00b: Normal 01b: Suspend 10b: Resume 11b: Abort	R/W
b5	command_datapresent	Data Present Select This bit is set to 1 to indicate that data is present and shall be transferred using the DAT line. If is set to 0 for the following: 1. Commands using only CMD line (ex. CMD52). 2. Commands with no data transfer but using busy signal on DAT[0] line (R1b or R5b ex. CMD38). 3. Resume Command. 0: Data not Present 1: Data present	R/W
b4	command_indexchkena	Command Index Check Enable If this bit is set to 1, the HC shall check the index field in the response to see if it has the same value as the command index. If it is not, it is reported as a Command Index Error. If this bit is set to 0, the Index field is not checked. 0: Disable 1: Enable	R/W

Table 9.10 reg_command Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b3	command_crcchkena	Command CRC Check Enable If this bit is set to 1, the HC shall check the CRC field in the response. If an error is detected, it is reported as a Command CRC Error. If this bit is set to 0, the CRC field is not checked. 0: Disable 1: Enable	R/W
b2	Reserved		R
b1, b0	command_responsetype	Response Type Select. 00b: No Response 01b: Response Length 136 10b: Response Length 48 11b: Response Length 48, check busy after response	R/W

9.4.9 reg_response[n] — Response Register [n] (n = 0..7)

This register is used to store responses from SD Cards.

Address: 4010 0010h + 2h × n (SDIO1)
4010 1010h + 2h × n (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	command_response															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.11 reg_response[n] Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b0	command_response	The command response from SD bus is stored to these registers according to response type. REP[15:0] → reg_response[0] ... REP[127:111] → reg_response[7]	R

Response Type	Meaning of Response	Response Field	REP[ ]
R1, R1b (Normal response)	Card Status	R[39:8]	REP[31:0]
R1b (Auto CMD12 response)	Card Status for Auto CMD12	R[39:8]	REP[127:96]
R1 (Auto CMD23 response)	Card Status for Auto CMD23	R[39:8]	REP[127:96]
R2 (CID, CSD register)	CID or CSD reg. incl.	R[127:8]	REP[119:0]
R3 (OCR register)	OCR reg. for memory	R[39:8]	REP[31:0]
R4 (OCR register)	OCR reg. for I/O, etc.	R[39:8]	REP[31:0]
R5, R5b	SDIO response	R[39:8]	REP[31:0]
R6 (Published RCA response)	New published RCA[31:16] etc.	R[39:8]	REP[31:0]

9.4.10 reg_dataport — Buffer Data Port Register

This register is used to access internal buffer.

Address: 4010 0020h (SDIO1)
4010 1020h (SDIO2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	sdhcdmactrl_piobufrddata															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	sdhcdmactrl_piobufrddata															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.12 reg_dataport Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	sdhcdmactrl_piobufrddata	The HC Buffer can be accessed through this 32-bit Data Port Register.	R/W

9.4.11 reg_presentstate — Present State Register

The HD can get status of the HC from this 32-bit read-only register.

Address: 4010 0024h (SDIO1)
4010 1024h (SDIO2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	sdif_dat7in_dsync	sdif_dat6in_dsync	sdif_dat5in_dsync	sdif_dat4in_dsync	sdif_cmdin_dsync	sdif_dat3in_dsync	sdif_dat2in_dsync	sdif_dat1in_dsync	sdif_dat0in_dsync	sdif_wp_dsync	sdif_cd_n_dsync	sdhccarddet_statestable_dsync	sdhccarddet_inserted_dsync
Value after reset	X	X	X	1	1	1	1	1	1	1	1	1	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	sdhcdmactrl_piobufrdena	sdhcdmactrl_piobufwrena	sdhcdmactrl_rdxferactive	sdhcdmactrl_wrxferactive	—	—	—	—	—	sdhcdmactrl_datalineactive	presentstate_inhibitdat	presentstate_inhibitcmd
Value after reset	X	X	X	X	0	0	0	0	X	X	X	X	X	0	0	0

Table 9.13 reg_presentstate Register Contents (1/3)

Bit Position	Bit Name	Function	R/W
b31 to b29	Reserved		R
b28	sdif_dat7in_dsync	This status is used to check DAT[7] line level to recover from errors, and for debugging.	R
b27	sdif_dat6in_dsync	This status is used to check DAT[6] line level to recover from errors, and for debugging.	R
b26	sdif_dat5in_dsync	This status is used to check DAT[5] line level to recover from errors, and for debugging.	R
b25	sdif_dat4in_dsync	This status is used to check DAT[4] line level to recover from errors, and for debugging.	R
b24	sdif_cmdin_dsync	This status is used to check CMD line level to recover from errors, and for debugging.	R
b23	sdif_dat3in_dsync	This status is used to check DAT[3] line level to recover from errors, and for debugging.	R
b22	sdif_dat2in_dsync	This status is used to check DAT[2] line level to recover from errors, and for debugging.	R
b21	sdif_dat1in_dsync	This status is used to check DAT[1] line level to recover from errors, and for debugging.	R
b20	sdif_dat0in_dsync	This status is used to check DAT[0] line level to recover from errors, and for debugging. This is especially useful in detecting the busy signal level from DAT[0].	R
b19	sdif_wp_dsync	Write Protect Switch Pin Level The Write Protect Switch is supported for memory and combo cards. This bit reflects the SDIO_WP pin. 0: Write protected (SDIO_WP = 1) 1: Write enabled (SDIO_WP = 0)	R
b18	sdif_cd_n_dsync	Card Detect Pin Level This bit reflects the inverse value of the SDIO_CD_N pin. 0: No Card present (SDIO_CD_N = 1) 1: Card present (SDIO_CD_N = 0)	R
b17	sdhccarddet_statestable_dsync	Card State Stable This bit is used for testing. If it is 0, the Card Detect Pin Level is not stable. If this bit is set to 1, it means the Card Detect Pin Level is stable. The Software Reset For All in the Software Reset Register shall not affect this bit. 0: Reset or Debouncing 1: No Card or Inserted	R

Table 9.13 reg_presentstate Register Contents (2/3)

Bit Position	Bit Name	Function	R/W
b16	sdhccarddet_inserted_dsync	Card Inserted This bit indicates whether a card has been inserted. Changing from 0 to 1 generates a Card Insertion interrupt in the Normal Interrupt Status Register and changing from 1 to 0 generates a Card Removal Interrupt in the Normal Interrupt Status Register. The Software Reset For All in the Software Reset Register shall not affect this bit. If a Card is removed while its power is on and its clock is oscillating, the HC shall clear SD Bus Power in the Power Control Register and SD Clock Enable in the Clock Control Register. In addition, the HD should clear the HC by the Software Reset For All in Software Register. The card detect is active regardless of the SD Bus Power. 0: Reset or Debouncing or No Card 1: Card Inserted	R
b15 to b12	Reserved		R
b11	sdhcdmactrl_piobufrdena	Buffer Read Enable This status is used for non-DMA read transfers. This read only flag indicates that valid data exists in the host side buffer status. If this bit is 1, readable data exists in the buffer. A change of this bit from 1 to 0 occurs when all the block data is read from the buffer. A change of this bit from 0 to 1 occurs when block data is ready in the buffer and generates the Buffer Read Ready Interrupt. 0: Read Disable 1: Read Enable	R
b10	sdhcdmactrl_piobufwrena	Buffer Write Enable This status is used for non-DMA write transfers. This read only flag indicates if space is available for write data. If this bit is 1, data can be written to the buffer. A change of this bit from 1 to 0 occurs when all the block data is written to the buffer. A change of this bit from 0 to 1 occurs when top of block data can be written to the buffer and generates the Buffer Write Ready Interrupt. 0: Write Disable 1: Write enable	R
b9	sdhcdmactrl_rdxferactive	Read Transfer Active This status is used for detecting completion of a read transfer. This bit is set to 1 for either of the following conditions: • After the end bit of the read command. • When writing a 1 to continue Request in the Block Gap Control Register to restart a read transfer. This bit is cleared to 0 for either of the following conditions: • When the last data block as specified by block length is transferred to the system. • When all valid data blocks have been transferred to the system and no current block transfers are being sent as a result of the Stop At Block Gap Request set to 1. A transfer complete interrupt is generated when this bit changes to 0. 0: No valid data 1: Transferring data	R

Table 9.13 reg_presentstate Register Contents (3/3)

Bit Position	Bit Name	Function	R/W
b8	sdhcdmactrl_wrxferactive	Write Transfer Active This status indicates a write transfer is active. If this bit is 0, it means no valid write data exists in the HC. This bit is set in either of the following cases: • After the end bit of the write command. • When writing a 1 to Continue Request in the Block Gap Control Register to restart a write transfer. This bit is cleared in either of the following cases: • After getting the CRC status of the last data block as specified by the transfer count (Single or Multiple) • After getting a CRC status of any block where data transmission is about to be stopped by a Stop At Block Gap Request. During a write transaction, a Block Gap Event interrupt is generated when this bit is changed to 0, as a result of the Stop At Block Gap Request being set. This status is useful for the HD in determining when to issue commands during write busy. 0: No valid data 1: Transferring data	R
b7 to b3	Reserved		R
b2	sdhcdmactrl_datalineactive	DAT Line Active This bit indicates whether one of the DAT line on SD bus is in use. 1: DAT line active 0: DAT line inactive	R
b1	presentstate_inhibitdat	Command Inhibit (DAT) This status bit is generated if either the DAT Line Active or the Read transfer Active is set to 1. If this bit is 0, it indicates the HC can issue the next SD command. Commands with busy signal belong to Command Inhibit (DAT) (ex. R1b, R5b type). Changing from 1 to 0 generates a Transfer Complete interrupt in the Normal Interrupt Status Register. ***Note*)** The SD Host Driver can save registers in the range of 000-00Dh for a suspend transaction after this bit has changed from 1 to 0. 0: Can issue command which uses the DAT line 1: Cannot issue command which uses the DAT line	R
b0	presentstate_inhibitcmd	Command Inhibit (CMD) If this bit is 0, it indicates the CMD line is not in use and the HC can issue a SD command using the CMD line. This bit is set immediately after the Command Register (00Fh) is written. This bit is cleared when the command response is received. Even if the Command Inhibit (DAT) is set to 1, Commands using only the CMD line can be issued if this bit is 0. Changing from 1 to 0 generates a Command complete interrupt in the Normal Interrupt Status Register. If the HC cannot issue the command because of a command conflict error or because of Command not Issued By Auto CMD12 Error, this bit shall remain 1 and the Command Complete is not set. Status issuing Auto CMD12 is not read from this bit. Auto CMD12 and Auto CMD23 consist of two responses. In this case, this bit is not cleared by the response of CMD12 or CMD23 but cleared by the response of a read/write command. Status issuing Auto CMD12 is not read from this bit. So if a command is issued during Auto CMD12 operation, HC shall manage to issue two commands: CMD12 and a command set by Command Register.	R

9.4.12 reg_hostcontrol1 — Host Control 1 Register

This register is used to program DMA modes, LED Control, Data Transfer Width, High Speed Enable, Card detect test level and signal selection.

Address: 4010 0028h (SDIO1)
4010 1028h (SDIO2)

Bit	b7	b6	b5	b4	b3	b2	b1	b0
	hostctrl1_cdsigselect	hostctrl1_cdtestlevel	hostctrl1_extdatawidth	hostctrl1_dmaselect		hostctrl1_highspeedena	hostctrl1_datawidth	hostctrl1_ledcontrol
Value after reset	0	0	0	0	0	0	0	0

Table 9.14 reg_hostcontrol1 Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b7	hostctrl1_cdsigselect	Card Detect Signal Selection This bit selects source for card detection. 0: SDIO_CD_N is selected (for normal use) 1: The card detect test level is selected	R/W
b6	hostctrl1_cdtestlevel	Card Detect Test Level This bit is enabled while the Card Detect Signal Selection is set to 1 and it indicates card inserted or not. Generates (card insert or card removal) interrupt when the bit of Normal Interrupt Status Enable Register is set. 0: No Card 1: Card Inserted	R/W
b5	hostctrl1_extdatawidth	Extended Data Transfer Width This bit controls 8-bit bus width mode for embedded device. Support of this function is indicated in 8-bit Support for Embedded Device in the Capabilities Register. If a device supports 8-bit bus mode, this bit may be set to 1. If this bit is 0, bus width is controlled by Data Transfer Width in the Host Control 1 Register. 0: Bus width is selected by Data Transfer Width bit 1: 8-bit bus width	R/W
b4, b3	hostctrl1_dmaselect	DMA Select One of supported DMA modes can be selected. The host driver shall check support of DMA modes by referring the Capabilities Register. 00b: SDMA 01b: 32-bit ADMA1 address 10b: 32-bit ADMA2 address 11b: 64-bit ADMA2 Address	R/W
b2	hostctrl1_highspeedena	High Speed Enable This bit is optional. Before setting this bit, the HD shall check the High Speed Support in the Capabilities Register. If this bit is set to 0 (default), the HC outputs CMD line and DAT lines at the falling edge of the SD clock (up to 25 MHz/20 MHz for MMC). If this bit is set to 1, the HC outputs CMD line and DAT lines at the rising edge of the SD clock (up to 50 MHz for SD/MMC). If Preset Value Enable in the Host Control 2 Register is set to 1, HD needs to reset SD Clock Enable before changing this field to avoid generating clock glitches. After setting this field, the HD sets SD Clock Enable again. 0: Normal Speed Mode 1: High Speed Mode	R/W
b1	hostctrl1_datawidth	Data Transfer Width (SD1 or SD4) This bit selects the data width of the HC. The HD shall select it to match the data width of the SD card. 0: 1-bit mode 1: 4-bit mode	R/W

Table 9.14 reg_hostcontrol1 Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b0	hostctrl1_ledcontrol	LED Control This bit is used to caution the user not to remove the card while the SD card is being accessed. If the software is going to issue multiple SD commands, this bit can be set during all transactions. It is not necessary to change for each transaction. 0: LED Off 1: LED On	R/W

9.4.13 reg_powercontrol — Power Control Register

This register is used to program the SD Bus power and voltage level.

Address: 4010 0029h (SDIO1)
4010 1029h (SDIO2)

Bit	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	emmc_hwreset	pwrctrl_sdbusvoltage			pwrctrl_sdbuspower
Value after reset	X	X	X	0	0	0	0	0

Table 9.15 reg_powercontrol Register Contents

Bit Position	Bit Name	Function	R/W
b7 to b5	Reserved		R
b4	emmc_hwreset	Hardware Reset Hardware reset signal is generated for eMMC card when this bit is set. Not used in this LSI.	R/W
b3 to b1	pwrctrl_sdbusvoltage	SD Bus Voltage Select By setting these bits, the HD selects the voltage level for the SD card. Before setting this register, the HD shall check the voltage support bits in the Capabilities Register. If an unsupported voltage is selected, the Host System shall not supply SD bus voltage. 111b: 3.3 V (Flattop.) Others: Reserved	R/W
b0	pwrctrl_sdbuspower	SD Bus Power Before setting this bit, the SD host driver shall set SD Bus Voltage Select. If the HC detects the No Card State, this bit shall be cleared. 0: Power Off 1: Power On This bit can be set to 1 only when the sdhccarddet_inserted_dsync bit in reg_presentstate register is 1.	R/W

9.4.14 reg_blockgapcontrol — Block Gap Control Register

This register is used to program the block gap request, read wait control and interrupt at block gap.

Address: 4010 002Ah (SDIO1)
4010 102Ah (SDIO2)

Bit	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	blkgapctrl_spimode	blkgapctrl_interrupt	blkgapctrl_rdwaitctrl	blkgapctrl_continue	blkgapctrl_stopatblkgap
Value after reset	1	0	0	0	0	0	0	0

Table 9.16 reg_blockgapcontrol Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b7 to b5	Reserved	Keep the initial value.	R/W
b4	blkgapctrl_spimode	SPI Mode Enable Bit. 0: SD Mode 1: SPI Mode	R/W
b3	blkgapctrl_interrupt	Interrupt At Block Gap This bit is valid only in 4-bit mode of the SDIO card and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer. If the SD card cannot signal an interrupt during a multiple block transfer, this bit should be set to 0. When the HD detects an SD card insertion, it shall set this bit according to the CCCR of the SDIO card.	R/W
b2	blkgapctrl_rdwaitctrl	Read Wait Control The read wait function is optional for SDIO cards. If the card supports read wait, set this bit to enable use of the read wait protocol to stop read data using DAT[2] line. Otherwise the HC has to stop the SD clock to hold read data, which restricts commands generation. When the HD detects an SD card insertion, it shall set this bit according to the CCCR of the SDIO card. If the card does not support read wait, this bit shall never be set to 1 otherwise DAT line conflict may occur. If this bit is set to 0, Suspend/Resume cannot be supported. 0: Disable Read Wait Control 1: Enable Read Wait Control	R/W
b1	blkgapctrl_continue	Continue Request This bit is used to restart a transaction which was stopped using the Stop At Block Gap Request. To cancel stop at the block gap, set Stop At block Gap Request to 0 and set this bit to restart the transfer. The HC automatically clears this bit in either of the following cases: 1) In the case of a read transaction, the DAT Line Active changes from 0 to 1 as a read transaction restarts. 2) In the case of a write transaction, the Write transfer active changes from 0 to 1 as the write transaction restarts. Therefore, it is not necessary for Host driver to set this bit to 0. If Stop At Block Gap Request is set to 1, any write to this bit is ignored. 0: Ignore 1: Restart	R/W

Table 9.16 reg_blockgapcontrol Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b0	blkgapctrl_stopatblkgap	Stop At Block Gap Request This bit is used to stop executing a transaction at the next block gap for non-DMA, SDMA and ADMA transfers. Until the transfer complete is set to 1, indicating a transfer completion the HD shall leave this bit set to 1. Clearing both the Stop At Block Gap Request and Continue Request shall not cause the transaction to restart. Read Wait is used to stop the read transaction at the block gap. The HC shall honor the Stop At Block Gap Request for write transfers, but for read transfers it requires that the SD card support Read Wait. Therefore, the HD shall not set this bit during read transfers unless the SD card supports Read Wait and the Read Wait Control has been set to 1. In case of write transfers in which the HD writes data to the Buffer Data Port Register, the HD shall set this bit after all block data is written. If this bit is set to 1, the HD shall not write data to Buffer Data Port Register. This bit affects Read Transfer Active, Write Transfer Active, DAT line active and Command Inhibit (DAT) in the Present State Register. 0: Transfer 1: Stop	R/W

9.4.15 reg_wakeupcontrol — Wake-up Control Register

This register is used to program the wakeup functionality.

Address: 4010 002Bh (SDIO1)
4010 102Bh (SDIO2)

Bit	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	wkupctrl_cardremoval	wkupctrl_cardinsertion	wkupctrl_cardinterrupt
Value after reset	X	X	X	X	X	0	0	0

Table 9.17 reg_wakeupcontrol Register Contents

Bit Position	Bit Name	Function	R/W
b7 to b3	Reserved		R
b2	wkupctrl_cardremoval	Wakeup Event Enable On SD Card Removal This bit enables wakeup event via Card Removal assertion in the Normal Interrupt Status Register. FN_WUS (Wake up Support) in CIS does not affect this bit. 0: Disable 1: Enable	R/W
b1	wkupctrl_cardinsertion	Wakeup Event Enable On SD Card Insertion This bit enables wakeup event via Card Insertion assertion in the Normal Interrupt Status Register. FN_WUS (Wake up Support) in CIS does not affect this bit. 0: Disable 1: Enable	R/W
b0	wkupctrl_cardinterrupt	Wakeup Event Enable On Card Interrupt This bit enables wakeup event via Card Interrupt assertion in the Normal Interrupt Status Register. This bit can be set to 1 if FN_WUS (Wake Up Support) in CIS is set to 1. 0: Disable 1: Enable	R/W

9.4.16 reg_clockcontrol — Clock Control Register

This register is used to program the Clock frequency select, generator select, Clock enable, Internal Clock state fields.

Address: 4010 002Ch (SDIO1)
4010 102Ch (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	clkctrl_sdclkfreqsel								clkctrl_sdclkfreqsel_upperbits		clkctrl_clkgensel	—	—	clkctrl_sdclkena	sdhcclkgen_intclkstable_dsync	clkctrl_intclkena
Value after reset	0	0	0	0	0	0	0	0	0	0	0	X	X	0	0	0

Table 9.18 reg_clockcontrol Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b15 to b8	clkctrl_sdclkfreqsel	SDIO_CLK Frequency Select This field is used to select the frequency of the SDIO_CLK pin. The frequency is not programmed directly; rather this register holds the divisor of the corecfg_baseclkfreq in the Capabilities Register. Only the following settings are allowed. The length of divider is extended to 10 bits and all divider values shall be supported. The highest 2-bit of clock divider is assigned to clkctrl_sdclkfreqsel_upperbits of this register. 3FFh: 1/2046 Divided Clock N: 1/2N Divided Clock (Duty 50%) 002h: 1/4 Divided Clock 001h: 1/2 Divided Clock 000h: Base Clock (10 MHz to 50 MHz) Setting 000h specifies the highest frequency of the SD clock. The divider values can be calculated by the frequency that is defined by the corecfg_baseclkfreq in the Capabilities Register. The frequency of the SDIO_CLK is set by the following formula: Clock Frequency = (Base Clock) / divisor. Thus, choose the smallest possible divisor which results in a clock frequency that is less than or equal to the target frequency.	R/W
b7, b6	clkctrl_sdclkfreqsel_upperbits	This field is assigned to Highest 2-bit of clock divider in SDIO_CLK Frequency Select.	R/W
b5	clkctrl_clkgensel	Clock Generator Select The Programmable Clock Mode is not supported in this HC. This bit attribute is Read Only and zero is read. 0: Divided Clock Mode	R
b4, b3	Reserved		R
b2	clkctrl_sdclkena	SD Clock Enable The HC shall stop SDIO_CLK when writing this bit to 0. SDIO_CLK frequency Select can be changed when this bit is 0. Then, the HC shall maintain the same clock frequency until SDIO_CLK is stopped (Stop at SDIO_CLK = 0). If the HC detects the No Card state, this bit shall be cleared. 0: Disable 1: Enable This bit is cleared when the card cannot be detected, or when the pwrctrl_sdbuspower bit in reg_powercontrol register is cleared.	R/W

Table 9.18 reg_clockcontrol Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b1	sdhcclkgen_intclkstable_dsync	Internal Clock Stable This bit is set to 1 when SD clock is stable after writing to Internal Clock Enable in this register to 1. The SD Host Driver shall wait to set SD Clock Enable until this bit is set to 1. ***Note*)** This is useful when using PLL for a clock oscillator that requires setup time. 0: Not Ready 1: Ready	R
b0	clkctrl_intclkena	Internal Clock Enable This bit is set to 0 when the HD is not using the HC or the HC awaits a wakeup event. The HC should stop its internal clock to go very low power state. Still, registers shall be able to be read and written. Clock starts to oscillate when this bit is set to 1. When clock oscillation is stable, the HC shall set Internal Clock Stable in this register to 1. This bit shall not affect card detection. 0: Stop 1: Oscillate	R/W

9.4.17 reg_timeoutcontrol — Timeout Control Register

The register sets the Data Timeout counter value.

Address: 4010 002Eh (SDIO1)
4010 102Eh (SDIO2)

Bit	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	timeout_ctrvalue			
Value after reset	X	X	X	X	0	0	0	0

Table 9.19 reg_timeoutcontrol Register Contents

Bit Position	Bit Name	Function	R/W
b7 to b4	Reserved		R
b3 to b0	timeout_ctrvalue	Data Timeout Counter Value This value determines the interval by which DAT line time-outs are detected. Refer to the Data Time-out Error in the Error Interrupt Status Register for information on factors that dictate time-out generation. Timeout frequency will be generated by dividing the base clock (TMCLK) with this setting. The base clock (TMCLK) is defined by corecfg_timeoutclkunit bit in Capabilities Register. When setting this register, prevent inadvertent time-out events by clearing the Data Time-out Error Status Enable (in the Error Interrupt Status Enable Register). 1111b: Reserved 1110b: TMCLK × 2^{27} ... ... 0001b: TMCLK × 2^{14} 0000b: TMCLK × 2^{13}	R/W

9.4.18 reg_softwarereset — Software Reset Register

This register is used to program the software reset for data, command and for all. A reset pulse is generated when writing 1 to each bit of this register. After completing the reset, the HC shall clear each bit. Because it takes some time to complete software reset, the HD shall confirm that these bits are 0.

Address: 4010 002Fh (SDIO1)
4010 102Fh (SDIO2)

Bit	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	swreset_for_dat	swreset_for_cmd	swreset_for_all
Value after reset	X	X	X	X	X	0	0	0

Table 9.20 reg_softwarereset Register Contents

Bit Position	Bit Name	Function	R/W
b7 to b3	Reserved		R
b2	swreset_for_dat	Software Reset for DAT Line Only part of data circuit is reset. The following registers and bits are cleared by this bit: • Buffer Data Port Register: Buffer is cleared and initialized. • Present State Register: Buffer read Enable Buffer write Enable Read Transfer Active Write Transfer Active DAT Line Active Command Inhibit (DAT) • Block Gap Control Register: Continue Request Stop At Block Gap Request • Normal Interrupt Status Register: Buffer Read Ready Buffer Write Ready Block Gap Event Transfer Complete 0: Work 1: Reset	R/W
b1	swreset_for_cmd	Software Reset for CMD Line Only part of command circuit is reset. The following registers and bits are cleared by this bit: Present State Register: Command Inhibit (CMD). Normal Interrupt Status Register: Command Complete. 0: Work 1: Reset	R/W
b0	swreset_for_all	Software Reset for All This reset affects the entire HC except for the card detection circuit. Register bits are cleared to 0. During its initialization, the HD shall set this bit to 1 to reset the HC. The HC shall reset this bit to 0. If this bit is set to 1, the SD card shall reset itself and must be re-initialized by the HD. 0: Work 1: Reset	R/W

9.4.19 reg_normalintrsts — Normal Interrupt Status Register

This register gives the status of all the interrupts.

The Normal Interrupt Status Enable Register affects read of this register, but Normal Interrupt Signal Enable Register does not affect these reads. An Interrupt is generated when the Normal Interrupt Signal Enable is enabled and at least one of the status bits is set to 1.

Address: 4010 0030h (SDIO1)
4010 1030h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	reg_errorintrsts	—	—	—	—	—	—	normalintrsts_cardintsts	normalintrsts_cardremsts	normalintrsts_cardinssts	normalintrsts_bufrdready	normalintrsts_bufwrready	normalintrsts_dmainterrupt	normalintrsts_blkgapevent	normalintrsts_xfercomplete	normalintrsts_cmdcomplete
Value after reset	0	X	X	X	X	X	X	0	0	0	0	0	0	0	0	0

Table 9.21 reg_normalintrsts Register Contents (1/3)

Bit Position	Bit Name	Function	R/W
b15	reg_errorintrsts	Error Interrupt If any of the bits in the Error Interrupt Status Register are set, then this bit is set. Therefore, the HD can test for an error by checking this bit first. 0: No error 1: Error	R
b14 to b9	Reserved		R
b8	normalintrsts_cardintsts	Card Interrupt Writing this bit to 1 does not clear this bit. It is cleared by resetting the SD card interrupt factor. In 1-bit mode, the HC shall detect the Card Interrupt without SD Clock to support wakeup. In 4-bit mode, the card interrupt signal is sampled during the interrupt cycle, so there are some sample delays between the interrupt signal from the card and the interrupt to the Host system. When this status has been set and the HD needs to start this interrupt service, Card Interrupt Status Enable in the Normal Interrupt Status Enable Register shall be set to 0 in order to clear the card interrupt statuses latched in the HC and stop driving the Host System. After completion of the card interrupt service (the reset factor in the SD card and the interrupt signal may not be asserted), set Card Interrupt Status Enable to 1 and start sampling the interrupt signal again. Interrupt detected by DAT[1] is supported when there is a card per slot. 0: No Card Interrupt 1: Generate Card Interrupt	R
b7	normalintrsts_cardremsts	Card Removal This status is set if the Card Inserted in the Present State Register changes from 1 to 0. When the HD writes this bit to 1 to clear this status, the status of the Card Inserted in the Present State Register should be confirmed. Because the card detect may possibly be changed when the HD clear this bit an Interrupt event may not be generated. 0: Card State Stable or Debouncing 1: Card Removed	R/W
b6	normalintrsts_cardinssts	Card Insertion This status is set if the Card Inserted in the Present State Register changes from 0 to 1. When the HD writes this bit to 1 to clear this status, the status of the Card Inserted in the Present State Register should be confirmed. Because the card detect may possibly be changed when the HD clear this bit an Interrupt event may not be generated. 0: Card State Stable or Debouncing 1: Card Inserted	R/W

Table 9.21 reg_normalintrsts Register Contents (2/3)

Bit Position	Bit Name	Function	R/W
b5	normalintrsts_bufrdready	Buffer Read Ready This status is set if the Buffer Read Enable changes from 0 to 1. This bit is cleared by writing 1b. 0: Not ready to read buffer 1: Ready to read buffer.	R/W
b4	normalintrsts_bufwrready	Buffer Write Ready This status is set if the Buffer Write Enable changes from 0 to 1. This bit is cleared by writing 1b. 0: Not ready to write buffer 1: Ready to write buffer	R/W
b3	normalintrsts_dmainterrupt	DMA Interrupt This status is set if the HC detects the Host SDMA Buffer Boundary in the Block Size Register. This bit is cleared by writing 1b. 0: No DMA Interrupt 1: DMA Interrupt is generated	R/W
b2	normalintrsts_blkgapevent	Block Gap Event If the Stop At Block Gap Request in the Block Gap Control Register is set, this bit is set. Read Transaction: This bit is set at the falling edge of the DAT Line Active Status (When the transaction is stopped at SD Bus timing. The Read Wait must be supported in order to use this function). Write Transaction: This bit is set at the falling edge of Write Transfer Active Status (After getting CRC status at SD Bus timing). This bit is cleared by writing 1b. 0: No Block Gap Event 1: Transaction stopped at Block Gap	R/W
b1	normalintrsts_xfercomplete	Transfer Complete This bit is set when a read/write transaction is completed. Read Transaction: This bit is set at the falling edge of Read Transfer Active Status. There are two cases in which the Interrupt is generated. The first is when a data transfer is completed as specified by data length (After the last data has been read to the Host System). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request in the Block Gap Control Register (After valid data has been read to the Host System). Write Transaction: This bit is set at the falling edge of the DAT Line Active Status. There are two cases in which the Interrupt is generated. The first is when the last data is written to the card as specified by data length and Busy signal is released. The second is when data transfers are stopped at the block gap by setting Stop At Block Gap Request in the Block Gap Control Register and data transfers completed. (After valid data is written to the SD card and the busy signal is released). This bit is cleared by writing 1b. 0: No Data Transfer Complete 1: Data Transfer Complete ***Note*)** Transfer Complete has higher priority than Data Time-out Error. If both bits are set to 1, the data transfer can be considered complete.	R/W

Table 9.21 reg_normalintrsts Register Contents (3/3)

Bit Position	Bit Name	Function	R/W
b0	normalintrsts_cmdcomplete	Command Complete This bit is set when we get the end bit of the command response (Except Auto CMD12 and Auto CMD23). This bit is cleared by writing 1b. 0: No Command Complete 1: Command Complete ***Note*)** Command Time-out Error has higher priority than Command Complete. If both are set to 1, it can be considered that the response was not received correctly.	R/W

9.4.20 reg_errorintrsts — Error Interrupt Status Register

This register gives the status of the error interrupts.

Status defined in this register can be enabled by the Error Interrupt Status Enable Register, but not by the Error Interrupt Signal Enable Register. The interrupt is generated when the Error Interrupt Signal Enable is enabled and at least one of the statuses is set to 1.

Address: 4010 0032h (SDIO1)
4010 1032h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	errorintrsts_hosterror	—	—	errorintrsts_admaerror	errorintrsts_autocmderror	errorintrsts_currlimiterror	errorintrsts_dataendbiterror	errorintrsts_datacrcerror	errorintrsts_datatimeouterror	errorintrsts_cmdindexerror	errorintrsts_cmdendbiterror	errorintrsts_cmdcrcerror	errorintrsts_cmdtimeouterror
Value after reset	X	X	X	0	X	X	0	0	0	0	0	0	0	0	0	0

Table 9.22 reg_errorintrsts Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b15 to b13	Reserved		R
b12	errorintrsts_hosterror	Target Response Error Occurs when detecting Host ERROR. This bit is cleared by writing 1b. 0: No error 1: Error	R/W
b11, b10	Reserved		R
b9	errorintrsts_admaerror	ADMA Error This bit is set when the HC detects errors during ADMA based data transfer. The state of the ADMA at an error occurrence is saved in the ADMA Error Status Register. This bit is cleared by writing 1b. 0: No error 1: Error	R/W
b8	errorintrsts_autocmderror	Auto CMD Error Auto CMD12 and Auto CMD23 use this error status. This bit is set when detecting that one of the bits (0 to 4) in Auto CMD Error Status Register has changed from 0 to 1. In case of Auto CMD12, this bit is set to 1, not only when the errors in Auto CMD12 occur but also when Auto CMD12 is not executed due to the previous command error. This bit is cleared by writing 1b. 0: No error 1: Error	R/W
b7	errorintrsts_currlimiterror	Current Limit Error By setting the SD Bus Power bit in the Power Control Register, the HC is requested to supply power for the SD Bus. If the HC supports the Current Limit Function, it can be protected from an illegal card by stopping power supply to the card in which case this bit indicates a failure status. Reading 1 means the HC is not supplying power to SD card due to some failure. Reading 0 means that the HC is supplying power and no error has occurred. This bit shall always set to be 0, if the HC does not support this function. This bit is cleared by writing 1b. 0: No error 1: Power Fail	R/W
b6	errorintrsts_dataendbiterror	Data End Bit Error Occurs when detecting 0 at the end bit position of read data which uses the DAT line or the end bit position of the CRC status. This bit is cleared by writing 1b. 0: No error 1: Error	R/W

Table 9.22 reg_errorintrsts Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b5	errorintrsts_datacrcerror	Data CRC Error Occurs when detecting CRC error when transferring read data which uses the DAT line or when detecting the Write CRC Status having a value of other than “010b”. This bit is cleared by writing 1b. 0: No error 1: Error	R/W
b4	errorintrsts_datatimeouterror	Data Timeout Error Occurs when detecting one of following timeout conditions: 1. Busy Timeout for R1b, R5b type. 2. Busy Timeout after Write CRC status 3. Write CRC status Timeout 4. Read Data Timeout. This bit is cleared by writing 1b. 0: No Error 1: Timeout	R/W
b3	errorintrsts_cmdindexerror	Command Index Error Occurs if a Command Index error occurs in the Command Response. This bit is cleared by writing 1b. 0: No Error 1: Error	R/W
b2	errorintrsts_cmdendbiterror	Command End Bit Error Occurs when detecting that the end bit of a command response is 0. This bit is cleared by writing 1b. 0: No Error 1: End bit error generated	R/W
b1	errorintrsts_cmdcrcerror	Command CRC Error Command CRC Error is generated in two cases. 1. If a response is returned and the Command Time-out Error is set to 0, this bit is set to 1 when detecting a CRC error in the command response. 2. The HC detects a CMD line conflict by monitoring the CMD line when a command is issued. If the HC drives the CMD line to 1 level, but detects 0 level on the CMD line at the next SDIO_CLK edge, then the HC shall abort the command (Stop driving CMD line) and set this bit to 1. The Command Timeout Error shall also be set to 1 to distinguish CMD line conflict. This bit is cleared by writing 1b. 0: No Error 1: CRC Error	R/W
b0	errorintrsts_cmdtimeouterror	Command Timeout Error Occurs only if the no response is returned within 64 SDIO_CLK cycles from the end bit of the command. If the HC detects a CMD line conflict, in which case Command CRC Error shall also be set. This bit shall be set without waiting for 64 SDIO_CLK cycles because the command will be aborted by the HC. This bit is cleared by writing 1b. 0: No Error 1: Timeout	R/W

9.4.21 reg_normalintrstsena — Normal Interrupt Status Enable Register

This register is used to enable the Normal Interrupt Status Register fields.

Address: 4010 0034h (SDIO1)
4010 1034h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	normalintrsts_enableregbit15	—	—	—	—	—	—	sdhcregset_cardintstsena	sdhcregset_cardremstsena	sdhcregset_cardinsstsena	normalintrsts_enableregbit5	normalintrsts_enableregbit4	normalintrsts_enableregbit3	normalintrsts_enableregbit2	normalintrsts_enableregbit1	normalintrsts_enableregbit0
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.23 reg_normalintrstsena Register Contents

Bit Position	Bit Name	Function	R/W
b15	normalintrsts_enableregbit15	This bit is fixed to 0b. The HD shall control error Interrupts using the Error Interrupt Status Enable Register.	R
b14 to b9	Reserved	Keep the initial value.	R/W
b8	sdhcregset_cardintstsena	Card Interrupt Status Enable If this bit is set to 0, the HC shall clear interrupt request to the System. The Card Interrupt detection is stopped when this bit is cleared and restarted when this bit is set to 1. The HD may clear the Card Interrupt Status Enable before servicing the Card Interrupt and may set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts. 0: Masked 1: Enabled	R/W
b7	sdhcregset_cardremstsena	Card Removal Status Enable 0: Masked 1: Enabled	R/W
b6	sdhcregset_cardinsstsena	Card Insertion Status Enable 0: Masked 1: Enabled	R/W
b5	normalintrsts_enableregbit5	Buffer Read Ready Status Enable 0: Masked 1: Enabled	R/W
b4	normalintrsts_enableregbit4	Buffer Write Ready Status Enable 0: Masked 1: Enabled	R/W
b3	normalintrsts_enableregbit3	DMA Interrupt Status Enable 0: Masked 1: Enabled	R/W
b2	normalintrsts_enableregbit2	Block Gap Event Status Enable 0: Masked 1: Enabled	R/W
b1	normalintrsts_enableregbit1	Transfer Complete Status Enable 0: Masked 1: Enabled	R/W
b0	normalintrsts_enableregbit0	Command Complete Status Enable 0: Masked 1: Enabled	R/W

9.4.22 reg_errorintrstsena — Error Interrupt Status Enable Register

This register is used to enable the Error Interrupt Status Register fields.

Address: 4010 0036h (SDIO1)
4010 1036h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	errorintrsts_enableregbit12	—	—	errorintrsts_enableregbit9	errorintrsts_enableregbit8	errorintrsts_enableregbit7	errorintrsts_enableregbit6	errorintrsts_enableregbit5	errorintrsts_enableregbit4	errorintrsts_enableregbit3	errorintrsts_enableregbit2	errorintrsts_enableregbit1	errorintrsts_enableregbit0
Value after reset	X	0	0	0	X	0	0	0	0	0	0	0	0	0	0	0

Table 9.24 reg_errorintrstsena Register Contents

Bit Position	Bit Name	Function	R/W
b15	Reserved		R
b14 to b13	Reserved	Keep the initial value.	R/W
b12	errorintrsts_enableregbit12	Target Response Error Status Enable 0: Masked 1: Enabled	R/W
b11	Reserved		R
b10	Reserved	Keep the initial value.	R/W
b9	errorintrsts_enableregbit9	ADMA Error Status Enable 0: Masked 1: Enabled	R/W
b8	errorintrsts_enableregbit8	Auto CMD Error Status Enable 0: Masked 1: Enabled	R/W
b7	errorintrsts_enableregbit7	Current Limit Error Status Enable 0: Masked 1: Enabled	R/W
b6	errorintrsts_enableregbit6	Data End Bit Error Status Enable 0: Masked 1: Enabled	R/W
b5	errorintrsts_enableregbit5	Data CRC Error Status Enable 0: Masked 1: Enabled	R/W
b4	errorintrsts_enableregbit4	Data Timeout Error Status Enable 0: Masked 1: Enabled	R/W
b3	errorintrsts_enableregbit3	Command Index Error Status Enable 0: Masked 1: Enabled	R/W
b2	errorintrsts_enableregbit2	Command End Bit Error Status Enable 0: Masked 1: Enabled	R/W
b1	errorintrsts_enableregbit1	Command CRC Error Status Enable 0: Masked 1: Enabled	R/W
b0	errorintrsts_enableregbit0	Command Timeout Error Status Enable 0: Masked 1: Enabled	R/W

9.4.23 reg_normalintrsigena — Normal Interrupt Signal Enable Register

This register is used to select which interrupt status is indicated to the Host System as the interrupt. These status bits all share the same 1-bit interrupt line (SDIF_Int). Setting any of these bits to 1 enables interrupt generation.

Address: 4010 0038h (SDIO1)
4010 1038h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	normalintrsig_enableregbit15	—	—	—	—	—	—	normalintrsig_enableregbit8	normalintrsig_enableregbit7	normalintrsig_enableregbit6	normalintrsig_enableregbit5	normalintrsig_enableregbit4	normalintrsig_enableregbit3	normalintrsig_enableregbit2	normalintrsig_enableregbit1	normalintrsig_enableregbit0
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.25 reg_normalintrsigena Register Contents

Bit Position	Bit Name	Function	R/W
b15	normalintrsig_enableregbit15	This bit is fixed to 0b. The HD shall control error Interrupts using the Error Interrupt Signal Enable Register.	R
b14 to b9	Reserved	Keep the initial value.	R/W
b8	normalintrsig_enableregbit8	Card Interrupt Signal Enable 0: Masked 1: Enabled	R/W
b7	normalintrsig_enableregbit7	Card Removal Signal Enable 0: Masked 1: Enabled	R/W
b6	normalintrsig_enableregbit6	Card Insertion Signal Enable 0: Masked 1: Enabled	R/W
b5	normalintrsig_enableregbit5	Buffer Read Ready Signal Enable 0: Masked 1: Enabled	R/W
b4	normalintrsig_enableregbit4	Buffer Write Ready Signal Enable 0: Masked 1: Enabled	R/W
b3	normalintrsig_enableregbit3	DMA Interrupt Signal Enable 0: Masked 1: Enabled	R/W
b2	normalintrsig_enableregbit2	Block Gap Event Signal Enable 0: Masked 1: Enabled	R/W
b1	normalintrsig_enableregbit1	Transfer Complete Signal Enable 0: Masked 1: Enabled	R/W
b0	normalintrsig_enableregbit0	Command Complete Signal Enable 0: Masked 1: Enabled	R/W

9.4.24 reg_errorintrsigena — Error Interrupt Signal Enable Register

This register is used to select which interrupt status is indicated to the Host System as the Interrupt. These status bits all share the same 1-bit interrupt line (SDIF_Int). Setting any of these bits to 1 enables Interrupt generation.

Address: 4010 003Ah (SDIO1)
4010 103Ah (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	errorintrsig_enableregbit12	—	—	errorintrsig_enableregbit9	errorintrsig_enableregbit8	errorintrsig_enableregbit7	errorintrsig_enableregbit6	errorintrsig_enableregbit5	errorintrsig_enableregbit4	errorintrsig_enableregbit3	errorintrsig_enableregbit2	errorintrsig_enableregbit1	errorintrsig_enableregbit0
Value after reset	X	0	0	0	X	0	0	0	0	0	0	0	0	0	0	0

Table 9.26 reg_errorintrsigena Register Contents

Bit Position	Bit Name	Function	R/W
b15	Reserved		R
b14 to b13	Reserved	Keep the initial value.	R/W
b12	errorintrsig_enableregbit12	Target Response Error Signal Enable 0: Masked 1: Enabled	R/W
b11	Reserved		R
b10	Reserved	Keep the initial value.	R/W
b9	errorintrsig_enableregbit9	ADMA Error Signal Enable 0: Masked 1: Enabled	R/W
b8	errorintrsig_enableregbit8	Auto CMD Error Signal Enable 0: Masked 1: Enabled	R/W
b7	errorintrsig_enableregbit7	Current Limit Error Signal Enable 0: Masked 1: Enabled	R/W
b6	errorintrsig_enableregbit6	Data End Bit Error Signal Enable 0: Masked 1: Enabled	R/W
b5	errorintrsig_enableregbit5	Data CRC Error Signal Enable 0: Masked 1: Enabled	R/W
b4	errorintrsig_enableregbit4	Data Timeout Error Signal Enable 0: Masked 1: Enabled	R/W
b3	errorintrsig_enableregbit3	Command Index Error Signal Enable 0: Masked 1: Enabled	R/W
b2	errorintrsig_enableregbit2	Command End Bit Error Signal Enable 0: Masked 1: Enabled	R/W
b1	errorintrsig_enableregbit1	Command CRC Error Signal Enable 0: Masked 1: Enabled	R/W
b0	errorintrsig_enableregbit0	Command Timeout Error Signal Enable 0: Masked 1: Enabled	R/W

9.4.25 reg_autocmderrsts — Auto CMD Error Status Register

This register is used to indicate CMD12 response error of Auto CMD12 and CMD23 response error of Auto CMD 23.

Address: 4010 003Ch (SDIO1)
4010 103Ch (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	autocmderrsts_nexterror	—	—	autocmderrsts_indexerror	autocmderrsts_endbiterror	autocmderrsts_crcerror	autocmderrsts_timeouterror	autocmderrsts_notexecerror
Value after reset	X	X	X	X	X	X	X	X	0	X	X	0	0	0	0	0

Table 9.27 reg_autocmderrsts Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b8	Reserved		R
b7	autocmderrsts_nexterror	Command Not Issued By Auto CMD12 Error Setting this bit to 1 means CMD_wo_DAT is not executed due to an Auto CMD12 error bits (4 to 1) in this register. This bit is set to 0 when Auto CMD Error is generated by Auto CMD23. 0: No Error 1: Not Issued	R
b6, b5	Reserved		R
b4	autocmderrsts_indexerror	Auto CMD Index Error Occurs if the Command Index error occurs in response to a command. 0: No Error 1: Error	R
b3	autocmderrsts_endbiterror	Auto CMD End Bit Error Occurs when detecting that the end bit of command response is 0. 0: No Error 1: End Bit Error Generated	R
b2	autocmderrsts_crcerror	Auto CMD CRC Error Occurs when detecting a CRC error in the command response. 0: No Error 1: CRC Error Generated	R
b1	autocmderrsts_timeouterror	Auto CMD Timeout Error Occurs if the no response is returned within 64 SDIO_CLK cycles from the end bit of the command. If this bit is set to 1, the other error status bits (4 to 2) are meaningless. 0: No Error 1: Timeout	R
b0	autocmderrsts_notexecerror	Auto CMD12 Not Executed If memory multiple block data transfer is not started due to command error, this bit is not set because it is not necessary to issue Auto CMD12. Setting this bit to 1 means the HC cannot issue Auto CMD12 to stop memory multiple block transfer due to some error. If this bit is set to 1, other error status bits (4 to 1) are meaningless. This bit is set to 0 when Auto CMD Error is generated by Auto CMD23. 0: Executed 1: Not Executed	R

9.4.26 reg_hostcontrol2 — Host Control 2 Register

This register is used to program Asynchronous Interrupt Enable and Preset value enable.

Address: 4010 003Eh (SDIO1)
4010 103Eh (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	hostctrl2_presetvalueenable	hostctrl2_asynchintrenable	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 9.28 reg_hostcontrol2 Register Contents

Bit Position	Bit Name	Function	R/W
b15	hostctrl2_presetvalueenable	Preset Value Enable As the operating SDIO_CLK frequency depend on the Host System implementation, it is difficult to determine these parameters in the Standard HD. When Preset Value Enable is set to 1, this bit enables the functions defined in the Preset Value Registers. 0: SDIO_CLK is controlled by HD. 1: Automatic Selection by Preset Value are Enabled. If this bit is set to 0, SDIO_CLK Frequency Select and Clock Generator Select in the Clock Control Register are set by HD. If this bit is set to 1, SDIO_CLK Frequency Select and Clock Generator Select in the Clock Control Register are set by HC as specified in the Preset Value Registers.	R/W
b14	hostctrl2_asynchintrenable	Asynchronous Interrupt Enable This bit can be set to 1 if a card support asynchronous interrupt and Asynchronous Interrupt Support is set to 1 in the Capabilities Register. Asynchronous interrupt is effective when DAT[1] interrupt is used in 4-bit SD mode. If this bit is set to 1, the HD can stop the SDIO_CLK during asynchronous interrupt period to save power. During this period, the HC continues to deliver Card Interrupt to the host when it is asserted by the Card. 0: Disabled 1: Enabled	R/W
b13 to b8	Reserved		R
b7 to b0	Reserved	Keep the initial value.	R/W

9.4.27 reg_capabilities — Capabilities Register

This register provides the host driver with information specific to the host controller implementation.

“slottype” and “baseclkfreq” are defined by CFG_SDIO[m] register of System Control.

Address: 4010 0040h (SDIO1)
4010 1040h (SDIO2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	corecfg_slottype		corecfg_asynchintrsupport	corecfg_64bitsupport	—	corecfg_1p8voltsupport	corecfg_3p0voltsupport	corecfg_3p3voltsupport	corecfg_suspressupport	corecfg_sdmasupport	corecfg_highspeedsupport	—	corecfg_adma2support	corecfg_8bitsupport	corecfg_maxblklength	
Value after reset	X	X	1	0	0	1	1	1	1	1	1	0	1	1	0	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	corecfg_baseclkfreq								corecfg_timeoutclkunit	—	corecfg_timeoutclkfreq					
Value after reset	X	X	X	X	X	X	X	X	1	0	0	0	0	0	0	1

Table 9.29 reg_capabilities Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31, b30	corecfg_slottype	This slot type reflects SLOTTYPE bits of CFG_SDIO[m] (m = 1 or 2) register of System Control. These bits should be set to an appropriate value since it is also used to determine the Card Detection time. 00b: Removable Card Slot 01b: Embedded Slot for One Device Others: Reserved	R
b29	corecfg_asynchintrsupport	Refer to SDIO Specification Version 3.00 about asynchronous interrupt. 0: Asynchronous Interrupt Not Supported 1: Asynchronous Interrupt Supported	R
b28	corecfg_64bitsupport	This bit indicates whether the HC supports 64-bit System Bus. 0: Does not support 64-bit system address 1: supports 64-bit system address	R
b27	Reserved		R
b26	corecfg_1p8voltsupport	This bit indicates whether the HC supports 1.8 V. 0: 1.8 V Not supported 1: 1.8 V supported (Not available in this LSI.)	R
b25	corecfg_3p0voltsupport	This bit indicates whether the HC supports 3.0 V. 0: 3.0 V Not supported 1: 3.0 V supported (Not available in this LSI.)	R
b24	corecfg_3p3voltsupport	This bit indicates whether the HC supports 3.3 V. 0: 3.3 V Not supported 1: 3.3 V supported	R
b23	corecfg_suspressupport	This bit indicates whether the HC supports Suspend/Resume functionality. If this bit is 0, the Suspend and Resume mechanism are not supported and the HD shall not issue either Suspend/Resume commands. 0: Not supported 1: supported	R
b22	corecfg_sdmasupport	This bit indicates whether the HC is capable of using DMA to transfer data between system memory and the HC directly. 0: SDMA Not Supported 1: SDMA Supported	R

Table 9.29 reg_capabilities Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b21	corecfg_highspeedsupport	This bit indicates whether the HC and the Host System support High Speed mode and they can supply SD clock frequency from 25 MHz to 50 MHz (for SD)/20 MHz to 50 MHz (for eMMC). 0: High Speed Not Supported 1: High Speed Supported	R
b20	Reserved		R
b19	corecfg_adma2support	0: ADMA2 Not Supported 1: ADMA2 Supported	R
b18	corecfg_8bitsupport	This bit indicates whether the HC is capable of using 8-bit bus width mode. 0: Extended Media Bus Not Supported 1: Extended Media Bus Supported	R
b17, b16	corecfg_maxblklength	This value indicates the maximum block size that the HD can read and write to the buffer in the HC. The buffer shall transfer this block size without wait cycles. 00b: 512 byte 01b: 1024 byte 10b: 2048 byte 11b: Reserved	R
b15 to b8	corecfg_baseclkfreq	This value indicates the base clock frequency for SDIO_CLK. (CFG_SDIO[m] register of System Control set is defined per slot.) Unit values are 1 MHz. The supported clock range is 10 MHz to 50 MHz. 32h: 50 MHz 02h: 2 MHz 01h: 1 MHz If the real frequency is 16.5 MHz, the lager value shall be set 11h (17 MHz) because the HD uses this value to calculate the clock divider value (Refer to the SDIO_CLK Frequency Select in the Clock Control Register.) and it shall not exceed upper limit of the SD clock frequency.	R
b7	corecfg_timeoutclkunit	This bit shows the unit of base clock (TMCLK) frequency used to detect Data Timeout Error. 0: kHz 1: MHz	R
b6	Reserved		R
b5 to b0	corecfg_timeoutclkfreq	This bit shows the base clock frequency used to detect Data Timeout Error. 000000b: Get Information via another method Others: 1 kHz to 63 kHz or 1 MHz to 63 MHz	R

9.4.28 reg_capabilities_cont — Capabilities Register (Continue)

This register provides the host driver with information specific to the host controller implementation.

Address: 4010 0044h (SDIO1)
4010 1044h (SDIO2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	corecfg_spiblkmode	corecfg_spisupport	corecfg_clockmultiplier							
Value after reset	X	X	X	X	X	X	0	1	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	X	0	0	0	0	X	0	0	0	X	0	0	0

Table 9.30 reg_capabilities_cont Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b26	Reserved		R
b25	corecfg_spiblkmode	This field indicates whether SPI Block Mode is supported or not. 0: Not Supported 1: Supported	R
b24	corecfg_spisupport	This field indicates whether SPI Mode is supported or not. 0: Not Supported 1: Supported	R
b23 to b16	corecfg_clockmultiplier	This field indicates clock multiplier value of programmable clock generator. Setting 00h means that HC does not support programmable clock generator. 00h: Clock Multiplier is Not Supported.	R
b15 to b0	Reserved		R

9.4.29 reg_maxcurrentcap — Maximum Current Capabilities Register

This register indicates maximum current capability for each voltage.

Address: 4010 0048h (SDIO1)
4010 1048h (SDIO2)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	corecfg_maxcurrent1p8v							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	corecfg_maxcurrent3p0v								corecfg_maxcurrent3p3v							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.31 reg_maxcurrentcap Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved		R
b23 to b16	corecfg_maxcurrent1p8v	Maximum Current for 1.8 V Not available in this LSI.	R
b15 to b8	corecfg_maxcurrent3p0v	Maximum Current for 3.0 V Not available in this LSI.	R
b7 to b0	corecfg_maxcurrent3p3v	Maximum Current for 3.3 V Not available in this LSI. See Note 1 in *Table 11.3, Current* in *the RZ/N1D Group, RZ/N1S Group, RZ/N1L Group User's Manual: System Introduction, Multiplexing, Electrical and Mechanical Information*.	R

9.4.30 reg_ForceEventforAUTOCMDErrorStatus — Force Event for Auto CMD Error Status Register

This register is not physically implemented, rather used for setting any bit of Auto CMD Error Status Register, which is difficult to set intentionally.

Address: 4010 0050h (SDIO1)
4010 1050h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	forcecmdnotissuedbyautocmd12err	—	—	forceautocmdindexerr	forceautocmdendbiterr	forceautocmdcrcerr	forceautocmdtimeouterr	forceautocmdnotexec
Value after reset	X	X	X	X	X	X	X	X	0	X	X	0	0	0	0	0

Table 9.32 reg_ForceEventforAUTOCMDErrorStatus Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b8	Reserved		R
b7	forcecmdnotissuedby autocmd12err	Force Event for Command Not Issued by Auto CMD12 Error. 1: Interrupt is generated 0: No Interrupt	W
b6, b5	Reserved		R
b4	forceautocmdindexerr	Force Event for Auto CMD Index Error. 1: Interrupt is generated 0: No Interrupt	W
b3	forceautocmdendbiterr	Force Event for Auto CMD End Bit Error. 1: Interrupt is generated 0: No Interrupt	W
b2	forceautocmdcrcerr	Force Event for Auto CMD CRC Error. 1: Interrupt is generated 0: No Interrupt	W
b1	forceautocmdtimeouterr	Force Event for Auto CMD Timeout Error. 1: Interrupt is generated 0: No Interrupt	W
b0	forceautocmdnotexec	Force Event for Auto CMD12 Not Executed. 1: Interrupt is generated 0: No Interrupt	W

9.4.31 reg_forceeventforerrintsts — Force Event for Error Interrupt Status Register

This register is not physically implemented, rather used for setting any bit of Error Interrupt Status Register, which is difficult to set intentionally.

Address: 4010 0052h (SDIO1)
4010 1052h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	forceadmaerr	forceautocmderr	forcecurrlimerr	forcedatendbiterr	forcedatcrcerr	forcedattimeouterr	forcecmdindexerr	forcecmdendbiterr	forcecmdcrcerr	forcecmdtimeouterr
Value after reset	X	X	X	X	X	X	0	0	0	0	0	0	0	0	0	0

Table 9.33 reg_forceeventforerrintsts Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b10	Reserved		R
b9	forceadmaerr	Force Event for ADMA Error. 1: Interrupt is generated 0: No Interrupt.	W
b8	forceautocmderr	Force Event for Auto CMD Error. 1: Interrupt is generated 0: No Interrupt.	W
b7	forcecurrlimerr	Force Event for Current Limit Error. 1: Interrupt is generated 0: No Interrupt.	W
b6	forcedatendbiterr	Force Event for Data End Bit Error. 1: Interrupt is generated 0: No Interrupt.	W
b5	forcedatcrcerr	Force Event for Data CRC Error. 1: Interrupt is generated 0: No Interrupt.	W
b4	forcedattimeouterr	Force Event for Data Timeout Error. 1: Interrupt is generated 0: No Interrupt.	W
b3	forcecmdindexerr	Force Event for Command Index Error 1: Interrupt is generated 0: No Interrupt.	W
b2	forcecmdendbiterr	Force Event for Command End Bit Error. 1: Interrupt is generated 0: No Interrupt.	W
b1	forcecmdcrcerr	Force Event for Command CRC Error. 1: Interrupt is generated 0: No Interrupt.	W
b0	forcecmdtimeouterr	Force Event for Command Timeout Error. 1: Interrupt is generated 0: No Interrupt.	W

9.4.32 reg_admaerrsts — ADMA Error Status Register

When the ADMA Error interrupt occur, this register holds the ADMA State in ADMA Error States field and ADMA System Address holds address around the error descriptor.

Address: 4010 0054h (SDIO1)
4010 1054h (SDIO2)

Bit	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	admaerrsts_admalenmismatcherr	admaerrsts_admaerrorstate	
Value after reset	X	X	X	X	X	0	0	0

Table 9.34 reg_admaerrsts Register Contents

Bit Position	Bit Name	Function	R/W
b7 to b3	Reserved		R
b2	admaerrsts_admalenmismatcherr	ADMA Length Mismatch Error This error occurs in the following 2 cases. • While Block Count Enable being set, the total data length specified by the Descriptor table is different from that specified by the Block Count and Block Length. • Total data length cannot be divided by the block length. 1: Error 0: No Error	R
b1, b0	admaerrsts_admaerrorstate	ADMA Error State This field indicates the state of ADMA when error is occurred during ADMA data transfer. This field never indicates “10b” because ADMA never stops in this state.	R

Bit 1 to 0	ADMA Error State When Error Occurred	Contents of SYS_SDR Register
00b	ST_STOP (Stop DMA)	Points to next of the error descriptor
01b	ST_FDS (Fetch Descriptor)	Points to the error descriptor
10b	Never set this state	(Not Used).
11b	ST_TFR (Transfer Data)	Points to the next of the error descriptor

9.4.33 reg_admasysaddr0 — ADMA System Address Register Low

This register contains the Lower 16-bit of physical system memory address used for ADMA data transfer.

Address: 4010 0058h (SDIO1)
4010 1058h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	adma_sysaddress0															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.35 reg_admasysaddr0 Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b0	adma_sysaddress0	This register holds byte address of executing command of the Descriptor table. 32-bit Address Descriptor uses system address 32-bit of this register. At the start of ADMA, the HD shall set start address of the Descriptor table. The ADMA increments this register address, which points to next line, when every fetching a Descriptor line. When the ADMA Error Interrupt is generated, this register shall hold valid Descriptor address depending on the ADMA state. The HD shall program Descriptor Table on 32-bit boundary and set 32-bit boundary address to this register. ADMA2 ignores lower 2-bit of this register and assumes it to be 00b. 32-bit Address ADMA	R/W

Register Value (reg_admasysaddr1, reg_admasysaddr0)	32-bit System Address
0000h, 0000h	00000000h
0000h, 0004h	00000004h
...	...
FFFFh, FFFCh	FFFFFFFCh

9.4.34 reg_admasysaddr1 — ADMA System Address Register High

This register contains the Higher 16-bit of physical system memory address used for ADMA data transfer.

Address: 4010 005Ah (SDIO1)
4010 105Ah (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	adma_sysaddress1															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 9.36 reg_admasysaddr1 Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b0	adma_sysaddress1	This register holds byte address of executing command of the Descriptor table. Details of this field is described in reg_admasysaddr0.	R/W

9.4.35 reg_presetvalue0 — Preset Value Register for Initialization

This register is used to read the SDIO_CLK Frequency Select Value, Clock Generator Select Value for Initialization.

Address: 4010 0060h (SDIO1)
4010 1060h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	ClockGeneratorSelectValue	SDCLKFrequencySelectValue									
Value after reset	X	X	X	X	X	0	0	0	0	0	0	0	0	0	0	0

Table 9.37 reg_presetvalue0 Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b11	Reserved		R
b10	ClockGeneratorSelectValue	This bit is effective when HC supports programmable clock generator. 0: Host Controller Ver2.00 Compatible Clock Generator 1: Programmable Clock Generator	R
b9 to b0	SDCLKFrequencySelectValue	10-bit preset value to set SDIO_CLK Frequency Select in the Clock Control Register is described by a host system.	R

9.4.36 reg_presetvalue1 — Preset Value Register for Default Speed

This register is used to read the SDIO_CLK Frequency Select Value, Clock Generator Select Value for Default Speed.

Address: 4010 0062h (SDIO1)
4010 1062h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	ClockGeneratorSelectValue	SDCLKFrequencySelectValue									
Value after reset	X	X	X	X	X	0	0	0	0	0	0	0	0	0	0	0

Table 9.38 reg_presetvalue1 Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b11	Reserved		R
b10	ClockGeneratorSelectValue	This bit is effective when HC supports programmable clock generator. 0: Host Controller Ver2.00 Compatible Clock Generator 1: Programmable Clock Generator	R
b9 to b0	SDCLKFrequencySelectValue	10-bit preset value to set SDIO_CLK Frequency Select in the Clock Control Register is described by a host system.	R

9.4.37 reg_presetvalue2 — Preset Value Register for High Speed

This register is used to read the SDIO_CLK Frequency Select Value, Clock Generator Select Value for High Speed.

Address: 4010 0064h (SDIO1)
4010 1064h (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	ClockGeneratorSelectValue	SDCLKFrequencySelectValue									
Value after reset	X	X	X	X	X	0	0	0	0	0	0	0	0	0	0	0

Table 9.39 reg_presetvalue2 Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b11	Reserved		R
b10	ClockGeneratorSelectValue	This bit is effective when HC supports programmable clock generator. 0: Host Controller Ver2.00 Compatible Clock Generator 1: Programmable Clock Generator	R
b9 to b0	SDCLKFrequencySelectValue	10-bit preset value to set SDIO_CLK Frequency Select in the Clock Control Register is described by a host system.	R

9.4.38 reg_slotintrsts — Slot Interrupt Status Register

This register is used to read the interrupt signal for each slot.

Address: 4010 00FCh (SDIO1)
4010 10FCh (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	sdhchostif_slotintrsts
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

Table 9.40 reg_slotintrsts Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b1	Reserved		R
b0	sdhchostif_slotintrsts	This status bit indicates the logical OR of Interrupt signal and Wakeup signal for slot.	R

9.4.39 reg_hostcontrollerver — Host Controller Version Register

This register is used to read the vendor version number and specification version number.

Address: 4010 00FEh (SDIO1)
4010 10FEh (SDIO2)

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SDHC_VENVERNUM								SpecificationVersionNumber							
Value after reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0

Table 9.41 reg_hostcontrollerver Register Contents

Bit Position	Bit Name	Function	R/W
b15 to b8	SDHC_VENVERNUM	The Vendor Version Number is set to 10h (1.0).	R
b7 to b0	SpecificationVersionNumber	The HC Version Number is set to 02h. (SD Host Specification Version 3.00)	R

9.5 Programming the SDIO

This chapter gives the details about various data transfer protocols and how to program the protocols.

- Non-DMA data transaction
- DMA data transaction
- ADMA data transaction
- Abort transaction (Sync and Async)

9.5.1 Non-DMA Transaction

The sequence for Not Using DMA [non-DMA] is shown below.

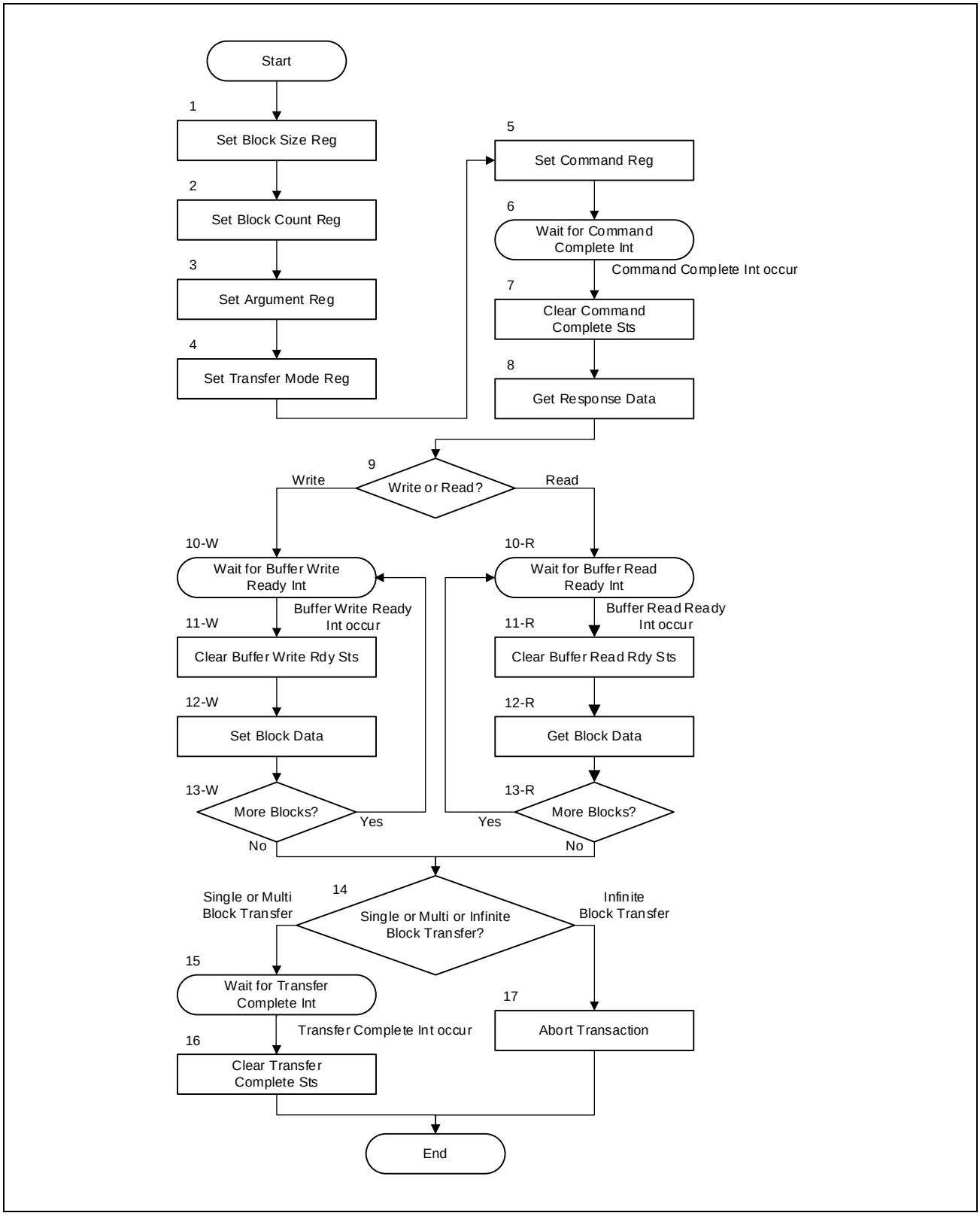


Figure 9.2 Data Transfer Using DAT Line Sequence (Not Using DMA)

Non-DMA Transaction Sequence

Table 9.42 Non-DMA Transaction

Step	Description
1	Set the value corresponding to the executed data byte length of one block to Block Size Register.
2	Set the value corresponding to the executed data block count to Block Count Register.
3	Set the value corresponding to the issued command to Argument Register.
4	Set the value to Multi/Single Block Select and Block Count Enable. Set the value corresponding to the issued command to Data Transfer Direction, Auto CMD12 Enable and DMA Enable.
5	Set the value corresponding to the issued command to Command Register. When writing the upper byte of Command Register, SD command is issued.
6	Wait for the Command Complete Interrupt
7	Write 1 to the Command Complete in the Normal Interrupt Status Register for clearing this bit.
8	Read Response Register and get necessary information in accordance with the issued command.
9	In the Case Where this sequence is for write to a card, go to step (10-W). In case of read from a card, go to step (10-R).
10-W	Wait for Buffer Write Ready Interrupt. **Non-DMA Write Transfer** On receiving the Buffer Write Ready interrupt the CPU will act as a master and start transferring the data via Buffer Data Port Register (fifo_1). Transmitter starts sending the data in SD bus when a block of data is ready in fifo_1. While transmitting the data in SD bus the buffer write ready interrupt is sent to the CPU for the second block of data. The CPU will act as a master and start sending the second block of data via Buffer Data Port Register to fifo_2. Buffer write ready interrupt will be asserted only when a FIFO is empty to receive a block of data.
11-W	Write 1 to the Buffer Write Ready in the Normal Interrupt Status Register for clearing this bit.
12-W	Write block data (in according to the number of bytes specified at the step (1)) to Buffer Data Port Register.
13-W	Repeat until all blocks are sent and then go to step (14).
10-R	**Non-DMA Read Transfer** Buffer Read Ready interrupt is asserted whenever a block of data is ready in one of the FIFO's. On receiving the Buffer Read Ready interrupt the CPU will act as a master and start reading the data via Buffer Data Port Register (fifo_1). Receiver start reading the data from SD bus only when a FIFO is empty to receive a block of data. When both the FIFO's are full the host controller will stop the data coming from the card through read wait mechanism (if card supports read wait) or through clock stopping. Wait for Buffer Read Ready Interrupt
11-R	Write 1 to the Buffer Read Ready in the Normal Interrupt Status Register for clearing this bit.
12-R	Read block data (in according to the number of bytes specified at the step (1)) from the Buffer Data Port Register.
13-R	Repeat until all blocks are received and then go to step (14).
14	If this sequence is for Single or Multiple Block Transfer, go to step (15). In case of Infinite Block Transfer, go to step (17).
15	Wait for Transfer Complete Interrupt.
16	Write 1 to the Transfer Complete in the Normal Interrupt Status Register for clearing this bit.
17	Perform the sequence for Abort Transaction.

CAUTION

Note: Step (1) and (2) can be executed at same time. Step (4) and (5) can be executed at same time.

9.5.2 DMA Transaction

The burst types like 8-beat incrementing burst or 4-beat incrementing burst or Single transfer is used to transfer or receive the data from the system memory mainly to avoid the hold of the Host/System bus by the master for a longer time. The sequence for Using DMA is shown below.

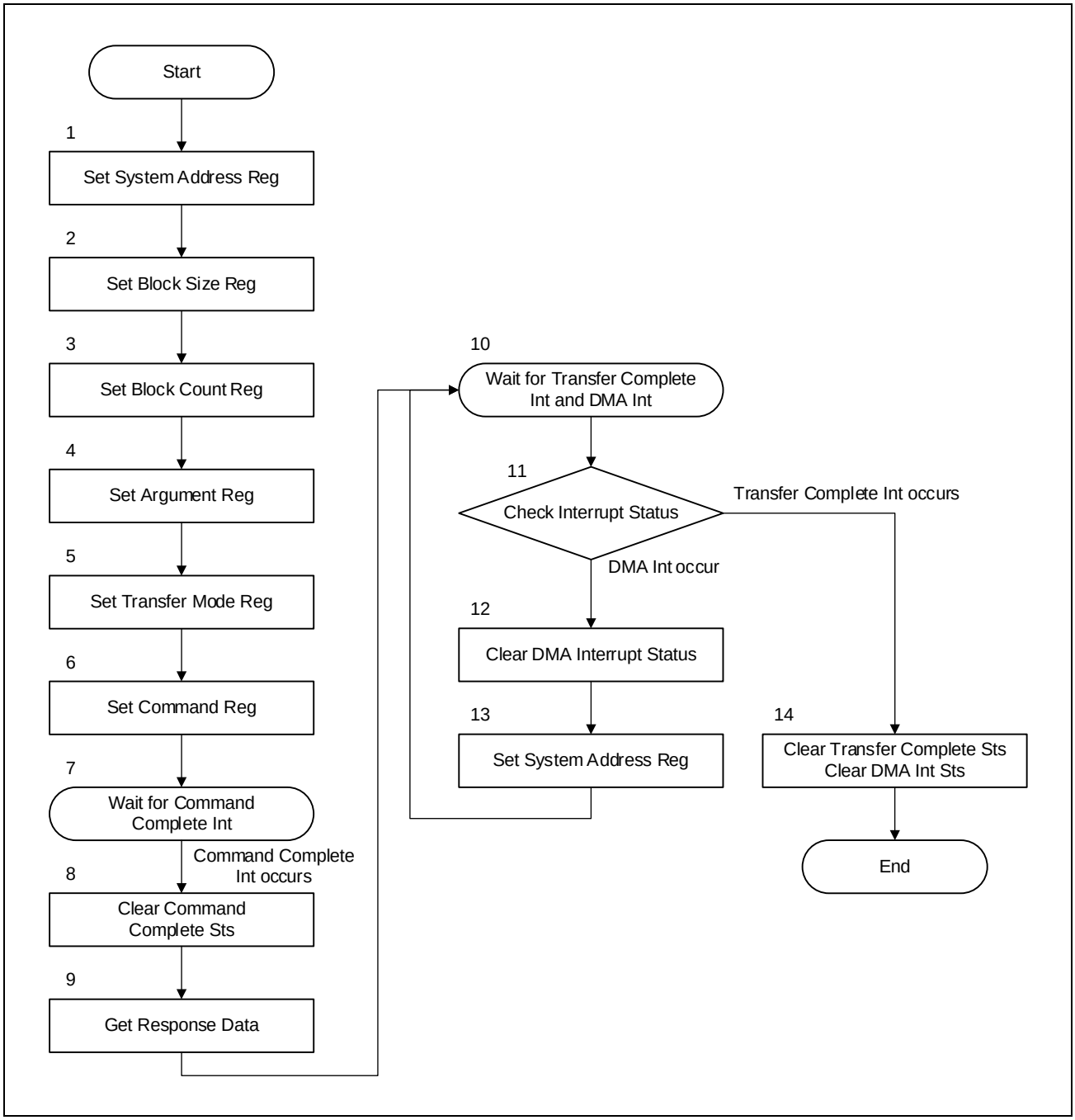


Figure 9.3 Data Transfer Using DAT Line Sequence (Using DMA)

DMA Transaction Sequence

Table 9.43 DMA Transaction

Step	Description
1	Set the system address for DMA in the SDMA System Address Register.
2	Set the value corresponding to the executed data byte length of one block in the Block Size Register.
3	Set the value corresponding to the executed data block count in the Block Count Register.
4	Set the value corresponding to the issued command to Argument Register.
5	Set the value to Multi/Single Block Select and Block Count Enable. Set the value corresponding to the issued command to Data Transfer Direction, Auto CMD12 Enable and DMA Enable.
6	Set the value corresponding to the issued command to Command Register. When writing the upper byte of Command Register, SD command is issued.
7	Wait for the Command Complete Interrupt.
8	Write 1 to the Command Complete in the Normal Interrupt Status Register for clearing this bit.
9	Read Response Register and get necessary information in accordance with the issued command. **DMA read transfer** On receiving the response end bit from the card for the write command (data flowing from Host to Card) the SD Host controller will act as the master and request the System/Host bus. After receiving the grant, the host controller will start reading a block of data from the system memory and fills the first FIFO. Whenever a block of data is ready the transmitter will start sending the data in SD bus. While transmitting the data in SD bus the host controller requests the bus to fill the second block in second FIFO. Ping Pong FIFO's are used to increase the throughput. Similarly, the host controller reads a block of data from the system memory whenever a FIFO is empty. This will continue till all the blocks are read from the System memory. Transfer complete Interrupt will be set only after transferring all the blocks of data to the card. **DMA write transfer** The block of data received from the Card (data flowing from Card to Host) is stored in first half of the FIFO. Whenever a block of data is ready the SD Host controller will act as the master and request the System/Host bus. After receiving the grant, the host controller will start writing a block of data into the system memory from the first FIFO. While transmitting the data into System memory the host controller will receive the second block of data and store in second FIFO. Similarly, the host controller write a block of data into the system memory whenever data is ready. This will continue till all the blocks are transferred to the System memory. Transfer complete Interrupt will be set only after transferring all the blocks of data to the System memory. ***Note*)** Host controller will receive a block of data from the card only when it has room to store a block of data in FIFO. When both the FIFO's are full the host controller will stop the data coming from the card through read wait mechanism (if card supports read wait) or through clock stopping.
10	Wait for the Transfer Complete Interrupt and DMA Interrupt.
11	If Transfer Complete is set 1, go to Step (14) else if DMA Interrupt is set to 1, go to Step (12). Transfer Complete is higher priority than DMA Interrupt.
12	Write 1 to the DMA Interrupt in the Normal Interrupt Status Register to clear this bit.
13	Set the next system address of the next data position to the SDMA System Address Register and go to Step (10).
14	Write 1 to the Transfer Complete and DMA Interrupt in the Normal Interrupt Status Register to clear this bit.

CAUTION

Note: Step (2) and Step (3) can be executed at same time. Step (5) and Step (6) can also be executed at same time.

For example, if the host wants to transfer 4 KB of data to the Card. Assume the maximum block size is 512 bytes. Then the host driver will program the Block Size Register as 512 and Block Count Register with the value 8.

The AHB Master and Transmitter residing inside the Host Controller will get the information (how much data to transfer) from these registers. Using the above information, the AHB master will act as a master and initiate a data read transaction (to read a block of data -512 bytes from the system memory). Whenever a block of data is ready in FIFO, the transmitter will start transmitting the block of data (512 bytes) in SD bus. After transmitting the entire block of data to the card, the transmitter will wait for a status response from the card. Transmitter will send the next block of data

only, when it receives a good status response from the card for the previous block of data otherwise the transaction will be aborted and the host will go for a fresh transaction.

9.5.3 ADMA Transactions

The following figure and table describe the ADMA transaction sequence.

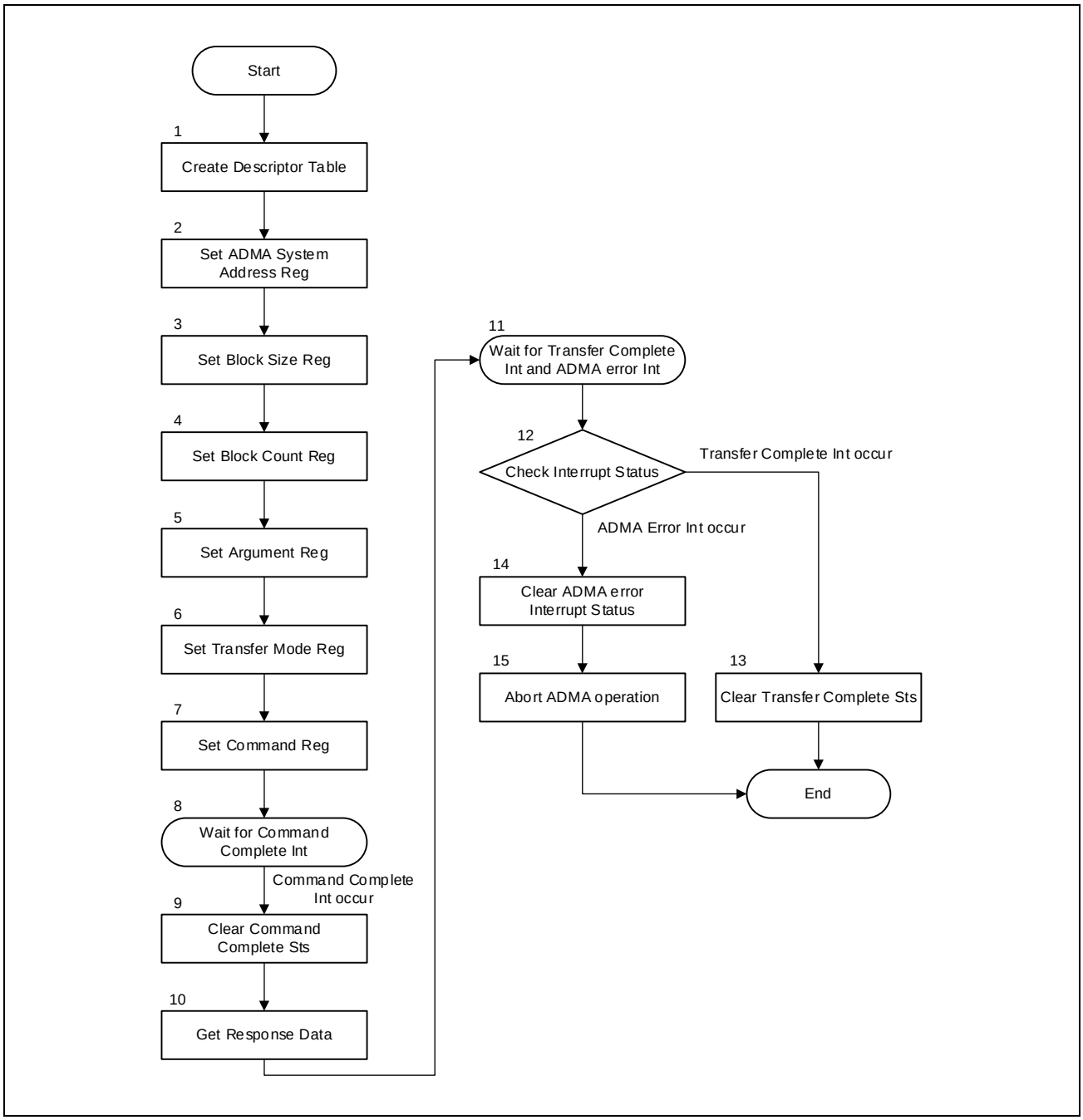


Figure 9.4 ADMA Transaction Flow

ADMA Transaction Flow

Table 9.44 ADMA Transaction

Step	Description
1	Create Descriptor table for ADMA in the system memory.
2	Set the Descriptor address for ADMA in the ADMA System Address Register.
3	Set the value corresponding to the executed data byte length of one block in the Block Size.
4	Set the value corresponding to the executed data block count in the Block Count Register. If the Block Count Enable in the Transfer Mode Register is set to 1, total data length can be designated by the Block Count Register and the Descriptor Table. These two parameters shall indicate same data length. However, transfer length is limited by the 16-bit Block Count Register. If the Block Count Enable in the Transfer Mode Register is set to 0, total data length is designated by not Block Count Register but the Descriptor Table. In this case, ADMA reads more data than length programmed in descriptor from SD card. Too much read operation is aborted asynchronously and extra read data is discarded when the ADMA is completed.
5	Set the argument value to the Argument Register.
6	Set the value to the Transfer Mode Register. The host driver determines Multi/Single Block Select, Block Count Enable, Data Transfer Direction, Auto CMD12 Enable and DMA Enable.
7	Set the value to the Command Register. ***Note*)** When writing to the upper byte [3] of the Command Register, the SD command is issued and DMA is started.
8	Wait for the Command Complete Interrupt.
9	Write 1 to the Command Complete in the Normal Interrupt Status Register to clear this bit.
10	Read Response Register and get necessary information of the issued command.
11	Wait for the Transfer Complete Interrupt and ADMA Error Interrupt.
12	If Transfer Complete is set 1, go to Step (13) else if ADMA Error Interrupt is set to 1, go to Step (14).
13	Write 1 to the Transfer Complete Status in the Normal Interrupt Status Register to clear this bit.
14	Write 1 to the ADMA Error Interrupt Status in the Error Interrupt Status Register to clear this bit.
15	Abort ADMA operation. SD card operation should be stopped by issuing abort command. If necessary, the HD checks ADMA Error Status Register to detect why ADMA error is generated.

CAUTION

Note: Step (3) and Step (4) can be executed simultaneously. Step (6) and Step (7) can also be executed simultaneously.

9.5.4 Abort Transaction

An Abort transaction is performed using CMD12 for a SD Memory Card and by using CMD52 for a SDIO Card. There are two cases where the HD needs to do an Abort Transaction.

(1) When the HD stops infinite block transfers.
(2) When HD stops transfers while a Multiple block transfer is exacting.

There are two ways to issue an Abort command. The first is an Asynchronous abort. The second is a Synchronous Abort. In an Asynchronous Abort sequence, the HD can issue an Abort Command at any time unless Command Inhibit (CMD) in the Present State Register is set to 1. In a Synchronous Abort, the HD shall issue an Abort command after the data transfer stopped by using Stop At Block Gap Request in the Block Gap Control Register.

9.5.4.1 Synchronous Abort

The flow for Synchronous Abort is shown below.

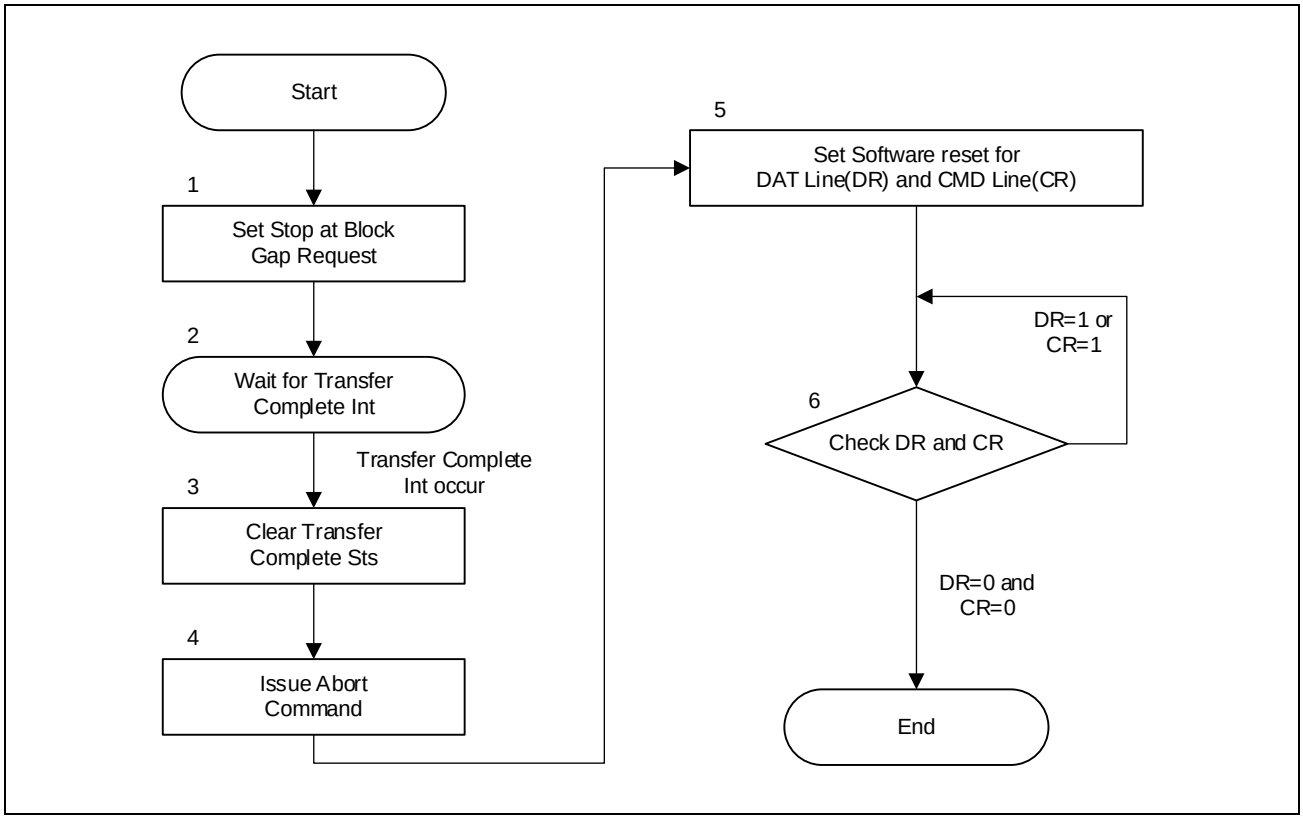


Figure 9.5 Synchronous Abort

Table 9.45 Synchronous Abort Transaction

Step	Description
1	Set the Stop At Block Gap Request in the Block Gap Control Register to 1 to stop SD transactions.
2	Wait for Transfer Complete Interrupt.
3	Set the Transfer Complete to 1 in the Normal Interrupt Status Register to clear this bit.
4	Issue Abort Command.
5	Set both Software Reset for DAT Line and Software Reset for CMD Line to 1 in the Software Reset Register to do software reset.
6	Check Software Reset for DAT Line and Software Reset for CMD Line in the Software Reset Register. If both Software Reset for DAT Line and Software Reset for CMD Line are 0, go to “END”. If either Software Reset for DAT Line or Software Reset for CMD Line is 1, repeat step (6).

9.5.4.2 Asynchronous Abort

The flow for asynchronous abort is shown below.

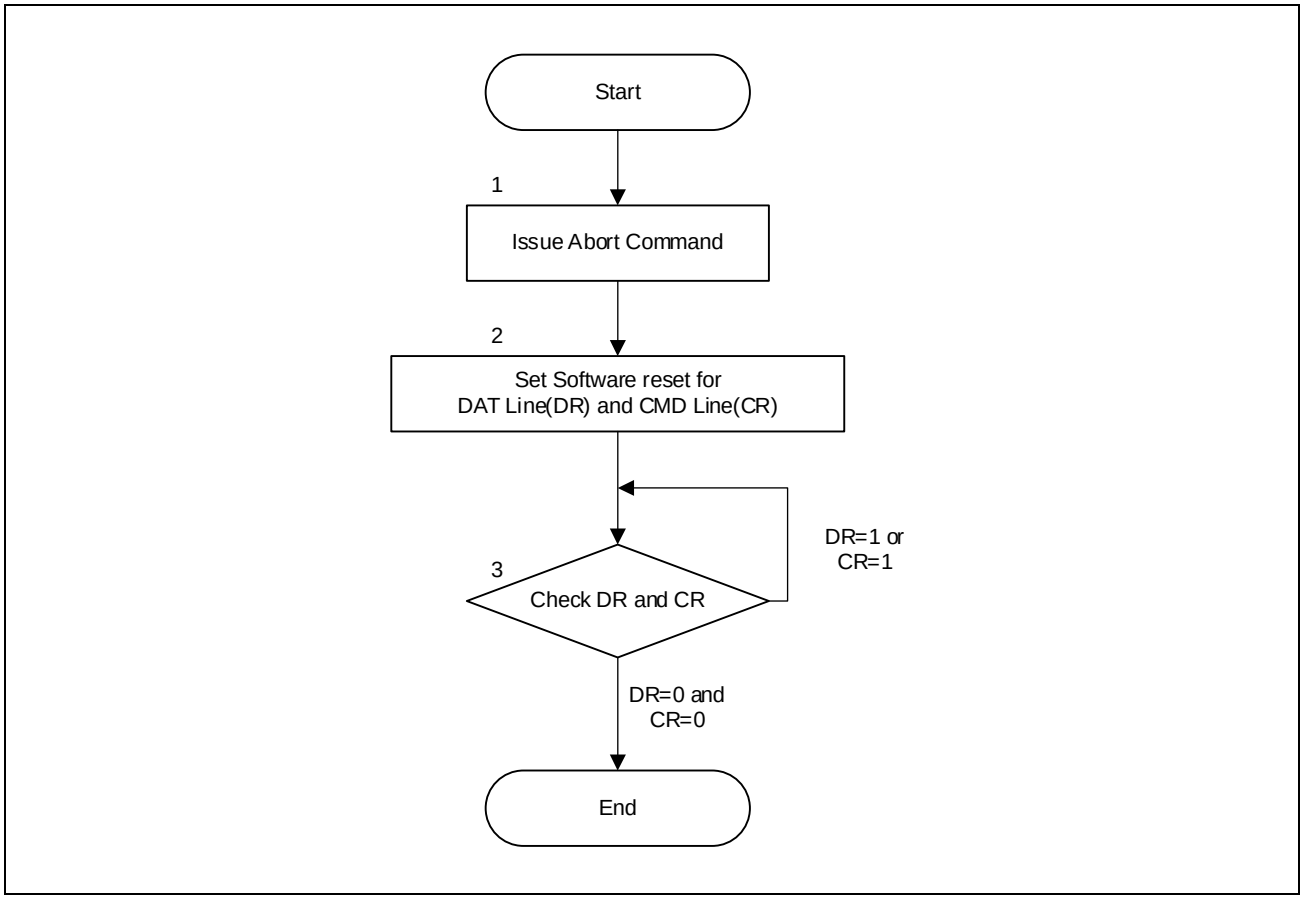


Figure 9.6 Asynchronous Abort

Asynchronous Abort Sequence

Table 9.46 Asynchronous Abort Transaction

Step	Description
1	Issue Abort Command.
2	Set both Software Reset For DAT Line and Software Reset For CMD Line to 1 in the Software Reset Register to do software reset.
3	Check Software Reset For DAT Line and Software Reset For CMD Line in the Software Reset Register. If both Software Reset For DAT Line and Software Reset For CMD Line are 0, go to “End”. If either Software Reset For DAT Line or Software Reset For CMD Line is 1, go to step (3).

Section 10 USB 2.0 HS Host/Function Controller (USBh/USBf)

10.1 Overview

The USB subsystem provides independent USB 2.0 Host controller and USB 2.0 Function controller which are fully compliant with the Universal Serial Bus specification (version 2.0). The USB subsystem can be used as 2-Host mode or 1-Host / 1-Function mode. It can be configured as one host port and one function port. The function port can be reconfigured by the firmware but the whole module must be reset (including the host port).

The USB PHY's PLL provides 48 MHz for RZ/N1 system.

Host Controller Feature

- Compliant with both the enhanced host controller interface (EHCI) specification (version 1.0) and the open host controller interface (OHCI) specification (version 1.0a)
- Supports for the following speeds:
  - High speed (HS): 480 Mbps (USB 2.0)
  - Full speed (FS): 12 Mbps (USB 1.1)
  - Low speed (LS): 1.5 Mbps (USB 1.1)
- A USB Plug Detect (UPD) which detects the connection of a device.
- Output port power switch management
- Port over current indication
- Integrated DMA
- USB PHYs (2 USB ports)
- The USB PHY shared with USB Function controller (Port1 only)
- Routing the clock of 48 MHz generated by USB PHY's PLL outside USB to reuse in the system
  - In case of change (from Host to Function mode or from Function to Host mode) via register bit (H2MODE) by firmware, this clock must be restarted.
- Transmit & Receive FIFO

Function Controller Feature

- USB PHY is shared with USB Host controller (Port1 only)
- 16 Endpoints are configured as below

Endpoint	Direction	Type	Buffer	Max Packet Size
EP0	IN/OUT	Control	Single	64 Byte
EP1	IN	Bulk	Double	512 Byte
EP2	OUT	Bulk	Double	512 Byte
EP3	IN	Bulk	Single	512 Byte
EP4	OUT	Bulk	Single	512 Byte
EP5	IN	Bulk	Single	512 Byte
EP6	IN	Interrupt	Single	1024 Byte
EP7	IN	Interrupt	Single	1024 Byte
EP8	IN	Interrupt	Single	1024 Byte
EP9	IN	Interrupt	Single	1024 Byte
EP10	IN	Isochronous	Double	1024 Byte
EP11	OUT	Isochronous	Double	1024 Byte
EP12	IN	Isochronous	Double	1024 Byte
EP13	OUT	Isochronous	Double	1024 Byte
EP14	IN	Isochronous	Double	1024 Byte
EP15	OUT	Isochronous	Double	1024 Byte

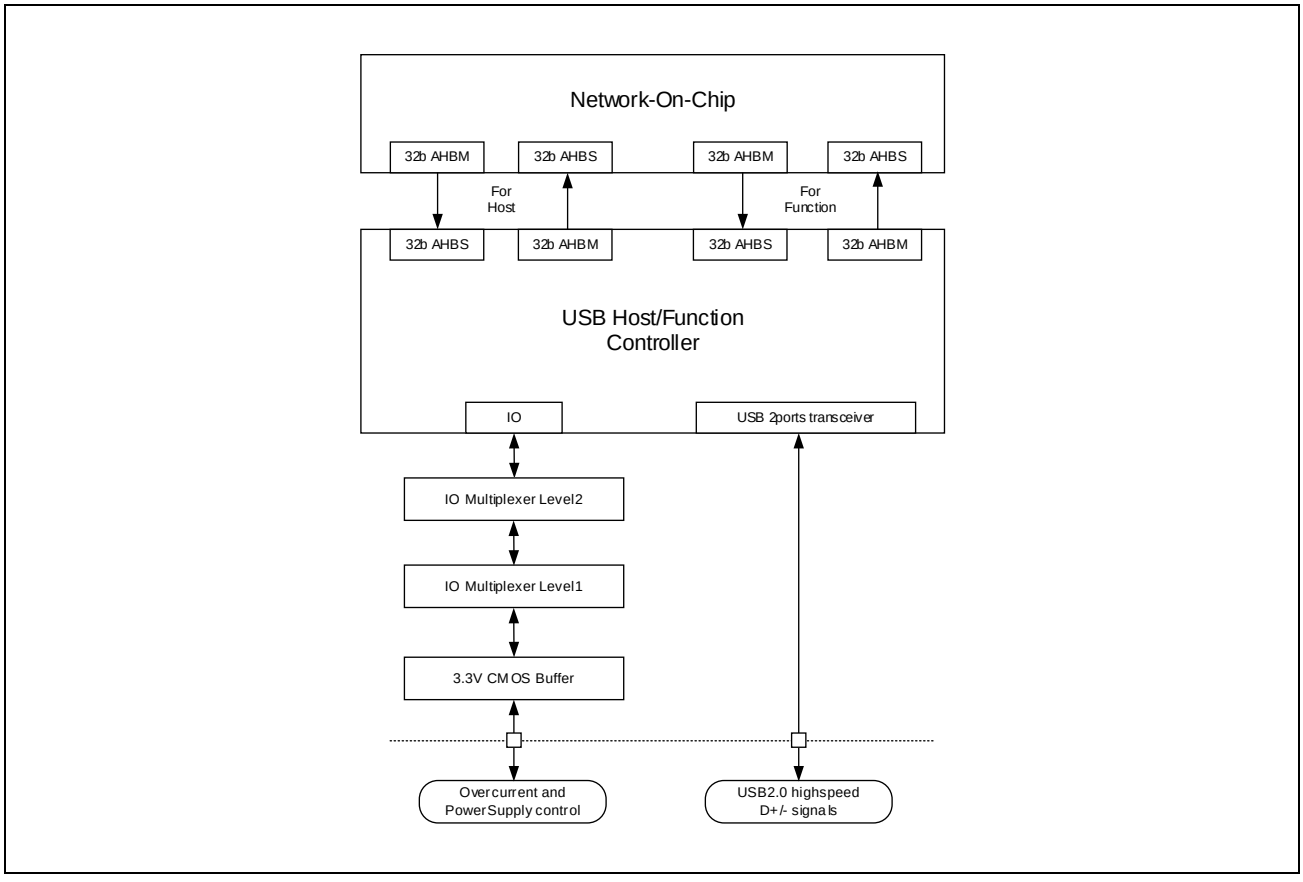


Figure 10.1 USB Subsystem Interfaces and Connections

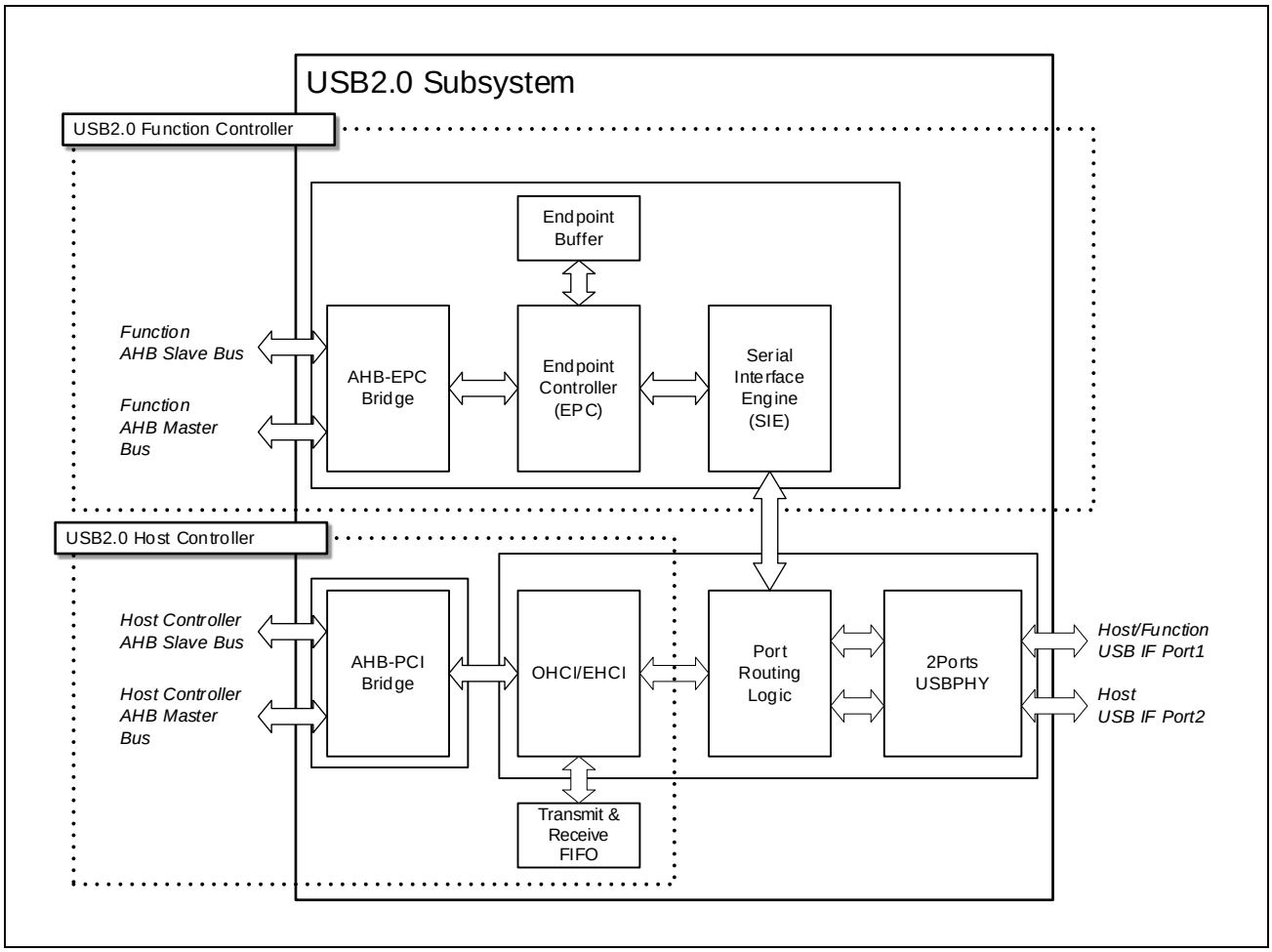


Figure 10.2 USB Subsystem Block Diagram

10.2 Signal Interfaces

Signal Name	Input Output	Description
Clock		
USB_HCLKH	Input	Internal bus clock (AHB) for HOST
USB_HCLKF	Input	Internal bus clock (AHB) for Function
USB_HCLKPM	Input	Internal bus clock (AHB) for Power Management
USB_PCICLK	Input	PCI clock for USB subsystem
USB_DCLK48	Output	48 MHz output for system, no clock gating
Interrupt		
USB2H_BIND_Int	Output	Level sensitive interrupt, Active High. (described as U2H_BIND_INT in this section)
USB2F_EPC_Int	Output	Level sensitive interrupt, Active High. (described as U2F_EPC_INT in this section)
USB2F_Int	Output	Level sensitive interrupt, Active High. (described as U2F_INT in this section)
External Signal		
USB_DP[m]	I/O	Bus High Speed D+ (port m)
USB_DM[m]	I/O	Bus High Speed D- (port m)
USB_OC[m]	Input	Overcurrent status (port m) for Host, active low
USB_PPON[m]	Output	Port Power Control (port m) for Host
USB_VBUS	Input	Port power detection pin (dedicated pin) for Function An external circuit should convert 5 V to 3 V, prevent chattering and input to USB_VBUS pin.

Note: m = 1 or 2

10.3 USBPLL Features

- The USBPLL is stopped by default after Power-On-Reset
- The USBPLL stopped if the HRESETn of the USB core is asserted
- The software needs to start USBPLL which means the DIRPD bit in CFG_USB register (in the system controller) needs to be de-asserted, the USB module needs to come out of reset (AHB HRESETn de-asserted), the software needs to release the PLL reset bit in the internal registers of the USB core, and the software needs to enable USB_DCLK48 clock (to enable the FRCLK48MOD bit in the CFG_USB register).
- Routing clock of 48 MHz outside USB to reuse in the system
- In case of changing CFG_USB.H2MODE (in the system controller), note the following:
  - In case of change (from Host to Function mode or from Function to Host mode) via register bit (H2MODE) by firmware, this clock must be restarted.
  - Port1 (Function) mode change while running port2 (Host) is not possible. Reset of the PHYs module is necessary and this module is common for both USB cores.
  - The USBPLL does not stop on H2MODE register bit but it does stop when the software resets the USB as a consequence of the mode change.

10.4 Register Map

10.4.1 OHCI Operation Register Map

Table 10.1 USB HC-OHCI Register Map

Address	Register Symbol	Register Name
4002 0000h	HCREVISION	HcRevision Register
4002 0004h	HCCONTROL	HcControl Register
4002 0008h	HCCOMMANDSTATUS	HcCommandStatus Register
4002 000Ch	HCINTERRUPTSTATUS	HcInterruptStatus Register
4002 0010h	HCINTERRUPTENABLE	HcInterruptEnable Register
4002 0014h	HCINTERRUPTDISABLE	HcInterruptDisable Register
4002 0018h	HCHCCA	HcHCCA Register
4002 001Ch	HCPERIODCURRENTED	HcPeriodicCurrentED Register
4002 0020h	HCCONTROLHEADED	HcControlHeadED Register
4002 0024h	HCCONTROLCURRENTED	HcControlCurrentED Register
4002 0028h	HCBULKHEADED	HcBulkHeadED Register
4002 002Ch	HCBULKCURRENTED	HcBulkCurrentED Register
4002 0030h	HCDONEHEAD	HcDoneHead Register
4002 0034h	HCFMINTERVAL	HcFrameInterval Register
4002 0038h	HCFMREMAINING	HcFrameRemaining Register
4002 003Ch	HCFMNUMBER	HcFrameNumber Register
4002 0040h	HCPERIODICSTART	HcPeriodicStart Register
4002 0044h	HCLSTHRESHOLD	HcLSThreshold Register
4002 0048h	HCRHDESCRIPTORA	HcRhDescriptorA Register
4002 004Ch	HCRHDESCRIPTORB	HcRhDescriptorB Register
4002 0050h	HCRHSTATUS	HcRhStatus Register
4002 0054h	HCRHPORTSTATUS1	HcRhPortStatus1 Register
4002 0058h	HCRHPORTSTATUS2	HcRhPortStatus2 Register

10.4.2 EHCI Operation Register Map

Table 10.2 USB HC-EHCI Register Map

Address	Register Symbol	Register Name
4002 1000h	CAPL_VERSION	HCIVERSION and CAPLENGTH Register (EHCI)
4002 1004h	HCSPARAMS	HCSPARAMS Register
4002 1008h	HCCPARAMS	HCCPARAMS Register
4002 100Ch	HCSP_PORTROUTE	HCSP_PORTROUTE Register
4002 1020h	USBCMD	USBCMD Register
4002 1024h	USBSTS	USBSTS Register
4002 1028h	USBINTR	USBINTR Register
4002 102Ch	FRINDEX	Frame Index Register
4002 1030h	CTRLDSSEGMENT	CTRLDSSEGMENT Register
4002 1034h	PERIODICLISTBASE	PERIODICLISTBASE Register
4002 1038h	ASYNCLISTADDR	ASYNCLISTADDR Register
4002 1060h	CONFIGFLAG	CONFIGFLAG Register
4002 1064h	PORTSC1	PORTSC1 Register
4002 1068h	PORTSC2	PORTSC2 Register

10.4.3 OHCI (PCI Configuration Space) Register Map

Table 10.3 USB CR-OHCI Register Map

Address	Register Symbol	Register Name
4003 0000h	VID_DID	Device ID - Vendor ID (OHCI)
4003 0004h	CMND_STS	Status - Command (OHCI)
4003 0008h	REVID_CC	Class Code - Revision ID (OHCI)
4003 000Ch	CLS_LT_HT_BIST	BIST - Header Type - Latency Timer - Cache Line Size (OHCI)
4003 0010h	BASEAD	OHCI Base Address
4003 002Ch	SSVID_SSID	Subsystem ID - Subsystem Vendor ID (OHCI)
4003 0030h	EROM_BASEAD	Expansion ROM Base Address (OHCI)
4003 0034h	CAPPTR	Capability Pointer (OHCI)
4003 003Ch	INTR_LINE_PIN	Max_Lat - Min_Gnt - Interrupt Pin - Interrupt Line (OHCI)
4003 0040h	CAPID_NIP_PMCAP	Capability Identifier - Next Item Pointer - Power Management Capabilities (OHCI)
4003 0044h	PMC_STS_PMCSR	Power Management Control and Status - PMCSR Bridge Support Extensions (OHCI)
4003 00E0h	EXT1	EXT1 Register (OHCI)
4003 00E4h	EXT2	EXT2 Register (OHCI)
4003 00F4h	UTMICTRL	USBPHY Operation Mode Control Register (OHCI)

10.4.4 EHCI (PCI Configuration Space) Register Map

Table 10.4 USB CR-EHCI Register Map

Address	Register Symbol	Register Name
4003 0100h	VID_DID	Device ID - Vendor ID (EHCI)
4003 0104h	CMND_STS	Status - Command (EHCI)
4003 0108h	REVID_CC	Class Code - Revision ID (EHCI)
4003 010Ch	CLS_LT_HT_BIST	BIST - Header Type - Latency Timer - Cache Line Size (EHCI)
4003 0110h	BASEAD	EHCI Base Address
4003 012Ch	SSVID_SSID	Subsystem ID - Subsystem Vendor ID (EHCI)
4003 0130h	EROM_BASEAD	Expansion ROM Base Address (EHCI)
4003 0134h	CAPPTR	Capability Pointer (EHCI)
4003 013Ch	INTR_LINE_PIN	Max_Lat - Min_Gnt - Interrupt Pin - Interrupt Line (EHCI)
4003 0140h	CAPID_NIP_PMCAP	Capability Identifier - Next Item Pointer - Power Management Capabilities (EHCI)
4003 0144h	PMC_STS_PMCSR	Power Management Control and Status - PMCSR Bridge Support Extensions (EHCI)
4003 0160h	SBRN_FLADJ_PW	SBRN - FLADJ - PORTWAKECAP
4003 01E0h	EXT1	EXT1 Register (EHCI)
4003 01E4h	EXT2	EXT2 Register (EHCI)
4003 01F4h	UTMICTRL	USBPHY Operation Mode Control Register (EHCI)

10.4.5 AHB-PCI Bridge (PCI Configuration Space) Register Map

Table 10.5 USB CR-AHBPCI Register Map

Address	Register Symbol	Register Name
4003 0000h	VID_DID	Device ID - Vendor ID (AHB-PCI Bridge)
4003 0004h	CMND_STS	Status - Command (AHB-PCI Bridge)
4003 0008h	REVID_CC	Class Code - Revision ID (AHB-PCI Bridge)
4003 000Ch	CLS_LT_HT_BIST	BIST - Header Type - Latency Timer - Cache Line Size (AHB-PCI Bridge)
4003 0010h	BASEAD	AHB-PCI Bridge Registers Base Address
4003 0014h	WIN1_BASEAD	PCI-AHB Window1 Base Address
4003 0018h	WIN2_BASEAD	PCI-AHB Window2 Base Address
4003 002Ch	SSVID_SSID	Subsystem ID - Subsystem Vendor ID (AHB-PCI Bridge)
4003 003Ch	INTR_LINE_PIN	Max_Lat - Min_Gnt - Interrupt Pin - Interrupt Line (AHB-PCI Bridge)

10.4.6 AHB-PCI Bridge (PCI Communication Space) Register Map

Table 10.6 USB HC-AHBPCI Register Map

Address	Register Symbol	Register Name
4003 0800h	PCIAHB_WIN1_CTR	PCIAHB Window1 Control Register
4003 0804h	PCIAHB_WIN2_CTR	PCIAHB Window2 Control Register
4003 0810h	AHBPCI_WIN1_CTR	AHBPCI Window1 Control Register
4003 0814h	AHBPCI_WIN2_CTR	AHBPCI Window2 Control Register
4003 0820h	PCI_INT_ENABLE	PCI Interrupt Enable Register
4003 0824h	PCI_INT_STATUS	PCI Interrupt Status Register
4003 0830h	AHB_BUS_CTR	AHB Bus Control Register
4003 0834h	USBCTR	USB Control Register
4003 0840h	PCI_ARBITER_CTR	PCI Arbiter Control Register

10.4.7 EPC Register Map

Table 10.7 USB FC-EPC Register Map

Address	Register Symbol	Register Name
4001 E000h	USB_CONTROL	USB Control Register
4001 E004h	USB_STATUS	USB Status Register
4001 E008h	USB_ADDRESS	Frame Number & USB Address Register
4001 E010h	TEST_CONTROL	Test Control Register
4001 E018h	SETUP_DATA0	Setup Data0 Register
4001 E01Ch	SETUP_DATA1	Setup Data1 Register
4001 E020h	USB_INT_STA	USB Interrupt Status Register
4001 E024h	USB_INT_ENA	USB Interrupt Enable Register
4001 E028h	EP0_CONTROL	EP0 Control Register
4001 E02Ch	EP0_STATUS	EP0 Status Register
4001 E030h	EP0_INT_ENA	EP0 Interrupt Enable Register
4001 E034h	EP0_LENGTH	EP0 OUT Data Length Register
4001 E038h	EP0_READ	EP0 Read Register
4001 E03Ch	EP0_WRITE	EP0 Write Register
4001 E040h + 20h × (m − 1)	EP[m]_CONTROL (m = 1..15)	EP[m] Control Register
4001 E044h + 20h × (m − 1)	EP[m]_STATUS (m = 1..15)	EP[m] Status Register
4001 E048h + 20h × (m − 1)	EP[m]_INT_ENA (m = 1..15)	EP[m] Interrupt Enable Register
4001 E04Ch + 20h × (m − 1)	EP[m]_DMA_CTRL (m = 1..15)	EP[m] DMA Control Register
4001 E050h + 20h × (m − 1)	EP[m]_PCKT_ADRS (m = 1..15)	EP[m] MaxPacket & BaseAddress Register
4001 E054h + 20h × (m − 1)	EP[m]_LEN_DCNT (m = 1..15)	EP[m] Length & DMA Count Register
4001 E058h + 20h × (m − 1)	EP[m]_READ (m = 1..15)	EP[m] Read Register
4001 E05Ch + 20h × (m − 1)	EP[m]_WRITE (m = 1..15)	EP[m] Write Register

10.4.8 AHB-EPC Bridge Register Map

Table 10.8 USB FC-AHBEPC Register Map

Address	Register Symbol	Register Name
4001 F000h	AHBSCTR	AHB Slave Controller Configuration Register
4001 F004h	AHBMCTR	AHB Master Controller Configuration Register
4001 F008h	AHBBINT	AHB-EPC Bridge Interrupt Source Register
4001 F00Ch	AHBBINTEN	AHB-EPC Bridge Interrupt Enable Register
4001 F010h	EPCTR	EPC and Transceiver Control Register
4001 F020h	USBSSVER	USBf Version Register
4001 F024h	USBSSCONF	Endpoint Configuration Register
4001 F110h + 10h × (m − 1)	EP[m]DCR1 (m=1..15)	Endpoint[m] DMA Setting Register1
4001 F114h + 10h × (m − 1)	EP[m]DCR2 (m=1..15)	Endpoint[m] DMA Setting Register2
4001 F118h + 10h × (m − 1)	EP[m]TADR (m=1..15)	Endpoint[m] DMA Start Address Register

10.5 Register Description

10.5.1 OHCI Operation Register Description

10.5.1.1 HCREVISION — HcRevision Register

Address: 4002 0000h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	REVISION							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Table 10.9 HCREVISION Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7 to b0	REVISION	The version of the HCI Specification implemented in this host controller. These bits show 10h, which indicates that this host controller is compliant with OHCI 1.0a.	R

10.5.1.2 HCCONTROL — HcControl Register

Address: 4002 0004h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	RWE	RWC	IR	HCFS		BLE	CLE	IE	PLE	CBSR	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.10 HCCONTROL Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b11	Reserved		R
b10	RWE	Remote Wakeup Enable If this bit is 1b, PME is asserted when bit 3 (RD) of the HcInterruptStatus register is 1b. 0: PME is not asserted when Resume is detected. (PME disabled) 1: PME is asserted when Resume is detected. (PME enabled).	R/W
b9	RWC	Remote WakeUp Connect. Set this bit during initialization if the system supports Remote WakeUp. 0: Remote WakeUp is not supported 1: Remote WakeUp is supported.	R/W
b8	IR	Interrupt Routing. Specify the path to report a generated interrupt source to the HcInterruptStatus register. Use this bit with the initial value because this module does not support SMI. 0: INTA 1: SMI (not available)	R/W
b7, b6	HCFS	Host Controller Functional State. When the host controller enters the USB operational state, it starts managing frames split into 1 ms. This state is always controlled by software except when moving from the USB suspend state to the USB resume state triggered by Remote WakeUp. This bit is set to 00b when a hardware reset ends, and 11b when a software reset ends. 00b: USB reset 01b: USB resume 10b: USB operational 11b: USB suspend.	R/W
b5	BLE	Bulk List Enable. Specify whether to process the bulk list. The setting of this bit is applied from the next frame. When correcting the bulk list, this bit must be 0b. 0: Do not process the bulk list 1: Process the bulk list.	R/W
b4	CLE	Control List Enable. Specify whether to process the control list. The setting of this bit is applied from the next frame. When correcting the control list, this bit must be 0b. 0: Do not process the control list 1: Process the control list.	R/W

Table 10.10 HCCONTROL Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b3	IE	Isochronous Enable. Specify whether to process the isochronous ED list. The setting of this bit is applied from the next frame. Specify whether to process the isochronous ED list when an isochronous ED is found during list processing. 0: Do not process Isochronous transfer 1: Process Isochronous transfer.	R/W
b2	PLE	Periodic List Enable. Specify whether to process the periodic frame list. The setting of this bit is applied from the next frame. 0: Do not process the periodic frame list, 1: Process the periodic frame list.	R/W
b1, b0	CBSR	Control Bulk Service Ratio. Specify the service ratio between control and bulk EDs. When processing the periodic frame list, the ratio specified by this bit is held and used for transfer. 00b: 1:1 = BulkED:ControlED 01b: 2:1 10b: 3:1 11b: 4:1	R/W

10.5.1.3 HCCOMMANDSTATUS — HcCommandStatus Register

Address: 4002 0008h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	SOC	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	OCR	BLF	CLF	HCR
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.11 HCCOMMANDSTATUS Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b18	Reserved		R
b17, b16	SOC	Scheduling Overrun Count. Indicates the number of times a scheduling overrun has occurred. This bit is incremented each time a scheduling overrun is detected and reset to 00b when the count reaches 11b. This bit is incremented when a scheduling overrun is detected even if bit 0 (SO) in the HcInterruptStatus register has already been set.	R
b15 to b4	Reserved		R
b3	OCR	Ownership Change Request. This bit is used to change ownership of the host controller.	R/W
b2	BLF	Bulk List Field. Indicates whether a TD exists in a bulk list. The host controller checks this bit when starting processing of the first ED in the bulk list. The host controller does not start list processing if this bit is 0b. If this bit is 1b, the host controller starts bulk list processing and then sets the bit to 0b. When the host controller detects a TD in the list, it sets this bit to 1b again and continues bulk list processing. The host controller sets this bit to 0b when list processing finishes. If no TD has been found in the list or if this bit is not set to 1b by software, this bit remains 0b and list processing stops. When restructuring the list before executing list processing, set bit 5 (BLE) of the HcCommand register and set this bit before starting list processing.	R/W
b1	CLF	Control List Field. Indicates whether a TD exists in a control list. The host controller checks this bit when starting processing of the first ED in the control list. The host controller does not start list processing if this bit is 0b. If this bit is 1b, the host controller starts control list processing and then sets the bit to 0b. When the host controller detects a TD in the list, it sets this bit to 1b again and continues control list processing. The host controller sets this bit to 0b when list processing finishes. If no TD has been found in the list or if this bit is not set to 1b by software, this bit remains 0b and list processing stops. When restructuring the list before executing list processing, set bit 4 (CLE) of the HcCommand register and set this bit before starting list processing.	R/W
b0	HCR	Host Controller Reset This bit is used to trigger a software reset for the host controller. If this bit is set, the host controller enters the USB suspend state, regardless of its functional state.	R/W

10.5.1.4 HCINTERRUPTSTATUS — HcInterruptStatus Register

Address: 4002 000Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	RHSC	FNO	UE	RD	SF	WDH	SO
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.12 HCINTERRUPTSTATUS Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b7	Reserved	Keep the initial value	R/W
b6	RHSC	Root Hub Status Change. This is an interrupt bit indicating that a setting of the HcRhStatus or HcRhPortStatus register has been changed. This bit is set when an HcRhStatus or HcRhPortStatus register setting has been changed by a hardware interrupt source. The interrupt source is cleared when this bit is set to 1b. 0: RHSC interrupt has not occurred 1: RHSC interrupt has occurred.	R/W
b5	FNO	Frame Number Overflow. This is an interrupt bit indicating that the MSB of bits [15:0] (FrameNumber) of the HcFrameNumber register has been changed. This bit is set after HccaFrameNumber is updated for the frame in which the MSB of the FrameNumber bit changes from 0b to 1b or 1b to 0b. The interrupt source is cleared when this bit is set to 1b. 0: No FNO interrupt has occurred 1: An FNO interrupt has occurred	R/W
b4	UE	Unrecoverable Error. This is an interrupt bit indicating that a system error not related to the USB has been detected on the PCI bus. The interrupt source is cleared when this bit is set to 1b. 0: UE interrupt has not occurred 1: UE interrupt has occurred.	R/W
b3	RD	Resume Detected This is an interrupt bit indicating that resume signaling has been detected. This bit is set when the host controller detects that a device on the USB is asserting resume signaling. This bit is not set when software asserts USB resume signaling. The interrupt source is cleared when this bit is set to 1b. 0: RD interrupt has not occurred 1: RD interrupt has occurred.	R/W
b2	SF	Start Of Frame This is an interrupt bit indicating that HccaFrameNumber was updated when a frame started. The host controller updates HccaFrameNumber when the SOF packet is transmitted and then sets this bit. The interrupt source is cleared when this bit is set to 1b. 0: SF interrupt has not occurred. 1: SF interrupt has occurred.	R/W
b1	WDH	Writeback Done Head This is an interrupt bit indicating that the host controller has updated the HccaDoneHead contents. The host controller sets this bit immediately after updating HccaDoneHead and does not update HccaDoneHead until it is cleared. The interrupt source is cleared when this bit is set to 1b. 0: WDH interrupt has not occurred 1: WDH interrupt has occurred.	R/W

Table 10.12 HCINTERRUPTSTATUS Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b0	SO	Scheduling Overrun. This is an interrupt bit indicating that USB scheduling has overrun in a frame. This bit is set after HccaFrameNumber of the next frame is updated, when the USB schedule overruns. When this bit is set, bits [17:16] (SOC) of the HcCommandStatus register are also incremented. The interrupt source is cleared when this bit is set to 1b. 0: SO interrupt has not occurred 1: SO interrupt has occurred.	R/W

10.5.1.5 HCINTERRUPTENABLE — HcInterruptEnable Register

Address: 4002 0010h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	MIE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	RHSCE	FNOE	UEE	RDE	SFE	WDHE	SOE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.13 HCINTERRUPTENABLE Register Contents

Bit Position	Bit Name	Function	R/W
b31	MIE	Master Interrupt Enable Enable bit for bits [6:0] in this register. If this bit is 0b, all interrupts are masked. Writing function is as follows. 0: Ignored 1: Enable the specified interrupt	R/W
b30 to b7	Reserved	Keep the initial value	R/W
b6	RHSCE	Root Hub Status Change Enable. Writing function is as follows. 0: Ignored 1: Enable the RHSC interrupt	R/W
b5	FNOE	Frame Number Overflow Enable. Writing function is as follows. 0: Ignored 1: Enable the FNO interrupt	R/W
b4	UEE	Unrecoverable Error Enable. Writing function is as follows. 0: Ignored 1: Enable the UE interrupt	R/W
b3	RDE	Resume Detected Enable. Writing function is as follows. 0: Ignored 1: Enable the RD interrupt	R/W
b2	SFE	Start Of Frame Enable. Writing function is as follows. 0: Ignored 1: Enable the SF interrupt	R/W
b1	WDHE	Writeback Done Head Enable. Writing function is as follows. 0: Ignored 1: Enable the WDH interrupt	R/W
b0	SOE	Scheduling Overrun Enable. Writing function is as follows. 0: Ignored 1: Enable the SO interrupt	R/W

NOTE

The interrupt enable bits are cleared by writing 1b to the corresponding bit in the HcInterruptDisable register.

10.5.1.6 HCINTERRUPTDISABLE — HcInterruptDisable Register

Address: 4002 0014h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	MID	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	RHSCD	FNOD	UED	RDD	SFD	WDHD	SOD
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.14 HCINTERRUPTDISABLE Register Contents

Bit Position	Bit Name	Function	R/W
b31	MID	Master Interrupt Disable. Disable bit for bits [6:0] in the HcInterruptEnable register. If this bit is 0b, all interrupts are masked. Writing function is as follows. 0: Ignored 1: Disable the specified interrupt	R/W
b30 to b7	Reserved	Keep the initial value	R/W
b6	RHSCD	Root Hub Status Change Disable. Writing function is as follows. 0: Ignored 1: Disable the RHSC interrupt.	R/W
b5	FNOD	Frame Number Overflow Disable. Writing function is as follows. 0: Ignored 1: Disable the FNO interrupt.	R/W
b4	UED	Unrecoverable Error Disable. Writing function is as follows. 0: Ignored 1: Disable the UE interrupt	R/W
b3	RDD	Resume Detected Disable. Writing function is as follows. 0: Ignored 1: Disable the RD interrupt	R/W
b2	SFD	Start Of Frame Disable. Writing function is as follows. 0: Ignored 1: Disable the SF interrupt	R/W
b1	WDHD	Writeback Done Head Disable. Writing function is as follows. 0: Ignored 1: Disable the WDH interrupt	R/W
b0	SOD	Scheduling Overrun Disable. Writing function is as follows. 0: Ignore 1: Disable the SO interrupt	R/W

10.5.1.7 HCHCCA — HcHCCA Register

Address: 4002 0018h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	HCHCCA															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	HCHCCA								—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.15 HCHCCA Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	HCHCCA	RAM base address setting for Host Controller Communication Area (HCCA) Set the value during an initialization. The host controller requests allocation of a 256-byte area for HCCA starting from the base address specified by this register.	R/W
b7 to b0	Reserved		R

10.5.1.8 HCPERIODCURRENTED — HcPeriodicCurrentED Register

Address: 4002 001Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	PERIODICCURRENTED															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	PERIODICCURRENTED												—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.16 HCPERIODCURRENTED Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	PERIODICCURRENTED	Physical address of current ED in the periodic list. The host controller updates these bits when ED list processing is finished.	R
b3 to b0	Reserved		R

10.5.1.9 HCCONTROLHEADED — HcControlHeadED Register

Address: 4002 0020h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	CONTROLHEADED															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	CONTROLHEADED												—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.17 HCCONTROLHEADED Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	CONTROLHEADED	Physical address of the first ED in the control list. To execute a control transfer, set this bit before setting bit 4 (CLE) of the HcControl register.	R/W
b3 to b0	Reserved		R

10.5.1.10 HCCONTROLCURRENTED — HcControlCurrentED Register

Address: 4002 0024h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	CONTROLCURRENTED															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	CONTROLCURRENTED												—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.18 HCCONTROLCURRENTED Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	CONTROLCURRENTED	Physical address of the current ED in the control list. The host controller updates this bit each time control ED list processing finishes. When structuring a new list, set this bit to 00000000h, which means the end of the list. Make sure that the ED pointed to by the link pointer of this field actually exists when suspending and resuming a transfer.	R/W
b3 to b0	Reserved		R

10.5.1.11 HCBULKHEADED — HcBulkHeadED Register

Address: 4002 0028h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	BULKHEADED															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	BULKHEADED												—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.19 HCBULKHEADED Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	BULKHEADED	Physical address of the first ED in the bulk list. To execute a bulk transfer, set these bits before setting bit 5 (BLE) of the HcControl register.	R/W
b3 to b0	Reserved		R

10.5.1.12 HCBULKCURRENTED — HcBulkCurrentED Register

Address: 4002 002Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	BULKCURRENTED															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	BULKCURRENTED												—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.20 HCBULKCURRENTED Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	BULKCURRENTED	Physical address of the current ED in the bulk list. The host controller updates this bit each time bulk ED list processing finishes. When structuring a new list, set this bit to 00000000h, which means the end of the list. Make sure that an ED which is pointed by the link pointer of these bits actually exists when it suspends and resumes a transfer.	R/W
b3 to b0	Reserved		R

10.5.1.13 HCDONEHEAD — HcDoneHead Register

Address: 4002 0030h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DONEHEAD															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DONEHEAD												—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.21 HCDONEHEAD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	DONEHEAD	Physical address of HcDoneHead in the host controller. This field shows the physical address of the TD that was added to a Done queue and finished last.	R
b3 to b0	Reserved		R

10.5.1.14 HCFMINTERVAL — HcFrameInterval Register

Address: 4002 0034h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	FIT	FSMPS														
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	FI													
Value after reset	0	0	1	0	1	1	1	0	1	1	0	1	1	1	1	1

Table 10.22 HCFMINTERVAL Register Contents

Bit Position	Bit Name	Function	R/W
b31	FIT	Frame Interval Toggle. Use this bit to synchronize the frame value between the software and the host controller. This bit should be toggled by software when it updates FI bit. When loading FI bits, the host controller reflects FIT value to bit 31 (FRT) of the HcFrameRemaining register. Software compares the value of this bit specified when writing a value to FI bits with the read FRT bit to confirm that the new FI bits value has been reflected.	R/W
b30 to b16	FSMPS	FS Largest Data Packet. Specify the maximum size of data that can be transmitted or received without causing a scheduling overrun. The host controller compares the current frame position with the specified value to determine up to which part of a frame can be transferred. The value is set by software, because it depends on system bus capacity.	R/W
b15, b14	Reserved		R
b13 to b0	FI	Frame Interval. Specify the frame length (bit time) for FS transfer. Set 2EDFh to satisfy one frame (= 1 ms) as defined in the USB specification	R/W

10.5.1.15 HCFMREMAINING — HcFrameRemaining Register

Address: 4002 0038h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	FRT	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	FR													
Value after reset	0	0	1	0	1	1	1	0	1	1	0	1	1	1	1	1

Table 10.23 HCFMREMAINING Register Contents

Bit Position	Bit Name	Function	R/W
b31	FRT	Frame Remaining Toggle. Use this bit to synchronize the frame value between the software and the host controller. When the FR bits are set to 0000h, the host controller copies the value of bits [13:0] (FI) of the HcFrameInterval register to the FR bits, and at the same time, copies the value of bit 31 (FIT) of the HcFrameInterval register to this bit. Software compares the value of the FIT bits with the value of this FRT bit to confirm that the new FI bit value has been applied to the FR bits.	R
b30 to b14	Reserved		R
b13 to b0	FR	Frame Remaining. 14-bit down counter that shows the current frame value. As time passes, the value of this bit is decremented. When the count reaches 0000h, the frame value is reloaded. The host controller then copies the value of bits [13:0] (FI) of the HcFrameInterval register to this bit, and counting down starts again.	R

10.5.1.16 HCFMNUMBER — HcFrameNumber Register

Address: 4002 003Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	FRAMENUMBER															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.24 HCFMNUMBER Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved		R
b15 to b0	FRAMENUMBER	Number of frames that have passed. These bits are incremented when bits [13:0] (FR) of the HcFrameRemaining register reach 0000h.	R

10.5.1.17 HCPERIODICSTART — HcPeriodicStart Register

Address: 4002 0040h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	PERIODICSTART													
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.25 HCPERIODICSTART Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b14	Reserved		R
b13 to b0	PERIODICSTART	The time when the host controller starts periodic frame list processing within a frame. Specify a value for these bits by using software during host controller initialization. If the value of bits [13:0] (FR) of the HcFrameRemaining register is larger than the value specified for these bits, the non-periodic frame list takes precedence over the periodic frame list. The OHCI standard recommends the value of this field to be about 90% of the value of bits [13:0] (FI) of the HcFrameInterval register. Therefore, the recommended value is 2A2Fh.	R/W

10.5.1.18 HCLSTHRESHOLD — HcLSThreshold Register

Address: 4002 0044h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	HCLSTHRESHOLD											
Value after reset	0	0	0	0	0	1	1	0	0	0	1	0	1	0	0	0

Table 10.26 HCLSTHRESHOLD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b12	Reserved		R
b11 to b0	HCLSTHRESHOLD	These bits are used to make a threshold value which shows whether a transferring is possible in comparison with remaining frame time in a LS transfer. A LS transfer can be started when the FmRemaining value is larger than this value.	R/W

10.5.1.19 HCRHDESCRIPTORA — HcRhDescriptorA Register

Address: 4002 0048h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	POTPGT								—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	NOCP	OCPM	DT	NPS	PSM	NDP							
Value after reset	0	0	0	0	1	0	0	1	0	0	0	0	0	0	X	X

Table 10.27 HCRHDESCRIPTORA Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	POTPGT	Power On To Power Good Time. Indicates the time software has to wait to access after the root hub port is turned on. The time is shown in 2 ms units. Therefore, the wait time is POTPGT x 2 ms.	R/W
b23 to b13	Reserved	Should be 0 at writing.	R
b12	NOCP	No Over Current Protection. Specify whether to support the overcurrent status on the root hub port. 0: Support overcurrent status 1: Not support overcurrent status.	R/W
b11	OCPM	Over Current Protection Mode. Specify how the overcurrent status on the root hub port is reported. This bit must reflect the mode specified for the PSM bit. This bit is valid only when the NOCP bit is cleared. 0: The overcurrent status is reported for all ports 1: The overcurrent status is reported on a per-port basis.	R/W
b10	DT	Device Type. Indicates that the root hub is not a compound device. Because the root hub is not permitted to be a compound device, this bit is always read as 0b.	R
b9	NPS	No Power Switching. Specify how to control the port power. 0: The power can be switched on or off 1: The power remains on while the host controller is operating.	R/W
b8	PSM	Power Switching Mode. Specify how to control the power switch for each port off the root hub. If bits [18:17] (PPCM) of the HcRhDescriptorB register are set, each port responds to port power commands (Set/ClearPortPower) only. If the bits are cleared, each port is controlled by the global power switch Set/ClearGlobalPower. This bit is valid only when the NPS bit is cleared. 0: All ports are powered at the same time 1: Each port is powered individually.	R/W
b7 to b0	NDP	Number Downstream Port. Specify the number of downstream ports supported by the root hub in the host controller. 01h: 1-Host mode 02h: 2-Host mode	R

10.5.1.20 HCRHDESCRIPTORB — HcRhDescriptorB Register

Address: 4002 004Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	PPCM		—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	X	X	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	DR		—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.28 HCRHDESCRIPTORB Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b19	Reserved		R
b18, b17	PPCM	Port Power Control Mask. Power control command (Set/ClearGlobalPower) setting bit. Bit 17 → port1, Bit18 → port2. This bit is valid when bit 8 (PSM) of the HcRhDescriptorA register is set. 0: All ports are controlled 1: Only a corresponding port is controlled Value after reset 01b: 1-Host mode, 11b: 2-Host mode	R/W
b16 to b3	Reserved		R
b2, b1	DR	Device Removable. Indicates whether each root hub port is removable. Bit 1 → port1, Bit2 → port2. 0: Not Removable 1: Removable	R/W
b0	Reserved		R

10.5.1.21 HCRHSTATUS — HcRhStatus Register

Address: 4002 0050h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	CRWE	—	—	—	—	—	—	—	—	—	—	—	—	—	OCIC	R_LPSC__W_SGP
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	R_DRWE__W_SRWE	—	—	—	—	—	—	—	—	—	—	—	—	—	OCI	R_LPS__W_CGP
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.29 HCRHSTATUS Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31	CRWE	Clear Remote Wakeup Enable. Use this bit to clear the DRWE bit. Setting this bit clears the DRWE bit. Writing 0b has no effect.	W
b30 to b18	Reserved	Should be 0 at writing.	R
b17	OCIC	Over Current Indicate Change. Indicates that the OCI bit setting has changed. This bit is set when the OCI bit setting has changed. If 1b is written to this bit when it is already 1b, this bit is cleared. Writing 0b has no effect. 0: The overcurrent status has not changed 1: The overcurrent status has changed.	R/W
b16	R_LPSC__W_SGP	[On Read] Local Power Status Change. Because the local power status feature is not supported, this bit is always read as 0b. [On Write] Set Global Power. When this bit is set to 1b, ports are turned on. bit 8 (PSM) of the HcRhDescriptorA register and bits [18:17] (PPCM) of the HcRhDescriptorB register defines which port will be turned on.	R/W
b15	R_DRWE__W_SRWE	[On Read] Device Remote Wakeup Enable. Specify whether to include a source specified by bit 16 (CSC) of the HcRhPortStatus1 register as a remote wakeup event. If a ConnectStatusChange event occurs while this bit is set, the operating status changes from USB Suspend to USB Resume, and then a Resume Detect interrupt occurs. 0: ConnectStatusChange is not a Remote Wakeup event 1: ConnectStatusChange is a Remote Wakeup event. [On Write] Set Remote Wakeup Enable. Use this bit to set the DRWE bit. Setting this bit sets the DRWE bit. Writing 0b has no effect.	R/W
b14 to b2	Reserved		R

Value	PSM	PPCM[1:0]	Note
0	—	—	No Change
1	0	—	PPS in HcRhPortStatus1/HcRhPortStatus2 are set
	1	00b	PPS in HcRhPortStatus1/HcRhPortStatus2 are set
		10b	PPS in HcRhPortStatus1 are set
		01b	PPS in HcRhPortStatus2 are set
		11b	No Change

Table 10.29 HCRHSTATUS Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b1	OCI	Over Current Indicator. Indicates the overcurrent status in global overcurrent detection mode. If per-port overcurrent protection is specified, this bit is always 0b. 0: The port is normal 1: The port is in the overcurrent state	R
b0	R_LPS__W_CGP	[On Read] Local Power Status. Because the local power status feature is not supported, this bit is always read as 0b. [On Write] Clear Global Power. When this bit is set to 1b, ports are turned off. Which port is turned on is determined by bit 8 (PSM) of the HcRhDescriptorA register and bits [18:17] (PPCM) of the HcRhDescriptorB register.	R/W

10.5.1.22 HCRHPORTSTATUS1/ HCRHPORTSTATUS2 — HcRhPortStatus1/ HcRhPortStatus2 Register

Address: HCRHPORTSTATUS1: 4002 0054h
HCRHPORTSTATUS2: 4002 0058h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	PRSC	OCIC	PSSC	PESC	CSC
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	R_LSDA__W_CPP	R_PPS__W_SPP	—	—	—	R_PRS__W_SPR	R_POCI__W_CSS	R_PSS__W_SPS	R_PES__W_SPE	R_CCS__W_CPE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.30 HCRHPORTSTATUS1/ HCRHPORTSTATUS2 Register Contents (1/3)

Bit Position	Bit Name	Function	R/W
b31 to b21	Reserved	Should be 0 at writing.	R
b20	PRSC	Port Reset Status Change. Indicates the end of a port reset. The host controller set this bit when a 10 ms hardware reset finishes. When the software sets this bit to 1b, this bit is cleared. 0: The port reset has not finished or PRS has not changed 1: The port reset has finished.	R/W
b19	OCIC	Over Current Indicate Change. Indicates that a port has been detected in the overcurrent state. This bit is valid when reporting the overcurrent state on a per-port basis is specified (OCPM of HcRhDescriptorA register = 1). When the software sets this bit to 1b, this bit is cleared. 0: The overcurrent status has not changed 1: The overcurrent status has changed.	R/W
b18	PSSC	Port Suspend Status Change. Indicates the end of a resume sequence. This bit is set when all resume processing by hardware has finished. When the software sets this bit to 1b, this bit is cleared. This bit is also cleared when the PRSC bit is set. 0: Resume processing has not finished 1: Resume processing has finished.	R/W
b17	PESC	Port Enable Status Change. Indicates that the PES bit setting has changed. This bit is set when the port status has changed due to a hardware event such as an overcurrent state, disconnection, power off, or babble error. When the software sets this bit to 1b, this bit is cleared. 0: The PES status has not changed 1: The PES status has changed.	R/W
b16	CSC	Connect Status Change. Indicates that the CCS bit setting has changed. The host controller sets this bit when the CCS changes when a device is connected or disconnected. The host controller also sets this bit when a request such as port reset, port suspend, and port enable is received when a device is being disconnected, in order to make software re-evaluate the device connection. When the software sets this bit to 1b, this bit is cleared. 0: CCS status has not changed 1: CCS status has changed.	R/W
b15 to b10	Reserved		R

Table 10.30 HCRHPORTSTATUS1/ HCRHPORTSTATUS2 Register Contents (2/3)

Bit Position	Bit Name	Function	R/W
b9	R_LSDA__W_CPP	Read: Low Speed Device Attached. Indicates the speed of the device connected to the port. This bit is valid when the CCS bit is set. 0: FS device 1: LS device. Write: Clear Port Power. Use this bit to turn off the port power when power control on a per-port basis is specified. The port power is turned off by writing 1b. Writing 0b has no effect.	R/W
b8	R_PPS__W_SPP	Read: Port Power Status. Indicates the port power status. This bit is cleared when an overcurrent is detected. 0: Port power is off 1: Port power is on. Write: Set Port Power. Use this bit to turn on the port power when power control on a per-port basis is specified. The port power is turned on by writing 1b. Writing 0b has no effect.	R/W
b7 to b5	Reserved		R
b4	R_PRS__W_SPR	Read: Port Reset Status. Indicates the port reset status. This bit is cleared when the PRSC bit is set on completion of a 10 ms port reset. When the CCS bit is cleared (device disconnected), this bit cannot be set. 0: Port reset not performed 1: Port reset in progress Write: Set Port Reset. Use this bit to request a port reset for the downstream port. Writing 1b to this bit starts a 10 ms port reset. If this bit is written while the CCS bit is cleared, the PRS bit cannot be written. The host controller sets the CSC bit and reports to software that a reset was attempted on a port to which no device was connected. Writing 0b has no effect.	R/W
b3	R_POCI__W_CSS	Read: Port Over Current Indicator. Indicates that an overcurrent has occurred on the downstream port. This bit is valid when reporting the overcurrent state on a per-port basis is specified (OCPM of HcRhDescriptorA register = 1). This bit is set to 0b when reporting the overcurrent state collectively for all ports is specified. 0: The port is normal 1: The port is in the overcurrent state. Write: Clear Suspend Status. Use this bit to end the suspend state and start the resume sequence. Writing 1b to this bit starts the resume sequence. Writing 0b has no effect. The resume sequence starts only when the PSS bit is set.	R/W
b2	R_PSS__W_SPS	Read: Port Suspend Status. Indicates that a port is in the suspend state or the resume sequence has started. This bit is set by writing to the SPS bit. This bit cannot be set while the CCS bit is cleared (no device is connected). This bit is cleared at the following timing: - When the resume sequence finishes and the PSSC bit is set - When a port reset ends and the PRSC bit is set - In the USB Resume state: 0: Port is used for transfer normally 1: Port is in the suspend state. Write: Set Port Suspend. Use this bit to shift the port to the suspend state. Writing 1b to this bit shifts the port to the suspend state. Writing 0b has no effect. If this bit is written while the CCS bit is cleared, the CSC bit is set to notify the driver that an attempt was made to suspend a disconnected port.	R/W

Table 10.30 HCRHPORTSTATUS1/ HCRHPORTSTATUS2 Register Contents (3/3)

Bit Position	Bit Name	Function	R/W
b1	R_PES__W_SPE	Read: Port Enable Status. Indicates the port enable/disable status. The host controller clears this bit when an event such as the overcurrent state, disconnection, power off, and babble error is detected. The PESC bit is set at this time. This bit cannot be set while the CCS bit is cleared (no device is connected). This bit is set by the host controller when a port reset ends or the port suspend state ends. 0: Port is disabled 1: Port is enabled. Write: Set Port Enable. Use this bit to set the PES bit. Writing 1b to this bit enables ports. Writing 0b has no effect. Note: Shift the port state by using bit 4 (SPR) of the HcRhPortStatus1 register. The OHCI standard supports enabling ports by using the SPE bit, but the USB specification does not.	R/W
b0	R_CCS__W_CPE	Read: Current Connect Status. Indicates the current status of connection on the downstream port. 0: No device is connected 1: A device is connected. Write: Clear Port Enable. Use this bit to clear the PES bit. Writing 1b to this bit disables ports. Writing 0b has no effect.	R/W

10.5.2 EHCI Operation Register Description

10.5.2.1 CAPL_VERSION — HCIVERSION and CAPLENGTH Register (EHCI)

Address: 4002 1000h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	INTERFACE_VERSION_NUMBER															
Value after reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	CAPABILITY_REGISTERS_LENGTH							
Value after reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Table 10.31 CAPL_VERSION Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	INTERFACE_VERSION_NUMBER	Indicates the version of the EHCI Specification implemented in this host controller. “0100h” means this host controller is compliant with the EHCI Rev. 1.0.	R
b15 to b8	Reserved		R
b7 to b0	CAPABILITY_REGISTERS_LENGTH	Indicates the start address of the host controller operational register. “20h” means that operational registers are allocated from address 20h.	R

10.5.2.2 HCSPARAMS — HCSPARAMS Register

Address: 4002 1004h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	DEBUG_PORT_NUMBER				—	—	—	P_INDICATOR
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	N_CC				N_PCC				PORT_ROUTING_RULES	—	—	PPC	N_PORTS			
Value after reset	0	0	0	1	0	0	X	X	1	0	0	1	0	0	X	X

Table 10.32 HCSPARAMS Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved		R
b23 to b20	DEBUG_PORT_NUMBER	Indicates that the host controller ports are provided for debugging. “0000b” means that there is no debugging port.	R
b19 to b17	Reserved		R
b16	P_INDICATOR	Indicates whether the host controller supports port indicator control. “0b” means that this host controller does not support port indicator control.	R
b15 to b12	N_CC	Indicates the number of OHCI host controllers related to the EHCI host controller. “1b” means that one OHCI host controller is embedded.	R
b11 to b8	N_PCC	Indicates the number of ports supported by one OHCI host controller. This bit reflects the setting of bits [1:0] (PORT_NO) of the PCI Configuration EXT1 register.	R
b7	PORT_ROUTING_RULES	Indicates how each port is mapped on the OHCI host controller. “1b” means that mapping is shown in the HCSP_PORTROUTE register.	R
b6, b5	Reserved		R
b4	PPC	Indicates how the host controller controls the port power. This bit reflects the setting of bit 2 (PPCNT) of the PCI Configuration EXT1 register. “1b” means that this USB subsystem supports power supply control.	R
b3 to b0	N_PORTS	Indicates the number of physical downstream ports used in this USB subsystem. This bit reflects the setting of bits [1:0] (PORT_NO) of the PCI Configuration EXT1 register.	R

10.5.2.3 HCCPARAMS — HCCPARAMS Register

Address: 4002 1008h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	EECP								ISOCHRONOUS_SCHEDULING_THRESHOLD				—	ASYNCHRONOUS_SCHEDULE_PARK_CAPABILITY	PROGRAMMING_FRAME_LIST_FLAG	BIT64_ADDRESSING_CAPABILITY
Value after reset	1	1	1	0	1	0	0	0	0	0	0	0	0	1	1	0

Table 10.33 HCCPARAMS Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved		R
b15 to b8	EECP	Indicates the offset address of the EHCI Extend Capabilities Registers. Indicates that an extend register is allocated to address E8h in the EHCI configuration space. Since the host controller does not support legacy features, these bits do not have a meaning.	R
b7 to b4	ISOCHRONOUS_SCHEDULING_THRESHOLD	“0b” means that the host controller does not support the isochronous data structure cache of the overall frame.	R
b3	Reserved		R
b2	ASYNCHRONOUS_SCHEDULE_PARK_CAPABILITY	Indicates whether Park mode for High Speed QH (Queue Head) in the asynchronous schedule is supported. “1b” means that the host controller supports the feature.	R
b1	PROGRAMMING_FRAME_LIST_FLAG	Indicates the setting for the frame list size that can be used by software. This bit shows 1b. If this bit is set to 1b, the usable frame list size can be specified by using bits [3:2] (FRAME_LIST_SIZE) of the USBCMD register. A frame list size of 4 KB or smaller can be specified.	R
b0	BIT64_ADDRESSING_CAPABILITY	Indicates whether the data structure uses memory pointers of 32-bit address or 64-bit address. This bit shows 0b, which indicates that the host controller supports the data structure that uses 32-bit address memory pointers. 64-bit address memory pointers are not supported.	R

10.5.2.4 HCSP_PORTROUTE — HCSP_PORTROUTE Register

Address: 4002 100Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	COMPANION_PORT_ROUTE															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	COMPANION_PORT_ROUTE															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.34 HCSP_PORTROUTE Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	COMPANION_PORT_ROUTE	Indicates the port for which the OHCI host controller is responsible. Since one OHCI controller is embedded in the host controller, these bits are ALL0.	R

10.5.2.5 USBCMD — USBCMD Register

Address: 4002 1020h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	INTERRUPT_THRESHOLD_CONTROL							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	ASYNCHRONOUS_SCHEDULE_PARK_MODE_ENABLE	—	ASYNCHRONOUS_SCHEDULE_PARK_MODE_COUNT		LIGHT_HOST_CONTROLLER_RESET	INTERRUPT_ON_ASYNC_ADVANCE_DOORBELL	ASYNCHRONOUS_SCHEDULE_ENABLE	PERIODIC_SCHEDULE_ENABLE	FRAME_LIST_SIZE		HCRESET	RS
Value after reset	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0

Table 10.35 USBCMD Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved		R
b23 to b16	INTERRUPT_THRESHOLD_CONTROL	Indicates the maximum rate until the host controller generates an interrupt. Setting a value other than those shown below is not guaranteed. 00h: Reserved 01h: 1 micro-frame 02h: 2 micro-frames 04h: 4 micro-frames 08h: 8 micro-frames (1 ms) 10h: 16 micro-frames (2 ms) 20h: 32 micro-frames (4 ms) 40h: 64 micro-frames (8 ms).	R/W
b15 to b12	Reserved		R
b11	ASYNCHRONOUS_SCHEDULE_PARK_MODE_ENABLE	Specify whether to enable or disable Park mode. 0: Disable 1: Enable.	R/W
b10	Reserved	Should be 0 at writing.	R
b9, b8	ASYNCHRONOUS_SCHEDULE_PARK_MODE_COUNT	Specify the number of transactions that can be executed successively from one QH (queue head). The valid setting value is 1h to 3h. This bit is valid when bit 11 (ASYNCHRONOUS_SCHEDULE_PARK_MODE_ENABLE) is 1b.	R/W
b7	LIGHT_HOST_CONTROLLER_RESET	Indicates the status of executing a light host controller reset. This bit is fixed to 0b because the host controller does not support light host controller reset.	R/W
b6	INTERRUPT_ON_ASYNC_ADVANCE_DOORBELL	This bit is used by software as a doorbell. This bit is set to 1b by software to generate an interrupt when advancing to the next QH (queue head) processing. When bit 5 (INTERRUPT_ON_ASYNC_ADVANCE_ENABLE) of the USBINTR register is 1b, an interrupt is generated at the first timing to generate an interrupt after this bit is set to 1. Operation is not guaranteed if this bit is set while bit 5 (INTERRUPT_ON_ASYNC_ADVANCE_ENABLE) of the USBINTR register is 0b. This bit is cleared by the host controller. When one QH process is completed, the host controller clears this bit to 0b, and then sets bit 5 (INTERRUPT_ON_ASYNC_ADVANCE) of the USBSTS register to 1b.	R/W

Table 10.35 USBCMD Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b5	ASYNCHRONOUS_SCHEDULE_ENABLE	Specify whether to continue or skip asynchronous list processing. 0: Do not continue (skip) asynchronous list processing 1: Continue (do not skip) asynchronous list processing by using the ASYNCLISTADDR register.	R/W
b4	PERIODIC_SCHEDULE_ENABLE	Specify whether to continue or skip periodic list processing. 0: Do not continue (skip) periodic list processing 1: Continue (do not skip) periodic list processing by using the PERIODICLISTBASE register.This bit should be set to 1 in this controller.	R/W
b3, b2	FRAME_LIST_SIZE	Specify the frame list size. The value set to this bit determines the size of the Frame List Current index in the FRINDEX register. 00b: 1024 elements (4096 bytes) 01b: 512 elements (2048 bytes) 10b: 256 elements (1024 bytes) 11b: Reserved.	R/W
b1	HCRESET	Host Controller Reset. Use this bit to initialize the host controller. If this bit is set to 1b, the host controller initializes the internal pipelines and state machines. Communication being performed on the USB stops immediately. At this time, USB reset signaling for the downstream port is not performed. PCI configuration registers are not initialized by this reset, but EHCI operational registers are, and the port owner returns to the OHCI host controller. The host controller resets this bit to 0b when the reset ends. Writing 0b to this bit by software cannot stop the reset. Set this bit when bit 12 (HCHALTED) of the USBSTS register is 1b.	R/W
b0	RS	Run/Stop. Use this bit to run and stop the EHCI host controller. If this bit is set to 1b, the host controller starts operating. The host controller continues operating while this bit is 1b. Set this bit to 1b when the host controller is in the halt state. Bit 12 (HCHALTED) of the USBSTS register indicates that the host controller finished a transaction and stopped. 0: Stop (The host controller finished a transaction and has halted.) 1: Run (The host controller executes scheduling.)	R/W

10.5.2.6 USBSTS — USBSTS Register

Address: 4002 1024h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	ASYNCHRONOUS_SCHEDULE_STATUS	PERIODIC_SCHEDULE_STATUS	RECLAMATION	HCHALTED	—	—	—	—	—	—	INTERRUPT_ON_ASYNC_ADVANCE	HOST_SYSTEM_ERROR	FRAME_LIST_ROLLOVER	PORT_CHANGE_DETECT	USBERRINT	USBINT
Value after reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.36 USBSTS Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved	Should be 0 at writing.	R
b15	ASYNCHRONOUS_SCHEDULE_STATUS	Indicates the current asynchronous scheduling status. Asynchronous scheduling is enabled (1b) or disabled (0b) when the value of this bit and bit 5 (ASYNCHRONOUS_SCHEDULE_ENABLE) of the USBCMD register is the same. 0: Invalid 1: Valid	R
b14	PERIODIC_SCHEDULE_STATUS	Indicates the current periodic scheduling status. Periodic scheduling is enabled (1b) or disabled (0b) when the value of this bit and bit 4 (PERIODIC_SCHEDULE_ENABLE) of the USBCMD register is the same. 0: Invalid 1: Valid	R
b13	RECLAMATION	Use this bit to detect an empty asynchronous schedule. After a reset or when fetching a queue head in which H = 1b, the host controller clears this bit to 0b. The host controller sets this bit to 1b when executing an asynchronous transaction or when a trigger event has been detected. When a queue head with H = 1b is fetched while this bit is 0b, the host controller shifts to Async Sched Sleeping mode.	R
b12	HCHALTED	This bit shows 0b when bit 0 (RS) of the USBCMD register is 1b. When the RS bit is set to 0b by the host controller or software, the EHCI host controller stops executing and this bit is set to 1b by the host controller. 0: EHCI host controller is running 1: EHCI host controller is in halt	R
b11 to b6	Reserved	Should be 0 at writing.	R
b5	INTERRUPT_ON_ASYNC_ADVANCE	Indicates the Async Advance interrupt status. When a queue head is fetched, the host controller checks bit 6 (INTERRUPT_ON_ASYNC_ADVANCE_DOORBELL (IAAD)) of the USBCMD register. If the IAAD bit is 1b, the host controller clears the IAAD bit when processing of the queue head finishes and then sets this bit. If bit 5 (INTERRUPT_ON_ASYNC_ADVANCE_ENABLE) of the USBINTR register is 1b, an Async Advance interrupt is generated by this interrupt source at the first timing to generate an interrupt after this bit is set to 1b. Writing 1b by software clears this bit. Writing 0b has no effect. 0: No Async Advance has occurred 1: An Async Advance has occurred	R/W

Table 10.36 USBSTS Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b4	HOST_SYSTEM_ERROR	This bit is set to 1b when a serious error, such as a parity error in the PCI system, occurs on the host controller. When such an error occurs, the host controller clears bit 0 (RS) of the USBCMD register to 0b so that subsequent TD processing is not executed. Writing 1b by software clears this bit. Writing 0b has no effect. 0: No system error has occurred 1: A system error has occurred	R/W
b3	FRAME_LIST_ROLLOVER	The host controller sets this bit to 1b when the Frame Index bit of the FRINDEX register rolled over from its maximum value to 000h. The maximum value at which a rollover occurs depends on the setting of bits [3:2] (FRAME_LIST_SIZE) of the USBCMD register. Writing 1b by software clears this bit. Writing 0b has no effect. 0: The frame list count has not rolled over to be 000h 1: The frame list count rolled over to be 000h	R/W
b2	PORT_CHANGE_DETECT	Indicates the change in the port status. The host controller sets this bit to 1b when a port for which bit 13 (PORT_OWNER) of the PORTSC[n] register is set to 0b satisfies any of the following conditions. – Bit 1 (Connect Status Change) of the PORTSC[n] changed from 0b to 1b (connection or disconnection of a device was detected) – Bit 3 (Port Enable/Disable Change) of the PORTSC[n] changed from 0b to 1b (the port enable status was changed) – Bit 5 (Over-current Change) of the PORTSC[n] changed from 0b to 1b (an overcurrent was detected) – Bit 6 (Force Port Resume) of the PORTSC[n] changed from 0b to 1b (a J-K transition was detected on a suspended port). Writing 1b by software clears this bit. Writing 0b has no effect.	R/W
b1	USBERRINT	USB Error Interrupt. Indicates the termination of a USB transaction due to an error. The host controller sets this bit to 1b when a USB transaction terminated due to an error such as an underflow of the error counter. Writing 1b by software clears this bit. Writing 0b has no effect. 0: The USB transaction normally completed 1: The USB transaction terminated due to an error	R/W
b0	USBINT	USB Interrupt. Indicates the normal completion of a USB transfer. The host controller sets this bit to 1b when any of the following events occurs. – Completion of a USB transfer – Reception of a short packet. This bit is also set to 1b if IOC (Interrupt On Complete) of the TD is 1b, even if a USB transfer terminated due to an error. Writing 1b by software clears this bit. Writing 0b has no effect. 0: The USB transfer has not completed 1: The USB transfer completed	R/W

10.5.2.7 USBINTR — USBINTR Register

Address: 4002 1028h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	INTERRUPT_ON_ASYNC_ADVANCE_ENABLE	HOST_SYSTEM_ERROR_ENABLE	FRAME_LIST_ROLLOVER_ENABLE	PORT_CHANGE_INTERRUPT_ENABLE	USB_ERROR_INTERRUPT_ENABLE	USB_INTERRUPT_ENABLE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.37 USBINTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b6	Reserved	Should be 0 at writing.	R
b5	INTERRUPT_ON_ASYNC_ADVANCE_ENABLE	Enable of bit 5 of USBSTS. To clear the interrupt, use the USBSTS register. 0: Disable 1: Enable	R/W
b4	HOST_SYSTEM_ERROR_ENABLE	Enable of bit 4 of USBSTS. To clear the interrupt, use the USBSTS register. 0: Disable 1: Enable	R/W
b3	FRAME_LIST_ROLLOVER_ENABLE	Enable of bit 3 of USBSTS. To clear the interrupt, use the USBSTS register. 0: Disable 1: Enable	R/W
b2	PORT_CHANGE_INTERRUPT_ENABLE	Enable of bit 2 of USBSTS. To clear the interrupt, use the USBSTS register. 0: Disable 1: Enable	R/W
b1	USB_ERROR_INTERRUPT_ENABLE	Enable of bit 1 of USBSTS. To clear the interrupt, use the USBSTS register. 0: Disable 1: Enable	R/W
b0	USB_INTERRUPT_ENABLE	Enable of bit 0 of USBSTS. To clear the interrupt, use the USBSTS register. 0: Disable 1: Enable	R/W

10.5.2.8 FRINDEX — Frame Index Register

Address: 4002 102Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	FRAME_INDEX													
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.38 FRINDEX Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b14	Reserved	Should be 0 at writing.	R
b13 to b0	FRAME_INDEX	These bits are used to allocate an index to periodic frame list. The bits are incremented at the end of a micro-frame. Bits [N:3] of these bits are used as the current frame list index. This means that, before moving to the next index, the current frame list is accessed eight times. The value of N is determined by bits [3:2] (FRAME_LIST_SIZE) of the USBCMD register.	R/W

Frame List Size	Number Elements	N
00b	(1024)	12
01b	(512)	11
10b	(256)	10
11b	Reserved	

Accessing this register only when the host controller stops (bit 12 (HCHALTED) of the USBSTS register = 1b).
The value of this bit is reflected to SOF frame number in the SOF token.

10.5.2.9 CTRLDSSEGMENT — CTRLDSSEGMENT Register

Address: 4002 1030h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	CTRLDSSEGMENT															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	CTRLDSSEGMENT															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.39 CTRLDSSEGMENT Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	CTRLDSSEGMENT	This register is not used because this host controller does not support 64-bit addressing.	R

10.5.2.10 PERIODICLISTBASE — PERIODICLISTBASE Register

Address: 4002 1034h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	BASEADDRESS															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	BASEADDRESS				—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.40 PERIODICLISTBASE Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b12	BASEADDRESS	Indicates the base address of the periodic frame list on a system memory. The value of this register is loaded by software before the host controller starts list processing. The host controller determines the frame list to process based on these bits and Frame Index (bit[13:0]) of the FRINDEX register. Align the periodic frame list address on 4 KB boundary. No guarantee if these bits are changed during operation.	R/W
b11 to b0	Reserved	Should be 0 at writing.	R

10.5.2.11 ASYNCLISTADDR — ASYNCLISTADDR Register

Address: 4002 1038h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	LPL															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	LPL											—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.41 ASYNCLISTADDR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b5	LPL	Link Pointer Low. Indicates a system memory address of asynchronous queue head which will be processed next. Align the asynchronous queue head address on 32-byte boundary.	R/W
b4 to b0	Reserved	Should be 0 at writing.	R

10.5.2.12 CONFIGFLAG — CONFIGFLAG Register

Address: 4002 1060h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	CF
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.42 CONFIGFLAG Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b1	Reserved	Should be 0 at writing.	R
b0	CF	Configure Flag: Use this bit to control default port routing, OHCI or EHCI. Software sets this bit to 1b at the end of host controller configuration. 0: OHCI host controller 1: EHCI host controller	R/W

10.5.2.13 PORTSC1/PORTSC2 — PORTSC1/PORTSC2 Register

Address: PORTSC1: 4002 1064h
PORTSC2: 4003 1068h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	WKOC_E	WKDSCNNT_E	WKCNNT_E	PORT_TEST_CONTROL			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	PORT_INDICATOR_CONTROL		PORT_OWNER	PP	LINE_STATUS		—	PORT_RESET	SUSPEND	FORCE_PORT_RESUME	OVER_CURRENT_CHANGE	OVER_CURRENT_ACTIVE	PORT_ENABLE_DISABLE_CHANGE	PORT_ENABLED_DISABLED	CONNECT_STATUS_CHANGE	CURRENT_CONNECT_STATUS
Value after reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.43 PORTSC1/PORTSC2 Register Contents (1/3)

Bit Position	Bit Name	Function	R/W
b31 to b23	Reserved		R
b22	WKOC_E	Wake on Over-current Enable. If this bit is set to 1, an overcurrent state can be detected as a wakeup event. This bit does not affect the host controller operation. This bit is 0 when PP=0	R/W
b21	WKDSCNNT_E	Wake on Disconnect Enable. If this bit is set to 1, disconnecting a device can be detected as a wakeup event. This bit does not affect the host controller operation. This bit is 0 when PP=0	R/W
b20	WKCNNT_E	Wake on Connect Enable. If this bit is set to 1, connecting a device can be detected as a wakeup event. This bit does not affect the host controller operation. This bit is 0 when PP=0	R/W
b19 to b16	PORT_TEST_CONTROL	Port Test Control[3:0] 0000b: Normal 0001b: Test J_STATE 0010b: Test K_STATE 0011b: Test SE0_NAK 0100b: Test Packet 0101b: Test FORCE_ENABLE Other: Reserved	R/W
b15, b14	PORT_INDICATOR_CONTROL	Port Indicator Control[1:0] These bits show 00b, which indicates that the host controller does not support port indicator control.	R
b13	PORT_OWNER	Port Owner 0: EHCI host controller 1: OHCI host controller. This bit is cleared when CF(bit0) of CONFIGFLAG register changes from 0 to 1. If CF = 0, this bit is set to 1. If the connected device is not a high-speed device, software set this bit to 1 to change port ownership to the OHCI host controller.	R/W

Table 10.43 PORTSC1/PORTSC2 Register Contents (2/3)

<table>
<tr><th>Bit Position</th><th>Bit Name</th><th>Function</th><th>R/W</th></tr>
<tr><td>b12</td><td>PP</td><td>Port Power
0: Not supply power to ports
1: Supply power to ports.
If this bit is set to 0, USB ports do not function and therefore do not detect the connection or disconnection of a device. If an overcurrent is detected when this bit is 1, the host controller clears this bit to 0, which stops power supply to the port.
The function of this bit depends on the value of PPC of the HCSPARAMS register.
<table>
<tr><th>PPC</th><th>PP</th><th>Function</th></tr>
<tr><td>0</td><td>1</td><td>This bit stays 1 and power is always supplied to the port</td></tr>
<tr><td>1</td><td>0 or 1</td><td>Power supply to the port depends on the setting of this bit</td></tr>
</table></td><td>R/W</td></tr>
<tr><td>b11, b10</td><td>LINE_STATUS</td><td>Line Status[1:0]
<table>
<tr><th>Bit11(D+)</th><th>Bit10(D-)</th><th>USB State</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>SE0</td><td>Not low-speed device</td></tr>
<tr><td>0</td><td>1</td><td>K-state</td><td>Low-speed device is connected
Releases port owner ship from EHCI to OHCI</td></tr>
<tr><td>1</td><td>0</td><td>J-state</td><td>Not low-speed device
Performs EHCI port reset</td></tr>
<tr><td>1</td><td>1</td><td>Undefined</td><td>Not low-speed device
Performs EHCI port reset</td></tr>
</table>
Use this bit to detect LS device before starting a port reset sequence or a port enable sequence. This bit is valid when PORT_ENABLE_DISABLE_CHANGE = 0 and CURRENT_CONNECT_STATUS = 1.
This bit is 0 when PP = 0.</td><td>R</td></tr>
<tr><td>b9</td><td>Reserved</td><td></td><td>R</td></tr>
<tr><td>b8</td><td>PORT_RESET</td><td>Port Reset
0: Port is not being reset
1: Port is being reset.
Writing 1 to this bit while the bit is 0 causes USB bus reset sequence. It is necessary to write 0 in order to finish the bus reset. Note that this bit must be kept as 1 until bus reset sequence finishes according to USB 2.0 specification.
If HCHALTED (bit12) of USBSTS register is 1, software must not reset the port.
This bit is set to 0 if any of the following conditions is satisfied.
– PP = 0
– PORT_OWNER = 1
– CURRENT_CONNECT_STATUS = 0</td><td>R/W</td></tr>
<tr><td>b7</td><td>SUSPEND</td><td>Port suspend status.
0: Not suspend state
1: Suspend state.
This bit and PORT_ENABLED_DISABLED determine the port status.
<table>
<tr><th>PORT_ENABLED_DISABLED</th><th>Suspend</th><th>Port state</th></tr>
<tr><td>0</td><td>X</td><td>Disable</td></tr>
<tr><td>1</td><td>0</td><td>Enable</td></tr>
<tr><td>1</td><td>1</td><td>Suspend</td></tr>
</table>
In suspend state, data transfer for the downstream port is blocked except for port reset. If this bit is set to 1 during a data transfer, the status reflection and data transfer will be blocked after the current transfer finished.
This bit is set to 1 by software, and it is possible only when PP=1, PORT_OWNER =0, and CURRENT_CONNECT_STATUS =1 in host controller registers.
This bit is unconditionally cleared under the following conditions:
– FORCE_PORT_RESUME is cleared by software.
– PORT_RESET is set to 1 by software.
This bit is 0 when PP = 0.</td><td>R/W</td></tr>
</table>

Table 10.43 PORTSC1/PORTSC2 Register Contents (3/3)

Bit Position	Bit Name	Function	R/W
b6	FORCE_PORT_RESUME	Force Port Resume 0: No resume signal (K state) has been detected or output 1: Resume signal (K state) has been detected or output. If the port state moves from J to K during the suspend state (Remote WakeUp), the host controller sets this bit and PORT_CHANGE_DETECT (bit2) of USBSTS register to 1. In addition, software sets this bit to 1 in order to output a resume signal. At this time, do not set the PORT_CHANGE_DETECT bit. While this bit is 1, a resume signal (FS K State) is driven to the USB port. This bit must be cleared to 0 after an appropriate time has passed. Writing 0 to this bit while the bit is 1, the port returns to HS Idle. This bit is 1 until the port returns to HS Idle. This bit is 0 when PP = 0.	R/W
b5	OVER_CURRENT_CHANGE	Over-Current Change 0: No change 1: OVER_CURRENT_ACTIVE has changed. This bit can be cleared by writing 1 by software. Writing 0 has no effect.	R/W
b4	OVER_CURRENT_ACTIVE	Over-Current Active 0: Not overcurrent state 1: Overcurrent state. The host controller clears the PP and related bits, then sets this bit to 1 when an overcurrent status is detected. This bit is cleared when the overcurrent state ends.	R
b3	PORT_ENABLE_DISABLE_CHANGE	Port Enable/Disable Change 0: No change 1: Port status has changed from Enable to Disable. The host controller disables the port, then sets this bit to 1 when a frame babble is detected. This bit can be cleared by writing 1 by software. Writing 0 has no effect. This bit is 0 when PP = 0.	R/W
b2	PORT_ENABLED_DISABLED	Port Enabled/Disabled 0: Port disable 1: Port enable When the host controller resets the port and recognizes that the connected device is a HS device, it enables the port and then sets this bit to 1. This bit can't be set to 1 by software. When the host controller detects a disconnection of a device or an error, it disables the ports and then clears this bit to 0. The port is also disabled when this bit is set to 0 by software. Note that writing to this bit is not reflected until the port status is changed in actual. Data transfer for the downstream port is blocked, except for port reset when the port is disabled. This bit is 0 when PP=0. Port is enabled regardless of the port status, and this bit is set to 1 when PORT_TEST_CONTROL [3:0] = 0101b (Test FORCE_ENABLE).	R/W
b1	CONNECT_STATUS_CHANGE	Connect Status Change 0: No change 1: CURRENT_CONNECT_STATUS has changed. This bit can be cleared by writing 1 by software. Writing 0 has no effect. This bit is 0 when PP=0.	R/W
b0	CURRENT_CONNECT_STATUS	Current Connect Status 0: No device is connected to the port 1: Device is connected to the port Even if no device is connected, this bit is 1 when PORT_TEST_CONTROL[3:0] = 0101b (Test FORCE_ENABLE). This bit is 0 when PP=0 or PORT_OWNER=0.	R

10.5.3 OHCI (PCI Configuration Space) Register Description

10.5.3.1 VID_DID — Device ID - Vendor ID (OHCI)

Address: 4003 0000h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DEVICE_ID															
Value after reset	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	VENDOR_ID															
Value after reset	0	0	0	1	0	0	0	0	0	0	1	1	0	0	1	1

Table 10.44 VID_DID Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	DEVICE_ID	Indicates the device type. This field is used to select a driver specified by the PCI specification. It is not necessary for this host controller.	R
b15 to b0	VENDOR_ID	Indicates the device vendor. This field is used to select a driver specified by the PCI specification. It is not necessary for this host controller.	R

10.5.3.2 CMND_STS — Status - Command (OHCI)

Address: 4003 0004h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DETECTED_PARITY_ERROR	SIGNALED_SYSTEM_ERROR	RECEIVED_MASTER_ABORT	RECEIVED_TARGET_ABORT	SIGNALED_TARGET_ABORT	DEVSEL_TIMING		DATA_PARITY_ERROR_DETECTED	FAST_BACK_TO_BACK_CAPABLE	—	—	CAPABILITIES_LIST	—	—	—	—
Value after reset	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	FAST_BACK_TO_BACK_ENABLE	SERR_ENABLE	WAIT_CYCLE_CONTROL	PARITY_ERROR_RESPONSE	VGA_PALETTE_SNOOP	MEMORY_WRITE_AND_INVALIDATE_ENABLE	SPECIAL_CYCLE	BUS_MASTER	MEMORY_SPACE	I_O_SPACE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.45 CMND_STS Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31	DETECTED_PARITY_ERROR	Indicates the parity error status. This bit is set when an address or data parity error is detected, cleared when 1b is written via the PCI bus.	R/W
b30	SIGNALED_SYSTEM_ERROR	Indicates the SERR status. This bit is set when a system error occurs, cleared when 1b is written via the PCI bus.	R/W
b29	RECEIVED_MASTER_ABORT	Indicates the Master-Master Abort status. This bit is set to 1b when the bus cycle executed by the host controller serving as a master terminates due to Master Abort signaling, cleared when 1b is written via the PCI bus.	R/W
b28	RECEIVED_TARGET_ABORT	Indicates the Master-Target Abort status. This bit is set to 1b when the bus cycle executed by the host controller serving as a master terminates due to Target Abort signaling, cleared when 1b is written via the PCI bus.	R/W
b27	SIGNALED_TARGET_ABORT	Indicates the Slave Target Abort status. This bit is set to 1b when the bus cycle of the host controller serving as a slave being accessed is terminated due to Target Abort signaling. This bit is cleared when 1b is written via the PCI bus.	R/W
b26, b25	DEVSEL_TIMING	Indicates the DEVSEL response speed. This bit is fixed to 01b (mid-speed response).	R
b24	DATA_PARITY_ERROR_DETECTED	This bit is set when the host controller serving as a master detects a parity error, cleared when 1b is written via the PCI bus. This bit is fixed to 0b when the Parity Error Response bit is set to 0.	R/W
b23	FAST_BACK_TO_BACK_CAPABLE	Indicates whether Fast Back to Back is supported. This bit is fixed to 0b because Fast Back to Back is not supported.	R
b22, b21	Reserved	Should be 0 at writing.	R
b20	CAPABILITIES_LIST	Indicates whether power management mode is supported. This bit is fixed to 1b.	R
b19 to b10	Reserved	Should be 0 at writing.	R
b9	FAST_BACK_TO_BACK_ENABLE	Use this bit to enable Fast Back to Back. This bit is fixed to 0b because the host controller does not support Fast Back to Back.	R
b8	SERR_ENABLE	Use this bit to enable system error response. 0: Do not assert SERR0 1: Assert SERR0 Set this bit to 1b to signal the system error by using the SERR signal.	R/W

Table 10.45 CMND_STS Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b7	WAIT_CYCLE_CONTROL	Use this bit to enable wait cycle control. This bit is fixed to 0b because the host controller does not support Address/Data Stepping.	R
b6	PARITY_ERROR_RESPONSE	Use this bit to enable parity error response. 0: Do not assert PERR0 1: Assert PERR0 The DETECTED_PARITY_ERROR bit is set to 1b even if this bit is 0 when a parity error is detected.	R/W
b5	VGA_PALETTE_SNOOP	Use this bit to enable VGA Palette Snoop. This bit is fixed to 0b (not supported)	R
b4	MEMORY_WRITE_AND_INVALIDATE_ENABLE	Use this bit to enable Memory Write and Invalidate. This bit should be remained as initial value,0	R/W
b3	SPECIAL_CYCLE	Use this bit to enable Special Cycle. This bit is fixed to 0b (not supported)	R
b2	BUS_MASTER	Use this bit to enable the bus master. This is a signal to enable master access to the PCI bus. Set this bit to 1b when accessing SRAM on the system bus. Set this bit to 1b during initialization.	R/W
b1	MEMORY_SPACE	Use this bit to enable accessing to the memory spaces. This is a signal to enable memory access prescribed in the PCI specification. Set this bit to 1b during initialization.	R/W
b0	I_O_SPACE	Use this bit to enable accessing to the I/O spaces. This bit is fixed to 0b because the host controller does not acknowledge I/O accesses.	R

10.5.3.3 REVID_CC — Class Code - Revision ID (OHCI)

Address: 4003 0008h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	BASE_CLASS								SUB_CLASS							
Value after reset	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	PROGRAMMING_I_F								REVISION_ID							
Value after reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1

Table 10.46 REVID_CC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	BASE_CLASS	Indicates the base class defined in the PCI specification. “0Ch” means that a serial peripheral bus controller.	R
b23 to b16	SUB_CLASS	Indicates the subclass defined in the PCI specification. “03h” means that USB device.	R
b15 to b8	PROGRAMMING_I_F	Indicates the program interface defined in the PCI specification. “10h” means that OHCI.	R
b7 to b0	REVISION_ID	Indicates the host controller revision. This bit is fixed to 01h.	R

10.5.3.4 CLS_LT_HT_BIST — BIST - Header Type - Latency Timer - Cache Line Size (OHCI)

Address: 4003 000Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	BIST								HEADER_TYPE							
Value after reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	LATENCY_TIMER								CACHE_LINE_SIZE							
Value after reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Table 10.47 CLS_LT_HT_BIST Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	BIST	This bit is used for self-testing. This bit is fixed to 00h because the host controller does not support self-testing.	R
b23 to b16	HEADER_TYPE	Use this bit to report the header type to the system. Bits [22:16] are fixed to 00h because the header type is Type0. Bit 23 is fixed to 1b because the multifunction device is used.	R
b15 to b8	LATENCY_TIMER	Use this bit to report the latency timer to the system. The lowest 2 bits are fixed to 00b.	R/W
b7 to b0	CACHE_LINE_SIZE	Use this bit to report the cache line size to the system.	R/W

10.5.3.5 BASEAD — OHCI Base Address

Address: 4003 0010h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	OHCI_BASE_ADDRESS															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	OHCI_BASE_ADDRESS												PREFETCHABLE	TYPE		MEMORY_SPACE_INDICATOR
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.48 BASEAD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	OHCI_BASE_ADDRESS	Use bits [31:12] to specify the base address of the operational register. Set the operational register base address value. Bits [11:4] is fixed to 00h. i.e. operational registers are allocated to a 4 KB address space.	R/W
b3	PREFETCHABLE	This bit is fixed to 0b because the host controller does not support prefetching in memory read cycles.	R
b2, b1	TYPE	Indicates that the base address of the OHCI operational registers is 32-bit width, and thus the registers can be allocated to any location in a 32-bit memory space. This field is fixed to 00h.	R
b0	MEMORY_SPACE_INDICATOR	Indicates that the OHCI operational registers are mapped on a system memory space. This bit is fixed to 0b.	R

10.5.3.6 SSVID_SSID — Subsystem ID - Subsystem Vendor ID (OHCI)

Address: 4003 002Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SUBSYSTEM_ID															
Value after reset	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SUBSYSTEM_VENDOR_ID															
Value after reset	0	0	0	1	0	0	0	0	0	0	1	1	0	0	1	1

Table 10.49 SSVID_SSID Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	SUBSYSTEM_ID	Indicates the device type. This field is used to select a driver specified by the PCI specification. It is not necessary for this host controller.	R
b15 to b0	SUBSYSTEM_VENDOR_ID	Indicates the device vendor. This field is used to select a driver specified by the PCI specification. It is not necessary for this host controller.	R

10.5.3.7 EROM_BASEAD — Expansion ROM Base Address (OHCI)

Address: 4003 0030h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	EXPANSION_ROM_BASE_ADDRESS															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	EXPANSION_ROM_BASE_ADDRESS						—	—	—	—	—	—	—	—	—	ROM_DECODE_ENABLE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.50 EROM_BASEAD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b10	EXPANSION_ROM_BASE_ADDRESS	This field is fixed to 000000h because decoding the expansion ROM is prohibited.	R
b9 to b1	Reserved		R
b0	ROM_DECODE_ENABLE	This bit is fixed to 0b because decoding the expansion ROM is prohibited.	R

10.5.3.8 CAPPTR — Capability Pointer (OHCI)

Address: 4003 0034h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	CAPABILITY_POINTER							
Value after reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Table 10.51 CAPPTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7 to b0	CAPABILITY_POINTER	This is a pointer to the capability identifier. “40h” means that the identifier is allocated to 40h in the host controller.	R

10.5.3.9 INTR_LINE_PIN — Max_Lat - Min_Gnt - Interrupt Pin - Interrupt Line (OHCI)

Address: 4003 003Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	MAX_LATENCY								MIN_GNT							
Value after reset	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	INTERRUPT_PIN								INTERRUPT_LINE							
Value after reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Table 10.52 INTR_LINE_PIN Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	MAX_LATENCY	Indicates the maximum acquisition frequency of the PCI bus. This host controller is implemented as 2Ah	R
b23 to b16	MIN_GNT	Indicates the minimum burst transfer time. This host controller is implemented as 01h	R
b15 to b8	INTERRUPT_PIN	Indicates the interrupt output pin. This bit is fixed at 01h since INTA is used.	R
b7 to b0	INTERRUPT_LINE	Indicates the interrupt line. Keep 00h in this host controller	R/W

10.5.3.10 CAPID_NIP_PMCAP — Capability Identifier - Next Item Pointer - Power Management Capabilities (OHCI)

Address: 4003 0040h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	PME_SUPPORT					D2_SUPPORT	D1_SUPPORT	AUX_CURRENT			DSI	—	PME_CLK	VERSION		
Value after reset	0	1	1	1	1	1	1	0	0	0	0	0	0	0	1	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	NEXT_ITEM_POINTER								CAPABILITY_IDENTIFIER							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 10.53 CAPID_NIP_PMCAP Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b27	PME_SUPPORT	[31]: Indicates whether the D3 Cold state is supported. This bit is fixed to 0b because the D3 Cold state is not supported [30:27]: Indicates that PME interrupt generation is supported in all PCI power states (D0 to D3). This bit is fixed to 1111b.	R
b26	D2_SUPPORT	Power Management Capabilities - Indicates that the PCI power state D2 is supported. This bit is fixed to 1b.	R
b25	D1_SUPPORT	Power Management Capabilities - Indicates that the PCI power state D1 is supported. This bit is fixed to 1b.	R
b24 to b22	AUX_CURRENT	Power Management Capabilities - Indicates the specified current value required for the 3.3 V auxiliary power supply. Generating PME interrupts in the D3 Cold state is not supported, therefore these bits are fixed to 000b.	R
b21	DSI	Power Management Capabilities - Indicates that no special initialization is required for using power management. This bit is fixed to 0b.	R
b20	Reserved		R
b19	PME_CLK	Power Management Capabilities - Indicates that USB_PCICLK is not required for generating PME interrupts. This bit is fixed to 0b.	R
b18 to b16	VERSION	Power Management Capabilities - Indicates that this system is compliant with PCI power management interface specification release 1.1. This bit is fixed to 010b.	R
b15 to b8	NEXT_ITEM_POINTER	Indicates that there is no subsequent item. This bit is fixed to 00h.	R
b7 to b0	CAPABILITY_IDENTIFIER	Indicates the PCI power management register ID. This bit is fixed to 01h.	R

10.5.3.11 PMC_STS_PMCSR — Power Management Control and Status - PMCSR Bridge Support Extensions (OHCI)

Address: 4003 0044h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DATA								BPCC_ENABLE	B2_B3	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	PME_STATUS	DATA_SCALE						PME_ENABLE	—	—	—	—	—	—	POWER_STATE	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.54 PMC_STS_PMCSR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	DATA	This bit is fixed to 00h. This is an optional field in the PCI specification and is not supported in this host controller.	R
b23	BPCC_ENABLE	This bit is fixed to 0b. This is a bit for the bridge and is not supported in this host controller.	R
b22	B2_B3	This bit is fixed to 0b. This is a bit for the bridge and is not supported in this host controller.	R
b21 to b16	Reserved	Should be 0 at writing.	R
b15	PME_STATUS	PME interrupt status. This bit is set to 1b when a PME generation condition is satisfied. PME generation condition: Bit 3 (RD) of the HcInterruptStatus register is set to 1b while bit 10 (RWE) of the HcControl register is 1b. This bit is cleared to 0b when 1b is written via the PCI bus.	R/W
b14 to b9	DATA_SCALE	This bit is fixed to 00b. This is an optional field in the PCI specification and is not supported in this host controller.	R
b8	PME_ENABLE	Enable of PME. If this bit is set to 1b, a PME interrupt is generated when the system returns from power management.	R/W
b7 to b2	Reserved	Should be 0 at writing.	R
b1, b0	POWER_STATE	Indicates the PCI power status. 00b: D0 State 01b: D1 State 10b: D2 State 11b: D3 hot State.	R/W

10.5.3.12 EXT1 — EXT1 Register (OHCI)

Address: 4003 00E0h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	POTPGT								HYPER_SPEED_TRANSFER_CONTROL_2					—	—	—
Value after reset	0	0	0	0	1	1	1	1	0	0	0	1	0	1	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	HYPER_SPEED_TRANSFER_CONTROL_1	—	—	—	—	—	ID_WRITE_ENABLE	—	—	—	—	PPCNT	PORT_NO	
Value after reset	0	0	1	1	0	0	1	1	0	0	0	1	1	1	X	X

Table 10.55 EXT1 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	POTPGT	Specify the setting for bits [31:24] (POTPGT) of the OHCI HcRhDescriptorA register. POTPGT is the required waiting time for software access after root hub is powered on.	R/W
b23 to b19	HYPER_SPEED_TRANSFER_CONTROL_2	Setting other than 02h (HS Asynchronous FIFO threshold = 64 bytes) is prohibited.	R/W
b18 to b14	Reserved	Keep the initial value	R/W
b13	HYPER_SPEED_TRANSFER_CONTROL_1	HS Async OUT advance Mode Specify the hyper-speed transfer mode feature used for asynchronous OUT transfer. Setting this bit to 1b enables this feature (thus increasing the transfer rate).	R/W
b12 to b8	Reserved	Keep the initial value	R/W
b7	ID_WRITE_ENABLE	Control write protection for parameters Subsystem ID, Subsystem Vendor ID, Max Latency, and Min Gnt. 0: Write protected 1: Enable write.	R/W
b6 to b3	Reserved	Keep the initial value	R/W
b2	PPCNT	Specify the setting for bit 4 (PPC) of the EHCI HCSPARAMS register. 0: Set the PPC bit to 0b. This indicates that the system that incorporates the host controller does not have port power control switches and port power is always on 1: Set the PPC bit to 1b. This indicates that the system that incorporates the host controller has a port power control switch. Set this bit to 0b if the port power is always on, or set bit 9 (NPS) of the OHCI HcRhDescriptorA register to 1b.	R/W
b1, b0	PORT_NO	Specify the number of valid USB downstream ports. Setting / Valid port 1h: Port 1 2h: Ports 1 and 2 Other: Reserved. Do not use a value other than 1h or 2h Value after reset 1-Host mode: 1h 2-Host mode: 2h	R/W

10.5.3.13 EXT2 — EXT2 Register (OHCI)

Address: 4003 00E4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	PLL_UNLOCK_ACCESS_MODE	—	—	—	—	—	RAM_CONNECT_CHECK_RESULT	RAM_CONNECT_CHECK_END_FLAG	RUN_RAM_CONNECT_CHECK
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	HYPER_SPEED_TRANSFER_CONTROL_3	EHCI_MASK
Value after reset	0	1	1	0	1	1	0	0	0	0	0	0	0	0	1	0

Table 10.56 EXT2 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	Should be 0 at writing.	R
b24	PLL_UNLOCK_ACCESS_MODE	Set the response mode for register access during USBPLL unlock 0: Wait until USBPLL lock 1: Return dummy value until USBPLL has been locked	R/W
b23 to b19	Reserved	Keep the initial value	R/W
b18	RAM_CONNECT_CHECK_RESULT	Indicates the result of RAM connection check. 0: NG 1: OK This bit is valid when the RAM_CONNECT_CHECK_END_FLAG bit is 1b. Once a connection check is performed, this flag is not cleared until the RUN_RAM_CONNECT_CHECK bit changes from 0b to 1b.	R
b17	RAM_CONNECT_CHECK_END_FLAG	Indicates the end of RAM connection check. 0: Connection check has not been performed/has not finished 1: Connection check finished. After the RUN_RAM_CONNECT_CHECK bit is set to 1b, RAM connection check starts, a certain period (about 2 us) passes, and then this bit is asserted.	R
b16	RUN_RAM_CONNECT_CHECK	Use this bit to trigger a RAM connection check. Set this bit to 1b to start a RAM connection check. This bit is not cleared automatically when the check finishes. To execute checking again, write 0b to this bit to clear it, and then write 1b again. The connection check circuit is reset when this bit is set to 1b, and the RAM_CONNECT_CHECK_END_FLAG bit and RAM_CONNECT_CHECK_RESULT bit are cleared.	R/W
b15 to b2	Reserved	Keep the initial value	R/W
b1	HYPER_SPEED_TRANSFER_CONTROL_3	Specify the hyper-speed transfer mode feature used for asynchronous IN/OUT transfer. Setting this bit to 1b enables this feature (thus increasing the transfer rate).	R/W
b0	EHCI_MASK	Mask (enable) of EHCI host controller. 0: Enable the EHCI host controller 1: Disable the EHCI host controller. If this bit is set to 1, accessing registers in PCI configuration space and memory space in the EHCI is disabled, so the EHCI host controller does not run.	R/W

10.5.3.14 UTMICTRL — USBPHY Operation Mode Control Register (OHCI)

Address: 4003 00F4h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	REPSEL	
Value after reset	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.57 UTMICTRL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b18	Reserved	Keep the initial value	R/W
b17, b16	REPSEL	Specify the interval of periodic terminal resistance adjustment. Set this bit to 00b during initialization of the host controller.	R/W
b15 to b0	Reserved	Keep the initial value	R/W

10.5.4 EHCI (PCI Configuration Space) Register Description

10.5.4.1 VID_DID — Device ID - Vendor ID (EHCI)

Address: 4003 0100h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DEVICE_ID															
Value after reset	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	VENDOR_ID															
Value after reset	0	0	0	1	0	0	0	0	0	0	1	1	0	0	1	1

Table 10.58 VID_DID Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	DEVICE_ID	Indicates the device type. This field is used to select a driver specified by the PCI specification. It is not necessary for this host controller.	R
b15 to b0	VENDOR_ID	Indicates the device vendor. This field is used to select a driver specified by the PCI specification. It is not necessary for this host controller.	R

10.5.4.2 CMND_STS — Status - Command (EHCI)

Address: 4003 0104h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DETECTED_PARITY_ERROR	SIGNALED_SYSTEM_ERROR	RECEIVED_MASTER_ABORT	RECEIVED_TARGET_ABORT	SIGNALED_TARGET_ABORT	DEVSEL_TIMING		DATA_PARITY_ERROR_DETECTED	FAST_BACK_TO_BACK_CAPABLE	—	MHZ66_CAPABLE	CAPABILITIES_LIST	—	—	—	—
Value after reset	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	FAST_BACK_TO_BACK_ENABLE	SERR_ENABLE	WAIT_CYCLE_CONTROL	PARITY_ERROR_RESPONSE	VGA_PALETTE_SNOOP	MEMORY_WRITE_AND_INVALIDATE_ENABLE	SPECIAL_CYCLE	BUS_MASTER	MEMORY_SPACE	I_O_SPACE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.59 CMND_STS Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31	DETECTED_PARITY_ERROR	Indicates the parity error status. This bit is set when an address or data parity error is detected, cleared when 1b is written via the PCI bus.	R/W
b30	SIGNALED_SYSTEM_ERROR	Indicates the SERR status. This bit is set when a system error occurs, cleared when 1b is written via the PCI bus.	R/W
b29	RECEIVED_MASTER_ABORT	Indicates the Master-Master Abort status. This bit is set to 1b when the bus cycle executed by the host controller serving as a master terminates due to Master Abort signaling, cleared when 1b is written via the PCI bus.	R/W
b28	RECEIVED_TARGET_ABORT	Indicates the Master-Target Abort status. This bit is set to 1b when the bus cycle executed by the host controller serving as a master terminates due to Target Abort signaling, cleared when 1b is written via the PCI bus.	R/W
b27	SIGNALED_TARGET_ABORT	Indicates the Slave Target Abort status. This bit is set to 1b when the bus cycle of the host controller serving as a slave being accessed is terminated due to Target Abort signaling. This bit is cleared when 1b is written via the PCI bus.	R/W
b26, b25	DEVSEL_TIMING	Indicates the DEVSEL response speed. This bit is fixed to 01b (mid-speed response).	R
b24	DATA_PARITY_ERROR_DETECTED	This bit is set when the host controller serving as a master detects a parity error, cleared when 1b is written via the PCI bus. This bit is fixed to 0b when the Parity Error Response bit is set to 0.	R/W
b23	FAST_BACK_TO_BACK_CAPABLE	Indicates whether Fast Back to Back is supported. This bit is fixed to 0b because Fast Back to Back is not supported.	R
b22	Reserved	Should be 0 at writing.	R
b21	MHZ66_CAPABLE	Indicates 66 MHz operation capability. This bit is fixed at 0b, i.e. 33 MHz only.	R
b20	CAPABILITIES_LIST	Indicates whether power management mode is supported. This bit is fixed to 1b.	R
b19 to b10	Reserved	Should be 0 at writing.	R
b9	FAST_BACK_TO_BACK_ENABLE	Use this bit to enable Fast Back to Back. This bit is fixed to 0b because the host controller does not support Fast Back to Back.	R

Table 10.59 CMND_STS Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b8	SERR_ENABLE	Use this bit to enable system error response. 0: Do not assert SERR0 1: Assert SERR0 Set this bit to 1b to signal the system error by using the SERR signal.	R/W
b7	WAIT_CYCLE_CONTROL	Enables Wait Cycle Control. This bit is always 0b since the host controller does not support Address or Data Stepping.	R
b6	PARITY_ERROR_RESPONSE	Use this bit to enable parity error response. 0: Do not assert PERR0 1: Assert PERR0 The DETECTED_PARITY_ERROR bit is set to 1b even if this bit is 0 when a parity error is detected.	R/W
b5	VGA_PALETTE_SNOOP	Use this bit to enable VGA Palette Snoop. This bit is fixed to 0b (not supported)	R
b4	MEMORY_WRITE_AND_INVALIDATE_ENABLE	Use this bit to enable Memory Write and Invalidate. This bit should be remained as initial value,0	R/W
b3	SPECIAL_CYCLE	Use this bit to enable Special Cycle. This bit is fixed to 0b (not supported)	R
b2	BUS_MASTER	Use this bit to enable the bus master. This is a signal to enable master access to the PCI bus. Set this bit to 1b when accessing SRAM on the system bus. Set this bit to 1b during initialization.	R/W
b1	MEMORY_SPACE	Use this bit to enable accessing to the memory spaces. This is a signal to enable memory access prescribed in the PCI specification. Set this bit to 1b during initialization.	R/W
b0	I_O_SPACE	Use this bit to enable accessing to the I/O spaces. This bit is fixed to 0b because the host controller does not acknowledge I/O accesses.	R

10.5.4.3 REVID_CC — Class Code - Revision ID (EHCI)

Address: 4003 0108h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	BASE_CLASS								SUB_CLASS							
Value after reset	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	PROGRAMMING_I_F								REVISION_ID							
Value after reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 10.60 REVID_CC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	BASE_CLASS	Indicates the base class defined in the PCI specification. “0Ch” means that a serial peripheral bus controller.	R
b23 to b16	SUB_CLASS	Indicates the subclass defined in the PCI specification. “03h” means that USB device.	R
b15 to b8	PROGRAMMING_I_F	Indicates the program interface defined in the PCI specification. “20h” means that EHCI.	R
b7 to b0	REVISION_ID	Indicates the host controller revision. This bit is fixed to 01h.	R

10.5.4.4 CLS_LT_HT_BIST — BIST - Header Type - Latency Timer - Cache Line Size (EHCI)

Address: 4003 010Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	BIST								HEADER_TYPE							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	LATENCY_TIMER								CACHE_LINE_SIZE							
Value after reset	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Table 10.61 CLS_LT_HT_BIST Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	BIST	This bit is used for self-testing. This bit is fixed to 00h because the host controller does not support self-testing.	R
b23 to b16	HEADER_TYPE	Use this bit to report the header type to the system. Bits [22:16] are fixed to 00h because the header type is Type0. Bit 23 is fixed to 0b because the multifunction device is not used.	R
b15 to b8	LATENCY_TIMER	Use this bit to report the latency timer to the system. The lowest 2 bits are fixed to 00b.	R/W
b7 to b0	CACHE_LINE_SIZE	Use this bit to report the cache line size to the system.	R/W

10.5.4.5 BASEAD — EHCI Base Address

Address: 4003 0110h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	EHCI_BASE_ADDRESS															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	EHCI_BASE_ADDRESS												PREFETCHABLE	TYPE		MEMORY_SPACE_INDICATOR
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.62 BASEAD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b4	EHCI_BASE_ADDRESS	Use bits [31:8] to specify the base address of the operational register. Set the operational register base address value. Bits [7:4] is fixed to 00h. i.e. operational registers are allocated to a 256-byte address space.	R/W
b3	PREFETCHABLE	This bit is fixed to 0b because the host controller does not support prefetching in memory read cycles.	R
b2, b1	TYPE	Indicates that the base address of the EHCI operational registers is 32-bit width, and thus the registers can be allocated to any location in a 32-bit memory space. This field is fixed to 00b.	R
b0	MEMORY_SPACE_INDICATOR	Indicates that the EHCI operational registers are mapped on a system memory space. This bit is fixed to 0b.	R

10.5.4.6 SSVID_SSID — Subsystem ID - Subsystem Vendor ID (EHCI)

Address: 4003 012Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SUBSYSTEM_ID															
Value after reset	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SUBSYSTEM_VENDOR_ID															
Value after reset	0	0	0	1	0	0	0	0	0	0	1	1	0	0	1	1

Table 10.63 SSVID_SSID Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	SUBSYSTEM_ID	Indicates the device type. This field is used to select a driver specified by the PCI specification. It is not necessary for this host controller.	R
b15 to b0	SUBSYSTEM_VENDOR_ID	Indicates the device vendor. This field is used to select a driver specified by the PCI specification. It is not necessary for this host controller.	R

10.5.4.7 EROM_BASEAD — Expansion ROM Base Address (EHCI)

Address: 4003 0130h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	EXPANSION_ROM_BASE_ADDRESS															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	EXPANSION_ROM_BASE_ADDRESS						—	—	—	—	—	—	—	—	—	ROM_DECODE_ENABLE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.64 EROM_BASEAD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b10	EXPANSION_ROM_BASE_ADDRESS	This field is fixed to 000000h because decoding the expansion ROM is prohibited.	R
b9 to b1	Reserved		R
b0	ROM_DECODE_ENABLE	This bit is fixed to 0b because decoding the expansion ROM is prohibited.	R/W

10.5.4.8 CAPPTR — Capability Pointer (EHCI)

Address: 4003 0134h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	CAPABILITY_POINTER							
Value after reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Table 10.65 CAPPTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7 to b0	CAPABILITY_POINTER	This is a pointer to the capability identifier. “40h” means that the identifier is allocated to 40h in the host controller.	R/W

10.5.4.9 INTR_LINE_PIN — Max_Lat - Min_Gnt - Interrupt Pin - Interrupt Line (EHCI)

Address: 4003 013Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	MAX_LATENCY								MIN_GNT							
Value after reset	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	INTERRUPT_PIN								INTERRUPT_LINE							
Value after reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Table 10.66 INTR_LINE_PIN Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	MAX_LATENCY	Indicates the maximum acquisition frequency of the PCI bus. This host controller is implemented as 22h	R
b23 to b16	MIN_GNT	Indicates the minimum burst transfer time. This host controller is implemented as 10h	R
b15 to b8	INTERRUPT_PIN	Indicates the interrupt output pin. This bit is fixed at 02h since INTB is used.	R
b7 to b0	INTERRUPT_LINE	Indicates the interrupt line. Keep 00h in this host controller	R/W

10.5.4.10 CAPID_NIP_PMCAP — Capability Identifier - Next Item Pointer - Power Management Capabilities (EHCI)

Address: 4003 0140h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	PME_SUPPORT					D2_SUPPORT	D1_SUPPORT	AUX_CURRENT			DSI	—	PME_CLK	VERSION		
Value after reset	0	1	1	1	1	1	1	0	0	0	0	0	0	0	1	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	NEXT_ITEM_POINTER								CAPABILITY_IDENTIFIER							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 10.67 CAPID_NIP_PMCAP Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b27	PME_SUPPORT	[31]: Indicates whether the D3 Cold state is supported. This bit is fixed to 0b because the D3 Cold state is not supported [30:27]: Indicates that PME interrupt generation is supported in all PCI power states (D0 to D3). This bit is fixed to 1111b.	R
b26	D2_SUPPORT	Power Management Capabilities - Indicates that the PCI power state D2 is supported. This bit is fixed to 1b.	R
b25	D1_SUPPORT	Power Management Capabilities - Indicates that the PCI power state D1 is supported. This bit is fixed to 1b.	R
b24 to b22	AUX_CURRENT	Power Management Capabilities - Indicates the specified current value required for the 3.3 V auxiliary power supply. Generating PME interrupts in the D3 Cold state is not supported. This bit is fixed to 000b.	R
b21	DSI	Power Management Capabilities - Indicates that no special initialization is required for using power management. This bit is fixed to 0b.	R
b20	Reserved		R
b19	PME_CLK	Power Management Capabilities - Indicates that USB_PCICLK is not required for generating PME interrupts. This bit is fixed to 0b.	R
b18 to b16	VERSION	Power Management Capabilities - Indicates that this system is compliant with PCI power management interface specification release 1.1. This bit is fixed to 010b.	R
b15 to b8	NEXT_ITEM_POINTER	Indicates that there is no subsequent item. This bit is fixed to 00h.	R
b7 to b0	CAPABILITY_IDENTIFIER	Indicates the PCI power management register ID. This bit is fixed to 01h.	R

10.5.4.11 PMC_STS_PMCSR — Power Management Control and Status - PMCSR Bridge Support Extensions (EHCI)

Address: 4003 0144h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DATA								BPCC_ENABLE	B2_B3	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	PME_STATUS	DATA_SCALE						PME_ENABLE	—	—	—	—	—	—	POWER_STATE	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.68 PMC_STS_PMCSR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	DATA	This bit is fixed to 00h. This is an optional field in the PCI specification and is not supported in this host controller.	R
b23	BPCC_ENABLE	This bit is fixed to 0b. This is a bit for the bridge and is not supported in this host controller.	R
b22	B2_B3	This bit is fixed to 0b. This is a bit for the bridge and is not supported in this host controller.	R
b21 to b16	Reserved	Should be 0 at writing.	R
b15	PME_STATUS	PME interrupt status. This bit is set to 1b when a PME generation condition is satisfied. PME generation condition: Bit 3 (RD) of the HcInterruptStatus register is set to 1b while bit 10 (RWE) of the HcControl register is 1b. This bit is cleared to 0b when 1b is written via the PCI bus.	R/W
b14 to b9	DATA_SCALE	This bit is fixed to 00b. This is an optional field in the PCI specification and is not supported in this host controller.	R
b8	PME_ENABLE	Enable of PME. If this bit is set to 1b, a PME interrupt is generated when the system returns from power management.	R/W
b7 to b2	Reserved	Should be 0 at writing.	R
b1, b0	POWER_STATE	Indicates the PCI power status. 00b: D0 State 01b: D1 State 10b: D2 State 11b: D3 hot State.	R/W

10.5.4.12 SBRN_FLADJ_PW — SBRN - FLADJ - PORTWAKECAP

Address: 4003 0160h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	PORTWAKECAP															
Value after reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	FLADJ								SBRN							
Value after reset	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0

Table 10.69 SBRN_FLADJ_PW Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	PORTWAKECAP	This bit is used to mask which port's wakeup event is used, among connected devices. The setting of this bit does not affect the host controller operation.	R/W
b15 to b8	FLADJ	Use this bit to adjust the length of 1 micro-frame in 16 HS bit time units. The initial value is 20h (60000d HS bit time).	R/W
b7 to b0	SBRN	Indicates the serial bus release number. This bit is fixed to 20h.	R

10.5.4.13 EXT1 — EXT1 Register (EHCI)

The entity of this register is identical to EXT1 Register in OHCI configuration.

10.5.4.14 EXT2 — EXT2 Register (EHCI)

The entity of this register is identical to EXT2 Register in OHCI configuration.

However, bit[0] (EHCI_MASK) cannot be accessed from the EHCI side.

10.5.4.15 UTMICTRL — USBPHY Operation Mode Control Register (EHCI)

The entity of this register is identical to USBPHY Operation Mode Control Register in OHCI configuration.

10.5.5 AHB-PCI Bridge (PCI Configuration Space) Register Description

10.5.5.1 VID_DID — Device ID - Vendor ID (AHB-PCI Bridge)

Address: 4003 0000h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DEVICE_ID															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	VENDOR_ID															
Value after reset	0	0	0	1	0	0	0	0	0	0	1	1	0	0	1	1

Table 10.70 VID_DID Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	DEVICE_ID	Device type. Use this field to select a driver specified by the PCI specification These bits are not used for embedded hosts.	R
b15 to b0	VENDOR_ID	Device vendor. Use this field to select a driver specified by the PCI specification These bits are not used for embedded hosts.	R

10.5.5.2 CMND_STS — Status - Command (AHB-PCI Bridge)

Address: 4003 0004h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DETPERR	SIGSERR	REMABORT	RETABORT	SIGTABORT	DEVTIM		MDPERR	FBTBCAP	—	M66_CAP	CAPLIST	—	—	—	—
Value after reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	FBTBEN	SERREN	STEPCTR	PERREN	VGAPSNP	MWINVEN	SPECIALC	MASTEREN	MEMEN	IOEN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.71 CMND_STS Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31	DETPERR	Parity error status bit. This bit is set when an address or data parity error has been detected, cleared by writing 1b.	R/W
b30	SIGSERR	SERR status bit. This bit is set when a system error occurs, cleared by writing 1b.	R/W
b29	REMABORT	Master Abort status bit. This bit is set when Master Abort has been received, cleared by writing 1b	R/W
b28	RETABORT	Master Target Abort status bit. This bit is set when Target Abort has been received, cleared by writing 1b.	R/W
b27	SIGTABORT	Slave Target Abort status bit. This bit is set when Target Abort has been sent, cleared by writing 1b.	R/W
b26, b25	DEVTIM	Indicates a DEVSEL response speed. It is implemented as 01b (Medium Mode).	R
b24	MDPERR	Parity error detection bit This bit is set when parity error has been detected during Master operation, cleared by writing 1b.	R/W
b23	FBTBCAP	Fast Back to Back capability. This bit is fixed at 0b. (Fast Back to Back is not supported)	R
b22	Reserved	Should be 0 at writing.	R
b21	M66_CAP	66 MHz operation capability. This bit is fixed at 0b. (66 MHz is no supported)	R
b20	CAPLIST	Support of Capabilities List. This bit is fixed at 0b. (Capabilities List is not supported)	R
b19 to b10	Reserved	Should be 0 at writing.	R
b9	FBTBEN	Enables Fast Back to Back. This bit is fixed at 0b.	R
b8	SERREN	Set the operation when a system error is detected. 0: Ignored (initial value) 1: SERR# is asserted. Set 1b at initialization of this host controller	R/W
b7	STEPCTR	Address Stepping control bit. This bit is fixed at 0b. (Address Stepping is not supported)	R
b6	PERREN	Set the operation when a parity error is detected. 0: Ignored (initial value) 1: PERR# is asserted. Set 1b at initialization of this host controller	R/W
b5	VGAPSNP	Enable of VGA Palette Snoop. This bit is fixed at 0b.	R
b4	MWINVEN	Enable of Memory Write and Invalidate. This bit is fixed at 0b.	R
b3	SPECIALC	Enable of Special Cycle. This bit is fixed at 0b.	R

Table 10.71 CMND_STS Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b2	MASTEREN	Enable of PCI Master operation. 0: Master operation is not permitted (initial value) 1: Master operation is permitted. Set 1b at initialization of this host controller	R/W
b1	MEMEN	Enable of PCI Slave operation. 0: Memory cycle reception is not possible (initial value) 1: Memory cycle reception is possible. Set 1b at initialization of this host controller	R/W
b0	IOEN	Enables access to the I/O space. This bit is fixed at 0b.	R

10.5.5.3 REVID_CC — Class Code - Revision ID (AHB-PCI Bridge)

Address: 4003 0008h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	CLASS_CODE															
Value after reset	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	CLASS_CODE								REVISION_ID							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 10.72 REVID_CC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	CLASS_CODE	Indicates 060000h.	R
b7 to b0	REVISION_ID	Indicates 01h.	R

10.5.5.4 CLS_LT_HT_BIST — BIST - Header Type - Latency Timer - Cache Line Size (AHB-PCI Bridge)

Address: 4003 000Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	BIST								HEADER_TYPE							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	LATENCY_TIMER								CACHE_LINE_SIZE							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.73 CLS_LT_HT_BIST Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	BIST	Indicates 00h (BIST is not implemented).	R
b23 to b16	HEADER_TYPE	Indicates 00h (Single Function Device).	R
b15 to b8	LATENCY_TIMER	This bit is used to notify Latency Timer to the system. Keep the initial value 00h in this host controller.	R/W
b7 to b0	CACHE_LINE_SIZE	Indicates 00h (Cache is not supported).	R

10.5.5.5 BASEAD — AHB-PCI Bridge Registers Base Address

Address: 4003 0010h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	PCICOM_BASEADR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	PCICOM_BASEADR						—	—	—	—	—	—	PREFETCH	TYPE		MEM
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.74 BASEAD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b10	PCICOM_BASEADR	Sets the base address of the AHB-PCI Bridge PCI Communication Register area. Upper 22 bits are used as the base address since the 1-KB space is required.	R/W
b9 to b4	Reserved	Should be 0 at writing.	R
b3	PREFETCH	Indicates whether or not data prefetch is possible. This bit is fixed at 0b. (Data prefetch disabled)	R
b2, b1	TYPE	Indicates the base address Type. “00b” means that the address can be allocated to any address of 4-GB space.	R
b0	MEM	Indicates that the bits specified with the base address are the memory space. This bit is fixed at 0b.	R

10.5.5.6 WIN1_BASEAD — PCI-AHB Window1 Base Address

Address: 4003 0014h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	PCI_WIN1_BASEADR				—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	PREFETCH	TYPE		MEM
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Table 10.75 WIN1_BASEAD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b28	PCI_WIN1_BASEADR	Specify the base address of the PCI-AHB Window 1 space. PCI-AHB Window 1 space can be accessible after setting of bits [11:10] (PCI_AHB_WIN1_SIZE) of the USBCTR register. Accessible size is defined as below. 256 MB: Bits [31:28] specify a base address 512 MB: Bits [31:29] specify a base address 1 GB: Bits [31:30] specify a base address 2 GB: Bits [31] specify a base address	R/W
b27 to b4	Reserved	Should be 0 at writing.	R
b3	PREFETCH	Indicates whether prefetching data is enabled or disabled. This bit is fixed to 1b. (Prefetching data is enabled.)	R
b2, b1	TYPE	Indicates the base address type. “00b” means that the address can be allocated to any address of 4-GB space.	R
b0	MEM	Indicates that the field specified by the base address is in the memory space. This bit is fixed to 0b.	R

10.5.5.7 WIN2_BASEAD — PCI-AHB Window2 Base Address

Address: 4003 0018h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	PCI_WIN2_BASEADR				—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	PREFETCH	TYPE		MEM
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.76 WIN2_BASEAD Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b28	PCI_WIN2_BASEADR	Specify the base address of the PCI-AHB Window 2 space. Fixed 256 MB is used in this host controller, therefore upper 4 bits are used for the base address.	R/W
b27 to b4	Reserved	Should be 0 at writing.	R
b3	PREFETCH	Indicates whether prefetching data is enabled or disabled.	R
b2, b1	TYPE	Indicates the base address type. “00b” means that the address can be allocated to any address of 4-GB space.	R
b0	MEM	Indicates that Indicates that the field specified by the base address is in the memory space. This bit is fixed to 0b.	R

10.5.5.8 SSVID_SSID — Subsystem ID - Subsystem Vendor ID (AHB-PCI Bridge)

Address: 4003 002Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SUBSYS_ID															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SUBSYS_VENDOR_ID															
Value after reset	0	0	0	1	0	0	0	0	0	0	1	1	0	0	1	1

Table 10.77 SSVID_SSID Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	SUBSYS_ID	Indicates 0000h.	R
b15 to b0	SUBSYS_VENDOR_ID	Indicates 1033h.	R

10.5.5.9 INTR_LINE_PIN — Max_Lat - Min_Gnt - Interrupt Pin - Interrupt Line (AHB-PCI Bridge)

Address: 4003 003Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	MAX_LAT								MIN_GNT							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	INT_PIN								INT_LINE							
Value after reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Table 10.78 INTR_LINE_PIN Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	MAX_LAT	“00h” means that no bus utilization request	R
b23 to b16	MIN_GNT	Indicates the maximum burst transfer time. “02h” means that latency timer request 16 burst	R
b15 to b8	INT_PIN	Indicates the interrupt output pin. This bit is fixed to 01h because INTA# is used.	R
b7 to b0	INT_LINE	Indicates the interrupt line. Keep the initial value, 00h, in this host controller.	R/W

10.5.6 AHB-PCI Bridge (PCI Communication Space) Register Description

10.5.6.1 PCIAHB_WIN1_CTR — PCIAHB Window1 Control Register

Address: 4003 0800h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	AHB_BASEADR				—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	ENDIAN_CTR			—	—	—	—	PREFETCH	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.79 PCIAHB_WIN1_CTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b28	AHB_BASEADR	Specify the base address of the AHB when the host controller access the PCI-AHB Window 1 space. PCI-AHB Window1 space (256 MB/512 MB/1 GB/2 GB) can be accessible after setting of bits [11:10] (PCI_AHB_WIN1_SIZE) of the USBCTR register. Accessible size is defined as below. 256 MB: Bits [31:28] specify a base address 512 MB: Bits [31:29] specify a base address 1 GB: Bits [31:30] specify a base address 2 GB: Bit [31] specify a base address.	R/W
b27 to b9	Reserved	Should be 0 at writing.	R/W
b8 to b6	ENDIAN_CTR	Specify convert type of endianness on AHB. This value should be changed during an initialization only. 000b: No conversion (little endian) 001b: Access type data swapping 010b: Byte data swapping (bit endian) 011b: Halfword swapping 100b: Address conversion Others: Prohibited.	R/W
b5 to b2	Reserved	Should be 0 at writing.	R
b1, b0	PREFETCH	Specify enable of prefetch on AHB for a read request from the host controller. 11b: Enable prefetch (Up to 16 burst) Others: Reserved (not for use)	R/W

10.5.6.2 PCIAHB_WIN2_CTR — PCIAHB Window2 Control Register

Address: 4003 0804h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	AHB_BASEADR				—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	ENDIAN_CTR			—	—	—	—	PREFETCH	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.80 PCIAHB_WIN2_CTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b28	AHB_BASEADR	Specify the base address of the AHB when the host controller access the PCI-AHB Window 2 space. PCI-AHB Window2 space (256 MB) can be accessible after setting of bits 9 (PCI_AHB_WIN2_EN) of the USBCTR register.	R/W
b27 to b9	Reserved	Should be 0 at writing.	R/W
b8 to b6	ENDIAN_CTR	Specify convert type of endianness on AHB. This value should be changed during an initialization only. 000b: No conversion (little endian) 001b: Access type data swapping 010b: Byte data swapping (bit endian) 011b: Halfword swapping 100b: Address conversion Others: Prohibited.	R/W
b5 to b2	Reserved	Should be 0 at writing.	R
b1, b0	PREFETCH	Specify enable of prefetch on AHB for a read request from the host controller. 11b: Enable prefetch (Up to 16 burst) Others: Reserved (not for use)	R/W

10.5.6.3 AHBPCI_WIN1_CTR — AHBPCI Window1 Control Register

Address: 4003 0810h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	PCIWIN1_BASEADR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	PCIWIN1_BASEADR					—	—	—	—	—	—	—	PCICMD			—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.81 AHBPCI_WIN1_CTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b11	PCIWIN1_BASEADR	Specify the base address of the PCI bus when it accesses AHB-PCI Window 1 space from AHB. Setting this register is required for accessing host controller and PCI configuration space in AHB-PCI bridge.	R/W
b10 to b4	Reserved	Should be 0 at writing.	R/W
b3 to b1	PCICMD	Specify the PCI bus cycle type. 101b: Configuration Read/Configuration Write Others: Reserved (not for use)	R/W
b0	Reserved	Should be 0 at writing.	R

10.5.6.4 AHBPCI_WIN2_CTR — AHBPCI Window2 Control Register

Address: 4003 0814h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	PCIWIN2_BASEADR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	BURST_EN	—	PCICMD			—
Value after reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Table 10.82 AHBPCI_WIN2_CTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	PCIWIN2_BASEADR	Specify the base address of the PCI bus when it accesses AHB-PCI Window 2 space from AHB. Setting this register is required for accessing OHCI operational register area.	R/W
b15 to b6	Reserved	Should be 0 at writing.	R/W
b5	BURST_EN	Enable of burst transfer on the PCI bus. 0b: Disable burst transfer Others: Reserved (not for use)	R/W
b4	Reserved	Should be 0 at writing.	R/W
b3 to b1	PCICMD	Specify the PCI bus cycle type. 011b: Memory Read/Memory Write Others: Reserved (not for use)	R/W
b0	Reserved	Should be 0 at writing.	R/W

10.5.6.5 PCI_INT_ENABLE — PCI Interrupt Enable Register

Address: 4003 0820h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	USBH_PMEEN	—	USBH_INTBEN	USBH_INTAEN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	PCIAHB_WIN2_INTEN	PCIAHB_WIN1_INTEN	—	—	—	—	—	—	RESERR_INTEN	SIGSERR_INTEN	PERR_INTEN	REMARBORT_INTEN	RETABORT_INTEN	SIGTABORT_INTEN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.83 PCI_INT_ENABLE Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b20	Reserved	Should be 0 at writing.	R
b19	USBH_PMEEN	Enable of USBH_PME 0: Disable 1: Enable	R/W
b18	Reserved	Should be 0 at writing.	R/W
b17	USBH_INTBEN	Enable of USBH_INTB 0: Disable 1: Enable	R/W
b16	USBH_INTAEN	Enable of USBH_INTA 0: Disable 1: Enable	R/W
b15, b14	Reserved	Should be 0 at writing.	R/W
b13	PCIAHB_WIN2_INTEN	Enable of PCIAHB_WIN2_INT 0: Disable 1: Enable	R/W
b12	PCIAHB_WIN1_INTEN	Enable of PCIAHB_WIN1_INT 0: Disable 1: Enable	R/W
b11 to b6	Reserved	Should be 0 at writing.	R/W
b5	RESERR_INTEN	Enable of RESERR_INT 0: Disable 1: Enable	R/W
b4	SIGSERR_INTEN	Enable of SIGSERR_INT 0: Disable 1: Enable	R/W
b3	PERR_INTEN	Enable of PERR_INT 0: Disable 1: Enable	R/W
b2	REMARBORT_INTEN	Enable of REMABORT_INT 0: Disable 1: Enable	R/W
b1	RETABORT_INTEN	Enable of RETABORT_INT 0: Disable 1: Enable	R/W

Table 10.83 PCI_INT_ENABLE Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b0	SIGTABORT_INTEN	Enable of SIGTABORT_INT 0: Disable 1: Enable	R/W

10.5.6.6 PCI_INT_STATUS — PCI Interrupt Status Register

Address: 4003 0824h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	USBH_PME	—	USBH_INTB	USBH_INTA
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	PCIAHB_WIN2_INT	PCIAHB_WIN1_INT	—	—	—	—	—	—	RESERR_INT	SIGSERR_INT	PERR_INT	REMABORT_INT	RETABORT_INT	SIGTABORT_INT
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.84 PCI_INT_STATUS Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b20	Reserved		R
b19	USBH_PME	Status of PME# interrupt from the host controller. It can be cleared in the host controller. 0: No PME has occurred 1: An PME has occurred	R
b18	Reserved	Should be 0 at writing.	R/W
b17	USBH_INTB	Status of INTB# interrupt from the host controller. It can be cleared in the host controller. 0: No INTB has occurred 1: An INTB has occurred	R
b16	USBH_INTA	Status of INTA# interrupt from the host controller. It can be cleared in the host controller. 0: No INTA has occurred 1: An INTA has occurred	R
b15, b14	Reserved	Should be 0 at writing.	R/W
b13	PCIAHB_WIN2_INT	Indicates that an AHB bus error has occurred in PCIAHB Window 2. This bit is cleared when 1b is written. 0: No AHB bus error has occurred 1: An AHB bus error has occurred	R/W
b12	PCIAHB_WIN1_INT	Indicates that an AHB bus error has occurred in PCIAHB Window 1. This bit is cleared when 1b is written. 0: No AHB bus error has occurred 1: An AHB bus error has occurred	R/W
b11 to b6	Reserved	Should be 0 at writing.	R/W
b5	RESERR_INT	Indicates the status of the interrupt caused by SERR# input. This bit is cleared when 1b is written. 0: SERR# assertion has not been detected 1: SERR# assertion has been detected	R/W
b4	SIGSERR_INT	Indicates the status of the interrupt caused by SERR# output. This bit is cleared when 1b is written. 0: SERR# has not been asserted 1: SERR# has been asserted	R/W
b3	PERR_INT	Indicates the status of the interrupt caused by PERR# input or output. This bit is cleared when 1b is written. 0: PERR# has not been asserted 1: PERR# has been asserted	R/W

Table 10.84 PCI_INT_STATUS Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b2	REMABORT_INT	Indicates that Master Abort is received during PCI master operation. This bit is cleared when 1b is written. 0: Master Abort has not been received 1: Master Abort has been received	R/W
b1	RETABORT_INT	Indicates that Target Abort is reported during PCI master operation. This bit is cleared when 1b is written. 0: Target Abort has not been reported 1: Target Abort has been reported	R/W
b0	SIGTABORT_INT	Indicates that Target Abort is reported during PCI slave operation. This bit is cleared when 1b is written. 0: Target Abort has not been reported 1: Target Abort has been reported	R/W

10.5.6.7 AHB_BUS_CTR — AHB Bus Control Register

Address: 4003 0830h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	SMODE_READY_CTR	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	MMODE_HBUSREQ	—	—	—	—	MMODE_WR_INCR	MMODE_BYTE_BURST	MMODE_HTRANS
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.85 AHB_BUS_CTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b18	Reserved	Should be 0 at writing.	R
b17	SMODE_READY_CTR	Wait cycle control type for AHB slave. 0: Reserved (not for use) 1: Wait cycle is controlled by HREADY=0	R/W
b16 to b8	Reserved	Should be 0 at writing.	R/W
b7	MMODE_HBUSREQ	HBUSREQ de-assert timing for the AHB master 0: Reserved (not for use) 1: Release at first cycle of HGRANT=1 & HREADY=1	R/W
b6 to b3	Reserved	Should be 0 at writing.	R/W
b2	MMODE_WR_INCR	AHB INCR burst use condition at write for the AHB master 0: Reserved (not for use) 1: INCR4/8/16, or INCR for 2/3 beat	R/W
b1	MMODE_BYTE_BURST	Burst mode setting in 16 bit/8 bit transfer for the AHB Master 0: Reserved (not for use) 1: No burst for 16 bit/8 bit transfer.	R/W
b0	MMODE_HTRANS	HTRANS behavior setting for the AHB Master 0: Reserved (not for use) 1: IDLE and HBUSREQ assertion during cycle separation	R/W

10.5.6.8 USBCTR — USB Control Register

Address: 4003 0834h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	PCI_AHB_WIN1_SIZE		PCI_AHB_WIN2_EN	DIRPD	—	—	—	—	—	PLL_RST	PCICLK_MASK	USBH_RST
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Table 10.86 USBCTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b12	Reserved	Should be 0 at writing.	R
b11, b10	PCI_AHB_WIN1_SIZE	Use this bit to control the PCI-AHB Window 1 area. Accessing Host Controller Registers. This value should be changed only in the initialization. 00b: 256 MB 01b: 512 MB 10b: 1 GB 11b: 2 GB	R/W
b9	PCI_AHB_WIN2_EN	Use this bit to enable the PCI-AHB Window 2. Accessing Host Controller Registers. This value should be changed only in the initialization. In this host controller, the PCI-AHB Window 2 area is fixed to 256 MB. 0: PCI-AHB Window 2 is not available 1: PCI-AHB Window 2 is available	R/W
b8	DIRPD	If this bit is set to 1b, USB subsystem moves to power-down state. Bit 12 (DIRPD) of the EPCTR register and the DIRPD of CFG_USB in the system controller have same function. 0: Normal operation 1: Direct power-down state	R/W
b7 to b3	Reserved	Keep the initial value	R/W
b2	PLL_RST	Reset of USBPLL The USBPLL is shared between the host and function controllers. Reset to the USBPLL is asserted when both PLL_RST in host and function controllers are 1b. 0: Release USBPLL reset 1: Assert USBPLL reset	R/W
b1	PCICLK_MASK	Use this bit to control PCI clock (USB_PCICLK) supply in the host controller. If this bit is set to 1b, the host controller is not accessible. 0: Supply the PCI clock 1: Stop the PCI clock	R/W
b0	USBH_RST	Use this bit to control the reset signal supplied to the host controller. Access to the host controller becomes valid after 3 USB_PCICLK cycles from end of the reset. 0: Release host controller reset 1: Assert host controller reset	R/W

10.5.6.9 PCI_ARBITER_CTR — PCI Arbiter Control Register

Address: 4003 0840h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	PCIBP_MODE	—	—	—	—	—	—	—	—	—	—	PCIREQ1	PCIREQ0
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.87 PCI_ARBITER_CTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b13	Reserved	Keep the initial value	R/W
b12	PCIBP_MODE	PCI bus master in the bus parking 0: This unit 1: Last accessed master Set 1b during initialization of this host controller and do not change the value.	R/W
b11 to b2	Reserved	Keep the initial value	R/W
b1	PCIREQ1	Enable of PCI bus request1 signal 0: Disable the request signal 1: Enable the request signal Set 1b during initialization of this host controller and do not change the value.	R/W
b0	PCIREQ0	Enable of PCI bus request0 signal 0: Disable the request signal 1: Enable the request signal Set 1b during initialization of this host controller and do not change the value.	R/W

10.5.7 EPC Register Description

10.5.7.1 USB_CONTROL — USB Control Register

Address: 4001 E000h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	USBTESTMODE		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	SOF_CLK_MODE	INT_SEL	FORCEFS	SOF_RCV	RSUM_IN	SUSPEND	CONF	DEFAULT	CONNECTB	PUE2	—	—
Value after reset	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0

Table 10.88 USB_CONTROL Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b19	Reserved		R
b18 to b16	USBTESTMODE	Set these bits after the status stage of the SET_FEATURE_TEST_MODE request has finished normally. After setting this field to other than 000b, set bit 1 (CS_TESTMODEEN) of Test Control Register. Test_J, Test_K, Test_SE0_NAK tests: After setting this field, the USBPHY enters test mode in HS mode as soon as CS_TESTMODEEN is set to 1b. Test_Packet test: When HS connection to the Host is established, Test_Packet is transmitted by setting CS_TESTMODEEN to 1b after setting this field. When HS Connection to the Host is not established, Test_Packet is transmitted by setting as follows USBTESTMODE = Test_J→CS_TESTMODEEN = 1→CS_TESTMODEEN = 0 →USBTESTMODE = Test_K→CS_TESTMODEEN = 1→CS_TESTMODEEN = 0 →USBTESTMODE = Test_SE0_NAK→CS_TESTMODEEN = 1 →CS_TESTMODEEN = 0 →USBTESTMODE = Test_Packet→CS_TESTMODEEN = 1 000b: Normal 001b: Test_J 010b: Test_K 011b: Test_SE0_NAK 100b: Test_Packet	R/W
b15 to b12	Reserved	Should be 0 at writing.	R
b11	SOF_CLK_MODE	Operating mode select for SOF output pin in HS mode. Set this bit during initialization. The operation is not guaranteed if this bit is set at any other time. 0: Invert signal when SOF/uSOF packet is received 1: Invert signal when SOF packet is received (no inversion by uSOF)	R/W
b10	INT_SEL	U2F_EPC_INT interrupt output type select Setting is common for all interrupt sources. 0: Pulse output. (not for use) 1: Level output. If there are multiple interrupt sources, the interrupt output keeps being asserted until the all sources are cleared.	R/W
b9	FORCEFS	0: Normal operation 1: Not for use	R/W

Table 10.88 USB_CONTROL Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b8	SOF_RCV	Enable bit for automatic recovery at SOF reception error 0: Disable (not for use) 1: Enable	R/W
b7	RSUM_IN	Remote wakeup trigger [Caution] In case of disabling clock supply to EPC, SIE and USBPHY in suspend state, the clocks should be resumed before setting this bit. 0: Not send the resume signal 1: Send the resume signal	R/W
b6	SUSPEND	When the state is Suspend, clock supply for EPC, SIE, USBPHY can be stopped by setting this bit to 1b. This bit is cleared to 0b automatically when RESUME is detected. 0: Supply clocks 1: Stop clocks	R/W
b5	CONF	Enable bit for Endpoints other than Endpoint 0. This bit is cleared to 0b automatically when it receives BusReset. If this bit is 0b, the state is Default or Address state except Suspend. If this bit is 1b, the state is Configured except Suspend 0: Disable Endpoints other than Endpoint 0. (No response is returned for tokens) 1: Enable Endpoints other than Endpoint 0.	R/W
b4	DEFAULT	Enable bit for Endpoint 0. This bit is set to 1b automatically when it receives BusReset. If this bit is 0b, the state is Attached or Powered as defined in the USB specification. If this bit is 1b, it depends on bit 5 (CONF). 0: Disable Endpoint0. (No response for tokens) 1: Enable Endpoint0.	R/W
b3	CONNECTB	By setting this bit to 1b when Un-Plugged, it can prevent the occurrence of a pseudo bus reset or suspend signal due to unstable D+/D- signals on the USB port. When this bit is set to 1b, the SIE enters the Suspend state and SPND_INT of USB Interrupt Status Register occurs. 0: Enable the USB signals to the SIE block 1: Disable the USB signals to the SIE block	R/W
b2	PUE2	Control bit for D+ pull-up. 0: Not pull up D+ signal 1: Pull up D+ signal	R/W
b1, b0	Reserved	Keep the initial value	R/W

10.5.7.2 USB_STATUS — USB Status Register

Address: 4001 E004h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SOF_DELAY_STATUS	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	SPEED_MODE	CONF	DEFAULT	USB_RST	SPND_OUT	RSUM_OUT	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.89 USB_STATUS Register Contents

Bit Position	Bit Name	Function	R/W
b31	SOF_DELAY_STATUS	This bit is set to 1 if a SOF is ignored during SOF unacceptable period. This bit is updated each time a SOF is received. For more details about SOF unacceptable period, see bit 31 (SOF_DELAY_MODE) of the Frame Number & USB Address Register. 0: SOF was received normally 1: SOF was ignored	R
b30 to b7	Reserved		R
b6	SPEED_MODE	Indicates USB port speed. 0: FS (Full-Speed) 1: HS (High-Speed)	R
b5	CONF	Status of Endpoints other than Endpoint 0 are enabled. This bit is equal to bit 5 (CONF) of the USB Control Register. 0: Disabled (No response for tokens) 1: Enabled	R
b4	DEFAULT	Status of Endpoint 0. This bit is equal to bit 4 (DEFAULT) of the USB Control Register. 0: Disabled (No response for tokens) 1: Enabled	R
b3	USB_RST	Indication of BusReset state 0: Not BusReset state 1: BusReset state	R
b2	SPND_OUT	Indication of Suspend state 0: Not Suspend state 1: Suspend state	R
b1	RSUM_OUT	Indication of Resume status 0: Resume is not being received 1: Resume is being received	R
b0	Reserved		R

10.5.7.3 USB_ADDRESS — Frame Number & USB Address Register

Address: 4001 E008h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SOF_DELAY_MODE	—	—	—	—	—	—	—	—	USB_ADDR						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SOF_STATUS	UFRAME			—	FRAME										
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.90 USB_ADDRESS Register Contents

Bit Position	Bit Name	Function	R/W
b31	SOF_DELAY_MODE	Setting of SOF unacceptable If this bit is set to 1b, SOF which comes within 64 cycle@60 MHz after normal SOF or SOF recovery is ignored as illegal SOF token. 0: SOF is valid regardless of when it is received 1: Ignore SOF during unacceptable period	R/W
b30 to b23	Reserved	Should be 0 at writing.	R
b22 to b16	USB_ADDR	The written value is used as USB address after status stage of SET_ADDRESS request is completed normally. Write these bits before the status stage of SET_ADDRESS request is completed. These bits are cleared to 0b when it receives BusReset.	R/W
b15	SOF_STATUS	SOF/uSOF reception status. This bit is updated each time a SOF or uSOF is received. 0: SOF or uSOF was received normally 1: SOF or uSOF reception error occurred	R
b14 to b12	UFRAME	The number of times for received uSOF within a frame. If bit 8 (SOF_RCV) of the USB Control Register is 1b, these bits are updated even if an automatic uSOF recovery occurs. These bits are valid only when bit 5 (CONF) of the USB Control Register is 1b.	R
b11	Reserved	Should be 0 at writing.	R
b10 to b0	FRAME	The frame number of the SOF. If bit 8 (SOF_RCV) of the USB Control Register is 1b, these bits are updated even if an automatic SOF recovery occurs. These bits are valid only when bit 5 (CONF) of the USB Control Register is 1b.	R

10.5.7.4 TEST_CONTROL — Test Control Register

Address: 4001 E010h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	FORCEHS	CS_TESTMODEEN	LOOPBACK
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.91 TEST_CONTROL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved	Should be 0 at writing.	R/W
b2	FORCEHS	By setting this bit to 1b the subsystem is set to HS mode by force. This bit is only used for LSI design. 0: Normal operation 1: Fix the operating mode to HS mode (not for use)	R/W
b1	CS_TESTMODEEN	If this bit is set to 1b, the settings of bits 18 to 16 (USBTESTMODE[2:0]) of the USB Control Register becomes valid promptly. This bit should be set to 1b after setting of USBTESTMODE, set to 0b after end of each test. 0: Normal operating mode 1: Enable USB test mode	R/W
b0	LOOPBACK	Enable bit for hardware loopback on logic side. 0: Normal operating mode 1: Hardware loopback on the logic side	R/W

10.5.7.5 SETUP_DATA0 — Setup Data0 Register

Address: 4001 E018h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SETUP4								SETUP3							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SETUP2								SETUP1							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.92 SETUP_DATA0 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	SETUP4	4th byte of the received SETUP data. These bits are updated when it receives SETUP data every time.	R
b23 to b16	SETUP3	3rd byte of the received SETUP data. These bits are updated when it receives SETUP data every time.	R
b15 to b8	SETUP2	2nd byte of the received SETUP data. These bits are updated when it receives SETUP data every time.	R
b7 to b0	SETUP1	1st byte of the received SETUP data. These bits are updated when it receives SETUP data every time.	R

10.5.7.6 SETUP_DATA1 — Setup Data1 Register

Address: 4001 E01Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SETUP8								SETUP7							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SETUP6								SETUP5							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.93 SETUP_DATA1 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	SETUP8	8th byte of the received SETUP data. These bits are updated when it receives SETUP data every time.	R
b23 to b16	SETUP7	7th byte of the received SETUP data. These bits are updated when it receives SETUP data every time.	R
b15 to b8	SETUP6	6th byte of the received SETUP data. These bits are updated when it receives SETUP data every time.	R
b7 to b0	SETUP5	5th byte of the received SETUP data. These bits are updated when it receives SETUP data every time.	R

10.5.7.7 USB_INT_STA — USB Interrupt Status Register

Address: 4001 E020h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	EP15_INT	EP14_INT	EP13_INT	EP12_INT	EP11_INT	EP10_INT	EP9_INT	EP8_INT
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	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	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.94 USB_INT_STA Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	Should be 0 at writing.	R
b23 to b8	EP[m]_INT/EP0_INT (m = 1..15)	Interrupt for Endpoint 0/[m]. Interrupt factors can be confirmed by EP0/EP[m] Status Register. These bits are cleared when interrupt factors in EP0/EP[m] Status Register are cleared 0: No interrupt of Endpoint0/[m] occurred 1: Interrupt of Endpoint0/[m] occurred	R
b7	Reserved	Should be 0 at writing.	R
b6	SPEED_MODE_INT	Interrupt for speed mode change This bit is cleared by writing 0b. 0: The speed mode has not changed from FS to HS 1: The speed mode has changed from FS to HS	R/W
b5	SOF_ERROR_INT	Interrupt for SOF/uSOF reception error. This bit is set to 1b if SOF or uSOF is not received within following period HS mode: 125 µs + 0.0625 µs. 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. 0: No SOF or uSOF reception error has occurred 1: An SOF or uSOF reception error has occurred	R/W
b4	SOF_INT	Interrupt for SOF/uSOF reception This bit is cleared by writing 0b. This bit is invalid when bit 5 (CONF) of the USB Control Register is 0b. 0: SOF or uSOF has not been received 1: SOF or uSOF has been received	R/W
b3	USB_RST_INT	Interrupt for BusReset. This bit is cleared by writing 0b. 0: BusReset has not been issued 1: BusReset has been issued	R/W
b2	SPND_INT	Interrupt for Suspend state. This bit is cleared by writing 0b. 0: Not in Suspend state 1: Entered to Suspend state	R/W
b1	RSUM_INT	Interrupt for Resume reception This bit is cleared by writing 0b. 0: Resume is not being received 1: Resume is being received	R/W
b0	Reserved	Should be 0 at writing.	R

10.5.7.8 USB_INT_ENA — USB Interrupt Enable Register

If each field is disabled, the corresponding interrupt is not asserted even if USB Interrupt Status Register is set.

Address: 4001 E024h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	EP15_EN	EP14_EN	EP13_EN	EP12_EN	EP11_EN	EP10_EN	EP9_EN	EP8_EN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	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	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.95 USB_INT_ENA Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	Should be 0 at writing.	R
b23 to b8	EP[m]_EN/EP0_EN (m = 1..15)	Enable bit for EP[m]_INT/EP0_INT 0: Disable 1: Enable	R/W
b7	Reserved	Should be 0 at writing.	R
b6	SPEED_MODE_EN	Enable bit for SPEED_MODE_INT 0: Disable 1: Enable	R/W
b5	SOF_ERROR_EN	Enable bit for SOF_ERROR_INT 0: Disable 1: Enable	R/W
b4	SOF_EN	Enable bit for SOF_INT 0: Disable 1: Enable	R/W
b3	USB_RST_EN	Enable bit for USB_RST_INT 0: Disable 1: Enable	R/W
b2	SPND_EN	Enable bit for SPND_INT 0: Disable 1: Enable	R/W
b1	RSUM_EN	Enable bit for RSUM_INT 0: Disable 1: Enable	R/W
b0	Reserved	Should be 0 at writing.	R

10.5.7.9 EP0_CONTROL — EP0 Control Register

Address: 4001 E028h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	EP0_STGSEL	EP0_OVERSEL	EP0_AUTO
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	EP0_PIDCLR	EP0_BCLR	EP0_DEND	EP0_DW		EP0_INAK_EN	EP0_PERR_NAK_CLR	EP0_STL	EP0_INAK	EP0_ONAK
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table 10.96 EP0_CONTROL Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b19	Reserved	Should be 0 at writing.	R
b18	EP0_STGSEL	Select the operation when data other than null data is received in status stage. If this bit is changed during a transfer, the operation is not guaranteed. 0: Normal reception. Data other than null data is discarded and it is handled as null data. No response for overrun, and it depends on EP0_OVERSEL 1: STALL response (not for use)	R/W
b17	EP0_OVERSEL	Select the operation for next OUT token when an overrun occurs in OUT transfer. (No response for overrun packet.) If this bit is changed during a transfer, the operation is not guaranteed. 0: STALL response for the next packet 1: Handle the next packet as a retry (not for use)	R/W
b16	EP0_AUTO	Select whether to send a packet automatically when maximum size (64 byte) is written to EP0 Write Register (transmission buffer). There is a case when this function should be disabled, maximum packet size − 1, 2, or 3 bytes, as shown in the example below. (If this function is not disabled, all data becomes valid at writing maximum packet size) If last data includes invalid data, clear this bit and use EP0_DW[1:0]. For example, when sending 61 to 63 bytes of data, it is handled as 64 bytes. 0: Not set EP0_DEND automatically 1: Set EP0_DEND automatically ***Caution*)** Change this bit while the buffer is empty.	R/W
b15 to b10	Reserved	Should be 0 at writing.	R/W
b9	EP0_PIDCLR	Use this bit to initialize DATA PID for Endpoint 0. Both transmission and reception PIDs are initialized to DATA1 by writing 1b. If this bit is written during a USB transaction for Endpoint 0, the write is held until the transaction finishes. DATA PID is initialized when it receives BusReset or SETUP token. This bit is write-only. Read as 0.	W
b8	EP0_BCLR	Use this bit to clear both EP0 Write and EP0 Read Registers. Both transmission and reception buffer are initialized by writing 1b. If this bit is written during a USB transaction for Endpoint 0, the write is held until the transaction finishes. In case of using this bit, it needs to check EP0_IN_DATA=0 and EP0_IN_EMPTY=1 of EP0 Status Register before writing next data. ***Caution*)** EP0 Write and EP0 Read Registers are not cleared even if USB BusReset is received.	W

Table 10.96 EP0_CONTROL Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b7	EP0_DEND	Transmission enable of EP0 Write Register The transmission data is enabled by writing 1b. When it sends null data, make sure bit 8 (EP0_IN_EMPTY) of EP0 Status Register is 1b, then write 1b to this bit. This bit is write-only. Read as 0.	W
b6, b5	EP0_DW	Valid data size for last data written in EP0 Write Register (transmission buffer). Set these bits together with EP0_DEND=1. 00b: 4 01b: 1 10b: 2 11b: 3 This field is write-only. Read as 0.	W
b4	EP0_INAK_EN	Write enable bit of EP0_INAK When writing to EP0_INAK, set 1b at the same time. This bit is write-only. Read as 0. ***Note*)** This bit is used to prevent EP0_INAK from being cleared unintentionally when set EP0_INAK by hardware and accessing this register by the software occurs at the same time.	W
b3	EP0_PERR_NAK_CLR	Write 1b to this bit to cancel forced NAK state by illegal token. Generally, this bit is not used since the STALL state is set by EP0_STL when it occurs forced NAK state. This bit is write-only. Read as 0.	W
b2	EP0_STL	STALL response control bit for IN/OUT/PING token to Endpoint0. If this bit is set to 1b, STALL is returned to all IN, OUT, and PING tokens during data stage and status stages. Even if this bit is cleared, STALL response is not canceled. This bit is cleared to 0b automatically when it receives SETUP token. This bit has a priority over EP0_ONAK and EP0_INAK. This bit is set to 1b in following case – Data other than null data is received at status stage while EP0_STGSEL is 1b, – Overrun occurs while EP0_OVERSEL is 0b. 0: Not return STALL. 1: Return STALL.	R/W
b1	EP0_INAK	NAK response control bit for IN token to Endpoint0. Generally ACK, NAK are issued automatically according to status of EP0 Read Register (reception buffer). Write 1b to this bit to send a NAK by force. This bit is set to 1b at the end of normal completion SETUP transaction. 0: Transmit data if data exists in the transmission buffer 1: Return a NAK even if data exists in the transmission buffer	R/W
b0	EP0_ONAK	NAK response control bit for OUT/PING token to Endpoint0. Generally ACK, NAK, and NYET are issued automatically according to status of EP0 Read Register (reception buffer). Write 1b to this bit to send a NAK by force. This bit is set to 1b at the end of normal completion SETUP transaction. 0: Receive data if there is available space in the reception buffer 1: Return a NAK even if there is available space in the reception buffer	R/W

10.5.7.10 EP0_STATUS — EP0 Status Register

Address: 4001 E02Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	EP0_PID	EP0_PERR_NAK	EP0_PERR_NAK_INT
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	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
Value after reset	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0

Table 10.97 EP0_STATUS Register Contents (1/3)

Bit Position	Bit Name	Function	R/W
b31 to b19	Reserved	Should be 0 at writing.	R
b18	EP0_PID	Next DATA PID 0: DATA0 1: DATA1	R
b17	EP0_PERR_NAK	This bit is set to 1b when it enters to forced NAK state due to reception of illegal token at Endpoint0. “illegal token” means • IN/OUT token reception before SETUP token is received (no setup stage) • OUT token reception at Control Read data stage • IN or invalid OUT token reception at Control Read status stage • IN or invalid OUT token reception at Control Write data stage • OUT or PING token reception at Control Write status stage • OUT token reception at No Data Control status stage While this bit is 1b, forced NAK is sent for IN/OUT/ PING token to Endpoint 0. If this bit is set, set bit 2 (EP0_STL) of the EP0 Control Register so that the Endpoint becomes STALL state. This bit is cleared to 0b when it receives SETUP token. This bit is also cleared to 0b when 1b is written to bit 3 (EP0_PERR_NAK_CLR) of the EP0 Control Register. 0: Not forced NAK state by illegal token 1: Forced NAK state by illegal token	R
b16	EP0_PERR_NAK_INT	This bit is set to 1b when forced NAK response due to reception of illegal token at Endpoint0. If this bit is set, set bit 2 (EP0_STL) of the EP0 Control Register so that the Endpoint becomes STALL state. This bit is cleared by writing 0b. 0: Illegal token was not received 1: Illegal token was received, and NAK was sent	R/W
b15	EP0_OUT_NAK_INT	This bit is set to 1b when NAK was sent for OUT/PING token to Endpoint 0. This bit is cleared by writing 0b. 0: NAK was not set for OUT/PING token 1: NAK was sent for OUT/PING token	R/W
b14	EP0_OUT_NULL	This bit is set to 1b when null data to Endpoint 0 is received. This bit is updated when valid OUT data is stored in EP0 Read Register (reception buffer). 0: Null data was not received 1: Null data was received.	R

Table 10.97 EP0_STATUS Register Contents (2/3)

Bit Position	Bit Name	Function	R/W
b13	EP0_OUT_FULL	EP0 Read Register (reception buffer) full status This bit is updated according to the buffer status. 0: Reception buffer is not full 1: Reception buffer is full, Maxpacketsize (64 Byte)	R
b12	EP0_OUT_EMPTY	EP0 Read Register (reception buffer) empty status This bit is updated according to the buffer status. 0: Reception buffer is not empty 1: Reception buffer is empty	R
b11	EP0_IN_NAK_INT	This bit is set to 1b when a NAK is sent in response to an IN token for Endpoint0. This bit is cleared by writing 0b. 0: NAK has not been sent in response to an IN token 1: NAK was sent in response to an IN token	R/W
b10	EP0_IN_DATA	EP0 Write Register (transmission buffer) data valid This bit is not set 1b while bit 7 (EP0_DEND) of the EP0 Control Register is 0b. This bit is updated according to the buffer status. 0: There is no waiting data in the transmission buffer. 1: There is a waiting data in the transmission buffer.	R
b9	EP0_IN_FULL	EP0 Write Register (transmission buffer) full status. This bit is updated according to the buffer status. 0: Transmission buffer is not full 1: Transmission buffer is full	R
b8	EP0_IN_EMPTY	EP0 Write Register (transmission buffer) empty status. This bit is updated according to the buffer status. 0: Transmission buffer is not empty 1: Transmission buffer is empty	R
b7	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. 0: Null data has not been received 1: Null data was received	R/W
b6	EP0_OUT_OR_INT	This bit is set to 1b if overrun occurs while Endpoint0 receives a data. This bit is cleared by writing 0b. 0: No overrun occurred. 1: Overrun occurred.	R/W
b5	EP0_OUT_INT	This bit is set to 1b when valid data is stored to EP0 Read Register (reception buffer) This bit is cleared by writing 0b. 0: Not ready to read from reception buffer 1: Ready to read from reception buffer	R/W
b4	EP0_IN_INT	This bit is set to 1b when data is sent from EP0 Write Register (transmission buffer) normally. This bit is cleared by writing 0b. 0: Transmission buffer has not become writable yet. 1: Transmission buffer becomes writable.	R/W
b3	EP0_STALL_INT	This bit is set to 1b when Endpoint0 moves to STALL state. 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. 0: Processing at Endpoint0 is not stalled 1: Processing at Endpoint0 is stalled	R/W
b2	STG_END_INT	This bit is set to 1b when the status stage of Control transfer completes normally. This bit is cleared by writing 0b. This bit is cleared when it receives next SETUP token. 0: Status stage has not completed normally 1: Status stage completed normally	R/W

Table 10.97 EP0_STATUS Register Contents (3/3)

Bit Position	Bit Name	Function	R/W
b1	STG_START_INT	This bit is set to 1b when the status stage of Control transfer starts. This bit is cleared by writing 0b, or cleared when it receives next SETUP token. 0: Status stage is not started 1: Status stage is started	R/W
b0	SETUP_INT	This bit is set to 1b when valid SETUP data is received. Clear this bit before starting request procedure to prepare next SETUP data reception during the procedure. This bit is cleared by writing 0b. 0: Valid SETUP data has not been received 1: Valid SETUP data was received	R/W

10.5.7.11 EP0_INT_ENA — EP0 Interrupt Enable Register

If each field is disabled, the corresponding interrupt is not asserted even if EP0 Status Register is set.

Address: 4001 E030h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	EP0_PERR_NAK_EN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	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
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.98 EP0_INT_ENA Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b17	Reserved	Should be 0 at writing.	R
b16	EP0_PERR_NAK_EN	Enable for bit 16 (EP0_PERR_NAK_INT) of EP0 Status Register. 0: Disable 1: Enable	R/W
b15	EP0_OUT_NAK_EN	Enable for bit 15 (EP0_OUT_NAK_INT) of EP0 Status Register. 0: Disable 1: Enable	R/W
b14 to b12	Reserved		R
b11	EP0_IN_NAK_EN	Enable for bit 11 (EP0_IN_NAK_INT) of EP0 Status Register. 0: Disable 1: Enable	R/W
b10 to b8	Reserved		R
b7	EP0_OUT_NULL_EN	Enable for bit 7 (EP0_OUT_NULL_INT) of EP0 Status Register. 0: Disable 1: Enable	R/W
b6	EP0_OUT_OR_EN	Enable for bit 6 (EP0_OUT_OR_INT) of EP0 Status Register. 0: Disable 1: Enable	R/W
b5	EP0_OUT_EN	Enable for bit 5 (EP0_OUT_INT) of EP0 Status Register. 0: Disable 1: Enable	R/W
b4	EP0_IN_EN	Enable for bit 4 (EP0_IN_INT) of EP0 Status Register. 0: Disable 1: Enable	R/W
b3	EP0_STALL_EN	Enable for bit 3 (EP0_STALL_INT) of EP0 Status Register. 0: Disable 1: Enable	R/W
b2	STG_END_EN	Enable for bit 2 (STG_END_INT) of EP0 Status Register. 0: Disable 1: Enable	R/W

Table 10.98 EP0_INT_ENA Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b1	STG_START_EN	Enable for bit 1 (STG_START_INT) of EP0 Status Register. 0: Disable 1: Enable	R/W
b0	SETUP_EN	Enable for bit 0 (SETUP_INT) of EP0 Status Register. 0: Disable 1: Enable	R/W

10.5.7.12 EP0_LENGTH — EP0 OUT Data Length Register

Address: 4001 E034h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	EP0_LDATA						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.99 EP0_LENGTH Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b7	Reserved		R
b6 to b0	EP0_LDATA	If OUT data is received in the EP0 Read Register (reception buffer) normally, this field indicates the number of received bytes. The value is decremented by reading EP0 Read Register, and it indicates number of the remaining data.	R

10.5.7.13 EP0_READ — EP0 Read Register

Address: 4001 E038h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	EP0_RDATA4								EP0_RDATA3							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	EP0_RDATA2								EP0_RDATA1							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.100 EP0_READ Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	EP0_RDATA4	Endpoint0 received data	R
b23 to b16	EP0_RDATA3	Endpoint0 received data	R
b15 to b8	EP0_RDATA2	Endpoint0 received data	R
b7 to b0	EP0_RDATA1	Endpoint0 received data	R

10.5.7.14 EP0_WRITE — EP0 Write Register

Address: 4001 E03Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	EP0_WDATA4								EP0_WDATA3							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	EP0_WDATA2								EP0_WDATA1							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.101 EP0_WRITE Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	EP0_WDATA4	Endpoint0 transmit data	W
b23 to b16	EP0_WDATA3	Endpoint0 transmit data	W
b15 to b8	EP0_WDATA2	Endpoint0 transmit data	W
b7 to b0	EP0_WDATA1	Endpoint0 transmit data	W

10.5.7.15 EP[m]_CONTROL — EP[m] Control Register (m = 1..15)

Address: 4001 E040h + 20h × (m − 1)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	EP[m]_EN	EP[m]_BUF_TYPE	—	—	—	EP[m]_DIR0	EP[m]_MODE		—	—	—	—	—	—	EP[m]_OVERSEL	EP[m]_AUTO
Value after reset	0	X	0	0	0	0	X	X	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	EP[m]_IPIDCLR	EP[m]_OPIDCLR	EP[m]_BCLR	EP[m]_CBCLR	EP[m]_DEND	EP[m]_DW		EP[m]_OSTL_EN	EP[m]_ISTL	EP[m]_OSTL	—	EP[m]_ONAK
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 10.102 EP[m]_CONTROL Register Contents (1/3)

Bit Position	Bit Name	Function	R/W
b31	EP[m]_EN	Enable of Endpoint[m] If this bit is set while bit 5 (CONF) of USB Control Register is 1b, the corresponding Endpoint can respond to USB transactions. When this bit is 0b, the EP[m] Write/Read Register (transmission/reception buffer) is cleared. If this bit is written during a USB transaction for Endpoint[m], the write is held until the transaction finishes. 0: Disable Endpoint[m] 1: Enable Endpoint[m] ***Caution*)** This bit is assumed to be changed only when it receives BusReset, USB request (such as Set Interface), or it initializes. No DATA PID initialization occurs even if this bit is cleared.	R/W
b30	EP[m]_BUF_TYPE	Buffer type of EP[m] buffering. Read-only. 0: Single buffering 1: Double buffering	R
b29 to b27	Reserved	Should be 0 at writing.	R
b26	EP[m]_DIR0	Direction of Endpoint[m] 0: IN 1: OUT ***Caution*)** This bit is assumed to be changed during initialization only. If this bit is changed during a transfer, the operation is not guaranteed.	R/W
b25, b24	EP[m]_MODE	Transfer type of Endpoint[m]. Read only. 00b: Bulk 01b: Interrupt 10b: Isochronous 11b: Reserved	R
b23 to b18	Reserved	Should be 0 at writing.	R
b17	EP[m]_OVERSEL	Select the operation for next OUT token when an overrun occurs in OUT transfer. If this bit is changed during a transfer, the operation is not guaranteed. Invalid for Isochronous transfer. 0: Return a STALL for the next packet 1: Handle the next packet as a retry (not for use)	R/W

Table 10.102 EP[m]_CONTROL Register Contents (2/3)

Bit Position	Bit Name	Function	R/W
b16	EP[m]_AUTO	Select whether to send a packet automatically when MaxPacketSize data is written to EP[m] Write Register (transmission buffer). There is a case when this function should be disabled, maximum packet size − 1, 2, or 3 bytes, as shown in the example below. (If this function is not disabled, all data becomes valid at writing maximum packet size) If last data includes invalid data, clear this bit and use EP[m]_DW[1:0]. ***Ex1)*** 32 bit CPU MaxPacketSize = 512, transmit size = 509 to 511 (Valid data size will be 512 bytes) ***Ex2)*** 32 bit CPU MaxPacketSize = 511, transmit size = 509,510 (Valid data size will be 511 bytes) 0: Not set EP[m]_DEND automatically 1: Set EP[m]_DEND automatically ***Caution)*** Setting this bit is prohibited if bit 4 (EP[m]_DMA_EN) of the EP[m] DMA Control Register is 1b. In addition, do not use this bit if MaxPacketSize=000h.	R/W
b15 to b12	Reserved	Should be 0 at writing.	R
b11	EP[m]_IPIDCLR	Use this bit to initialize transmission DATA PID for Endpoint[m]. The transmission DATA PID is initialized by writing 1b. If this bit is written during a USB transaction for Endpoint[m], the write is held until the transaction finishes. This bit is write-only. Read as 0. ***Caution)*** DATA PID is initialized when it receives BusReset.	W
b10	EP[m]_OPIDCLR	Use this bit to initialize receive DATA PID for Endpoint[m]. The receive DATA PID is initialized by writing 1b. If this bit is written during a USB transaction for Endpoint[m], the write is held until the transaction finishes. This bit is write-only. Read as 0. ***Caution)*** DATA PID is initialized when it receives BusReset.	W
b9	EP[m]_BCLR	Use this bit to clear both EP[m] Write and EP[m] Read Registers on CPU side and USB side. Both transmission and reception buffers on CPU side and USB side are initialized by writing 1b. If this bit is written during a USB transaction for Endpoint[m], the write is held until the transaction finishes. In case of EP[m]_DIR0=0, it needs to check EP[m]_IN_DATA =0 and EP[m]_IN_EMPTY=1 of EP[m] Status Register after setting this bit. This bit is write-only. Read as 0. ***Caution)*** Setting this bit together with EP[m]_CBCLR is prohibited. In addition, setting this bit is prohibited if bit 4 (EP[m]_DMA_EN) of the EP[m] DMA Control Register is 1b. EP[m] Write and EP[m] Read Registers are not cleared even if USB BusReset is received.	W
b8	EP[m]_CBCLR	Use this bit to clear both EP[m] Write and EP[m] Read Registers on CPU side. Both transmission and reception buffer on CPU side are initialized by writing 1b. This bit is invalid in single buffer case. This bit is write-only. Read as 0. ***Caution)*** Setting this bit together with EP[m]_BCLR is prohibited. In addition, setting this bit is prohibited if bit 4 (EP[m]_DMA_EN) of the EP[m] DMA Control Register is 1b.	W

Table 10.102 EP[m]_CONTROL Register Contents (3/3)

Bit Position	Bit Name	Function	R/W
b7	EP[m]_DEND	Transmission enable of EP[m] write Register The transmission data is enabled by writing 1b. When it sends null data, make sure bit 0 (EP[m]_IN_EMPTY) of the EP[m] Status Register is 1b, then write 1b to this bit. This bit is write-only. Read as 0. ***Caution*)** Setting this bit is prohibited if bit 4 (EP[m]_DMA_EN) of the EP[m] DMA Control Register is 1b.	W
b6, b5	EP[m]_DW	Valid data size for last data written in EP[m] Write Register (transmission buffer). Set these bits together with EP[m]_DEND=1. 00b: 4 01b: 1 10b: 2 11b: 3 This field is write-only. Read as 0.	W
b4	EP[m]_OSTL_EN	Write enable bit of EP[m]_OSTL When writing to EP[m]_OSTL, set 1b at the same time. This bit is write-only. Read as 0.	W
b3	EP[m]_ISTL	STALL response control bit for IN token to Endpoint[m] 0: Not return STALL for IN token 1: Return STALL for IN token	R/W
b2	EP[m]_OSTL	STALL response control bit for OUT/PING token to Endpoint[m] EP[m]_OSTL_EN should be 1 together when it writes 1 to this bit. This bit is set to 1b if overrun occur when EP[m]_OVERSEL=0. 0: Not return STALL for OUT/PING token 1: Return STALL for OUT/PING token	R/W
b1	Reserved	Should be 0 at writing.	R
b0	EP[m]_ONAK	NAK response control bit for OUT/PING token to Endpoint[m]. Generally ACK, NAK, and NYET are issued automatically according to status of EP[m] Read Register (reception buffer). Write 1b to this bit to send a NAK by force. 0: Receive data if there is available space in the reception buffer 1: Return a NAK even if there is available space in the reception buffer	R/W

10.5.7.16 EP[m]_STATUS — EP[m] Status Register (m = 1..15)

Address: 4001 E044h + 20h × (m − 1)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	EP[m]_ISO_PIDERR	EP[m]_OPID	EP[m]_OUT_NOTKN	EP[m]_ISO_OR	—	EP[m]_ISO_CRC	EP[m]_OUT_END_INT	EP[m]_OUT_OR_INT	EP[m]_OUT_NAK_ERR_INT	EP[m]_OUT_STALL_INT	EP[m]_OUT_INT	EP[m]_OUT_NULL_INT	EP[m]_OUT_FULL	EP[m]_OUT_EMPTY
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	EP[m]_IPID	EP[m]_IN_NOTKN	EP[m]_ISO_UR	EP[m]_IN_END_INT	—	EP[m]_IN_NAK_ERR_INT	EP[m]_IN_STALL_INT	EP[m]_IN_INT	EP[m]_IN_DATA	EP[m]_IN_FULL	EP[m]_IN_EMPTY
Value after reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1

Table 10.103 EP[m]_STATUS Register Contents (1/4)

Bit Position	Bit Name	Function	R/W
b31, b30	Reserved	Should be 0 at writing.	R
b29	EP[m]_ISO_PIDERR	• For Isochronous Endpoint: This bit is set if it receives illegal DATA PID. In HS mode, this bit is set when received DATA PID is DATA1, DATA2, or MDATA. In FS mode, this bit is set when received DATA PID is DATA1. 0: No invalid DATA PID has been received 1: Invalid DATA PID was received • For Interrupt/Bulk Endpoint: Always 0	R
b28	EP[m]_OPID	Indicates next expected DATA PID. 0: DATA0 1: DATA1	R
b27	EP[m]_OUT_NOTKN	• For Isochronous Endpoint: This bit is set if no OUT token is received in the interval between SOFs or uSOFs. This bit is updated each time SOF or uSOF is received. 0: OUT token was received between SOFs or uSOFs 1: No OUT tokens have been received between SOFs or uSOFs • For Interrupt/Bulk Endpoint: Always 0	R
b26	EP[m]_ISO_OR	• For Isochronous Endpoint: This bit is set when OUT token is received while there is no room in EP[m] Read Register (reception buffer) and OUT data is discarded. This bit is updated each time data is received. 0: No OUT data has been discarded 1: The received OUT data was discarded • For Interrupt/Bulk Endpoint: Always 0	R
b25	Reserved	Should be 0 at writing.	R

Table 10.103 EP[m]_STATUS Register Contents (2/4)

Bit Position	Bit Name	Function	R/W
b24	EP[m]_ISO_CRC	• For Isochronous Endpoint: This bit is set if the received data includes a CRC error. If the data is not required, set bit 8 (EP[m]_CBCLR) of the EP[m] Control Register to clear the received data. Note that received null data is cleared automatically. This bit is updated each time data is received. This bit is cleared when the reception buffer is cleared. 0: The received data does not include a CRC error 1: The received data includes a CRC error • For Interrupt/Bulk Endpoint: Always 0	R
b23	EP[m]_OUT_END_INT	This bit is set when buffer read DMA for OUT-direction Endpoint[m] completes. If this bit is set, bit 4 (EP[m]_DMA_EN) of the EP[m] DMA Control Register is cleared together. This bit is set when it receives a short packet if bit 11 (EP[m]_STOP_MODE) and bit 8 (EP[m]_STOP_SET) of the EP[m] DMA Control Register are 1b This bit is cleared by writing 0b. 0: Buffer read DMA not completed 1: Buffer read DMA completed	R/W
b22	EP[m]_OUT_OR_INT	This bit is set to 1b if an overrun occurs when it receives a data at Endpoint[m] This bit is cleared by writing 0b. 0: No overrun occurred 1: Overrun occurred	R/W
b21	EP[m]_OUT_NAK_ERR_INT	• For Interrupt/Bulk Endpoint: 0: No NAK response for OUT/PING token 1: NAK was sent for OUT/PING token • For Isochronous Endpoint: 0: No reception error occurred 1: Reception error occurred Detail of the error can be checked by EP[m]_ISO_PIDERR, EP[m]_OUT_NOTKN, EP[m]_ISO_OR, EP[m]_ISO_CRC This bit is cleared by writing 0b.	R/W
b20	EP[m]_OUT_STALL_INT	This bit is set when Endpoint[m] becomes STALL while bit 26 (EP[m]_DIR0) of USB Control Register is 1b (OUT direction). This bit is cleared by writing 0b. 0: EP[m] is OUT and not in STALL 1: EP[m] is OUT and in STALL	R/W
b19	EP[m]_OUT_INT	This bit is set when valid data other than null has been received normally in EP[m] Read Register (reception buffer) and CPU side buffer becomes readable. If null data is received, EP[m]_OUT_NULL_INT is set. This bit is cleared by writing 0b. 0: Reception buffer has not become readable yet. 1: Reception buffer becomes readable.	R/W
b18	EP[m]_OUT_NULL_INT	This bit is set when null data is received in EP[m] Read Register (reception buffer) normally. In case of double-buffer, this bit is set when null data packets are toggled to CPU side. The null data packet is cleared when this bit is set. (The reception buffer become ready to receive next packet.) This bit is cleared by writing 0b. 0: Null data has not been received 1: Null data was received	R/W
b17	EP[m]_OUT_FULL	EP[m] Read Register (reception buffer) full status. In case of double-buffer, this bit is set if CPU side buffer is full. This bit is updated according to the buffer status. 0: Reception buffer is not full 1: Reception buffer is full	R

Table 10.103 EP[m]_STATUS Register Contents (3/4)

Bit Position	Bit Name	Function	R/W
b16	EP[m]_OUT_EMPTY	EP[m] Read Register (reception buffer) empty status. In case of double-buffer, this bit is set if CPU side buffer is empty. This bit is updated according to the reception buffer status. 0: Reception buffer is not empty 1: Reception buffer is empty	R
b15 to b11	Reserved	Keep the initial value	R/W
b10	EP[m]_IPID	Indicates next DATA PID. 0: DATA0 1: DATA1	R
b9	EP[m]_IN_NOTKN	• For Isochronous Endpoint: This bit is set if no IN token is received in the interval between SOFs or uSOFs This bit is updated each time SOF or uSOF is received. 0: IN token was received between SOFs or uSOFs 1: No IN tokens have been received between SOFs or uSOFs	R
b8	EP[m]_ISO_UR	• For Isochronous Endpoint: This bit is set when IN token is received before data has prepared in EP[m] Write Register (transmission buffer) and null data is sent automatically. This bit is updated each time data is sent. 0: Null data has not been sent automatically 1: Null data was sent automatically • For Interrupt/Bulk Endpoint: Always 0	R
b7	EP[m]_IN_END_INT	This bit is set when buffer write DMA for IN-direction Endpoint[m] completes. If this bit is set, bit 4 (EP[m]_DMA_EN) of the EP[m] DMA Control Register is cleared together. This bit is cleared by writing 0b. 0: Buffer write DMA not completed 1: Buffer write DMA completed	R/W
b6	Reserved	Should be 0 at writing.	R
b5	EP[m]_IN_NAK_ERR_INT	• For Interrupt/Bulk Endpoint: 0: No NAK response for IN token 1: NAK was sent for IN token • For Isochronous Endpoint: 0: No transmission error occurred 1: Transmission error occurred Detail of the error can be checked by EP[m]_IN_NOTKN and EP[m]_ISO_UR This bit is cleared by writing 0b.	R/W
b4	EP[m]_IN_STALL_INT	This bit is set when Endpoint[m] becomes STALL while bit 26 (EP[m]_DIR0) of USB Control Register is 0b (IN direction). 0: EP[m] is IN and not in STALL 1: EP[m] is IN and in STALL This bit is cleared by writing 0b.	R/W
b3	EP[m]_IN_INT	This bit is set when next data can be written to CPU side of EP[m] Write Register (transmission buffer). In case of single buffer, this bit is set when data transmitted normally for IN transactions. In case of double-buffer, this bit is set when CPU side transmission buffer data that granted permission has been toggled to USB side. 0: Transmission buffer has not become writable yet. 1: Transmission buffer becomes writable.	R/W
b2	EP[m]_IN_DATA	EP[m] Write Register (transmission buffer) data valid This bit is updated according to the buffer status. 0: There is no waiting data in the transmission buffer. 1: There is a waiting data in the transmission buffer.	R

Table 10.103 EP[m]_STATUS Register Contents (4/4)

Bit Position	Bit Name	Function	R/W
b1	EP[m]_IN_FULL	EP[m] Write Register (transmission buffer) full status. This bit is cleared when bit 7 (EP[m]_DEND) of the EP[m] Control Register is set to 1b and data transmission is enabled. This bit is updated according to the buffer status. 0: Transmission buffer is not full 1: Transmission buffer is full	R
b0	EP[m]_IN_EMPTY	EP[m] Write Register (transmission buffer) empty status. This bit is updated according to the buffer status. 0: Transmission buffer is not empty 1: Transmission buffer is empty	R

10.5.7.17 EP[m]_INT_ENA — EP[m] Interrupt Enable Register (m = 1..15)

If each field is disabled, the corresponding interrupt is not asserted even if EP[m] Status Register is set.

Address: 4001 E048h + 20h × (m − 1)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	EP[m]_OUT_END_EN	EP[m]_OUT_OR_EN	EP[m]_OUT_NAK_ERR_EN	EP[m]_OUT_STALL_EN	EP[m]_OUT_EN	EP[m]_OUT_NULL_EN	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	EP[m]_IN_END_EN	—	EP[m]_IN_NAK_ERR_EN	EP[m]_IN_STALL_EN	EP[m]_IN_EN	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.104 EP[m]_INT_ENA Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	Should be 0 at writing.	R
b23	EP[m]_OUT_END_EN	Enable for bit 23 (EP[m]_OUT_END_INT) of EP[m] Status Register. 0: Disable 1: Enable	R/W
b22	EP[m]_OUT_OR_EN	Enable for bit 22 (EP[m]_OUT_OR_INT) of EP[m] Status Register. 0: Disable 1: Enable	R/W
b21	EP[m]_OUT_NAK_ERR_EN	Enable for bit 21 (EP[m]_OUT_NAK_ERR_INT) of EP[m] Status Register. 0: Disable 1: Enable	R/W
b20	EP[m]_OUT_STALL_EN	Enable for bit 20 (EP[m]_OUT_STALL_INT) of EP[m] Status Register. 0: Disable 1: Enable	R/W
b19	EP[m]_OUT_EN	Enable for bit 19 (EP[m]_OUT_INT) of EP[m] status Register. 0: Disable 1: Enable	R/W
b18	EP[m]_OUT_NULL_EN	Enable for bit 18 (EP[m]_OUT_NULL_INT) of EP[m] Status Register. 0: Disable 1: Enable	R/W
b17 to b8	Reserved	Should be 0 at writing.	R
b7	EP[m]_IN_END_EN	Enable for bit 7 (EP[m]_IN_END_INT) of EP[m] Status register. 0: Disable 1: Enable	R/W
b6	Reserved	Should be 0 at writing.	R
b5	EP[m]_IN_NAK_ERR_EN	Enable for bit 5 (EP[m]_IN_NAK_ERR_INT) of EP[m] Status Register. 0: Disable 1: Enable	R/W
b4	EP[m]_IN_STALL_EN	Enable for bit 4 (EP[m]_IN_STALL_INT) of EP[m] Status Register. 0: Disable 1: Enable	R/W
b3	EP[m]_IN_EN	Enable for bit 3 (EP[m]_IN_INT) of EP[m] Status Register. 0: Disable 1: Enable	R/W
b2 to b0	Reserved	Should be 0 at writing.	R

10.5.7.18 EP[m]_DMA_CTRL — EP[m] DMA Control Register (m = 1..15)

Address: 4001 E04Ch + 20h × (m − 1)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	EP[m]_STOP_MODE	EP[m]_DEND_SET	EP[m]_BURST_SET	EP[m]_STOP_SET	—	—	—	EP[m]_DMA_EN	—	—	—	EP[m]_DMAMODE0
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.105 EP[m]_DMA_CTRL Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b12	Reserved	Should be 0 at writing.	R/W
b11	EP[m]_STOP_MODE	DMA stop conditions setting when bit 8 (EP[m]_STOP_SET) is 1b. 0: DMA stops when a short packet is received, and the DMA transfer completed. 1: DMA stops when a short packet is received and it is ready to read. (DMA does not stop if EP[m]_DMA_EN is set after the received short packet is ready to read)	R/W
b10	EP[m]_DEND_SET	Specify whether to enable setting bit 7 (EP[m]_DEND) of the EP[m] Control Register to 1b when a DMA completion from AHB-EPC bridge is received when bit 26 (EP[m]_DIR0) of the USB Control Register is 0. This bit is invalid for OUT transactions. Note that data less than 32 bit cannot be transferred, if this bit is used. 0: Not set EP[m]_DEND automatically 1: Set EP[m]_DEND automatically	R/W
b9	EP[m]_BURST_SET	Specify whether to clear EP[m]_DMA_EN each time one DMA packet completion 0: Clear EP[m]_DMA_EN automatically 1: Not clear EP[m]_DMA_EN automatically **[Notes on DMA write]** Set this bit and bit 16 (EP[m]_AUTO) of the EP[m] Control Register to 1b when successively transferring data of the maximum packet size by DMA. Clear this bit when transferring a short packet. If a last packet contains fractional data, write the data by PIO. **[Notes on DMA read]** If a short packet that includes null data is received when bit 8 (EP[m]_STOP_SET) is 1b, EP[m]_DMA_EN is cleared even if this bit is 1b. Set this bit to 1b when successively transferring data of the MaxPacketSize by DMA. Clear this bit when transferring a short packet. If a last packet contains fractional data, read the data by PIO	R/W
b8	EP[m]_STOP_SET	Select whether to clear bit 4 (EP[m]_DMA_EN) 0b and send the DMA completion notice to AHB-EPC bridge for stopping DMA transfer when a short packet that includes null data is received while bit 26 (EP[m]_DIR0) of the USB Control Register is 1b (OUT direction). 0: Reserved (not for use) 1: Clear EP[m]_DMA_EN and send DMA completion notice to AHB-EPC bridge	R/W
b7 to b5	Reserved	Should be 0 at writing.	R

Table 10.105 EP[m]_DMA_CTRL Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b4	EP[m]_DMA_EN	Endpoint[m] DMA enable 0: Not use DMA 1: Use DMA This bit is cleared to 0b under the following conditions: A DMA transfer specified by EP[m]DCR2 has completed. One DMA packet completion while bits EP[m]_DMACNT of EP[m]_LEN_DCNT is 001h. One DMA packet completion while EP[m]_BURST_SET= 0. (If a last data contains fractional data, this bit is cleared before transferring last data.) If a short packet that includes null data is received while EP[m]_STOP_SET=1 and the data has been transferred by DMA. (If EP[m]_STOP_MODE is 1b, it stops when short packet that includes null data is ready to read.) This bit is not cleared even if the Endpoint becomes STALL condition due to overrun. If this bit is 1b, PIO access for EP[m] Write and Read Registers is not possible. Similarly, setting bit 7 (EP[m]_DEND) of the EP[m] Control Register is prohibited. Be sure to clear this bit before using PIO access after a DMA transfer or when setting the EP[m]_DEND bit. Before clearing this bit, stop the DMA master in advance so that no transfer is performed. If this bit is cleared during a DMA transfer, the transferred data is not guaranteed.	R/W
b3 to b1	Reserved	Should be 0 at writing.	R
b0	EP[m]_DMAMODE0	DMA mode select 0: Single (Not for use) 1: Demand	R/W

10.5.7.19 EP[m]_PCKT_ADRS — EP[m] MaxPacket & BaseAddress Register (m = 1..15)

Address: 4001 E050h + 20h × (m − 1)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	EP[m]_BASEAD												
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	EP[m]_MPKT										
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.106 EP[m]_PCKT_ADRS Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b29	Reserved	Should be 0 at writing.	R
b28 to b16	EP[m]_BASEAD	Base address RAM mapping for Endpoint[m] This field should be set while bit 31 (EP[m]_EN) of the EP[m] Control Register is 0b.	R/W
b15 to b11	Reserved	Should be 0 at writing.	R
b10 to b0	EP[m]_MPKT	MaxPacketSize for Endpoint[m]. This field should be set while bit 31 (EP[m]_EN) of the EP[m] Control Register is 0b.	R/W

10.5.7.20 EP[m]_LEN_DCNT — EP[m] Length & DMA Count Register (m = 1..15)

Address: 4001 E054h + 20h × (m − 1)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	EP[m]_DMACNT								
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	EP[m]_LDATA										
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.107 EP[m]_LEN_DCNT Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b25	Reserved	Should be 0 at writing.	R/W
b24 to b16	EP[m]_DMACNT	Arbitrary number of DMA transfer can be specified by these bits. Set DMA packet number to these bits. The value is decremented each time of one DMA packet completion. Bit 4 (EP[m]_DMA_EN) of the EP[m] DMA Control Register is cleared to 0b at when these bits are decremented to 000h. To use this function, set bit 9 (EP[m]_BURST_SET) of EP[m] DMA Control Register. This function is disabled when this field is set to 000h. ***Caution*)** Maximum available value is 100h (= 256 packets) Do not specify the number of transferred bytes.	R/W
b15 to b11	Reserved	Should be 0 at writing.	R
b10 to b0	EP[m]_LDATA	Indicates the number of readable bytes stored in the EP[m] Read Register (reception buffer) on the CPU side. The value is decremented by reading EP[m] Read Register.	R

10.5.7.21 EP[m]_READ — EP[m] Read Register (m = 1..15)

Address: 4001 E058h + 20h × (m − 1)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	EP[m]_RDATA4								EP[m]_RDATA3							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	EP[m]_RDATA2								EP[m]_RDATA1							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.108 EP[m]_READ Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	EP[m]_RDATA4	Endpoint[m] received data	R
b23 to b16	EP[m]_RDATA3	Endpoint[m] received data	R
b15 to b8	EP[m]_RDATA2	Endpoint[m] received data	R
b7 to b0	EP[m]_RDATA1	Endpoint[m] received data	R

10.5.7.22 EP[m]_WRITE — EP[m] Write Register (m = 1..15)

Address: 4001 E05Ch + 20h × (m − 1)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	EP[m]_WDATA4								EP[m]_WDATA3							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	EP[m]_WDATA2								EP[m]_WDATA1							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.109 EP[m]_WRITE Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	EP[m]_WDATA4	Endpoint[m] transmit data	W
b23 to b16	EP[m]_WDATA3	Endpoint[m] transmit data	W
b15 to b8	EP[m]_WDATA2	Endpoint[m] transmit data	W
b7 to b0	EP[m]_WDATA1	Endpoint[m] transmit data	W

10.5.8 AHB-EPC Bridge Register Description

10.5.8.1 AHBSCTR — AHB Slave Controller Configuration Register

Address: 4001 F000h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	WAIT_MODE
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 10.110 AHBSCTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b1	Reserved	Should be 0 at writing.	R/W
b0	WAIT_MODE	Wait behavior control setting for AHB slave 0: Reserved (not for use) 1: Wait control by HREADY	R/W

10.5.8.2 AHBMCTR — AHB Master Controller Configuration Register

Address: 4001 F004h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	ARBITER_CTR	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	MCYCLE_RST	—	—	ENDIAN_CTR		—	—	—	—	—	WBURST_TYPE	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

Table 10.111 AHBMCTR Register Contents

Bit Position	Bit Name	Function	R/W
b31	ARBITER_CTR	Select the arbitration type for the Endpoint which uses DMA transfer. 0: Round Robin priority (EP1 → ... → EP15 → EP1) 1: Fixed priority (EP1 > EP2 > ... > EP15)	R/W
b30 to b13	Reserved	Should be 0 at writing.	R
b12	MCYCLE_RST	Reserved in this system, should be 0 at writing.	R/W
b11, b10	Reserved	Should be 0 at writing.	R
b9, b8	ENDIAN_CTR	Select the data conversion type during DMA transfer. 00: Little endian Other: Reserved (not for use)	R/W
b7 to b3	Reserved	Keep the initial value	R/W
b2	WBURST_TYPE	Variable-length burst use condition for AHB master write 0: Reserved (not for use) 1: INCR4/8/16 + INCR for 2 to 3 burst	R/W
b1, b0	Reserved	Should be 0 at writing.	R/W

10.5.8.3 AHBBINT — AHB-EPC Bridge Interrupt Source Register

Address: 4001 F008h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DMA_ENDINT_EP[15:1]															—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	VBUS_INT	—	—	—	—	—	—	MBUS_ERRINT	—	SBUS_ERRINT0	ERR_MASTER			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.112 AHBBINT Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b17	DMA_ENDINT_EP [15:1]	These bits are set when DMA transfer for corresponding Endpoint is completed This field is cleared by writing 1b.	R/W
b16 to b14	Reserved	Should be 0 at writing.	R/W
b13	VBUS_INT	This bit is set when VBUS signal level has changed. This field is cleared by writing 1b.	R/W
b12 to b7	Reserved	Should be 0 at writing.	R/W
b6	MBUS_ERRINT	This bit is set when the AHB master received error response. This bit is cleared by writing 1b.	R/W
b5	Reserved	Should be 0 at writing.	R/W
b4	SBUS_ERRINT0	This bit is set when the AHB slave issued error response for over 32 bit access. This bit is cleared by writing 1b.	R/W
b3 to b0	ERR_MASTER	This field stores the master number at error response when SBUS_ERRINT0 is 1b. This field retains a value until the SBUS_ERRINT0 bit is cleared to 0b.	R

10.5.8.4 AHBBINTEN — AHB-EPC Bridge Interrupt Enable Register

If each field is disabled, the corresponding interrupt is not asserted even if AHB-EPC Bridge Interrupt Source Register is set.

Address: 4001 F00Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DMA_ENDINTEN_EP[15:1]															—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	VBUS_INTEN	—	—	—	—	—	—	MBUS_ERRINTEN	—	SBUS_ERRINT0EN	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.113 AHBBINTEN Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b17	DMA_ENDINTEN_EP[15:1]	Enable for DMA_ENDINT_EP[15:1] of AHBBINT register 0: Disable 1: Enable	R/W
b16 to b14	Reserved	Should be 0 at writing.	R/W
b13	VBUS_INTEN	Enable for bit 13 (VBUS_INT) of the AHBBINT register. 0: Disable 1: Enable	R/W
b12 to b7	Reserved	Should be 0 at writing.	R/W
b6	MBUS_ERRINTEN	Enable for bit 6 (MBUS_ERRINT) of the AHBBINT register. 0: Disable 1: Enable	R/W
b5	Reserved	Should be 0 at writing.	R/W
b4	SBUS_ERRINT0EN	Enable for bit 4 (SBUS_ERRINT0) of the AHBBINT register. 0: Disable 1: Enable	R/W
b3 to b0	Reserved	Should be 0 at writing.	R

10.5.8.5 EPCTR — EPC and Transceiver Control Register

Address: 4001 F010h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	DIRPD	—	—	—	VBUS_LEVEL	—	—	PLL_RESUME	PLL_LOCK	—	PLL_RST	—	EPC_RST
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Table 10.114 EPCTR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b13	Reserved	Should be 0 at writing.	R
b12	DIRPD	If this bit is set to 1b, USB subsystem moves to power-down state. Bit 8 (DIRPD) of the USBCTR register and DIRPD of CFG_USB in the system controller have same function. 0: Normal operation 1: Direct power-down mode	R/W
b11 to b9	Reserved	Should be 0 at writing.	R
b8	VBUS_LEVEL	Indicates the status of the VBUS input pin. 0: VBUS = 0 1: VBUS = 1	R
b7, b6	Reserved	Should be 0 at writing.	R
b5	PLL_RESUME	By setting this bit, the clock supply can be resumed, when SIE is in Suspend state and the clock for the function controller is stopped. Be sure to clear this bit to 0b after the clock supply is resumed. 0: Normal operation 1: Resume clock supply	R/W
b4	PLL_LOCK	Indicates USBPLL lock status 0: USBPLL has not been locked 1: USBPLL is locked	R
b3	Reserved	Should be 0 at writing.	R/W
b2	PLL_RST	Reset of USBPLL The USBPLL is shared between the host and function controllers. Reset to the USBPLL is asserted when both PLL_RST in host and function controllers are 1b. 0: Release USBPLL reset 1: Assert USBPLL reset	R/W
b1	Reserved	Should be 1 at writing.	R/W
b0	EPC_RST	Reset of EPC block. 0: Release EPC reset 1: Assert EPC reset	R/W

10.5.8.6 USBSSVER — USBf Version Register

Address: 4001 F020h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	AHBB_VER							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	EPC_VER								SS_VER							
Value after reset	0	0	1	0	0	1	0	0	x	x	x	x	x	x	x	x

Table 10.115 USBSSVER Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved		R
b23 to b16	AHBB_VER	Indicates the AHB bridge version.	R
b15 to b8	EPC_VER	Indicates the EPC version.	R
b7 to b0	SS_VER	Indicates the USB function controller version. RZ/N1D: 31h RZ/N1S, RZ/N1L: 41h	R

10.5.8.7 USBSSCONF — Endpoint Configuration Register

Address: 4001 F024h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	EP_AVAILABLE															
Value after reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DMA_AVAILABLE															
Value after reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Table 10.116 USBSSCONF Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	EP_AVAILABLE	Indicates the implemented Endpoint modules. Each bit in this field corresponds to Endpoint number. 0: Not available 1: Available	R
b15 to b0	DMA_AVAILABLE	Indicates an Endpoint that can be used for DMA transfers. Each bit in this field corresponds to Endpoint number. Since Endpoint0 is not available for DMA transfers, bit 0 always indicates 0b. 0: Not Available 1: Available	R

10.5.8.8 EP[m]DCR1 — Endpoint[m] DMA Setting Register1 (m = 1..15)

Address: 4001 F110h + 10h × (m − 1)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	EP[m]_DMACNT							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	EP[m]_DIR0	EP[m]_REQEN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.117 EP[m]DCR1 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved	Should be 0 at writing.	R
b23 to b16	EP[m]_DMACNT	Set the number of packets (not bytes) in DMA transfer. Set same the value as EP[m]_DMACNT[7:0] of the EP[m] Length & DMA Count Register, except for handling a data packet that cannot be divided by 32 bits. However, if the EP[m]_DMACNT[8:0] bits are set to 100h, set this field to 00h. This field is decremented at every one packet DMA transfer completion. ***Caution*)** Don't change this bit while EP[m]_REQEN is 1b.	R/W
b15 to b2	Reserved	Should be 0 at writing.	R
b1	EP[m]_DIR0	DMA direction setting Set same the value as EP[m]_DIR0 of the EP[m] Control Register. 0: IN direction (from AHB to EPC) 1: OUT direction (from EPC to AHB) ***Caution*)** Don't change this bit while EP[m]_REQEN is 1b.	R/W
b0	EP[m]_REQEN	Setting for DMA transfer requests from EPC (Endpoint controller) This bit is cleared to 0b when the number of packets specified by the EP[m]_DMACNT have been transferred or when the EPC receives a short packet and it ends the DMA transfer. 0: Ignore (DMA transfer not available) 1: Permit (DMA transfer is available)	R/W

10.5.8.9 EP[m]DCR2 — Endpoint[m] DMA Setting Register2 (m = 1..15)

Address: 4001 F114h + 10h × (m − 1)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	EP[m]_LMPKT										
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	EP[m]_MPKT										
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.118 EP[m]DCR2 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b27	Reserved	Should be 0 at writing.	R
b26 to b16	EP[m]_LMPKT	Byte length setting of last packet for DMA transfer IN transfer (EP[m]_DIR0 of the EP[m]DCR1 Register = 0b) Set the number of bytes to be sent by DMA in the last packet. DMA transfer completes when the specified number of bytes been transferred. Since data is transferred in 32-bit unit, the lower 2 bits are ignored. PIO transfer should be used for 3 bytes or less. <*Example*> 512 bytes (maximum packet size) → 200h 511 bytes (short packet) → 1FCh ***Caution*)** Don't change this bit while EP[m]_REQEN is 1b. OUT transfer (EP[m]_DIR0 of the EP[m]DCR1 Register = 1b) Indicates the number of bytes transferred in the last packet by DMA. Since data is transferred in 32-bit unit, the lower 2 bits are ignored. Writing to this field is invalid during OUT transfer.	R/W
b15 to b11	Reserved	Should be 0 at writing.	R
b10 to b0	EP[m]_MPKT	Set maximum packet size of Endpoint[m] Set same the value as EP[m]_MPKT[10:0] of the EP[m] MaxPacket & BaseAddress Register. ***Caution*)** Don't change this bit while EP[m]_REQEN is 1b.	R/W

10.5.8.10 EP[m]TADR — Endpoint[m] DMA Start Address Register (m = 1..15)

Address: 4001 F118h + 10h × (m − 1)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	EP[m]_TADR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	EP[m]_TADR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 10.119 EP[m]TADR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	EP[m]_TADR	DMA start address for AHB (System address) Since data is transferred in 32-bit unit, the lower 2 bits are ignored. ***Caution*)** Don't change this bit while EP[m]_REQEN is 1b.	R/W

10.6 Usage Notes

10.6.1 Accessing Function Controller Registers

10.6.1.1 Notes on Accessing EPC Registers

When accessing EPC registers, make sure the USB_HCLKF is supplied to the function controller. If not, the operation might deadlock on the AHB bus. Read bit 4 (PLL_LOCK) of the EPCTR register of AHB-EPC Bridge 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 de-asserted
- When the resume signal is de-asserted by using the remote wakeup feature. (See **Section 10.6.7.2(2), Power Up (Function Controller)** for details.)

10.6.1.2 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.

10.6.2 Accessing Host Controller Registers

The host controller's registers are accessed via the internal PCI bus. For the AHB bus to perform access properly, therefore, the AHB bus memory space must be correctly mapped to the PCI bus memory space in this subsystem. Note that the PCI has two memory spaces: the PCI configuration space for storing the settings for PCI bus transfer and the base address settings for the PCI memory space; and the PCI memory space for actually transferring data.

Accesses from AHB to the host controller and vice versa are performed via the window area in the AHB-PCI bridge. The relationship between the register areas and each window area is shown in the figure and table below.

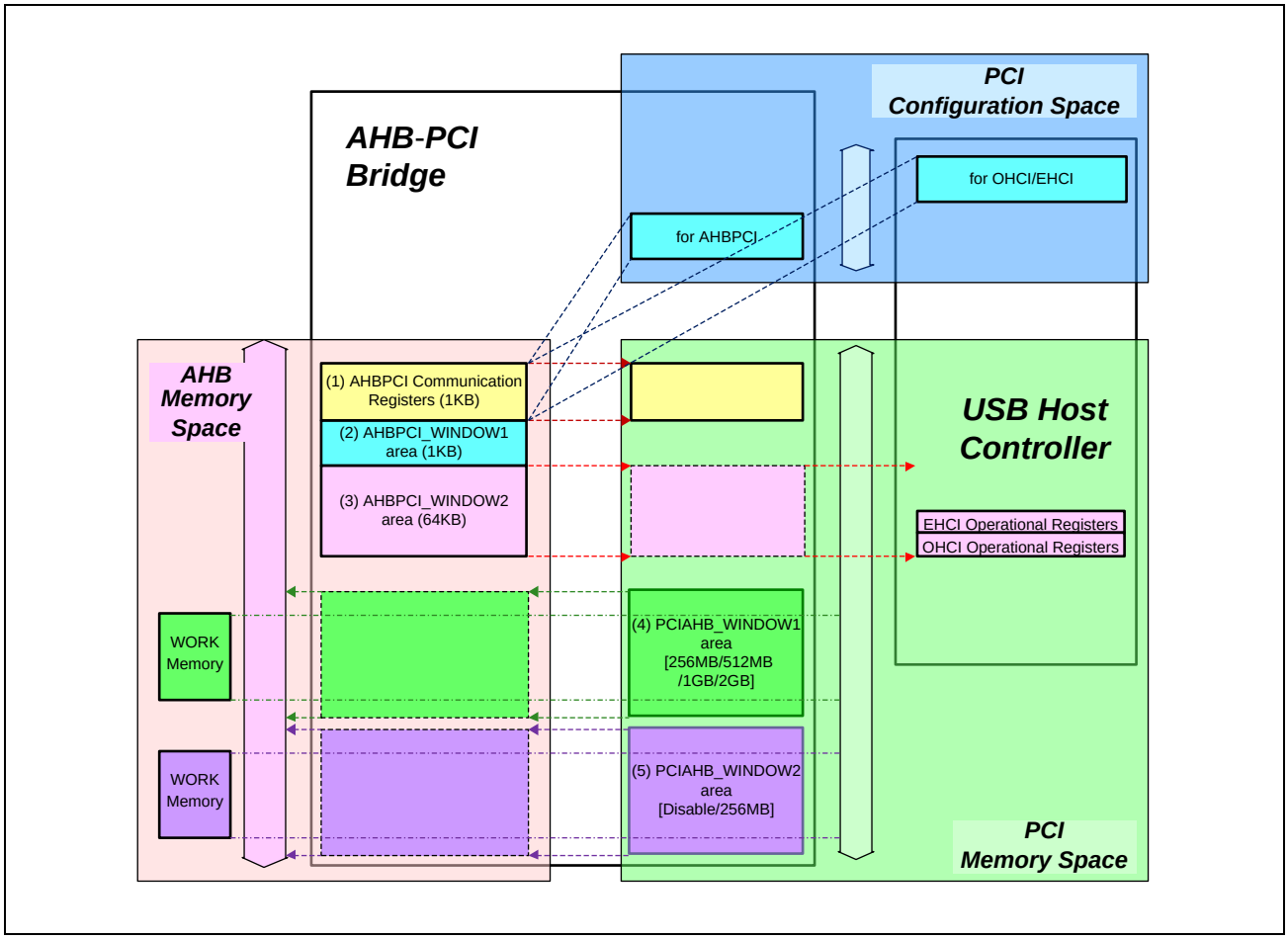


Figure 10.3 AHB and PCI Space Mapping Image

Table 10.120 Description of Each Area

AHB Area	Size	Description
(1) AHBPCI communication registers	1 KB	This area stores the registers for specifying the AHB settings and window area base address settings. This area is also mapped on the PCI memory space, so avoid overlapping other areas when mapping this area.
(2) AHBPCI_WINDOW1 area	1 KB	The PCI configuration registers are accessed via this area. Whether to access the OHCI/EHCI configuration registers or AHB-PCI bridge registers is specified by using the AHBPCI_WIN1_CTR register.
(3) AHBPCI_WINDOW2 area	64 KB	The OHCI/EHCI operational registers are accessed via this area.
(4) PCIAHB_WINDOW1 area	256 MB, 512 MB, 1 GB, 2 GB	The host controller accesses the work memory on the AHB bus via this area. The size of this area can be changed by using the USBCTR register.
(5) PCIAHB_WINDOW2 area	Disable, 256 MB	The host controller accesses the work memory on the AHB bus via this area. Whether to use this area or not can be specified by using the USBCTR register.

Specify these areas on the PCI memory space so that the area for the AHBPCI communication registers does not overlap the AHBPCI_WINDOW2 area (OHCI/EHCI operational register area), and the PCIAHB_WINDOW1 area does not overlap the PCIAHB_WINDOW2 area.

Although normally the simplest way to perform mapping is to map the AHB memory space to the same addresses as the PCI memory space, if the above areas will overlap on the AHB memory map, it is necessary to specify a base address that prevents overlapping with the PCIAHB_WINDOW1/2 areas by using a PCI configuration register (the OHCI/EHCI/AHBPCI base address register). This is illustrated in the figure below.

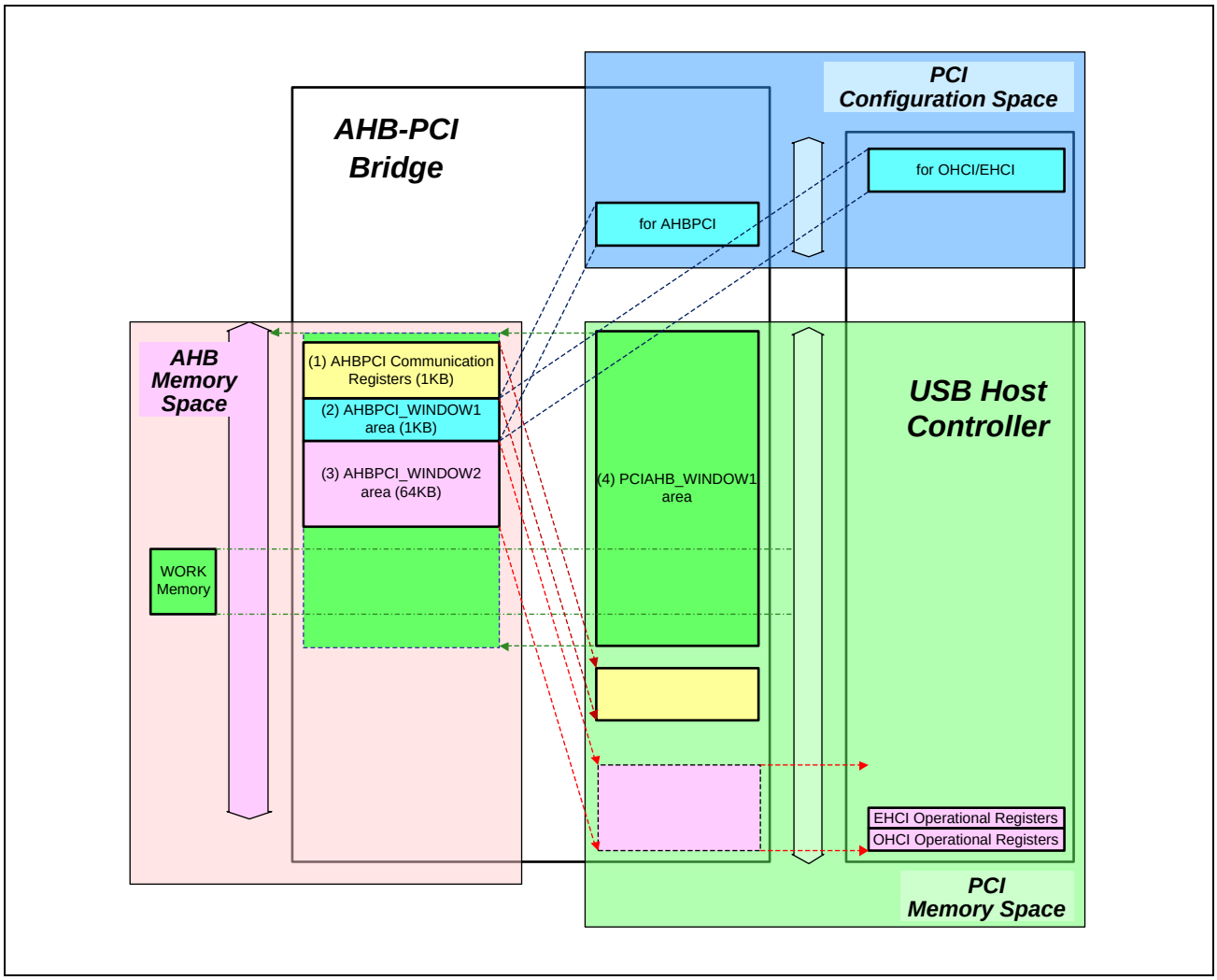


Figure 10.4 AHB and PCI Space Mapping Image (Areas Overlapping)

The relationship between the registers used to map the AHB and PCI spaces and the setting value is shown in the Figure and Table below.

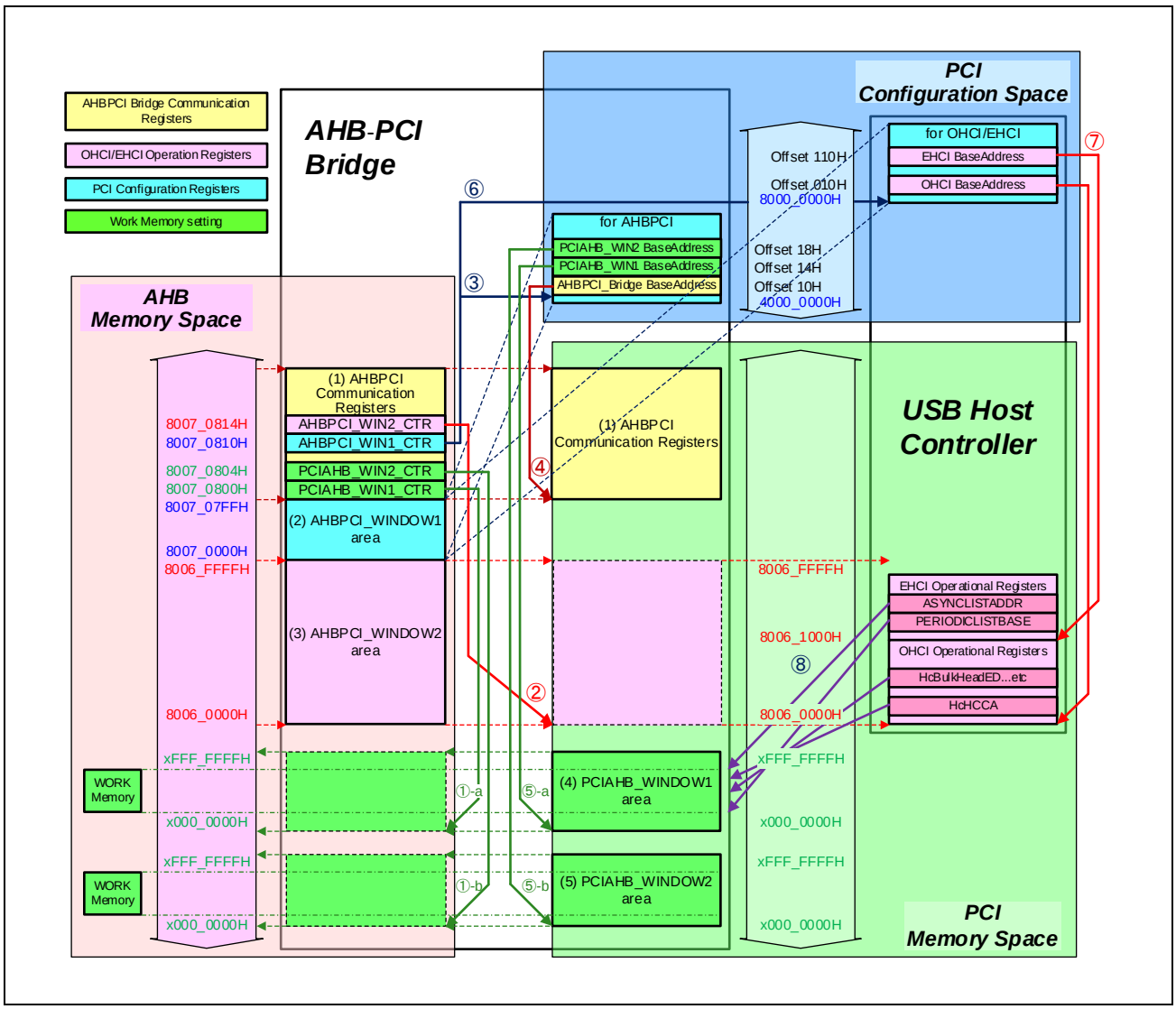


Figure 10.5 AHB and PCI Space Mapping Image (Areas Overlapping)

Table 10.121 Address Setting Register Values

Register		Description
(1)-a	PCIAHB_WIN1_CTR	When the host controller accesses the PCIAHB_WINDOW1 area as a bus master, the AHB bus address is translated to the base address specified in this register. Generally, specify the area including the work memory used.
(1)-b	PCIAHB_WIN2_CTR	When the host controller accesses the PCIAHB_WINDOW2 area as a bus master, the AHB bus address is translated to the base address specified in this register. Generally, specify the area including the work memory used.
(2)	AHBPCI_WIN2_CTR	When the AHBPCI_WINDOW2 area is accessed, the PCI bus address is translated to the base address specified in this register. Generally, specify the same value as that specified for the AHBPCI_WINDOW2 area. However, make sure that the specified area does not overlap the area including the work memory, which is specified in (1).
(3)	AHBPCI_WIN1_CTR	If the base address is specified as 4000_0000h in this register, PCI configuration registers for AHBPCI are accessed.
(4)	AHBPCI Bridge base address	Specify the base address of the AHB-PCI bridge in a PCI space. This register is not accessed via the PCI bus, but make sure that the area does not overlap other areas.
(5)-a	PCIAHB_WIN1 base address	Specify the base address of the PCIAHB_WINDOW1 area in a PCI space. Generally, specify the same address as that specified in (1)-a. Make sure that the areas specified by (5)-a and (5)-b do not overlap.
(5)-b	PCIAHB_WIN2 base address	Specify the base address of the PCIAHB_WINDOW2 area in a PCI space. Generally, specify the same address as that specified in (1)-a. Make sure that the areas specified by (5)-a and (5)-b do not overlap.
(6)	AHBPCI_WIN1_CTR	If the base address is specified as 8000_0000h in this register, PCI configuration registers for OHCI/EHCI are accessed.
(7)	OHCI /EHCI base address	Specify the base address of the OHCI/EHCI operational registers in a PCI space. Generally, specify the same address as that specified in (2) for the OHCI operational registers. For EHCI operational registers, specify the address of {address specified in (2) + offset 1000h}.
(8)	OHCI/EHCI operational registers	After the settings of (1) to (7) are specified, the host controller can access data copied to the AHB work RAM, such as descriptors, via the PCI bus. The following registers are used to specify the work RAM address at which data exists. • OHCI/EHCI Operational Register – HcHCCA register – HcPeriodicCurrentED register – HcControlHeadED register – HcControlCurrentED register – HcBulkHeadED register – HcBulkCurrentED register – HcDoneHead register • EHCI Operational Register – PERIODICLISTBASE register – ASYNCLISTADDR register

10.6.2.1 Accessing PCI Configuration Registers

PCI configuration space registers are accessed via the AHB-PCI Window 1 area (host controller addresses 10000h to 107FFh: 2 KB space). At this time, be sure to specify appropriate settings for the AHBPCI_WIN1_CTR register. The following table shows the AHBPCI_WIN1_CTR register setting used to access each PCI configuration space on the OHCI/EHCI and AHB-PCI bridges.

Table 10.122 AHBPCI_WIN1_CTR Register Settings

	AHBPCI_WIN1_CTR Register Setting	
Area to Access	PCIWIN1_BASEADR[31:11]	PCICMD[2:0]
OHCI/EHCI PCI configuration space	Set only bit 31 to 1b.	101b
AHB-PCI bridge PCI configuration space	Set only bit 30 to 1b	

10.6.2.2 Accessing OHCI/EHCI Operational Registers

To access OHCI/EHCI operational registers, OHCI/EHCI PCI configuration space and AHBPCI_WIN2_CTR register settings are required, as is the setting of PCI space address mapping. The following shows the required settings.

Table 10.123 Settings Required for Accessing OHCI/EHCI Operational Registers

Setting bits	Setting
Bit 1 (MEMORY_SPACE) of OHCI/EHCI PCI Configuration Space, Offset 04h (Status - Command)	1b (Enable accessing to memory space)
Bits [3:1] (PCICMD[2:0]) of AHBPCI_WIN2_CTR register	011b (Memory read/write)

10.6.3 Reset Control

10.6.3.1 Reset Configuration

USB subsystem is reset by the HRESETn signal. When the HRESETn signal is asserted, all the circuits within the USB subsystem are reset. The reset signal of USB subsystem is an asynchronous reset that is directly connected to the F/F reset signal.

The reset is divided between USB host controller and USB function controller.

The reset signals of USB subsystem are shown in Table below.

Table 10.124 Reset Signals

Reset Signal	Supplied From	Description
HRESETn	System Control	Power-on Reset Signal of the USB subsystem. Resets the entire USB subsystem. HRESETn is controlled from System Control as software reset.
PLL_RST	Internal	USBPHY Reset signal. The USBPLL is shared between the USB Host and Function Controllers. Do not assert this signal when either USB Host or Function Controller is active.
EPC_RST	Internal	USB Function Controller (EPC/SIE) Reset Signal
USBH_RST	Internal	USB Host Controller Reset Signal

When the HRESETn is asserted, the all internal reset signals are also asserted. The internal USB host controller reset can be operated from the AHB-PCI Bridge register. The internal USB function controller reset can be operated from the AHB-EPC Bridge register. After the HRESETn de-assertion, use the same registers to cancel the reset state. For the reset sequence, refer to **Section 10.6.9.1, Reset Sequence**.

10.6.3.2 Reset System Diagram

The reset system in USB subsystem is shown below.

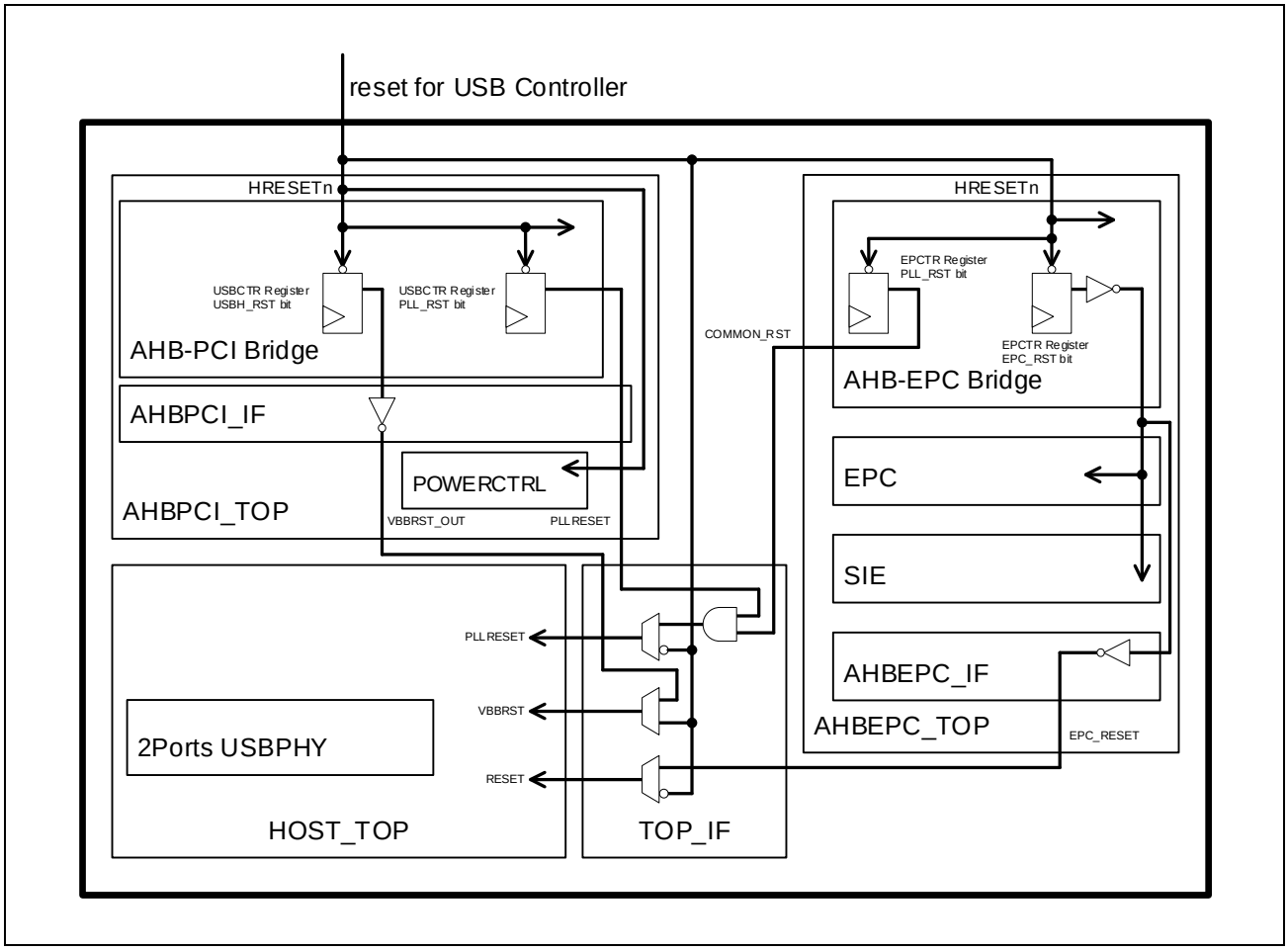


Figure 10.6 Reset System Diagram

10.6.4 Interrupts

10.6.4.1 Interrupt Control Registers

(1) U2H_INT Control Register

This is an interrupt generated by the AHB-PCI bridge. Use the AHB-PCI Bridge registers to check the status of, clear, and enable the interrupts.

Table 10.125 U2H_INT Control Register

Control	Target Registers
Checking the status of and clearing the interrupts	PCI_INT_STATUS register
Enabling the interrupts	PCI_INT_ENABLE register

(2) U2H_OHCI_INT Control Register

This is an INTA interrupt signal generated by the host controller. Interrupts are controlled basically by using host controller registers, but to assert an interrupt signal, the interrupt must be enabled in the relevant AHB-PCI Bridge register.

Table 10.126 U2H_OHCI_INT Control Register

Control	Target Registers
Checking the status of and clearing the interrupts	HcInterruptStatus register
Enabling the interrupts	HcInterruptEnable register HcInterruptDisable register PCI_INT_ENABLE register (bit 16 (USBH_INTAEN))

(3) U2H_EHCI_INT Control Register

This is an INTB interrupt signal generated by the host controller. Interrupts are controlled basically by using host controller registers, but to assert an interrupt signal, the interrupt must be enabled in the relevant AHB-PCI Bridge register.

Table 10.127 U2H_EHCI_INT Control Register

Control	Target Registers
Checking the status of and clearing the interrupts	USBSTS register
Enabling the interrupts	USBINTR register PCI_INT_ENABLE register (bit 17 USBH_INTBEN)

(4) U2H_PME_INT Control Register

This is a PME interrupt signal generated by the host controller. Interrupts are controlled basically by using host controller registers, but to assert an interrupt signal, the interrupt must be enabled in the relevant AHB-PCI Bridge register.

Table 10.128 U2H_PME_INT Control Register

Control	Target Registers
Checking the status of and clearing the interrupts	PCI Configuration register for OHCI/EHCI offset 44h
Enabling the interrupts	PCI Configuration register for OHCI/EHCI offset 44h PCI_INT_ENABLE register (bit 19 USBH_PMEEN)

(5) U2F_INT 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 10.129 U2F_INT Control Register

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

(6) U2F_EPC_INT 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 10.130 U2F_EPC_INT Control Register

Control	Target Registers
Checking the status of and clearing the interrupts	USB Interrupt Status register EP0/EP[m] Status register
Enabling the interrupts	USB Interrupt Enable register EP0/EP[m] Interrupt Enable register

10.6.4.2 Interrupt Signal Overview

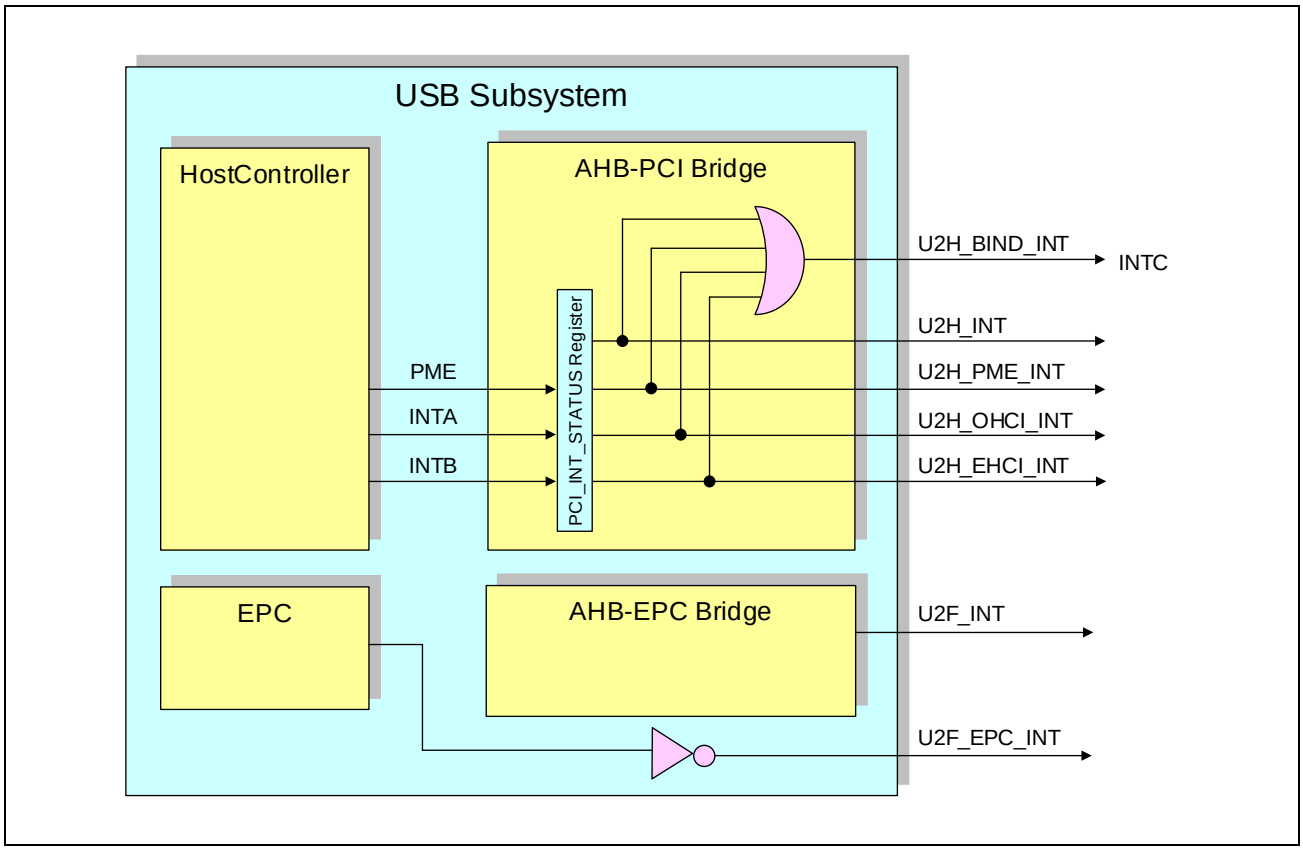


Figure 10.7 Interrupt Signal Overview

10.6.4.3 Interrupt Signal Clear Time

The AHB-PCI Bridge implements a posted write, therefore, an interrupt generated by the host controller or function controller may not be cleared soon after setting the clear register and same interrupt state may be recognized many times.

The user must take a countermeasure to avoid such false recognition.

For example, access to the host controller or function controller register (b) after accessing to an Interrupt Clear Register (a). The operation of (b) is waited until the operation of (a) is completed. As the result, the interrupt is cleared without fail at the completion of the access of (b).

(1) Host Controller

The U2H_OHCI_INT, U2H_EHCI_INT, and U2H_PME_INT may not be cleared soon after setting the clear register.
When USB_HCLKH = 125 MHz and USB_PCICLK = 25 MHz, the interrupt is typically cleared in about 360 ns after setting the clear register.
When the host controller is executing a transfer as a master on the internal PCI-bus, it takes about 1.6 us (36 CLK@USB_PCICLK + 3 CLK@USB_HCLKH + 2 CLK@12 MHz) at worst to clear the interrupt.

(2) Function Controller

When USB_HCLKF = 125 MHz, the interrupt is typically cleared in about 180 ns after setting the clear register. When the function controller is executing a DMA transfer between AHB-EPC and EPC, it takes about 610 ns (35 CLK@60 MHz + 3 CLK@USB_HCLKF) at worst to clear the interrupt.

10.6.5 Overcurrent Control and VBUS Control

10.6.5.1 Overcurrent Control

This section describes the behavior of the USB_OC and USB_PPON signals used for controlling the external circuit that is used for USB port overcurrent detection and VBUS control.

(1) Meanings of USB_OC and USB_PPON Signals

The meanings of USB_OC and USB_PPON signals are shown in Table below.

The USB_OC2 and USB_PPON2 signals perform overcurrent control of the port 2 host only when this subsystem is used as USB interface port 1 host/port 2 host. When used as USB interface port 1 function/port 2 host, because the number of ports used for the host is one only, overcurrent of the port 2 host is controlled by USB_OC1 and USB_PPON1 signals.

Table 10.131 USB_OC and USB_PPON

USB Interface	Pin	I/O	Level	Meaning
Port 1 function/port 2 host	USB_OC1	Input	1	Port 2 overcurrent not detected.
			0	Port 2 overcurrent detected
	USB_OC2	Input	PU	
	USB_PPON1	Output	1	Power supply to port 2 VBUS ON
			0	Power supply to port 2 VBUS OFF
	USB_PPON2	Output	OPEN	
Port 1 host/port 2 host	USB_OC1	Input	1	Port 1 overcurrent not detected
			0	Port 1 overcurrent detected
	USB_OC2	Input	1	Port 2 overcurrent not detected
			0	Port 2 overcurrent detected
	USB_PPON1	Output	1	Power supply to port 1 VBUS ON
			0	Power supply to port 1 VBUS OFF
	USB_PPON2	Output	1	Power supply to port 2 VBUS ON
			0	Power supply to port 2 VBUS OFF

(2) PPON Output Signal Assert/De-assert Conditions

A timing chart showing the conditions under which USB_OC1 and USB_PPON1 are asserted and de-asserted is shown in Figure below.

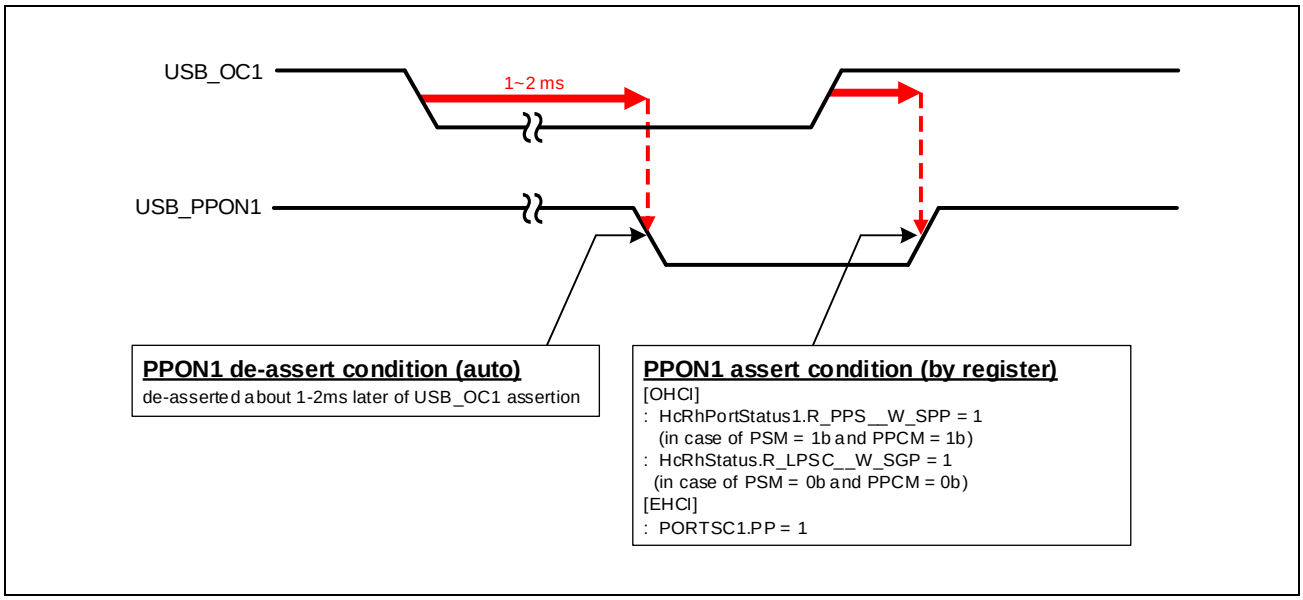


Figure 10.8 USB_OC1 and USB_PPON1 Assert, De-assert Timing Chart

The same timing chart applies to when USB_OC2 and USB_PPON2 are asserted and de-asserted.

The figure above shows a timing chart for USB interface port 1. When using port 2, specify the same settings for the following port 2 registers:

- HcRhPortStatus2 register
- PORTSC2 register

Even if USB_OC1/OC2 is de-asserted, USB_PPON1/PPON2 is not asserted automatically. USB_PPON1/PPON2 is asserted when the software sets the port power bit after USB_OC1/OC2 is de-asserted.

10.6.5.2 VBUS Control

Depending on the connections with peripheral circuits, it might be possible to stop VBUS to reduce the power consumption when the USB port is not used by connecting the USB_OC1/PPON1 and USB_OC2/PPON2 pins to a high-side switch. The relationship between USB_PPON1/PPON2 and VBUS is as follows:

Table 10.132 Relationship between PPON1/PPON2 and VBUS

USB Interface	PPON	VBUS
Port 1 function/port 2 host	USB_PPON1 = 0b	Port 2 VBUS stops
	USB_PPON1 = 1b	Port 2 VBUS operates
Port 1 host/port 2 host	USB_PPON1 = 0b	Port 1 VBUS stops
	USB_PPON1 = 1b	Port 1 VBUS operates
	USB_PPON2 = 0b	Port 2 VBUS stops
	USB_PPON2 = 1b	Port 2 VBUS operates

The behavior of USB_PPON1/PPON2 when USB_OC1/OC2 is asserted differs depending on the PCI configuration register and OHCI operational register settings. The relationship between the register settings and USB_PPON1/PPON2 is shown below.

Table 10.133 Relationship between Register Settings and USB_PPON1/PPON2

PCI Configuration Register	OHCI Operational Register					PPON	
EXT1 Register	HcRhDescriptorA Register			HcRhDescriptorB Register			
				PPCM		Output Pin Behavior	
PPCNT Bit	NOCP Bit	NPS Bit	PSM Bit	Bit 1	Bit 0	USB_PPON2	USB_PPON1
0	—	—	—	—	—	Fixed to 1b	
—	1	—	—	—	—	Fixed to 1b	
—	—	1	—	—	—	Fixed to 1b	
1	0	0	0	—	—	USB_PPON1 and USB_PPON2 are de-asserted (1b to 0b) by asserting USB_OC1 or USB_OC2 to 0b.	
			1	0	0	USB_PPON1 and USB_PPON2 are de-asserted (1b to 0b) by asserting USB_OC1 or USB_OC2 to 0b.	
				0	1	USB_PPON2 is de-asserted (1b to 0b) by asserting USB_OC1 or USB_OC2 to 0b.	USB_PPON1 is de-asserted (1b to 0b) by asserting USB_OC1 to 0b.
				1	0	USB_PPON2 is de-asserted (1b to 0b) by asserting USB_OC2 to 0b.	USB_PPON1 is de-asserted (1b to 0b) by asserting USB_OC1 or USB_OC2 to 0b.
				1	1	USB_PPON2 is de-asserted (1b to 0b) by asserting USB_OC2 to 0b.	USB_PPON1 is de-asserted (1b to 0b) by asserting USB_OC1 to 0b.

Note: If NPS = 1b, the host controller detects an overcurrent. But it does not de-assert USB_PPON1/PPON2.

10.6.5.3 Overcurrent Detection Using PPON

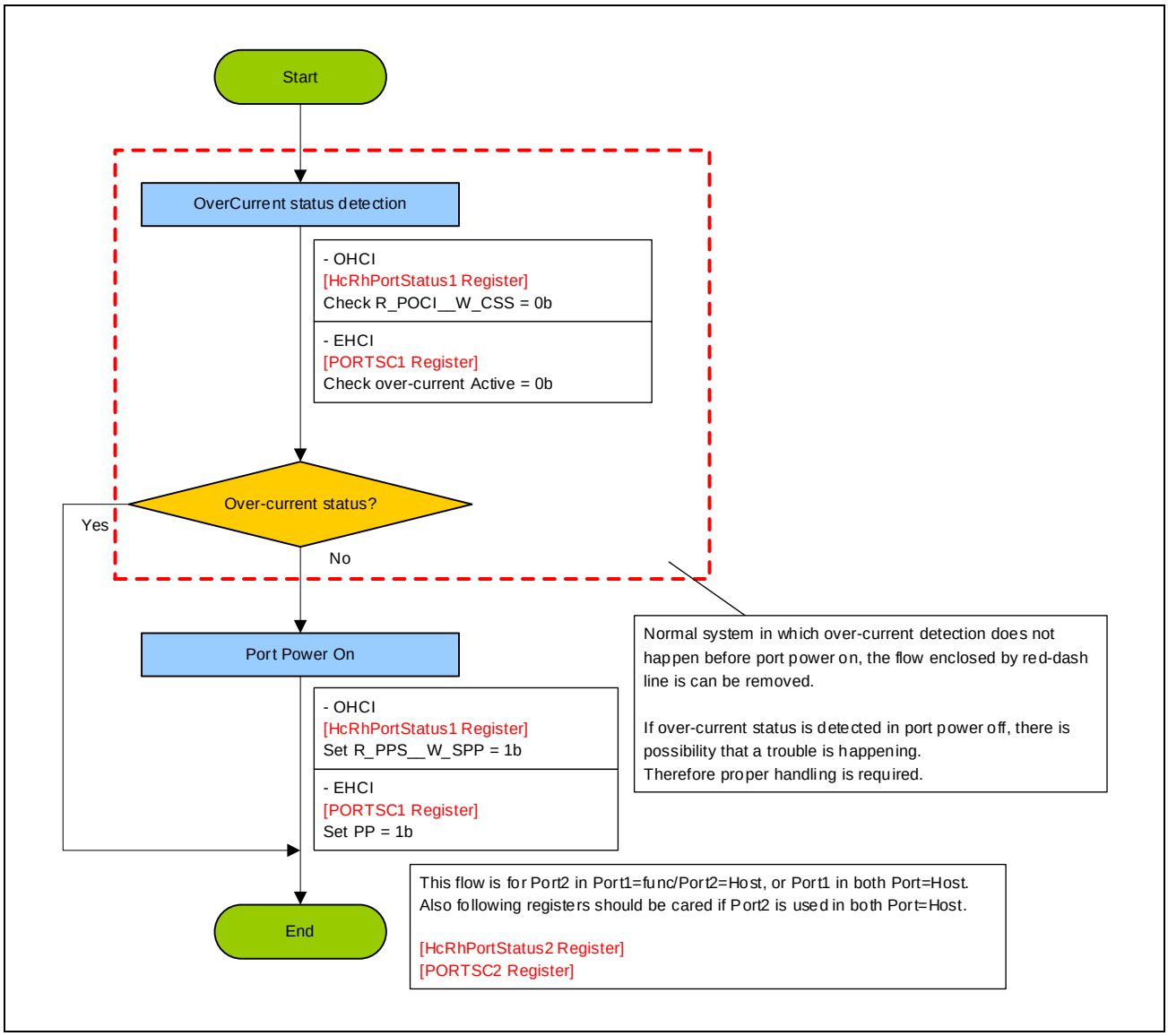


Figure 10.9 Overcurrent Detection Using PPON

10.6.5.4 PPON Setting Flow

Shown below is the PPON setting flow in a system where OCI might be active (0b) at system startup.

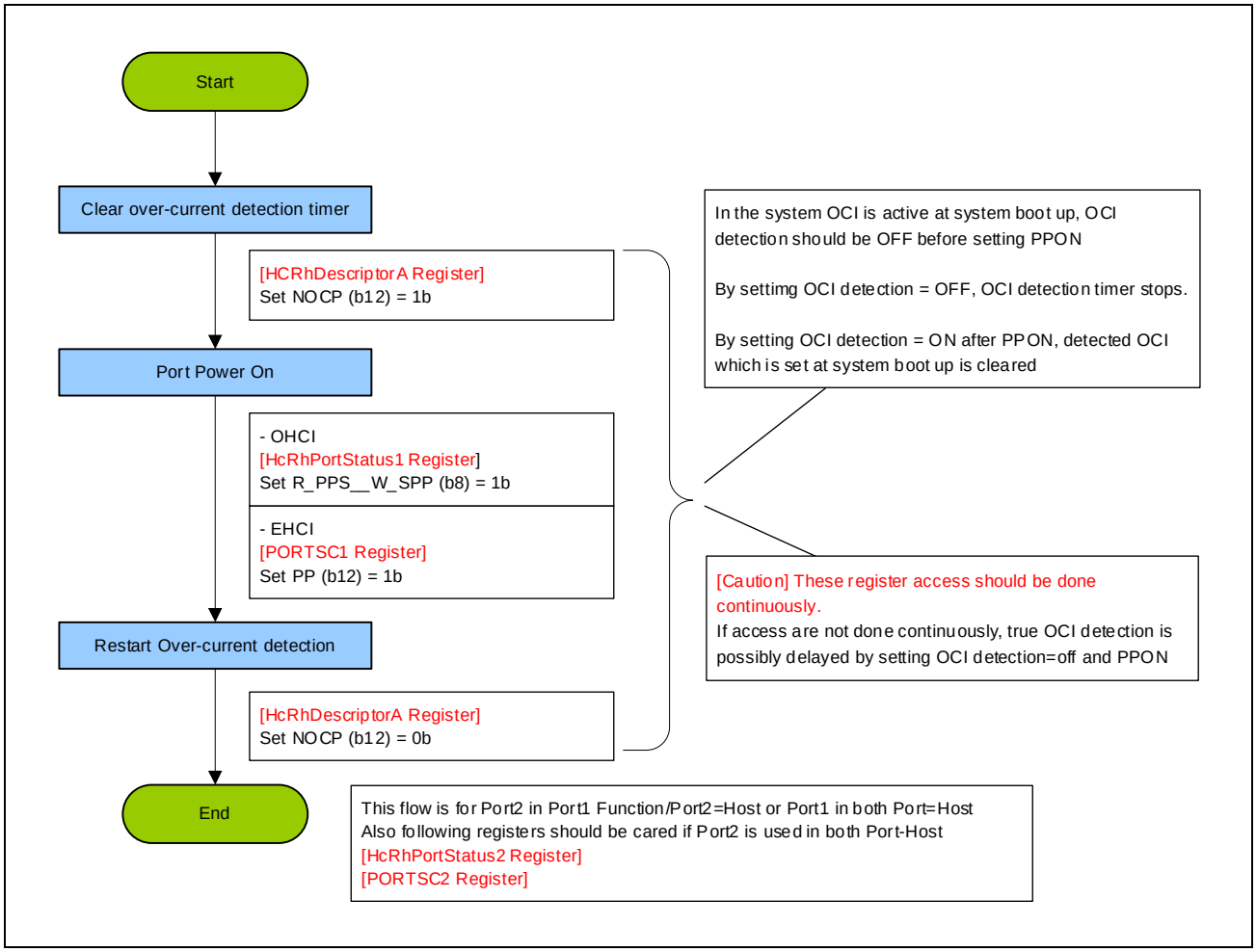


Figure 10.10 PPON Setting Flow

10.6.6 VBUS Detection

10.6.6.1 External Circuit for VBUS Detection

An external circuit is required to detect VBUS level in USB Function port.

It should convert 5 V to 3 V, prevent chattering and input to USB_VBUS pin.

10.6.6.2 VBUS Detection Part

VBUS detection is performed by a register in AHB-EPC Bridge. With the EPCTR register bit8 VBUS_LEVEL, the status of the USB_VBUS signal can be checked. Also, when the status of the USB_VBUS signal is changed, AHBBINT register bit 13 VBUS_INT interrupt can be generated.

In addition, since the VBUS detection circuit in AHB-EPC Bridge is synchronized with the USB_HCLKPM signal, it is possible to generate an interrupt even when clock supply to the AHB clock (USB_HCLKF) or the function controller is stopped.

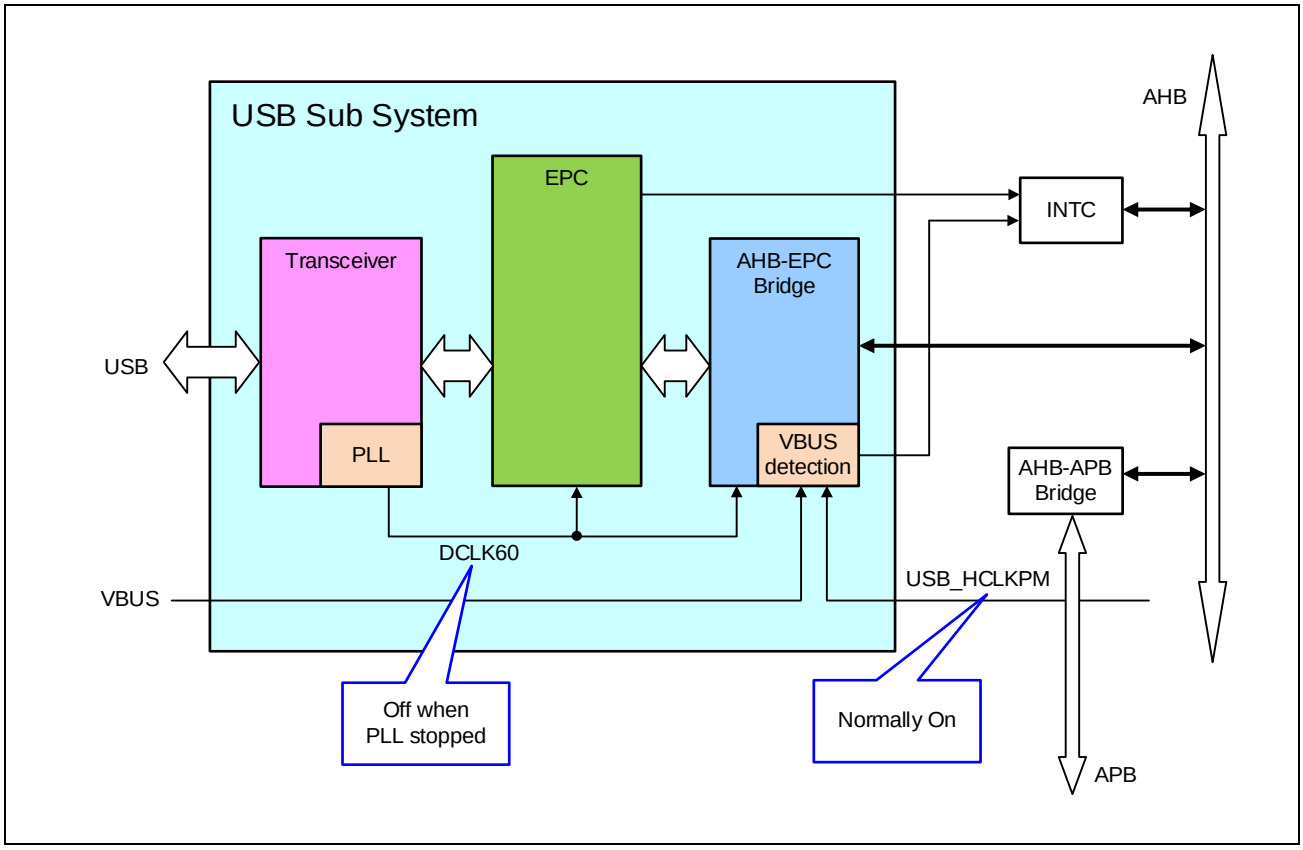


Figure 10.11 VBUS Detection Part Image

10.6.6.3 VBUS Detection Flow

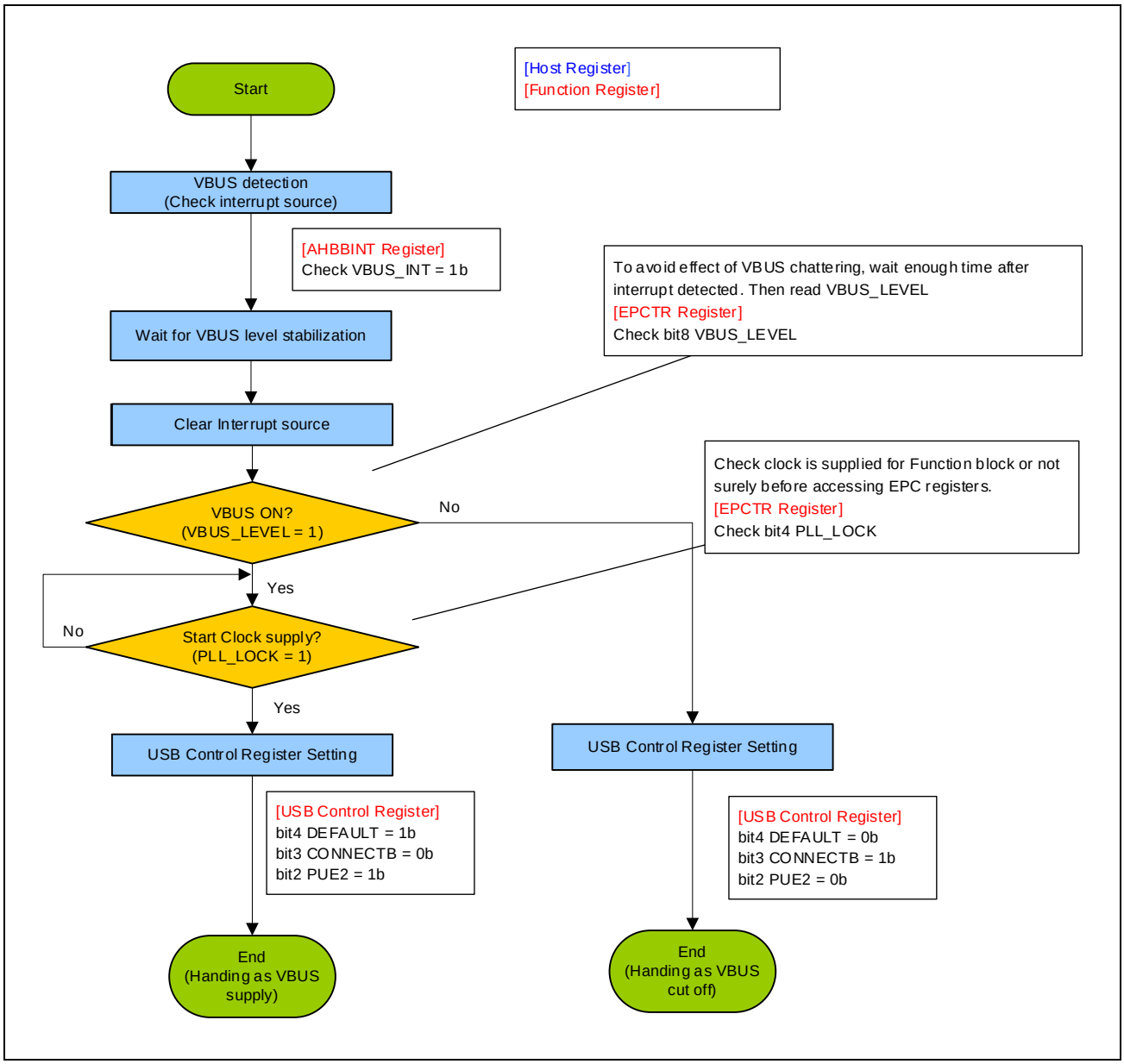


Figure 10.12 VBUS Detection Flow

CAUTIONS

1. If USB is Disconnected in the SUSPEND state (in which clock supply to the function controller is stopped), reset the function controller by the EPCTR Register bit0 EPC_RST bit.
2. The chattering time may be maximum 100 ms or so when an external circuit is not installed.
   Even when an external circuit is installed, time to be taken until USB_VBUS becomes steady differs depending on the customer's system. So, we recommend that the chattering time be evaluated on the customer's system.

10.6.7 Power Management

This chapter describes power management in the host and function controllers and the direct power-down (PLL stop) feature used when a USB port event does not need to be detected.

10.6.7.1 Power Management in the Host Controller

When host operations are paused, the power consumption can be reduced by powering down the host controller and stopping the PCI clock (USB_PCICLK) in the subsystem. The subsystem contains a USB_PCICLK mask circuit that can be used to control the PCI clock by using the registers in the AHB-PCI bridge.

(1) Power Down (Host Controller)

An example of the procedure used to power down the host controller is shown below.

(a) When using OHCI

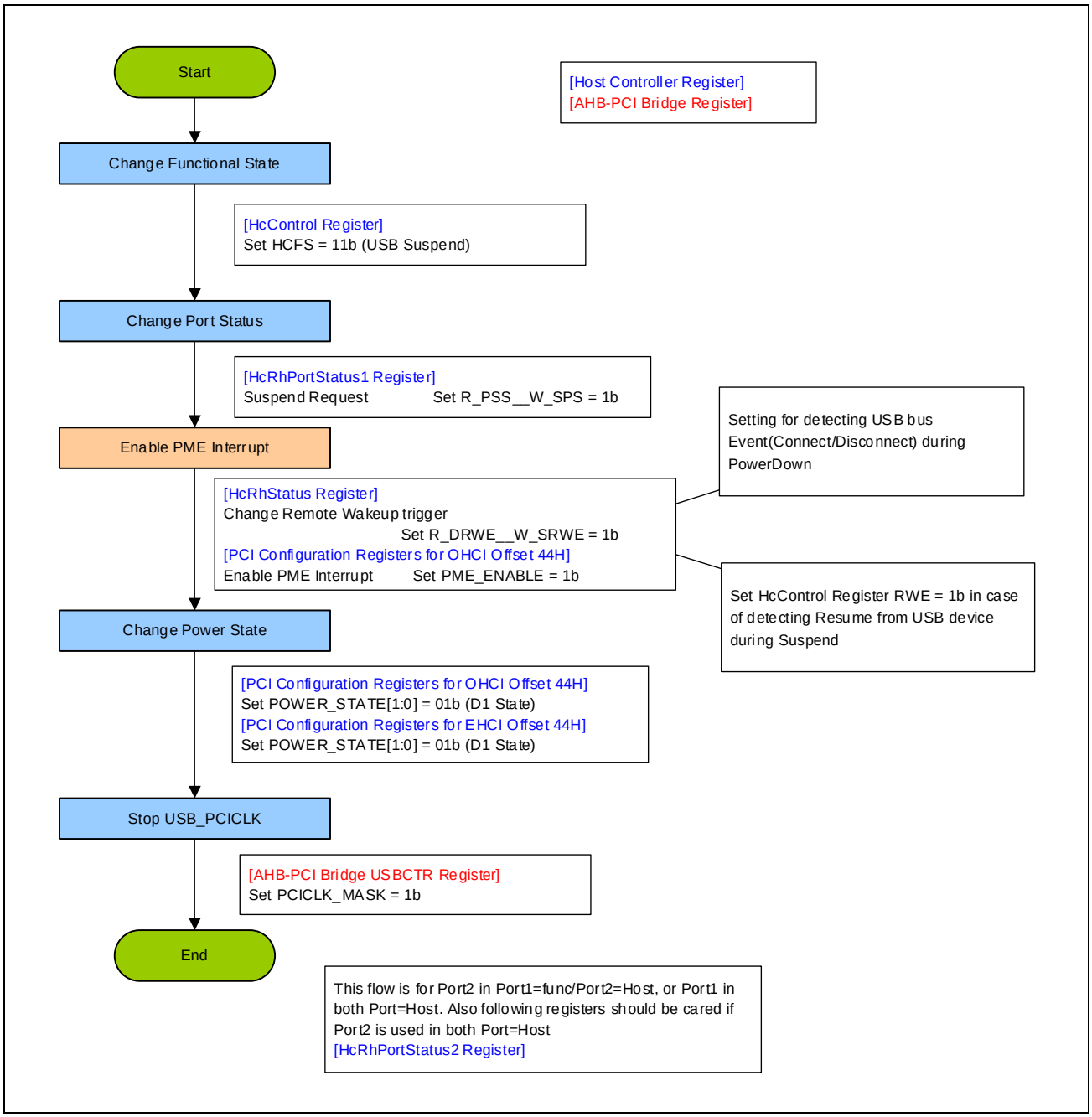


Figure 10.13 Host Controller Power-Down Flow (When Using OHCI)

(b) When using EHCI

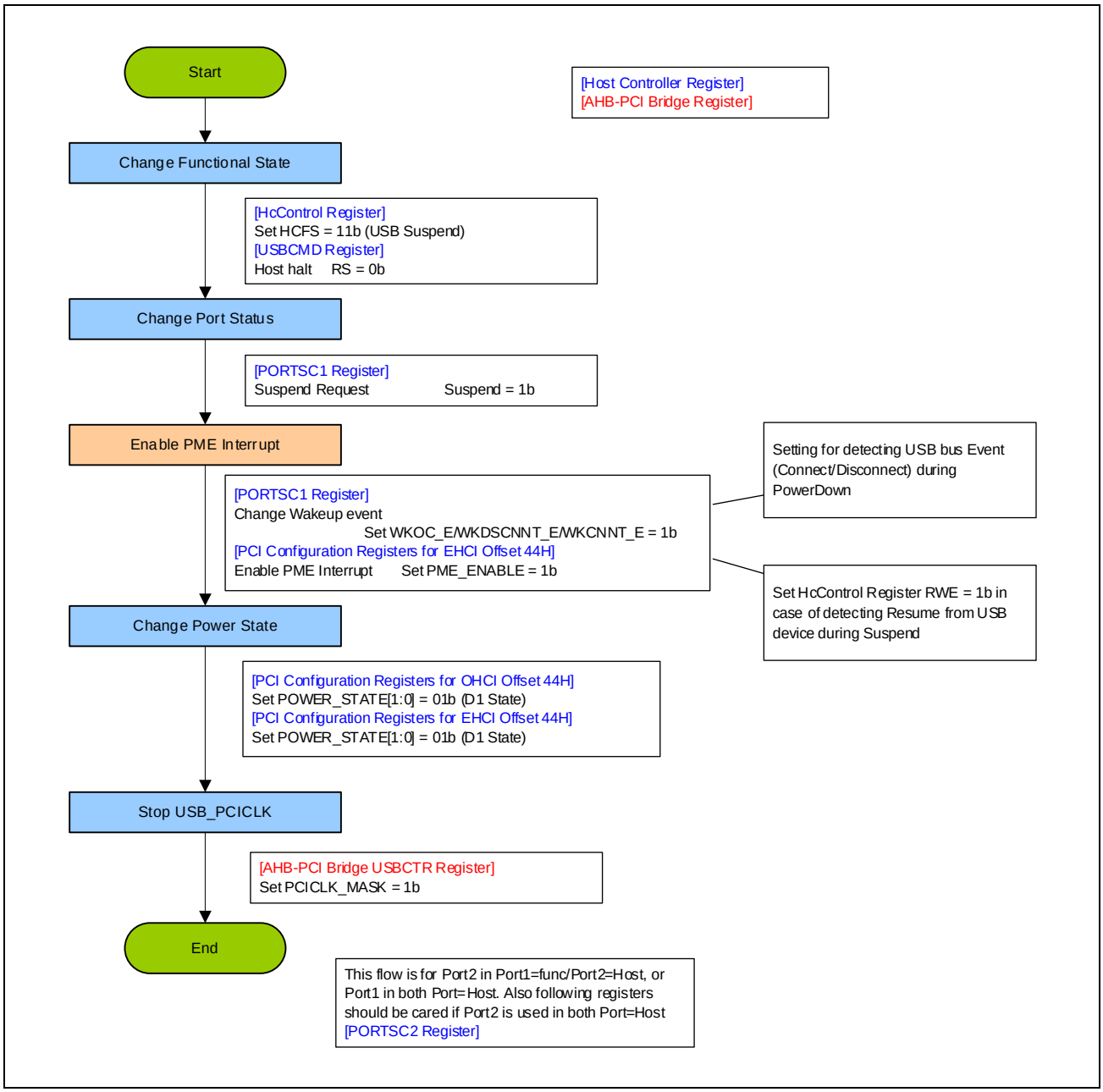


Figure 10.14 Host Controller Power-Down Flow (When Using EHCI)

(2) Power Up (Host Controller)

An example of the procedure used to power up the host controller is shown below. When restoring power to the host controller after it has been powered down, set the registers using the reverse procedure to the power down procedure.

(a) When using OHCI

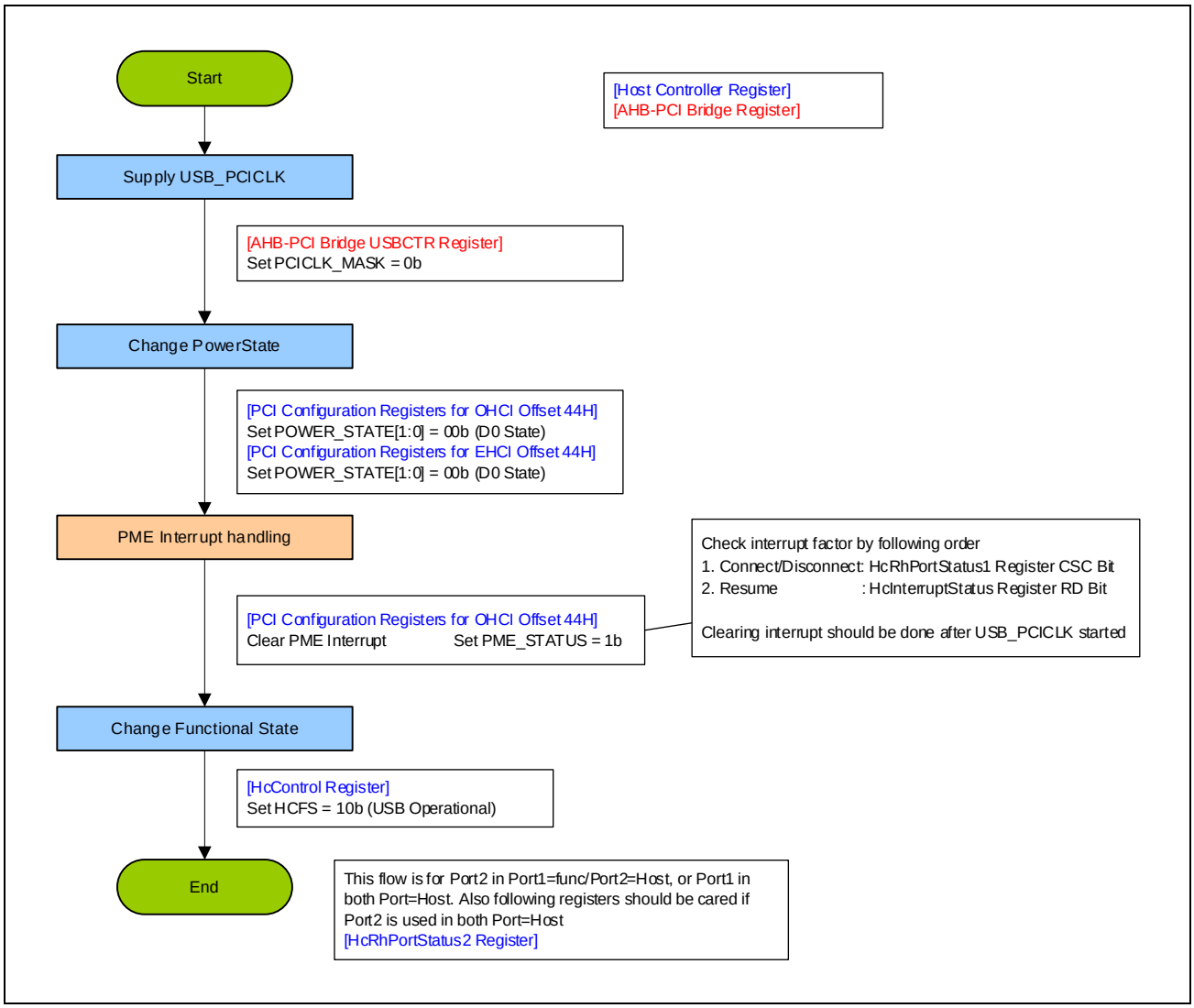


Figure 10.15 Host Controller Power-Up Flow (When Using OHCI)

(b) When using EHCI

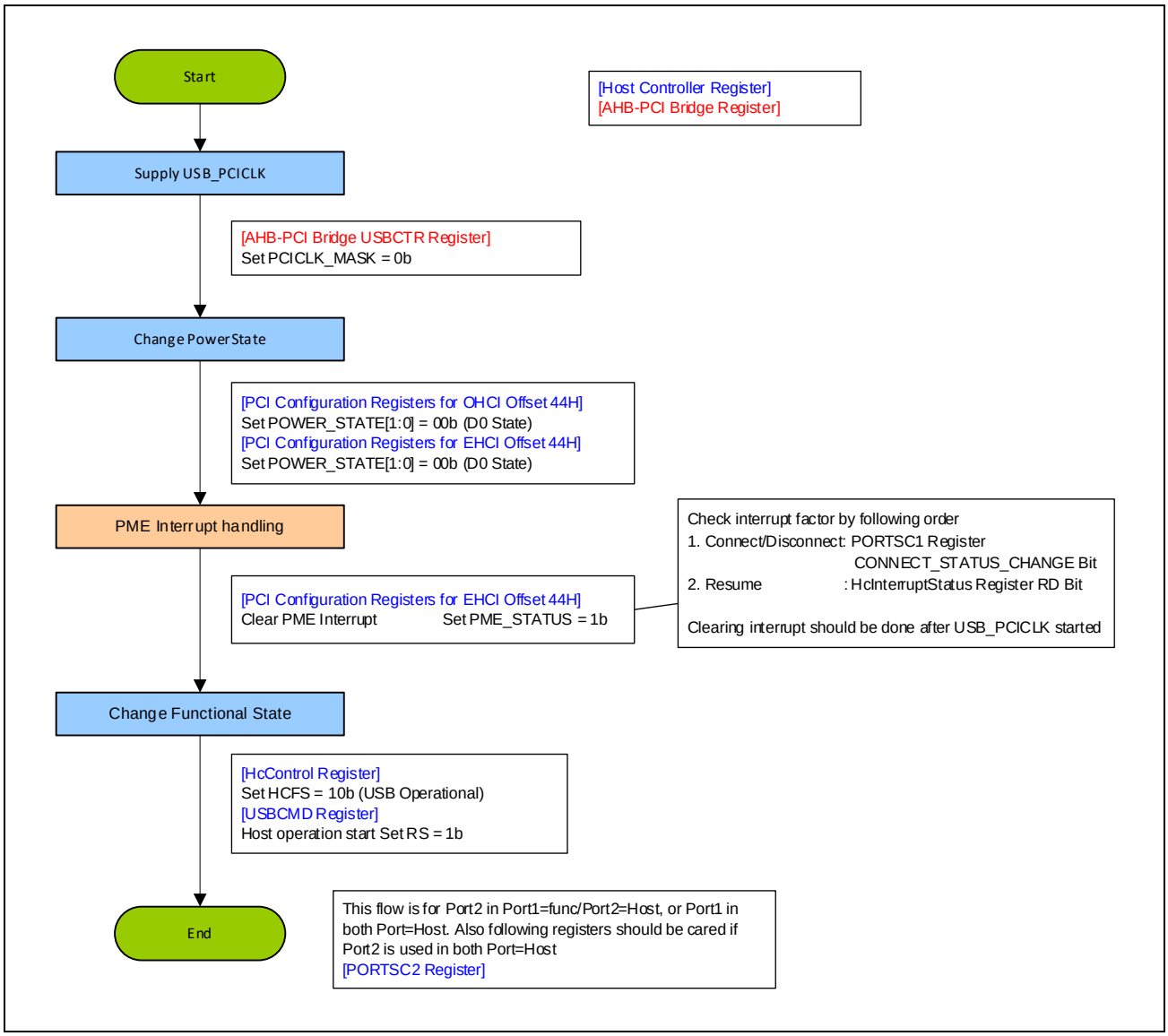


Figure 10.16 Host Controller Power-Up Flow (When Using EHCI)

NOTE

When power-up, please set PERIODIC_SCHEDULE_ENABLE bit of the USBCMD register to “1”

10.6.7.2 Power Management in the Function Controller

(1) Power Down (Function Controller)

An example of the procedure used to power down the function controller is shown below.

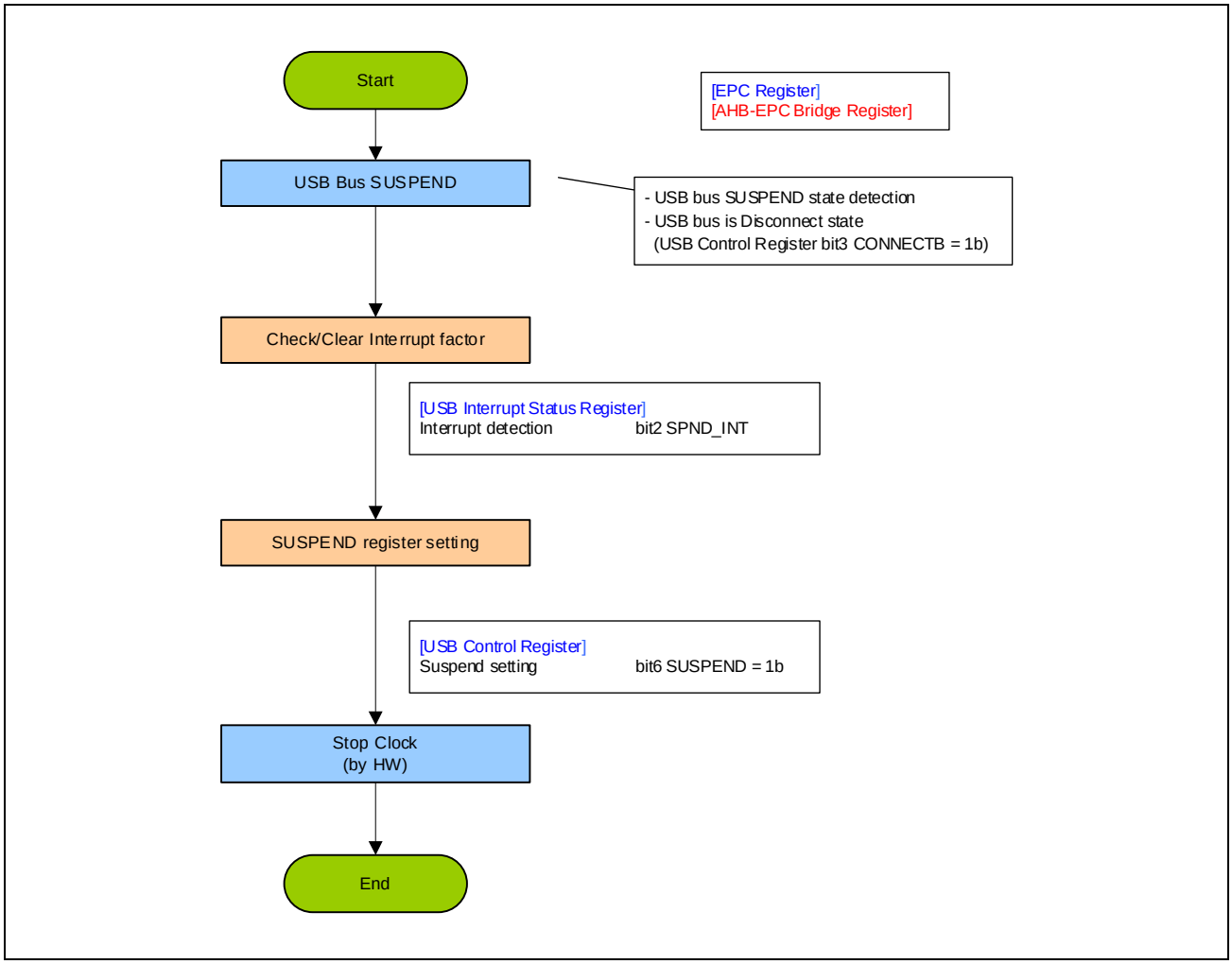


Figure 10.17 Function Controller Power-Down Flow

(2) Power Up (Function Controller)

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

(a) RESUME

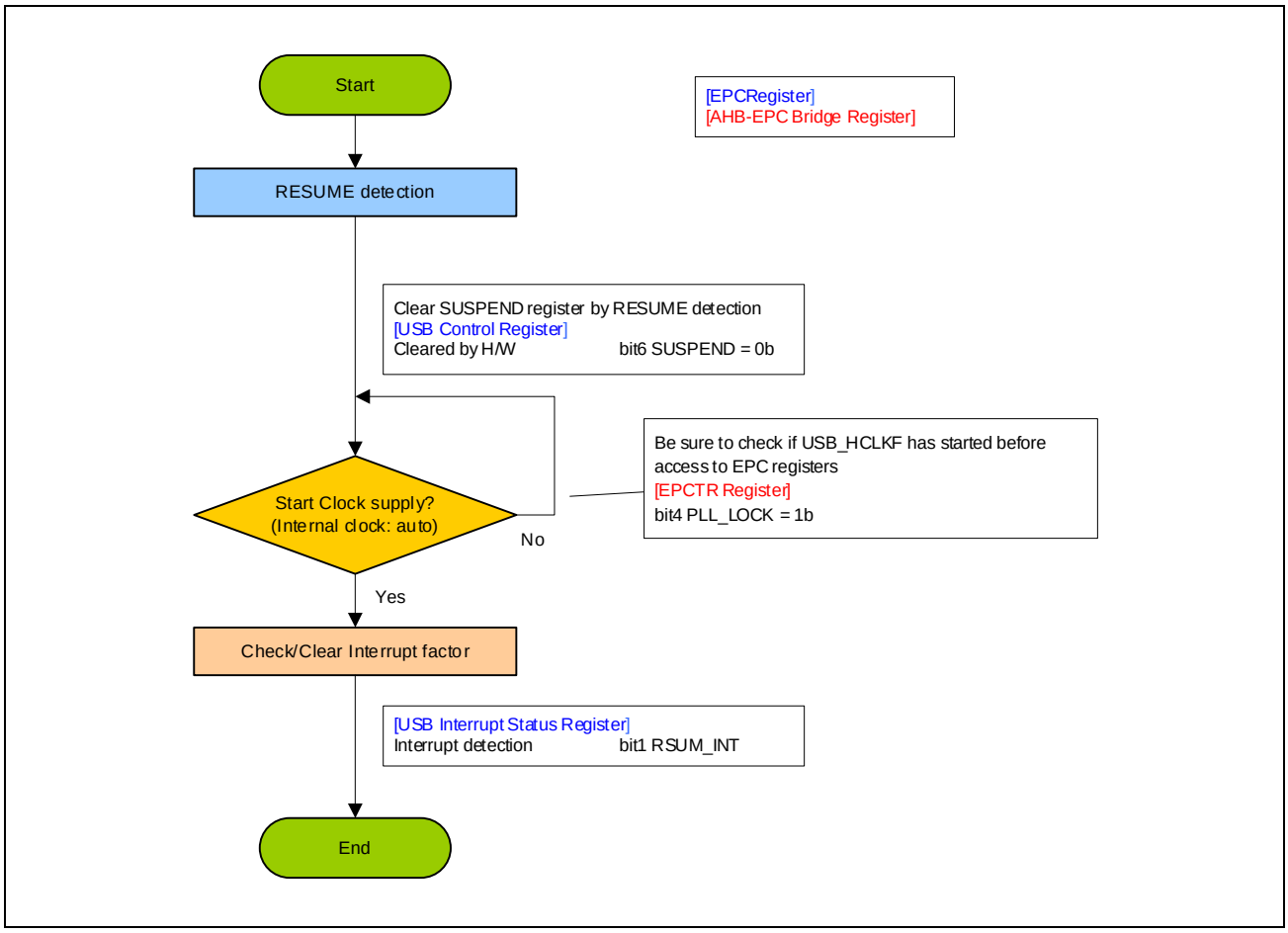


Figure 10.18 Power Up by Detecting the Resume Signal

(b) Remote WakeUp

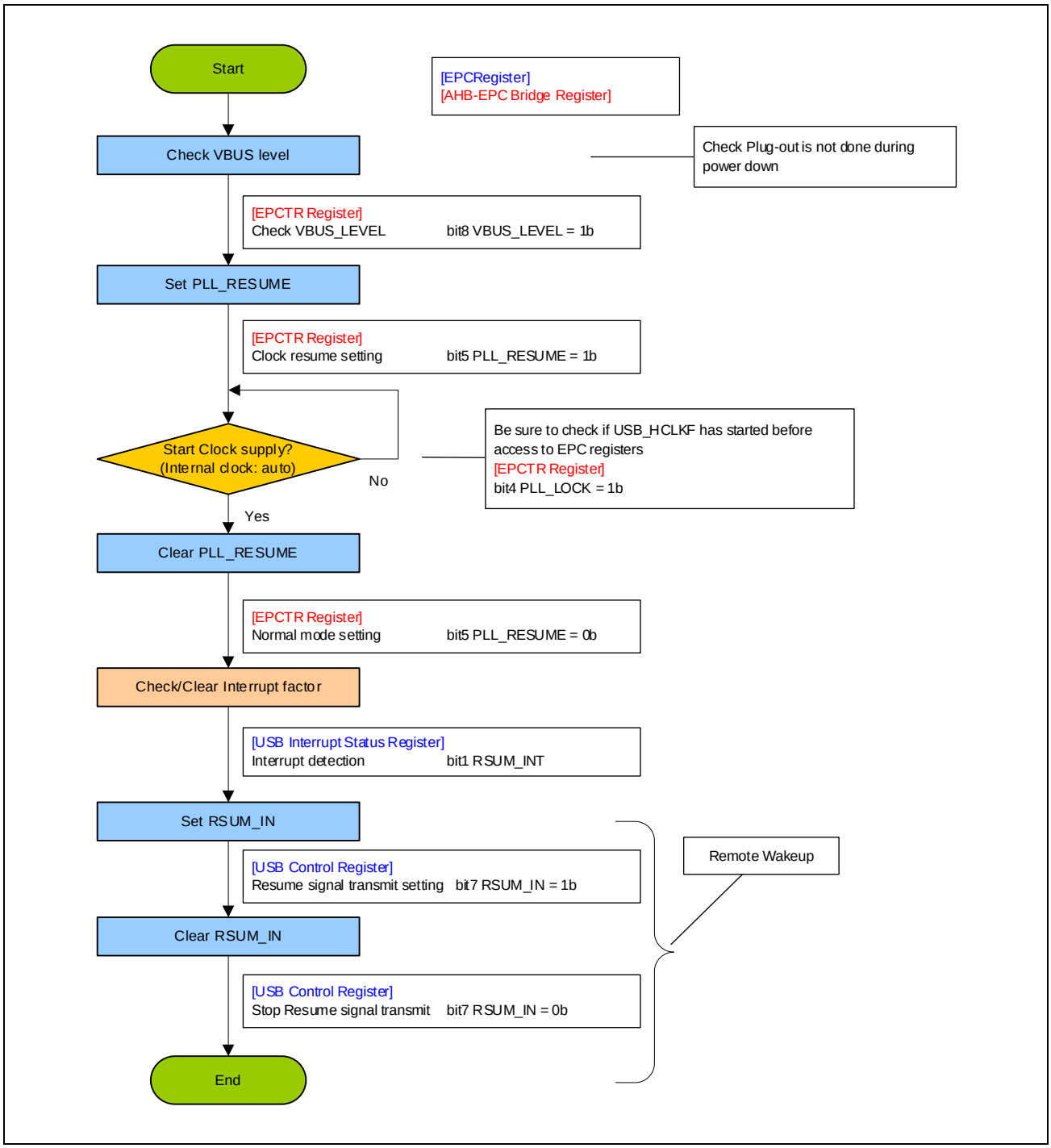


Figure 10.19 Power Up by Detecting the Resume Signal (Remote WakeUp)

10.6.7.3 Direct Power-Down Feature

When this subsystem is not used in the system, the power consumption can be reduced by stopping PLL by using the direct power-down feature.

(1) Entering the Direct Power-Down Mode

The following shows how to enter the direct power-down mode. To reduce power consumption, stopping the USB_PCICLK supply is recommended.

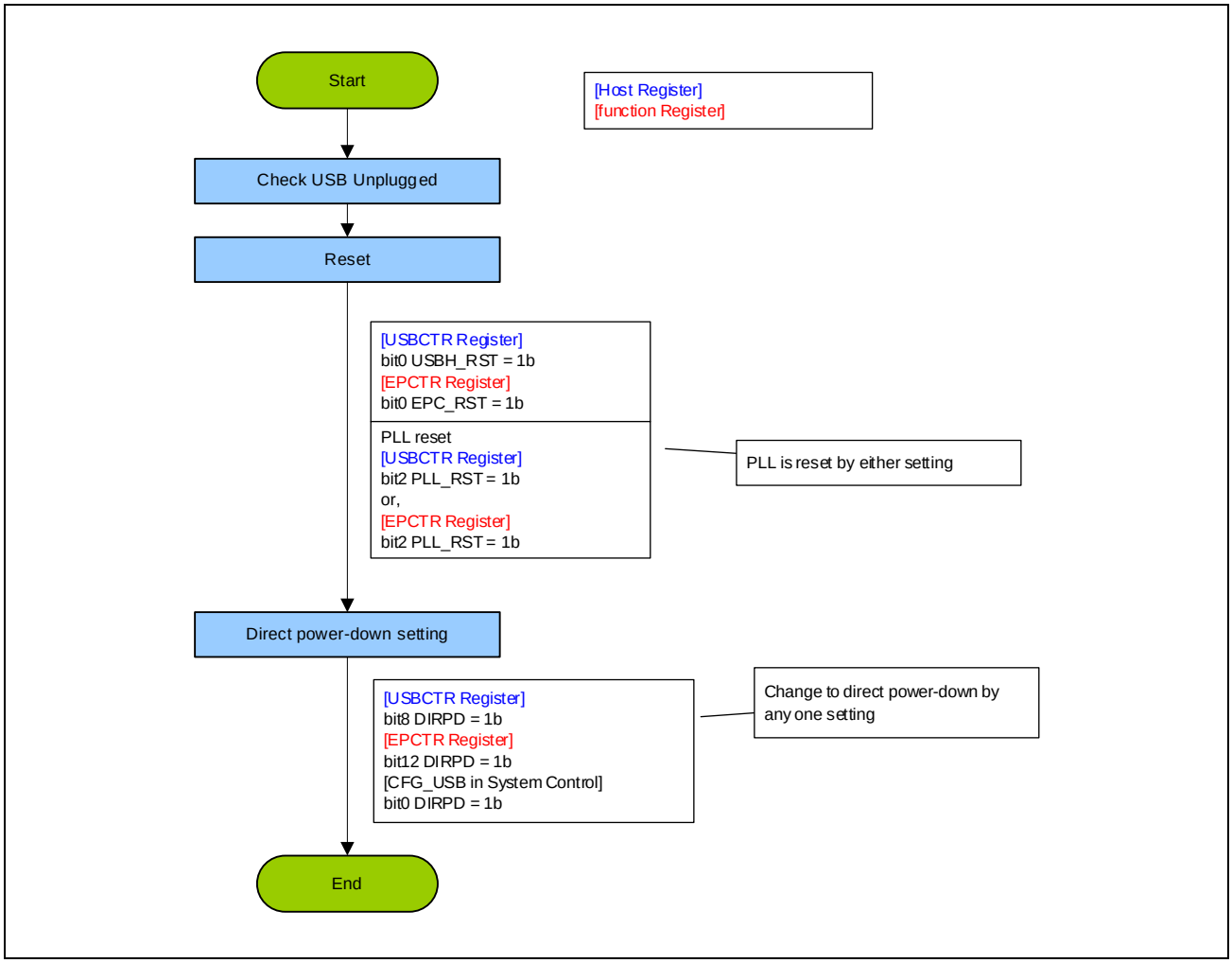


Figure 10.20 Entering the Direct Power-Down Mode

The USB Subsystem has DIRPD input and DIRPD register bits respectively. The relation is described in following figure.

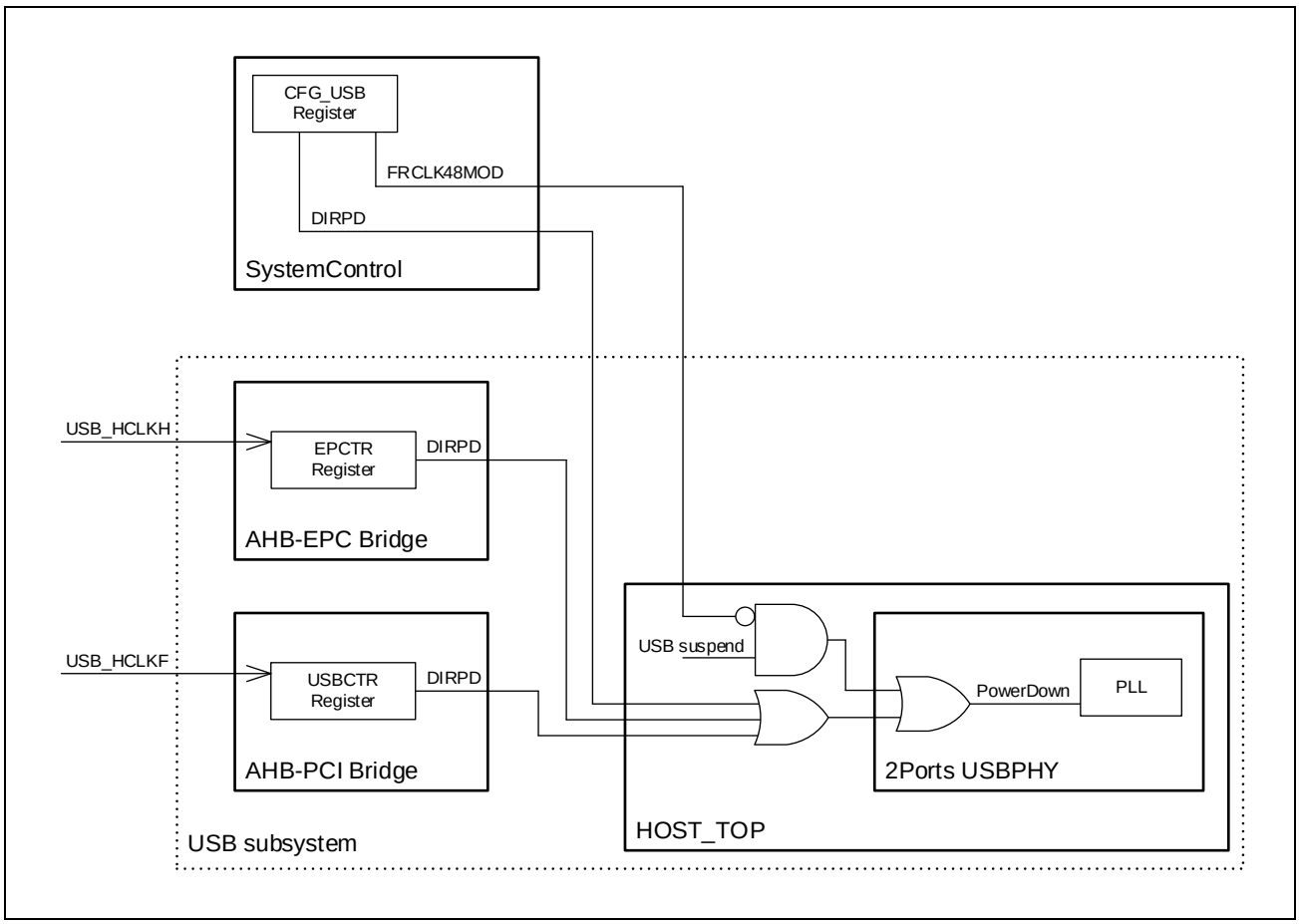


Figure 10.21 USB Subsystem - CFG_USB Register

(2) Exiting the Direct Power-Down Mode

The following shows how to exit the direct power-down mode. When doing so, cancel the direct power-down mode when requesting a software reset.

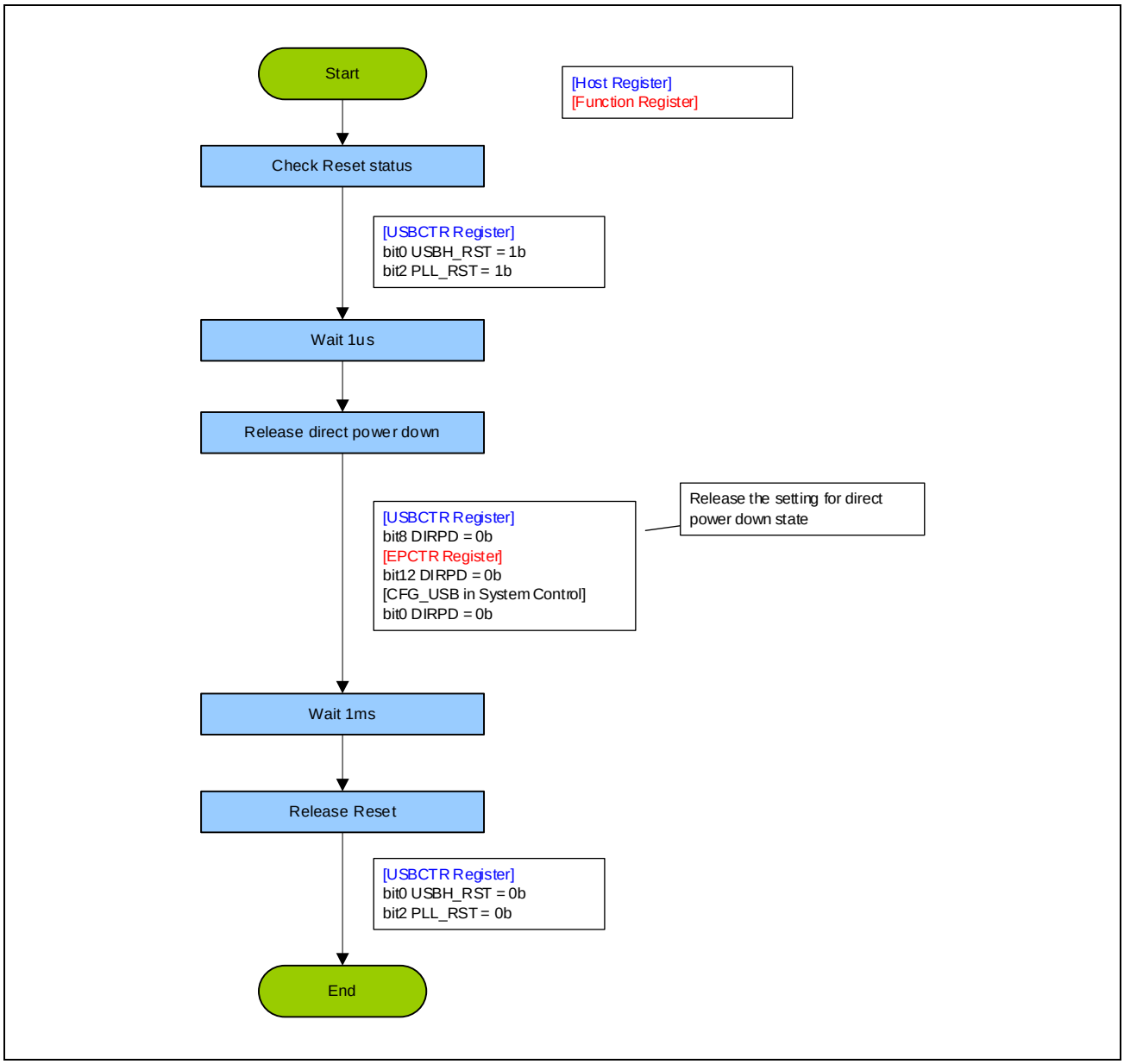


Figure 10.22 Exiting the Direct Power-Down Mode (Host Controller)

NOTE

The USB Subsystem has DIRPD input and DIRPD register bits respectively.

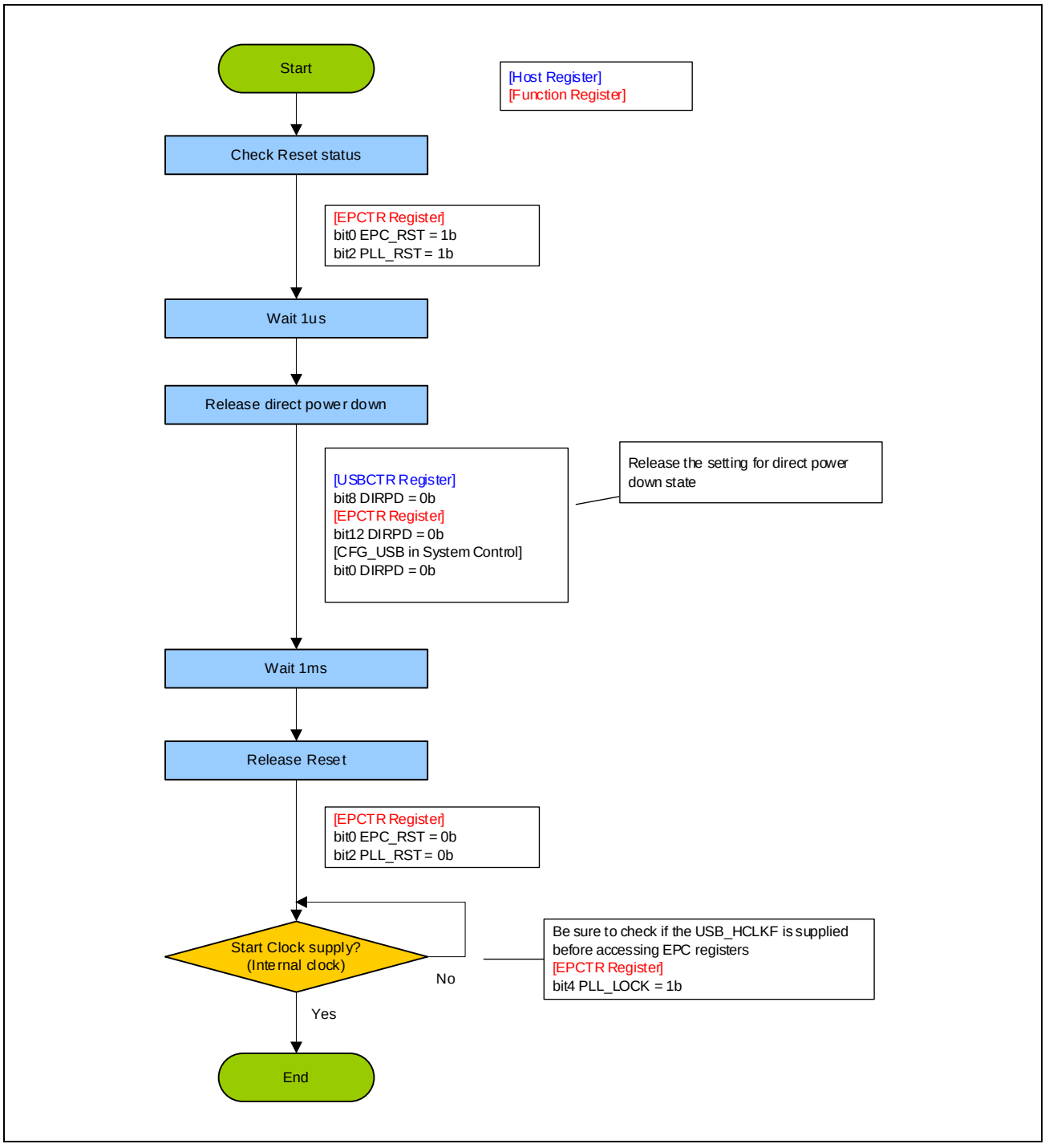


Figure 10.23 Exiting the Direct Power-Down Mode (Function Controller)

NOTE

The USB Subsystem has DIRPD input and DIRPD register bits respectively.

10.6.7.4 Notes on Suspend state transition

When receiving Resume signal or BusReset signal from the USB Host Controller immediately after Suspend state transition, it may not recover from Suspend state.

If it is required to avoid timing conflicts, do as follows.

- Not to stop the clock to USBPHY during Suspend state

  Even if Suspend state is detected, do not set 1b to the bit 6 (SUSPEND) of the USB Control Register.

10.6.8 USB Function Endpoints Configuration

In this subsystem, the Endpoints are implemented as follows.

Type and Buffer cannot be changed. For Direction and MaxPacketSize, the settings in the table below are recommended.

Table 10.134 Configuration for RZ/N1

Endpoint	Direction	Type	Buffer	MaxPacketSize
EP0	IN/OUT	Control	Single	64 Byte
EP1	IN	Bulk	Double	512 Byte
EP2	OUT	Bulk	Double	512 Byte
EP3	IN	Bulk	Single	512 Byte
EP4	OUT	Bulk	Single	512 Byte
EP5	IN	Bulk	Single	512 Byte
EP6	IN	Interrupt	Single	1024 Byte
EP7	IN	Interrupt	Single	1024 Byte
EP8	IN	Interrupt	Single	1024 Byte
EP9	IN	Interrupt	Single	1024 Byte
EP10	IN	Isochronous	Double	1024 Byte
EP11	OUT	Isochronous	Double	1024 Byte
EP12	IN	Isochronous	Double	1024 Byte
EP13	OUT	Isochronous	Double	1024 Byte
EP14	IN	Isochronous	Double	1024 Byte
EP15	OUT	Isochronous	Double	1024 Byte

10.6.8.1 Specifying the Base Address

To specify which Endpoint buffer is allocated to which RAM area, specify the RAM base address by using the EP[m]_BASEAD[12:0] bits of the EP[m] MaxPacket & BaseAddress Register. This setting is required for each Endpoint.

For the case shown in table before the table above, 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.
- Specify the EP1 MaxPacket & BaseAddress 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.

The following table lists the EP[m] MaxPacket & BaseAddress Register settings example.

Table 10.135 BaseAddress settings example

	EP[m]_BASEAD[12:0]	EP[m]_MPKT[10:0]	Value Specified for EP[m] MaxPacket & BaseAddress Register
EP0 (fixed)	—	—	—
EP1	020h	200h(HS)/040h(FS)	0020_0200h (HS) 0020_0040h (FS)
EP2	120h	200h(HS)/040h(FS)	0120_0200h (HS) 0120_0040h (FS)
EP3	220h	008h	0220_0008h
EP4	222h	200h(HS)/040h(FS)	0222_0200h (HS) 0222_0040h (FS)
EP5	322h	200h(HS)/040h(FS)	0322_0200h (HS) 0322_0040h (FS)

10.6.9 Operating Procedures

10.6.9.1 Reset Sequence

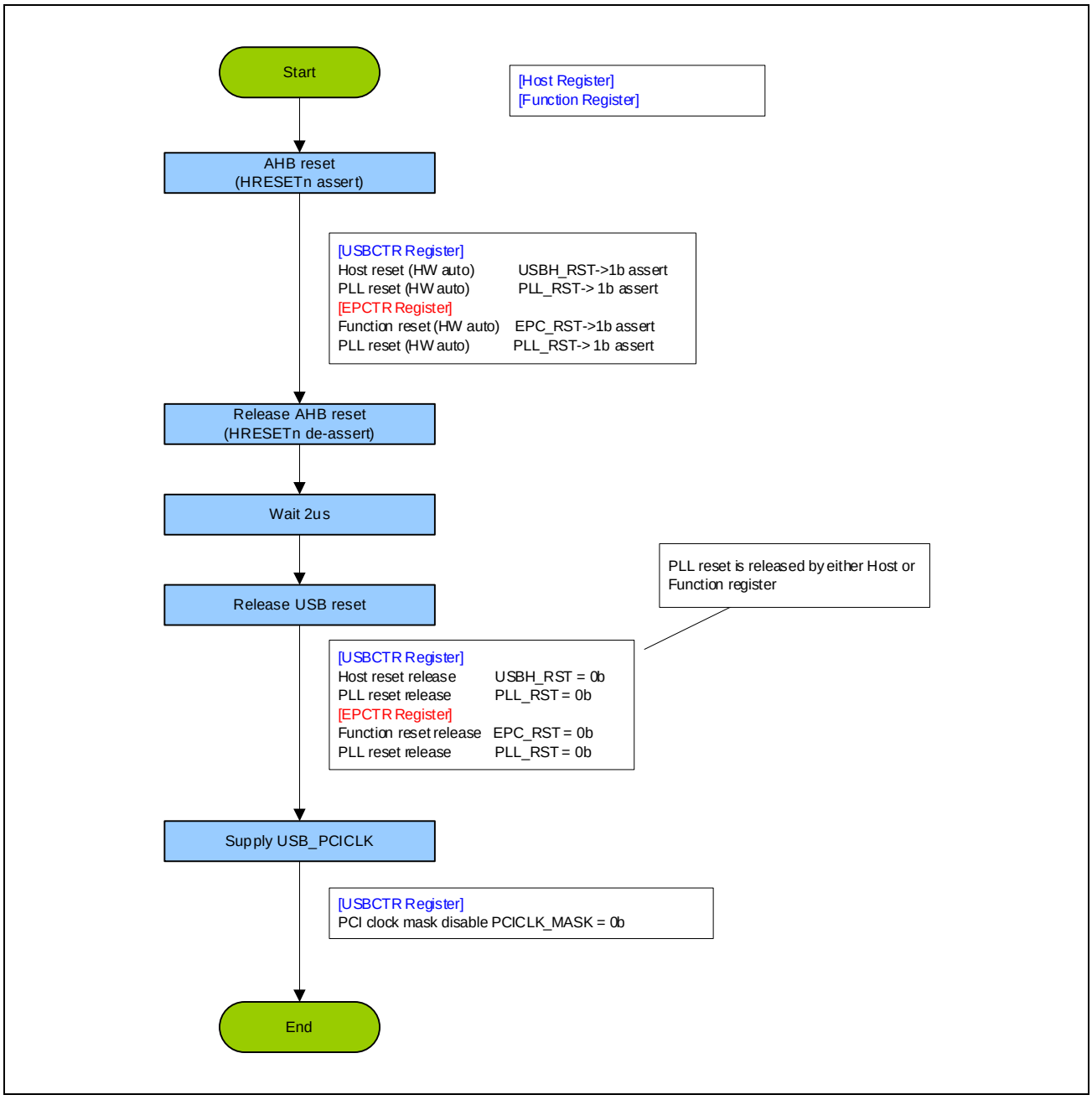


Figure 10.24 Reset Sequence

10.6.9.2 Initial Setup Sequence

(1) Example of Initial Setup of Host Controller

The following shows an example of implementing the following features:

- Accessing OHCI/EHCI operational registers via the AHB-PCI Window 2 register
- Transferring data from the host controller to the AHB bus

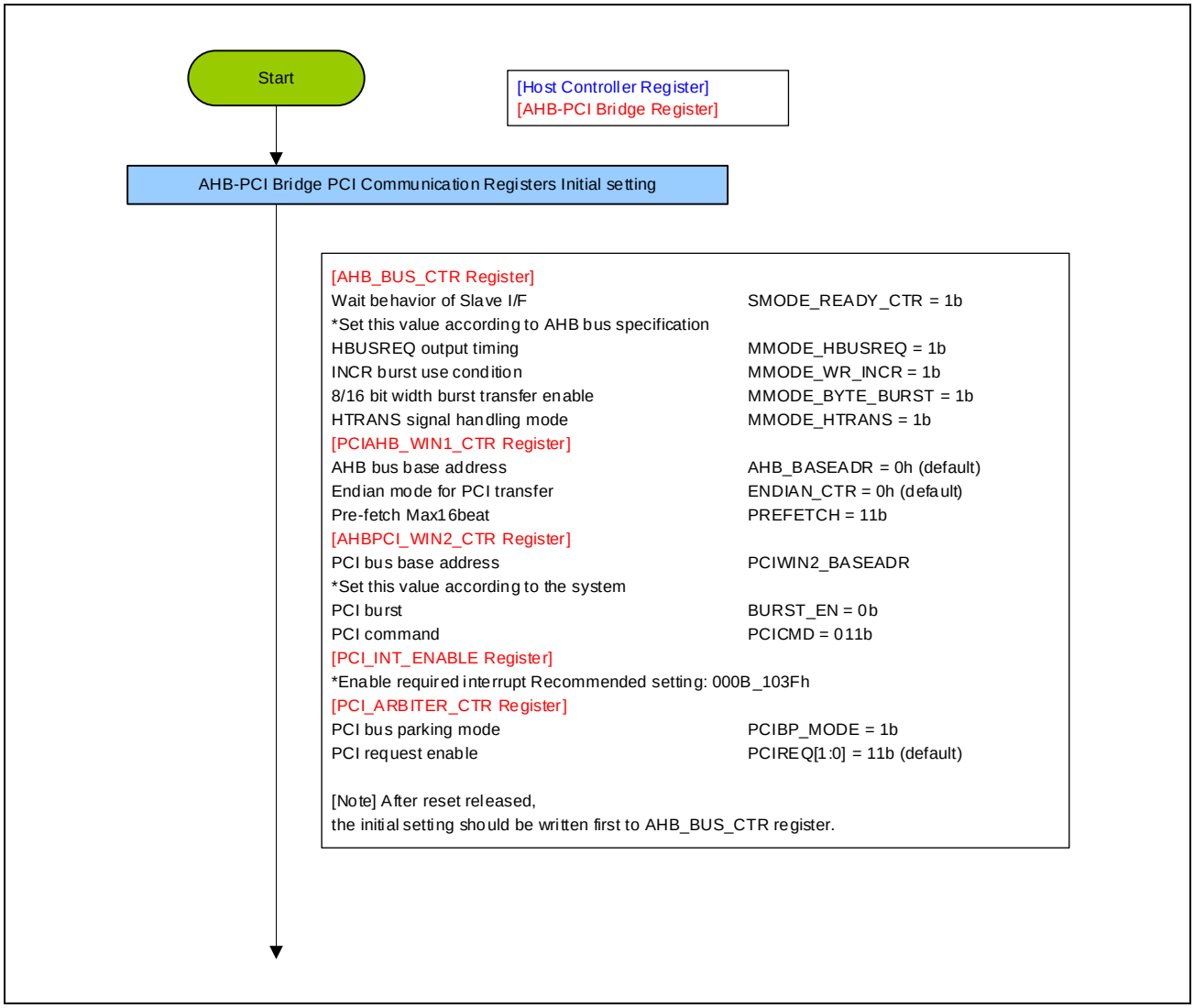


Figure 10.25 Host Controller Initial Setup Sequence (1/2)

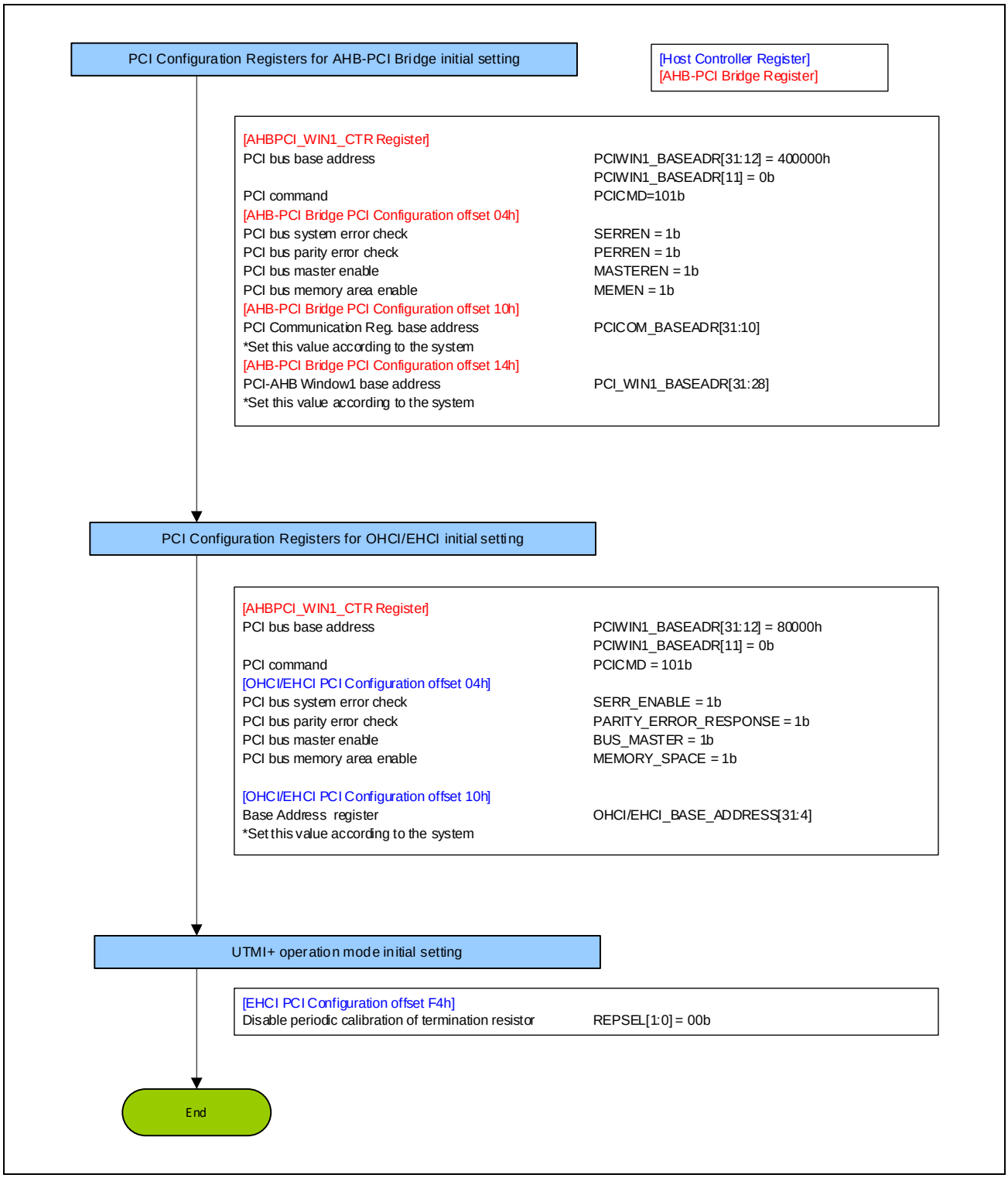


Figure 10.25 Host Controller Initial Setup Sequence (2/2)

(2) Example of Initial Setup of the Function Controller

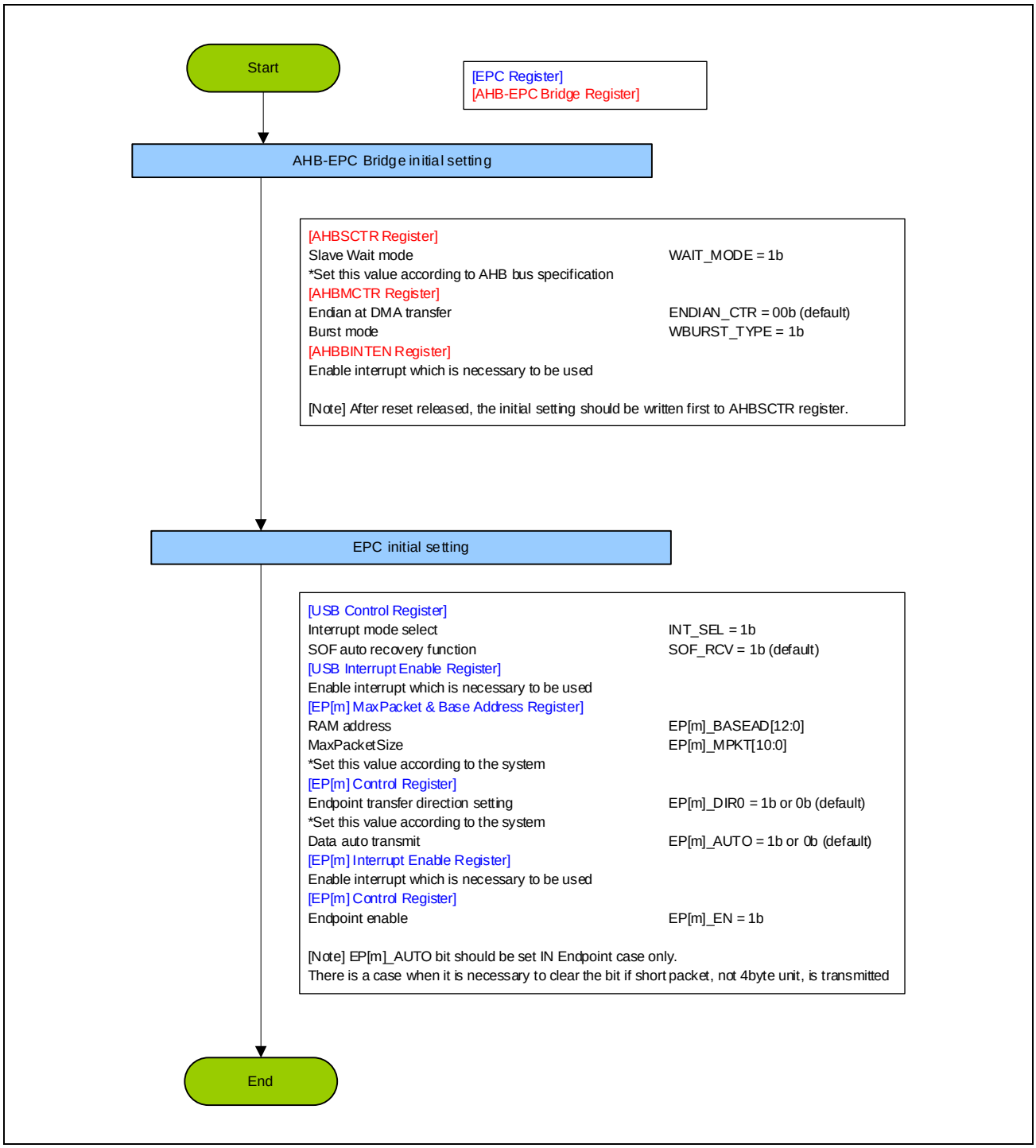


Figure 10.26 Function Controller Initial Setup Sequence

10.6.9.3 USB Host Transfer Flow

Control the USB host transfer flow according to the following OHCI/EHCI specifications.

- Open Host Controller Interface Specification for USB Rev 1.0a
- Enhanced Host Controller Interface Specification for Universal Serial Bus Revision 1.0

The following describes supplementary information about stopping DMA.

(1) Stopping DMA Transfer

The AHB-PCI bridge does not have a feature to enable or disable DMA. The PCI bus cycle in which the host controller activated as a master is output to the AHB bus as is by using DMA.

The host controller performs DMA transfer for the following purposes:

(a) To write the current frame number to the memory
(b) To read or write a descriptor and data copied to the memory to perform list processing

NOTE

Writing a frame number is automatically performed in each frame cycle if the USB state is “Operational”.

To stop a DMA transfer, set the USB to suspend or reset state.

To abort only the list processing in (b), clear the list processing enable bit (BLE, CLE, IE, and PLE in the HcControl register); the list processing stops at the next frame.

10.6.9.4 Function Transfer Overview

(1) PIO OUT Transfer

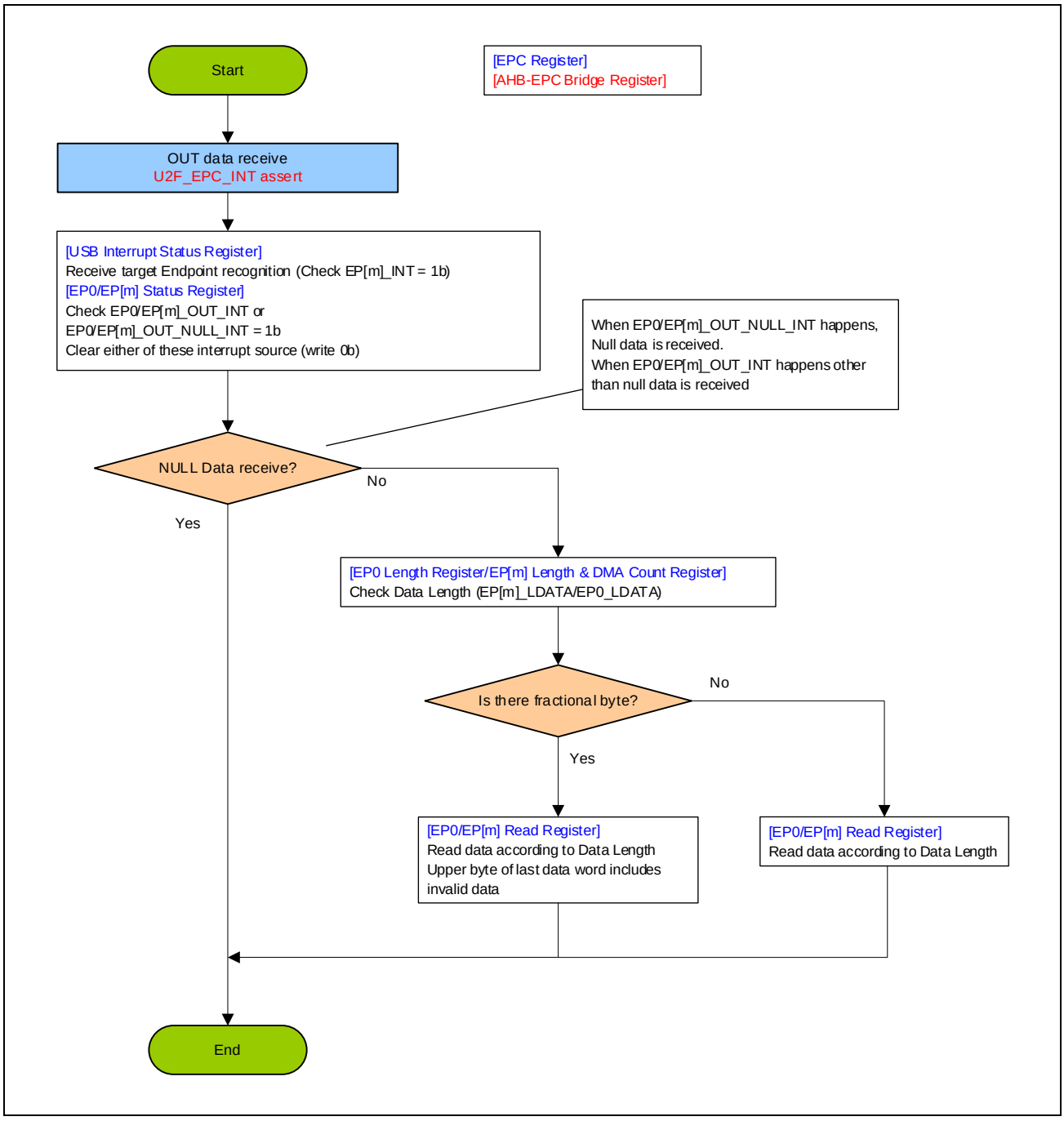
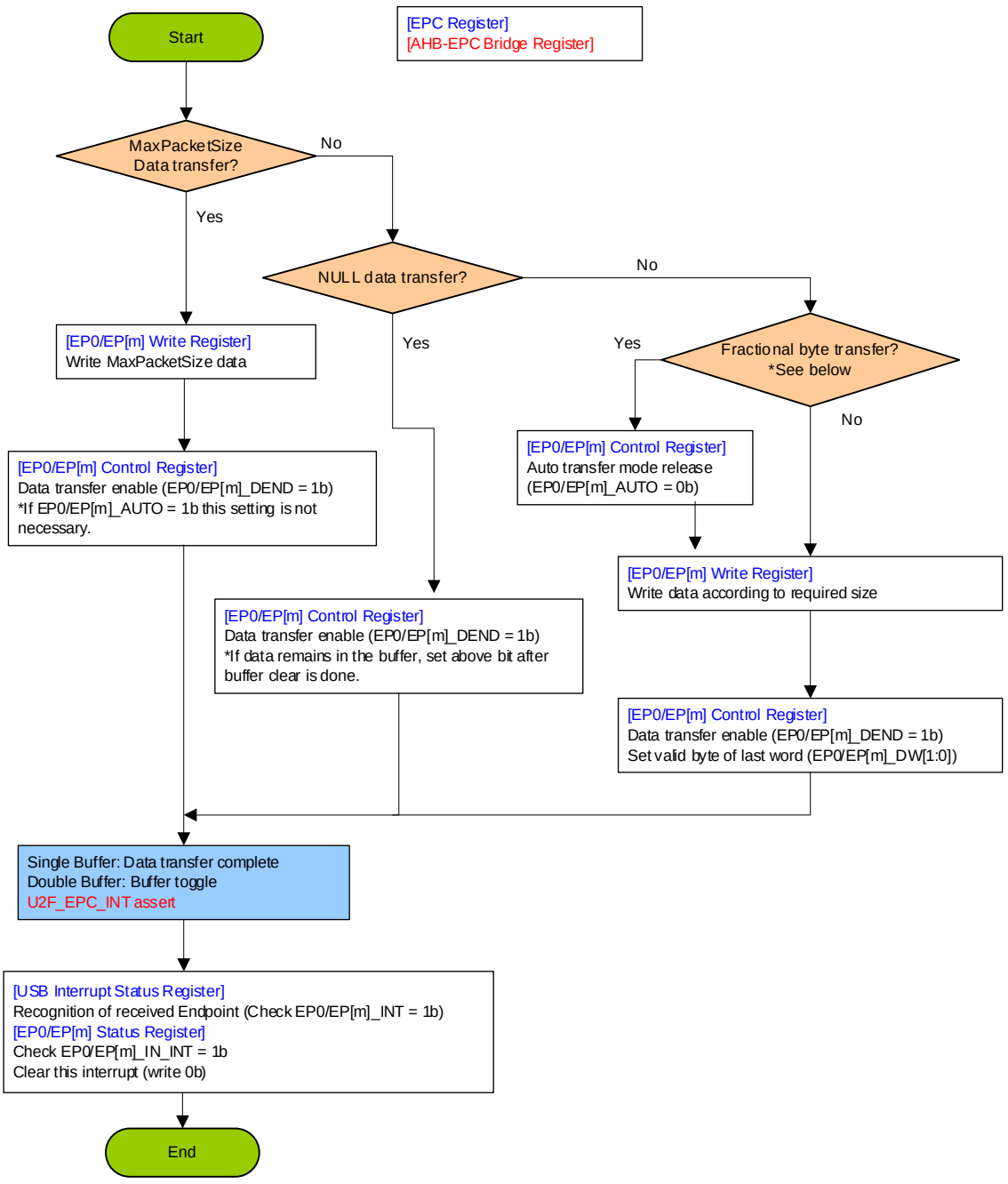


Figure 10.27 PIO OUT Transfer Overview

(2) PIO IN Transfer



Note 1. 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/EP[m]_AUTO bit. Otherwise, a packet of the maximum packet size will be transmitted.

Figure 10.28 PIO IN Transfer Overview

(3) DMA OUT Transfer

To perform a DMA transfer, specify the number of transferred packets by using the EP[m]_DMACNT bit of the EP[m]DCR1 register and EP[m]_LEN_DCNT register.

When DMA transfer is performed the number of times specified by the EP[m]_DMACNT bit of the EP[m]_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 U2F_INT or U2F_EPC_INT interrupt.

A DMA OUT transfer assumes the following conditions:

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

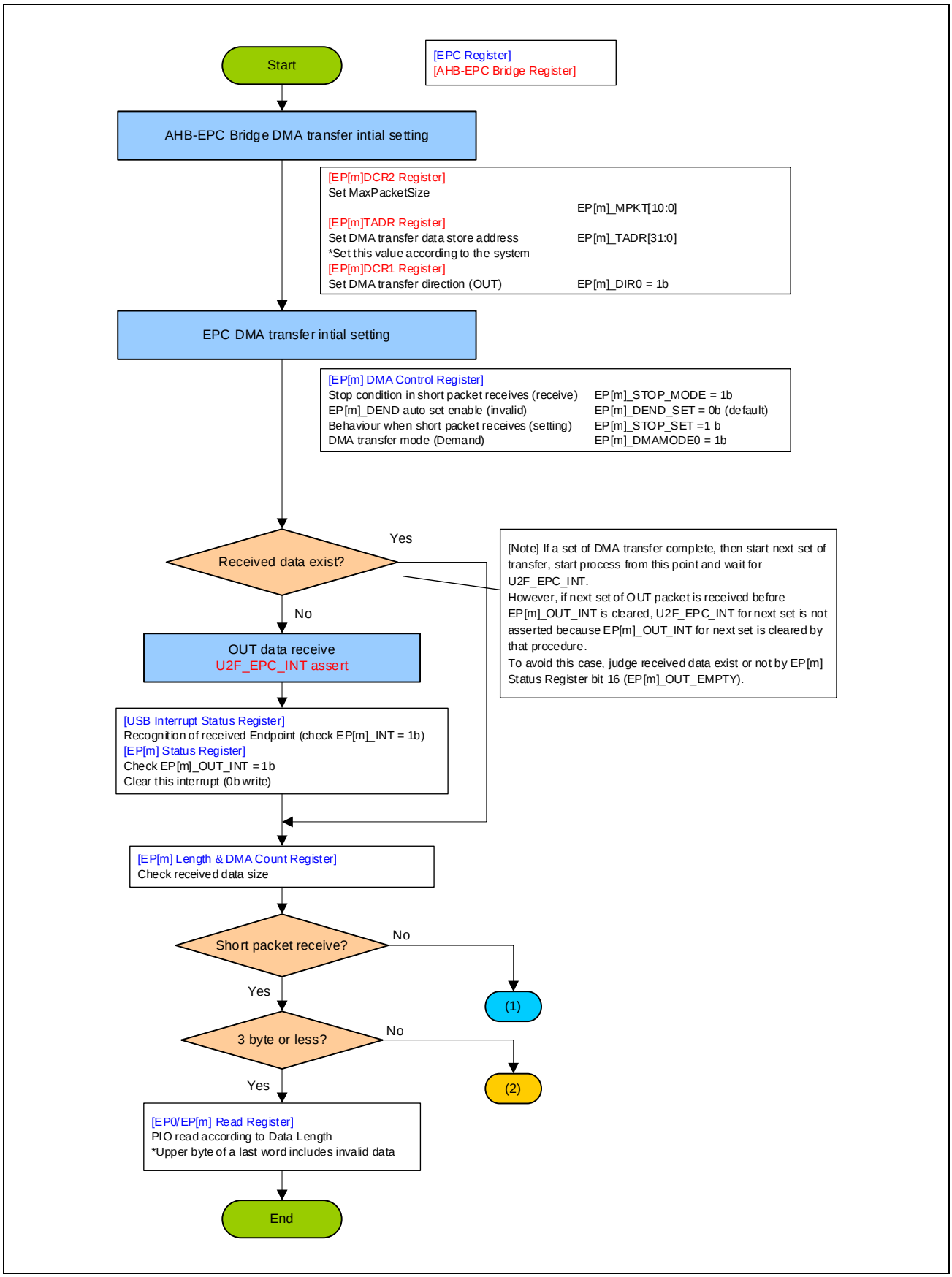


Figure 10.29 DMA OUT Transfer Overview (1/3)

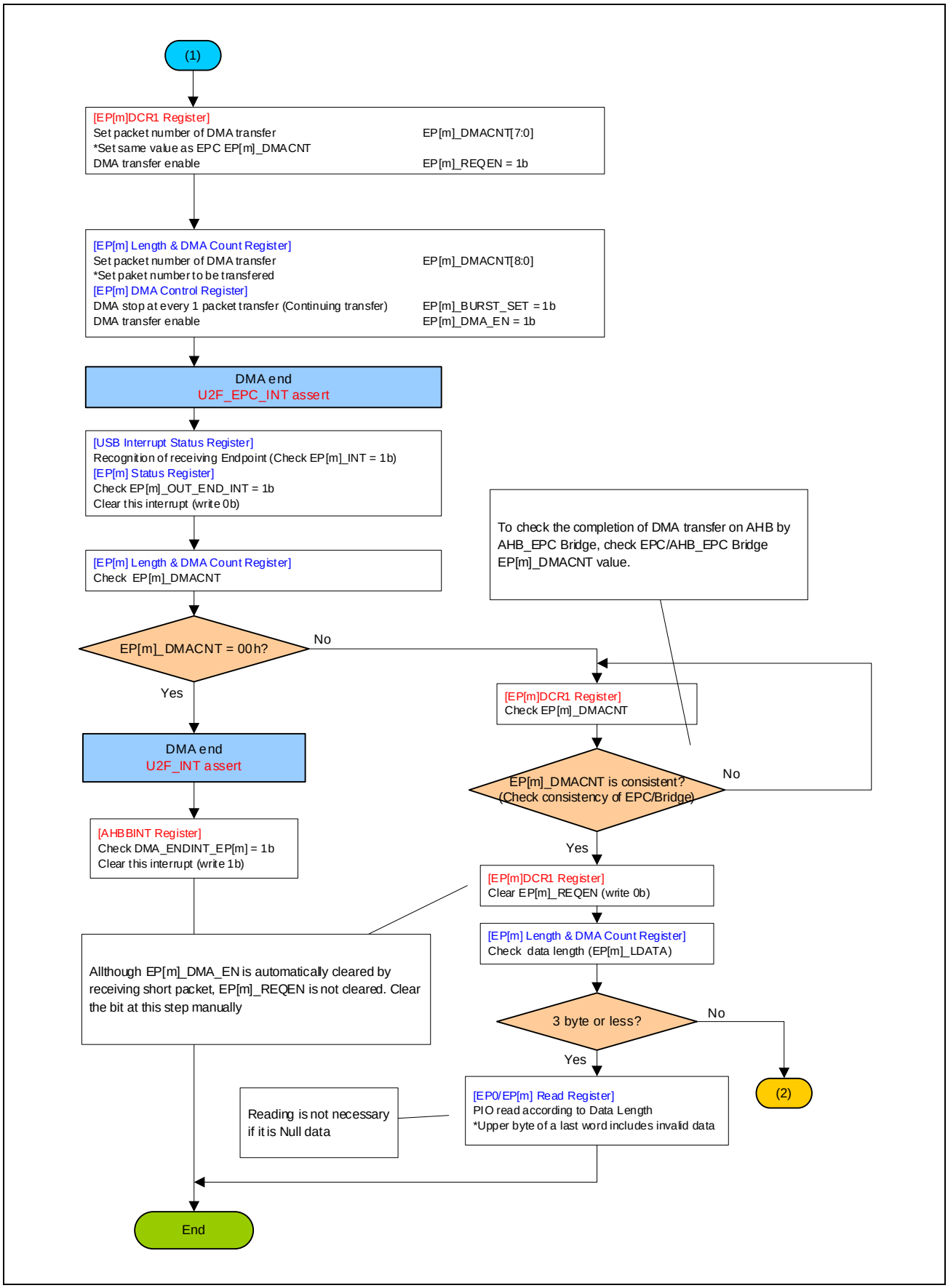


Figure 10.29 DMA OUT Transfer Overview (2/3)

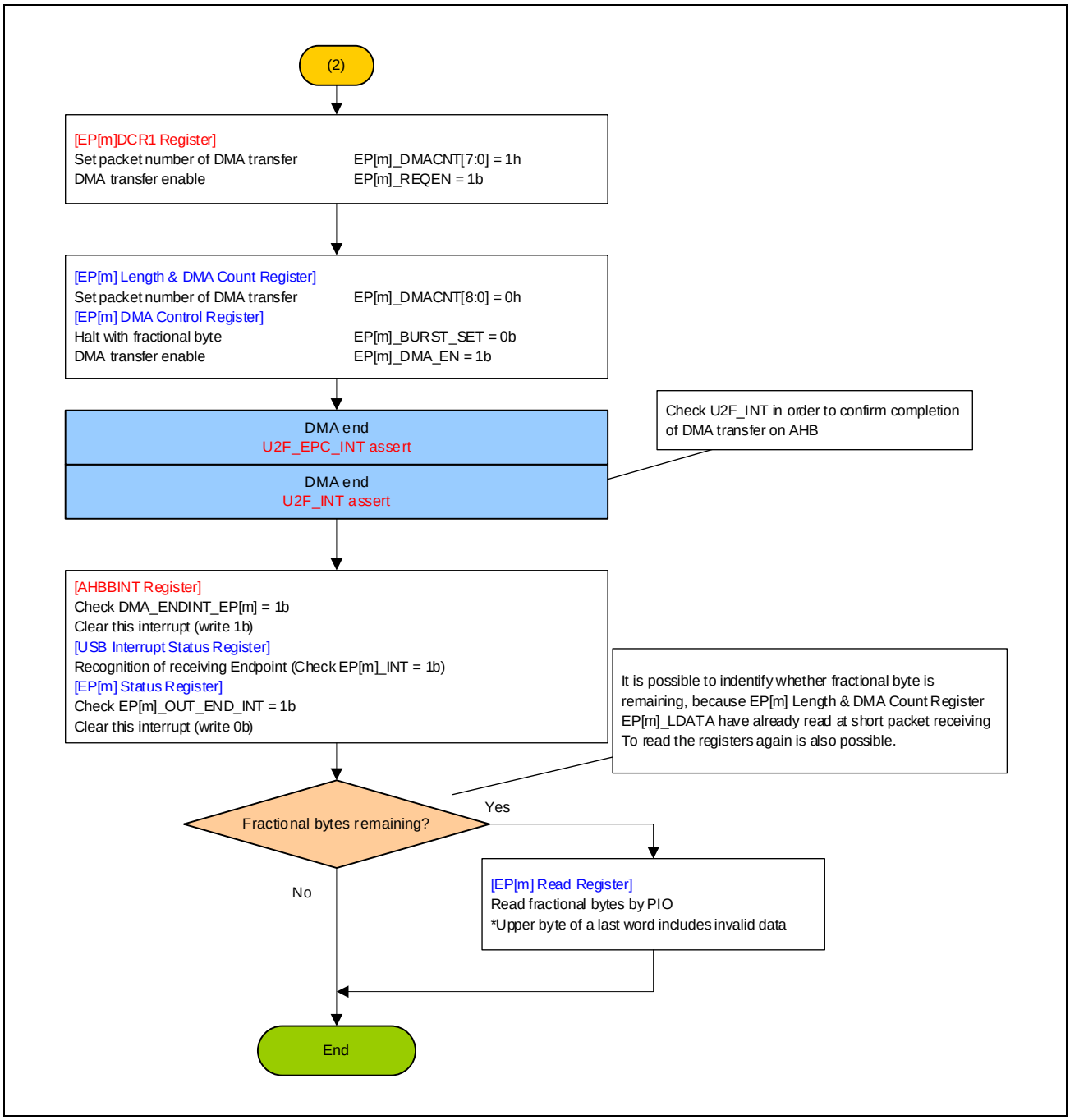


Figure 10.29 DMA OUT Transfer Overview (3/3)

(4) DMA IN Transfer

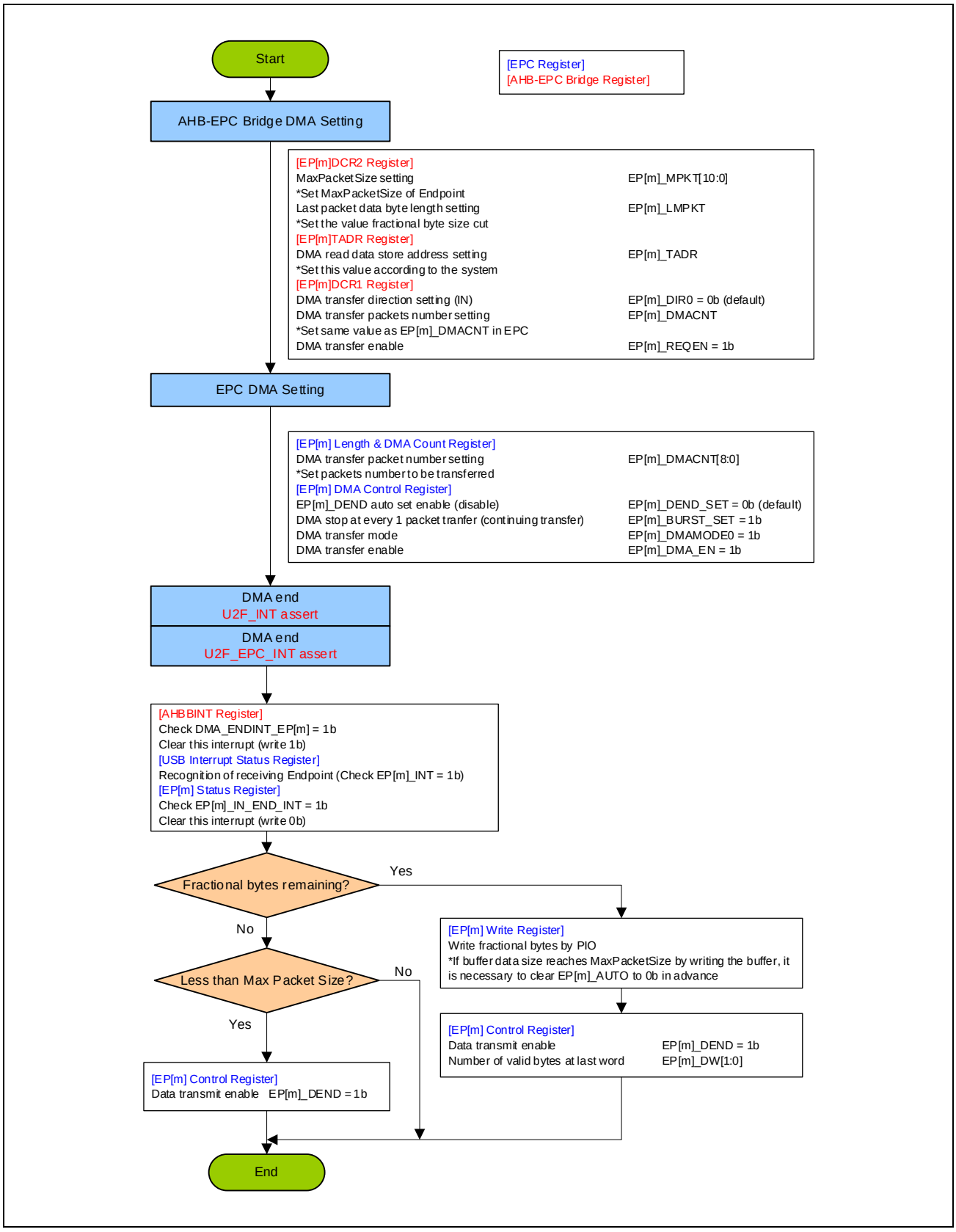


Figure 10.30 DMA IN Transfer Overview

(5) Stopping a DMA Transfer

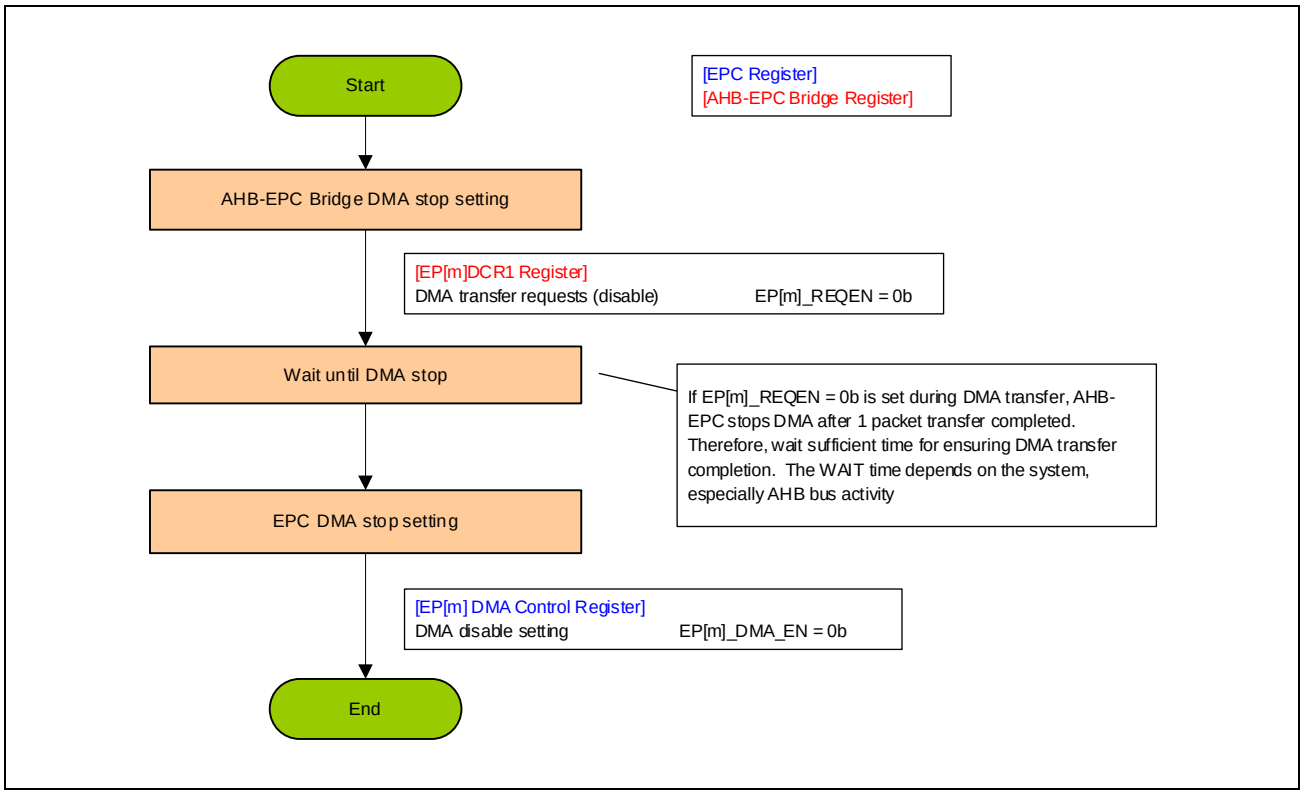


Figure 10.31 Stopping a DMA Transfer

(6) Control Transfer

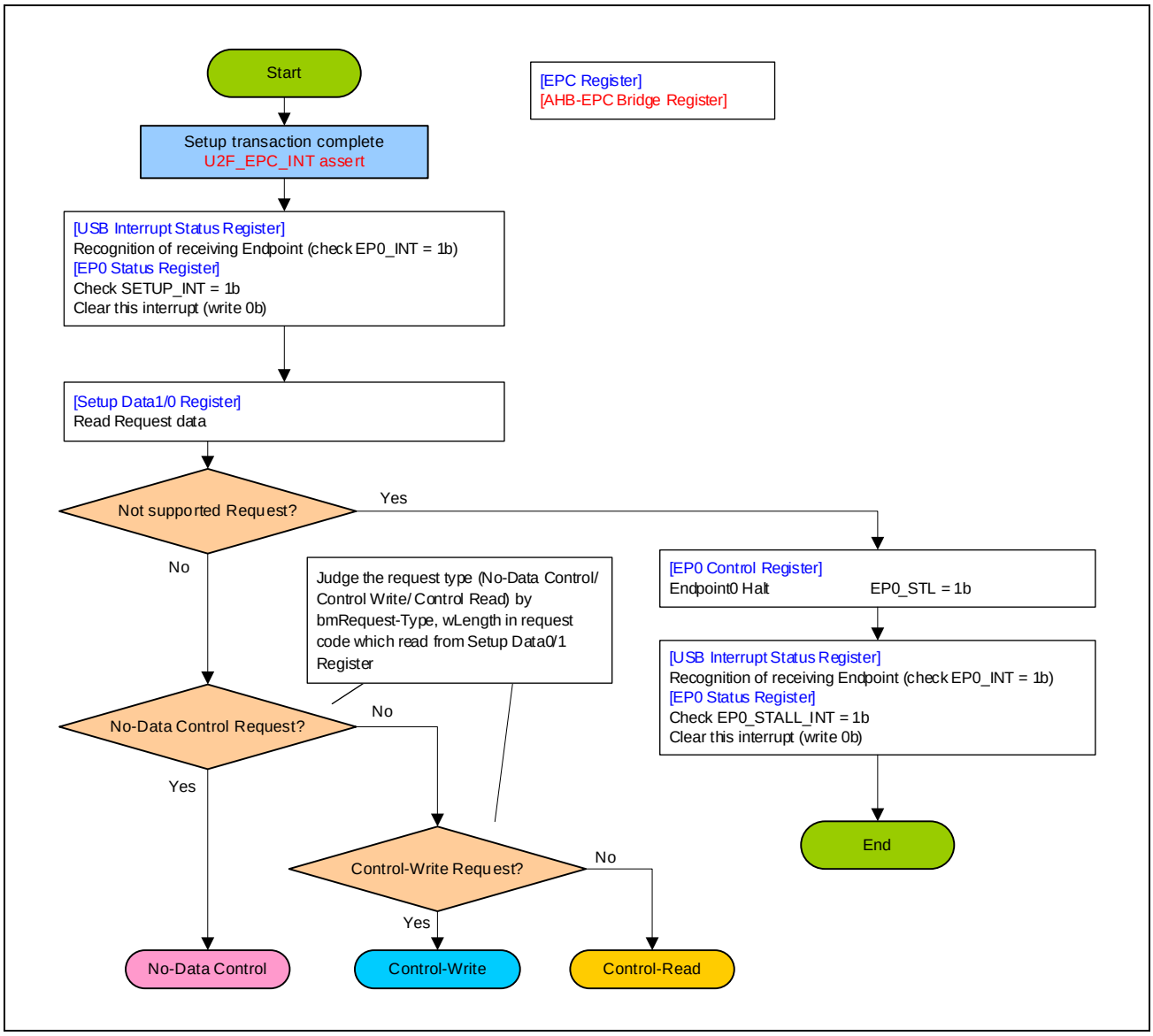


Figure 10.32 Control Transfer Overview (1/3)

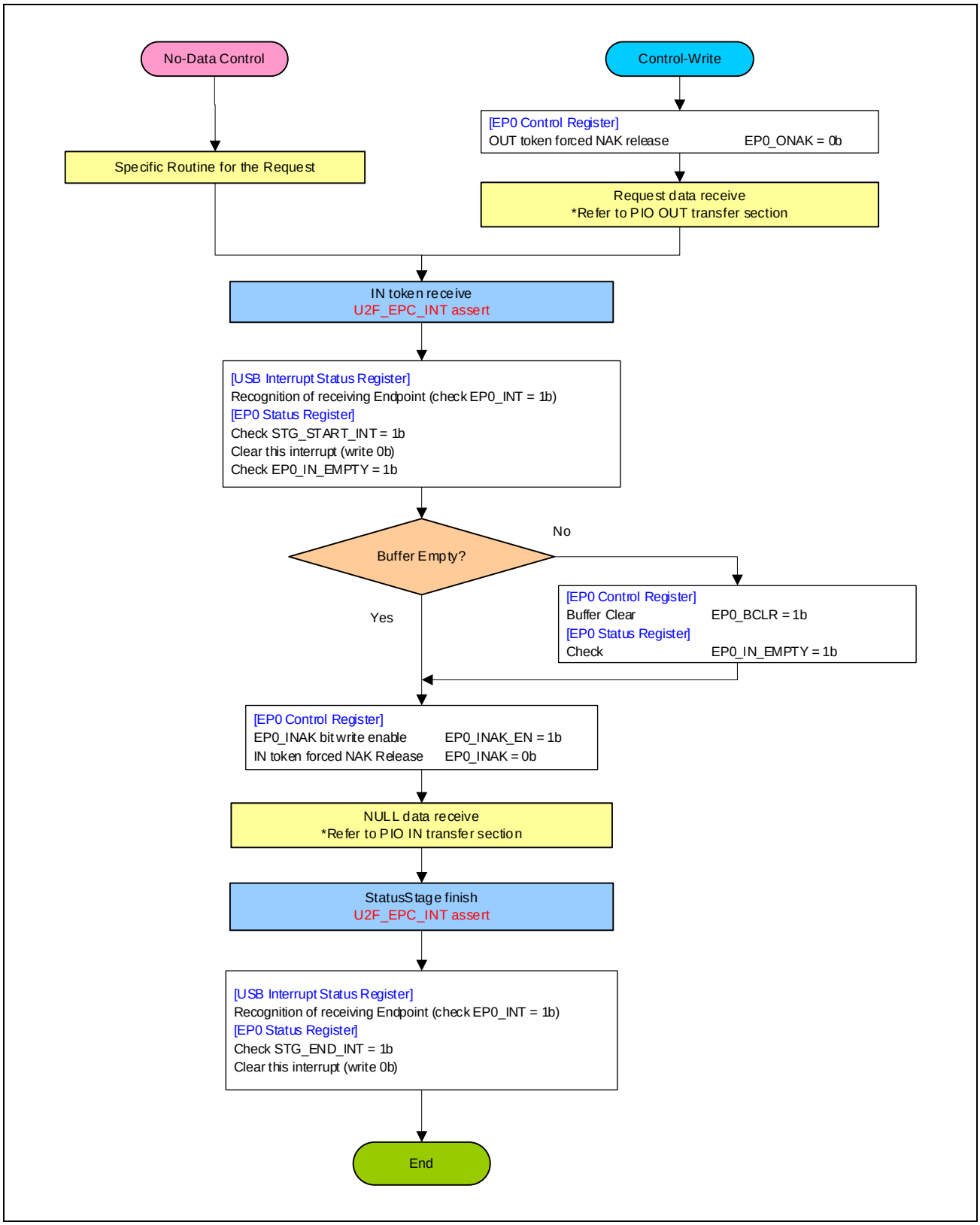


Figure 10.32 Control Transfer Overview (2/3)

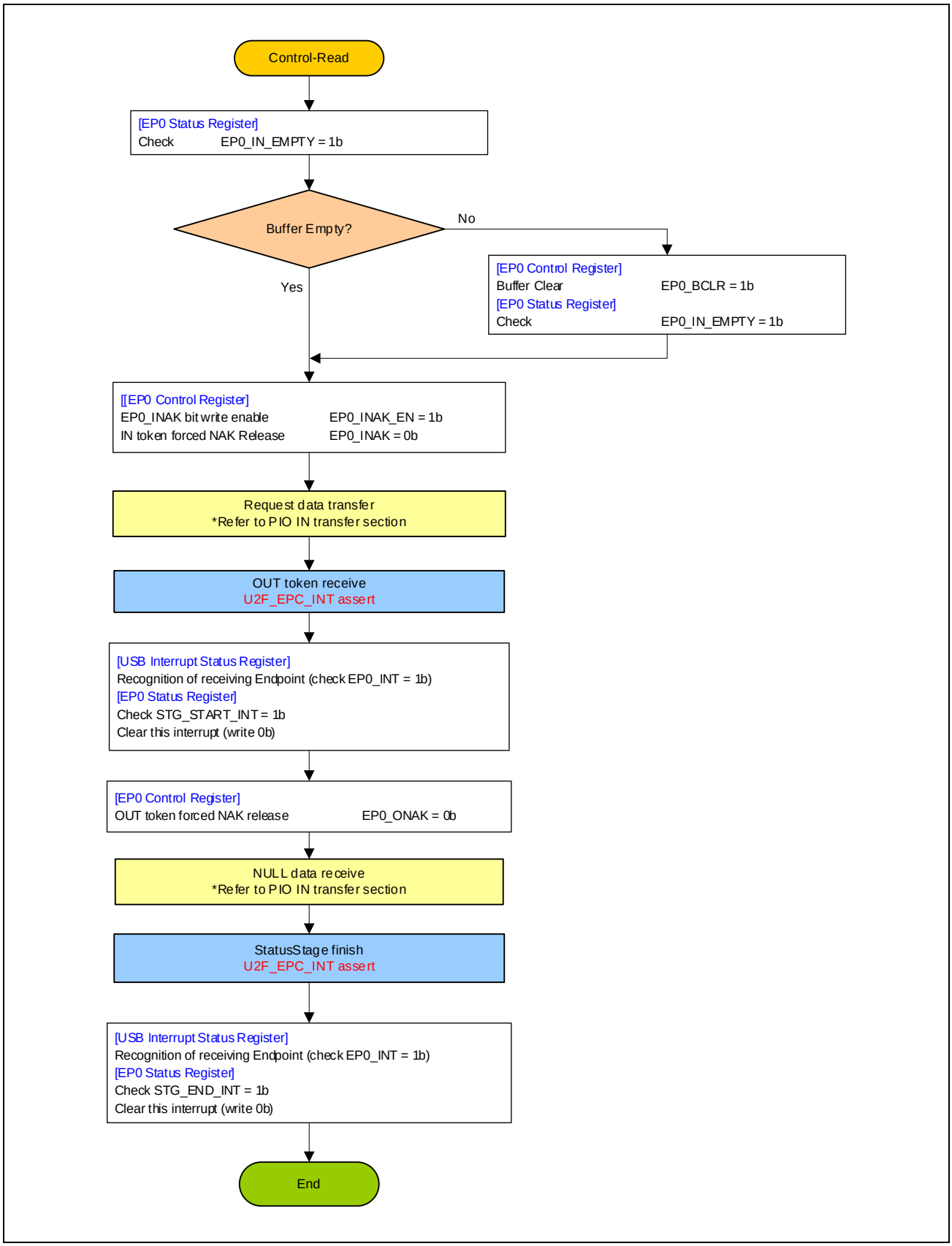


Figure 10.32 Control Transfer Overview (3/3)

(7) 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.

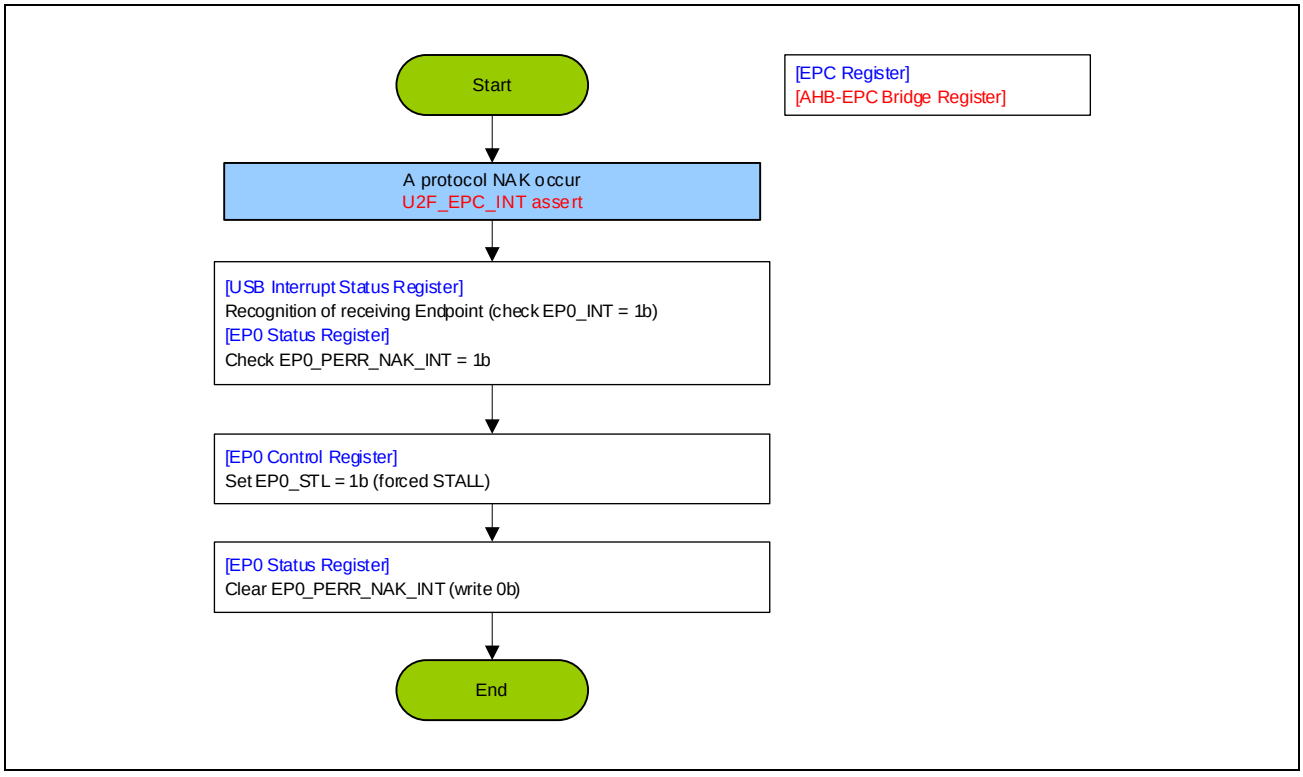


Figure 10.33 Protocol Error NAK Processing Overview

(8) 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 10.136 Device States

Default	This is the state in which the USB device address is reset to 00h after a USB bus reset 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.

(a) Clear Feature

Table 10.137 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)*1	Response (1)*1	Response (1)*1
01h (Interface)	01h	0000h	0000h	0000h	STALL response	STALL response	STALL response
02h (Endpoint)	01h	0000h (Endpoint Halt)	Endpoint No.	0000h	Response (2)	Response (2)	Response (3)

Note 1. 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 10.138 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 Endpoint0. (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 Endpoint0. (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 EP[m]_OPIDCLR, EP[m]_IPIDCLR, EP[m]_OSTL, EP[m]_ISTL, and EP[m]_BCLR bits of the EP[m] Control register to 0b.) • If wIndex does not indicate a supported Endpoint, send a STALL.

(b) Get Configuration

Table 10.139 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 10.140 Get Configuration Request Processing

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

(c) GET DESCRIPTOR

Table 10.141 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 10.142 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.

(d) GET INTERFACE

Table 10.143 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 10.144 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.

(e) Get Status

Table 10.145 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 No.	0002h	Response (3)	Response (3)	Response (4)

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

<Processing details>

Table 10.146 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 (EP[m]_ISTL or EP[m]_OSTL) of the EP[m] Control register is 1b. If 0b, send 0000b.) • If wIndex does not indicate a supported Endpoint, send a STALL.

(f) SET ADDRESS

Table 10.147 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 10.148 Set Address Request Processing

Response (1)	• If wValue is less than or equal to 127, set the wValue value to the Frame Number & USB Address Register. • If wValue is greater than or equal to 128, send a STALL.

(g) Set Configuration

Table 10.149 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 10.150 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.

(h) SET DESCRIPTOR

Table 10.151 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 10.152 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.)

(i) Set Feature

Table 10.153 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
		0002h (TEST MODE)	Test Selector	0000h	Response (2)	Response (2)	Response (2)
01h (Interface)	03h	0000h	0000h	0000h	STALL response	STALL response	STALL response
02h (Endpoint)	03h	0000h (Endpoint Halt)	Endpoint No.	0000h	Response (3)	Response (3)	Response (4)

Transfer type: No Data Control (SETUP-IN)

<Processing details>

Table 10.154 Set Feature Request Processing

Response (1)	• Set the remote wakeup flag. (The USB function controller does not need to perform special processing.)
Response (2)	• Write the Test Selector value indicated by wIndex to the USBTESTMODE bit of the USB Control register.
Response (3)	• 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 (4)	• If wIndex is 0000h, clear the buffer for Endpoint0. (Perform the same processing as that of *Response (2)*). • When wIndex indicates the supported Endpoint, set the target Endpoint to the Halt state. (Set the EP[m]_OSTL and EP[m]_ISTL bit of the EP[m] Control register to 1b.) • If wIndex does not indicate a supported Endpoint, send a STALL.

(j) SET INTERFACE

Table 10.155 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 10.156 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 EP[m] MaxPacket & BaseAddress register etc. • 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.

(k) Sync Frame

Table 10.157 Sync Frame Request

Request Code Field					Action		
bmRequest - Type	bRequest	wValue	wIndex	wLength	Default	Address	Configured
82h	0Ch	0000h	EP No.	0000h	STALL response	STALL response	STALL response

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

(9) Descriptors

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

(a) Device descriptor

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

Table 10.158 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	Specify the USB standard version the device conforms to. 0200h indicates that the device conforms to USB 2.0.
bDeviceClass	4	Byte	xxh	Specify the class the device belongs to.
bDeviceSubClass	5	Byte	xxh	Specify the subclass the device belongs to.
bDeviceProtocol	6	Byte	00h	Specify the protocol defined by the class or subclass the device belongs to.
bMaxPacketSize0	7	Byte	40h	Specify the maximum packet size (64 bytes) to be transferred at Endpoint 0.
idVendor	8	Word	xxxxh	Specify the vendor ID.
idProduct	10	Word	xxxxh	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.

(b) Device qualifier descriptor

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

Table 10.159 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	Specify the USB standard version the device conforms to. 0200h indicates that the device conforms to USB 2.0.
bDeviceClass	4	Byte	xxh	Specify the class the device belongs to.
bDeviceSubClass	5	Byte	xxh	Specify the subclass the device belongs to.
bDeviceProtocol	6	Byte	00h	Specify the protocol defined by the class or subclass the device belongs to.
bMaxPacketSize0	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.
bReserved	9	Byte	00h	Reserved for future use, must be zero.

(c) Configuration descriptor and Other speed configuration descriptor

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 10.160 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*
bNumInterfaces	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.
bMaxPower	8	Byte	00h	Specify the current value required by the device. Value = current consumption value/2 (mA).

(d) Interface descriptor

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

Table 10.161 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	xxh	Specify the number of Endpoints other than Endpoint 0 supported by this interface.
bInterfaceClass	5	Byte	xxh	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	xxh	Specify the protocol defined by the class or subclass the interface belongs to. This is equivalent to bDeviceProtocol in device descriptors.
iInterface	8	Byte	xxh	Specify the index for the string descriptor that describes this interface.

(e) Endpoint descriptor

Endpoint descriptor is used to send information about the Endpoint used.

Table 10.162 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	xxh	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	xxh	Specify the transfer type at the Endpoint. • Bits 7 and 6: Always set to 00b. • Bits 5 to 2 are fixed to 0000b do not support isochronous transfers. • Bits 1 and 0: Set the transfer type. (00b: Control, 01b: Isochronous, 10b: Bulk, 11b: Interrupt)
wMaxPacketSize	4	Word	0200h	Specify the maximum packet size to be sent at the Endpoint. • Bits 15 to 11: Always set to 000b. • Bits 10 to 0: Set 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.

(f) String descriptor

String descriptor is used to send the text for description.

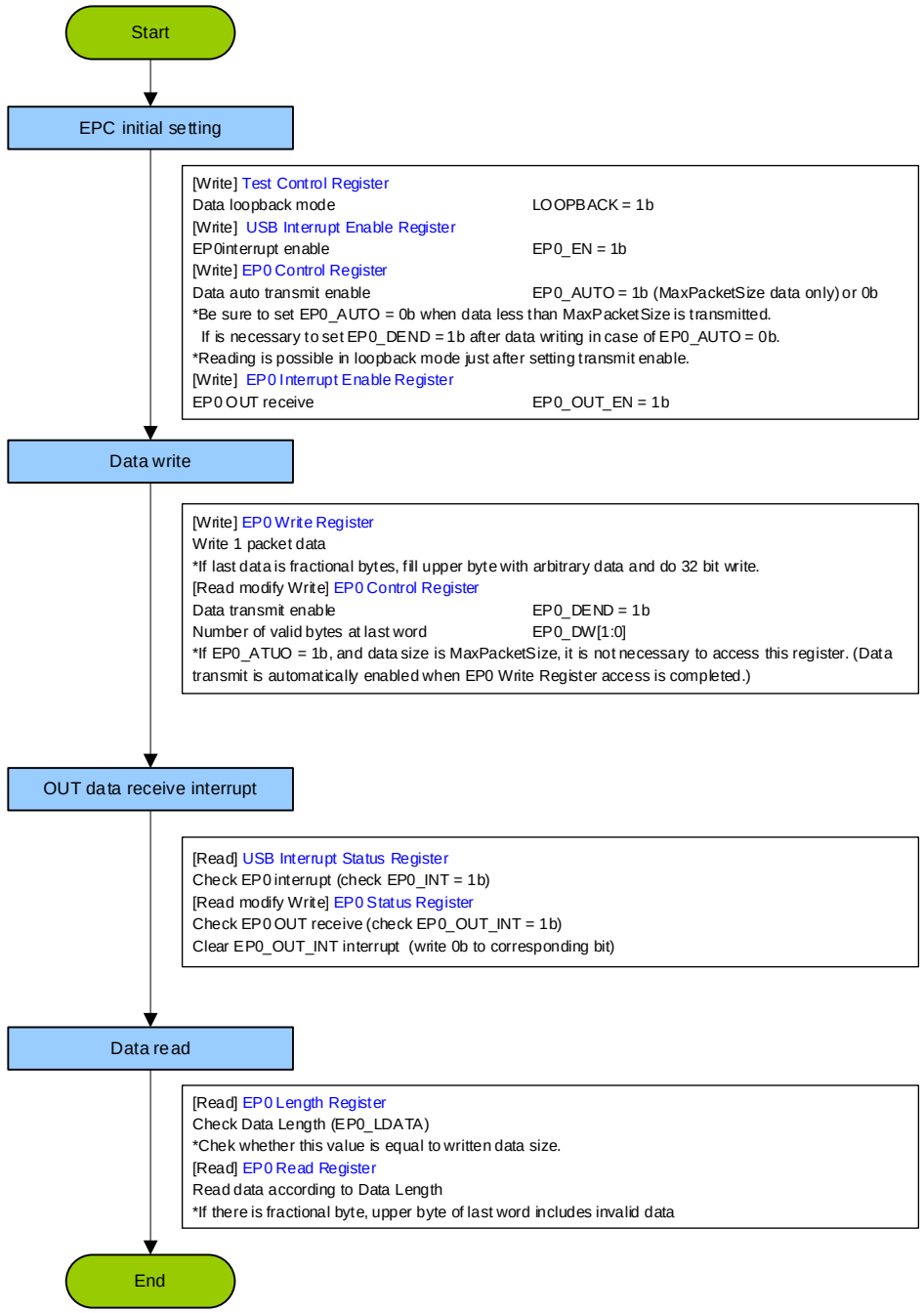
Table 10.163 String Descriptor

Field	Offset	Size	Value (Example)	Description
bLength	0	Byte	xxh	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	2	Word	0409h	• When wIndex is 0, specify the language of the string for wLANGID.
bString		Byte	xxh	• 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.

(10) EP0 Data Loopback

This section describes how to loop back data at EP0.

Execute data loopback while no other operations are being performed after reset release.



Note 1. Be sure to clear the buffer (by writing 1b to the EP0_BCLR of the EP0 Control register) before looping back the next data.

Note 2. In the above flow, the EPC initial setting is retained, so registers whose settings do not change do not need to be set.

Figure 10.34 EP0 Data Loopback

(11) EP[m] Data Loopback

This section describes how to loop back data at EP[m].

Execute data loopback while no other operations are being performed after reset release.

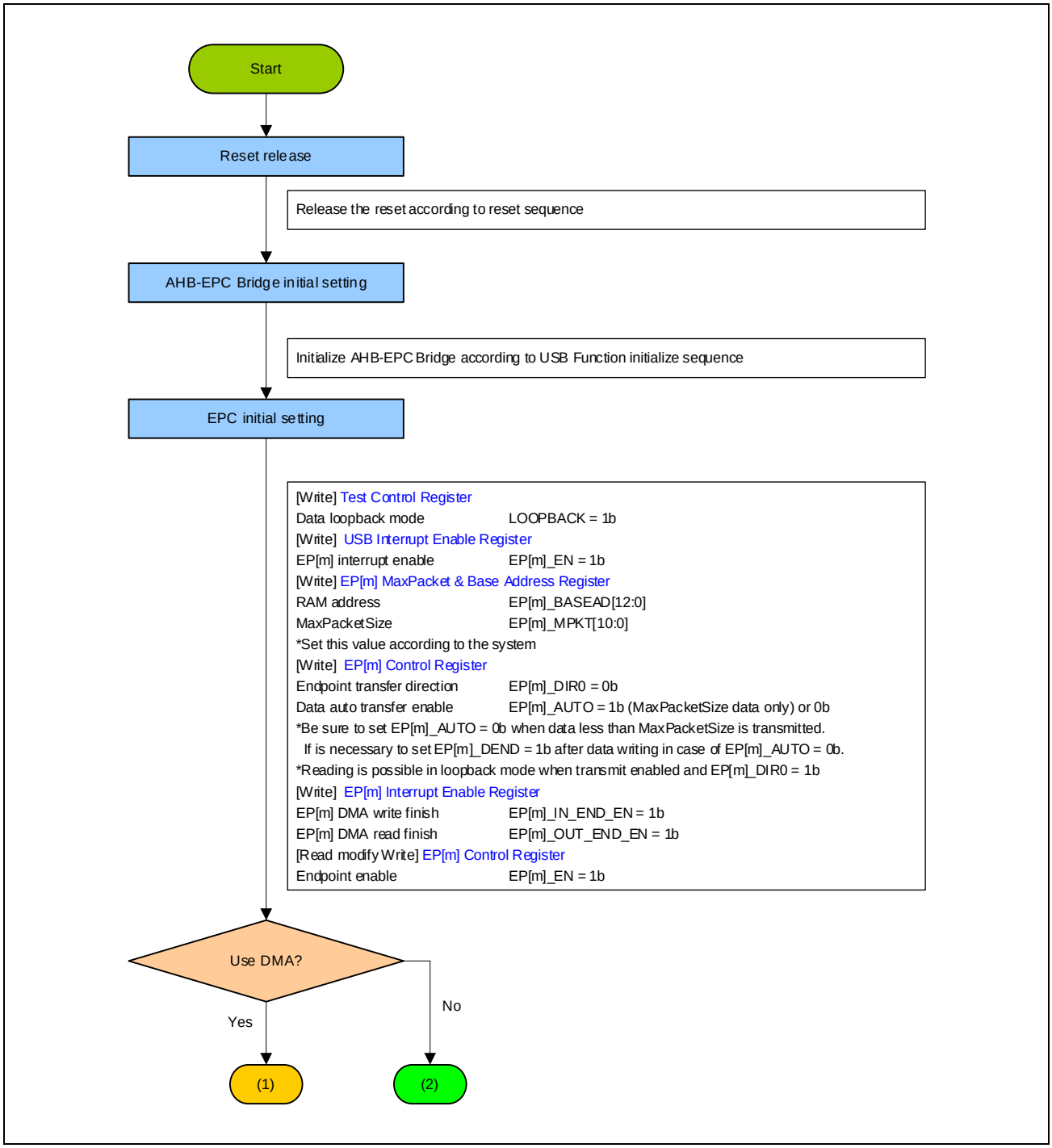


Figure 10.35 EP[m] Data Loopback (1/4)

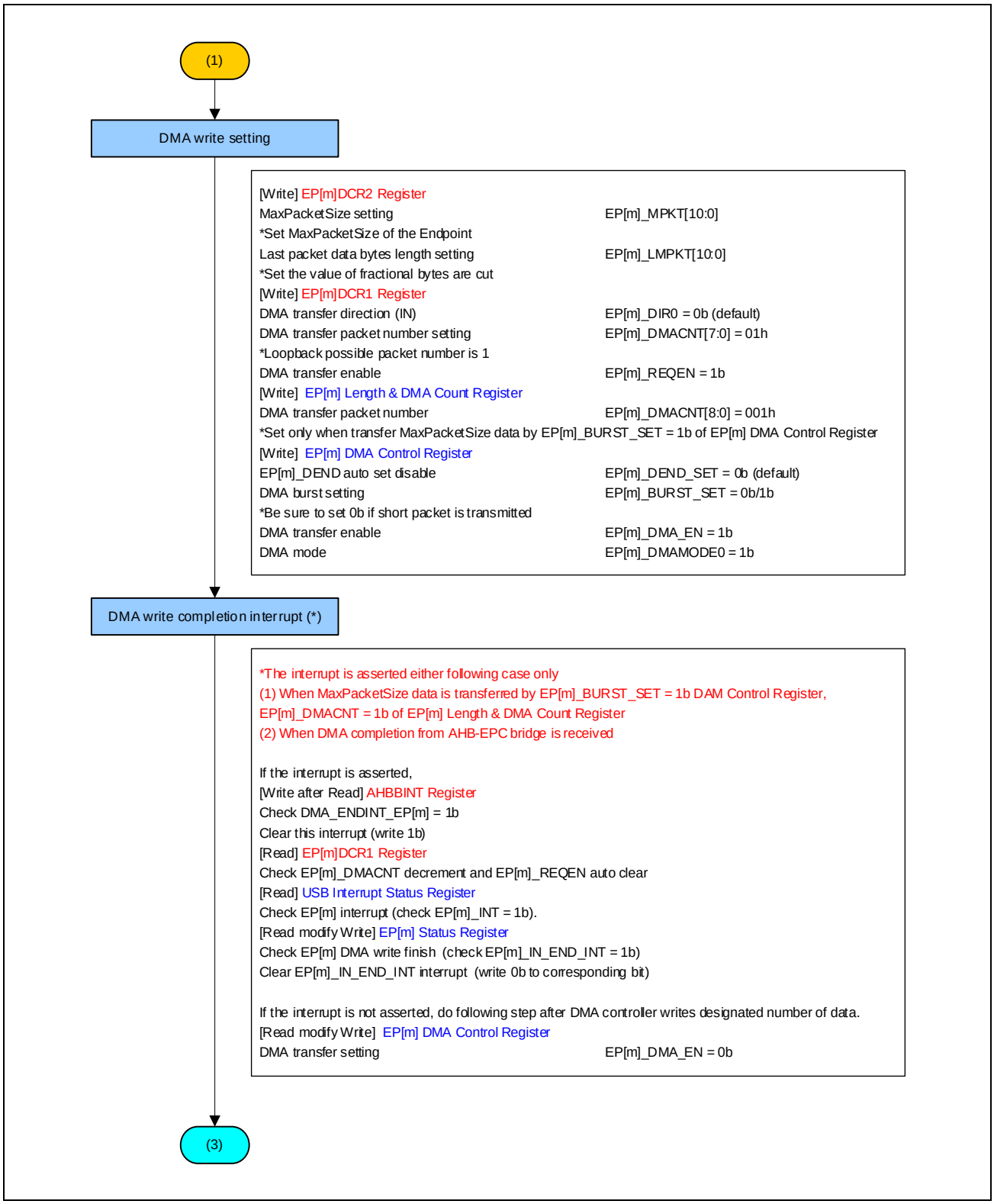


Figure 10.35 EP[m] Data Loopback (2/4)

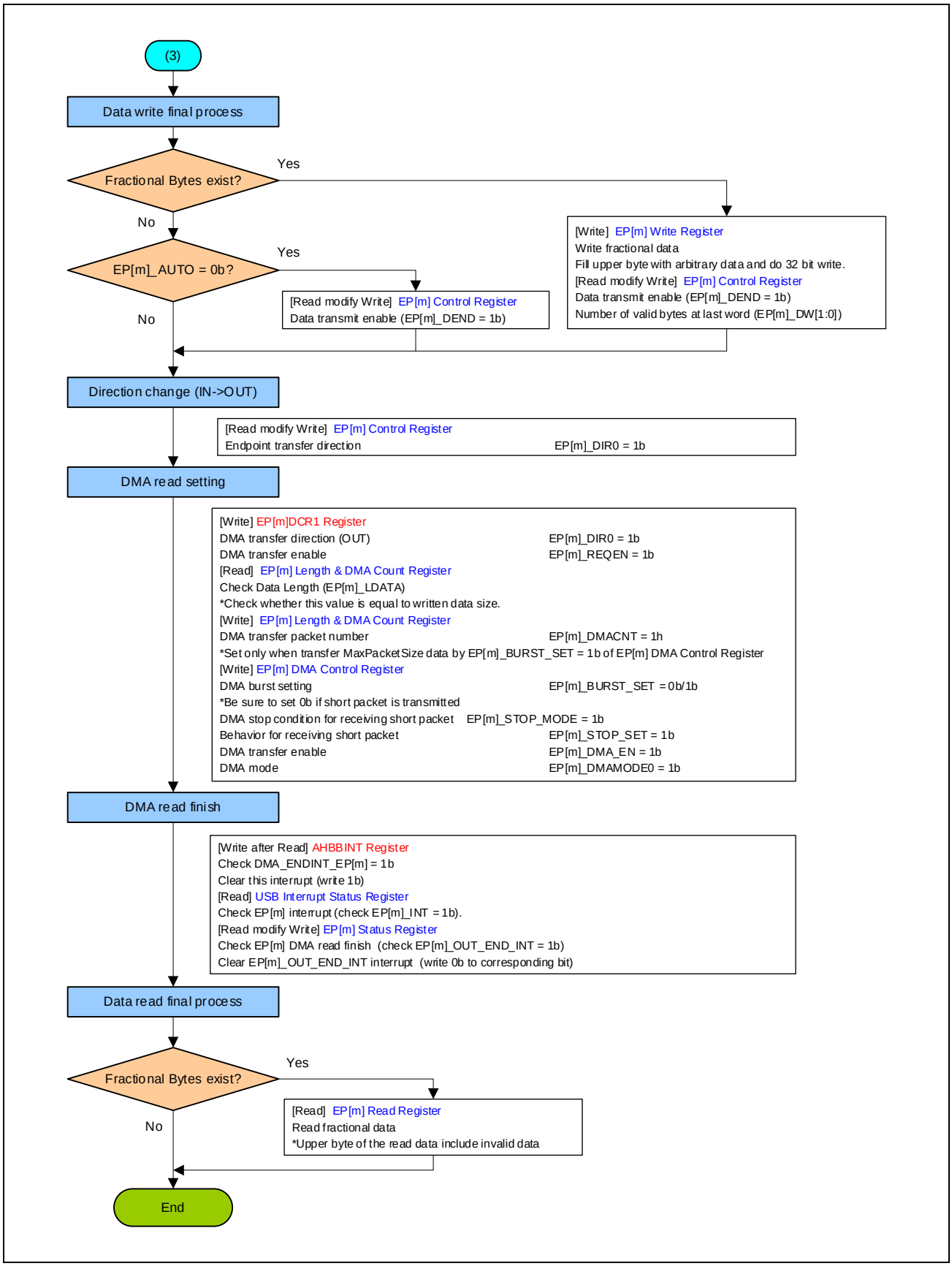
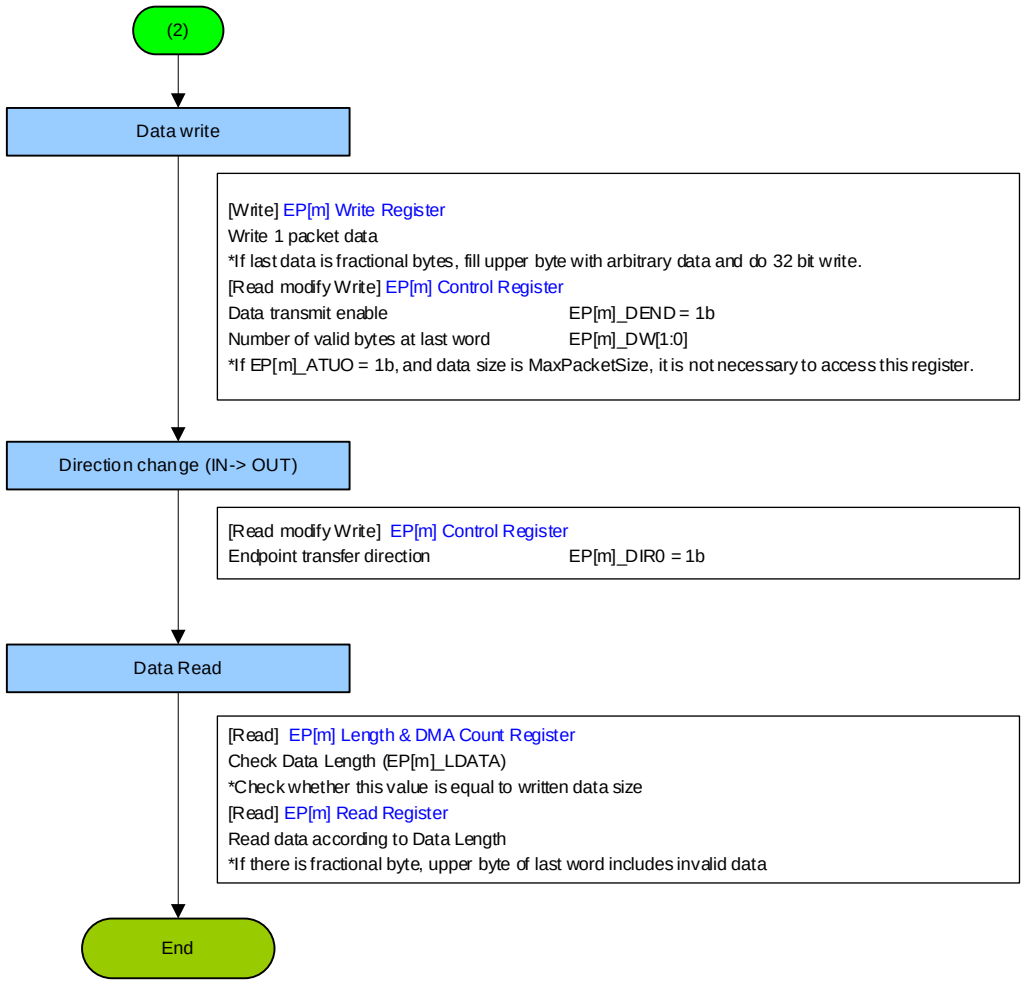


Figure 10.35 EP[m] Data Loopback (3/4)



Note 1. Be sure to clear the buffer (by writing 1b to the EP[m]_BCLR of the EP[m] Control register) before looping back the next data.

Note 2. In the above flow, the EPC initial setting is retained, so registers whose settings do not change do not need to be set.

Figure 10.35 EP[m] Data Loopback (4/4)

Section 11 DMA Controller

11.1 Overview

RZ/N1 provides two instances of DMA Controller (DMAC1 and DMAC2). Each one is able to service up to 8 independent DMA channels, for data transfers between single source and destination.

- 2 units
  - 8 channels, 16 request sources (request interfaces) for DMAC1
  - 8 channels, 16 request sources (request interfaces) for DMAC2
- Programmable DMA burst size
- Transfer width 8, 16, 32, 64 bits
- Programmable addressing, increment or decrement
- Memory-to-memory, memory-to-peripheral, and peripheral-to-memory transfers
- Internal four-word × 64 bits FIFO per channel
- Multi-block transfers achieved through:
  - Linked Lists (block chaining)
  - Auto-reloading of channel registers
  - Contiguous address between blocks
  - Independent source and destination selection of multi-block transfer method
  - Scatter/Gather

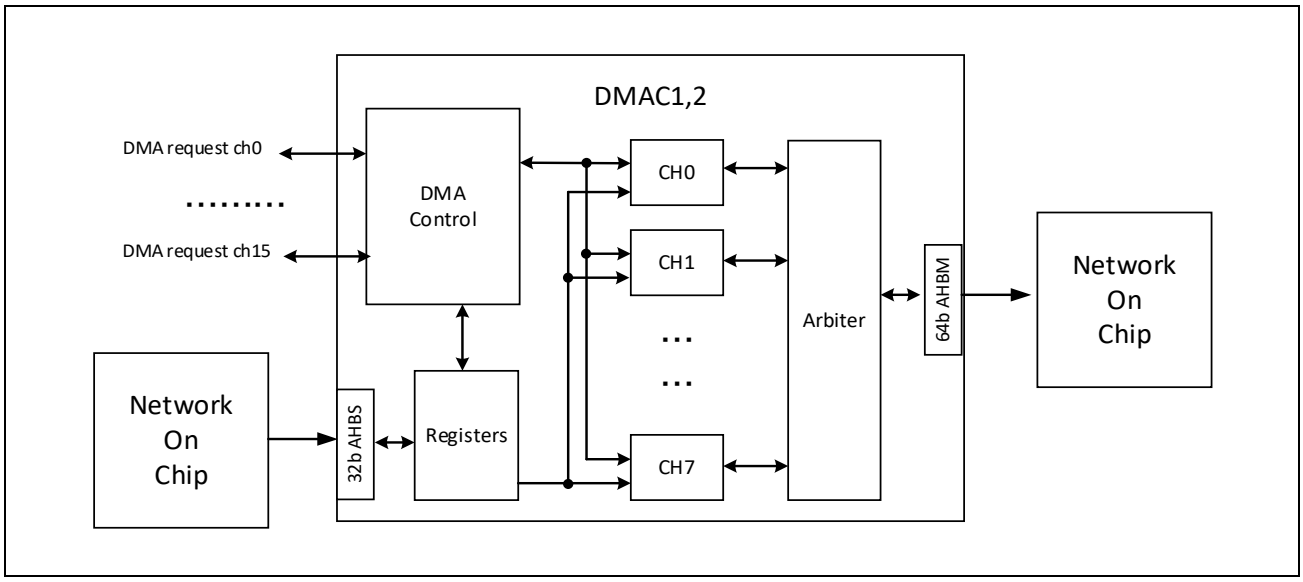


Figure 11.1 DMA Controller Interfaces and Connections

11.2 Signal Interfaces

Signal Name	Input Output	Description
Clock		
DMA[m]_HCLK	Input	Internal bus clock (AHB)
Interrupt		
DMA[m]_Int	Output	Level sensitive interrupt, Active High

Note: m = 1 or 2
Index removed style is used in this chapter.
Ex) DMA_Int

11.3 Basic Definitions

The following terms are concise definitions used throughout this chapter.

■ **Peripheral:**

- Area that requires a request interface (DMA request) to the DMAC when executing a DMA transfer.

■ **Source peripheral:**

- Device from which the DMAC reads data (area to read data by DMA request).

■ **Destination peripheral:**

- Device to which the DMAC writes the stored data (area to write data by DMA request).

■ **Memory:**

- Source or destination that is always “ready” for a DMA transfer and does not require a request interface to interact with the DMAC.

■ **Channel:**

- Read/write data path between a source and a destination.
- If the source is peripheral, then a source request interface is assigned to the channel for DMAC transfer from source.
- If the destination is peripheral, then a destination request interface is assigned to the channel for DMAC transfer to destination.
- Request interfaces can be assigned dynamically by programming the channel registers.

■ **Request interface:**

- A set of signals or software registers that controls a transfer request between the DMAC and source/destination peripheral.
- This interface is used to control a DMAC transaction.
- A channel can receive a request from either hardware or software request interface.

■ **Hardware request interface:**

- Hardware signals to control transferring a single or burst transaction between the DMAC and the source/destination peripheral.

■ **Software request interface:**

- Software registers to control transferring a single or burst transaction between the DMAC and the source/destination peripheral. Signals on the peripheral are not used.

■ **Flow controller:**

- Device that determines the DMA block transfer size and terminates it. Peripheral or DMAC can be the flow controller.
- If the peripheral can control the block transfer size, the peripheral will be the flow controller. Otherwise, the DMAC will be the flow controller.

■ **Flow control mode (CFG[n].FCMODE):**

- Special mode that only applies when the destination peripheral is the flow controller. It controls the data pre-fetching from the source peripheral.

■ **Transaction:**

- Basic data processing of a DMA transfer per one request.
- There are two types of transactions.

■ **Single transaction:**

- The transaction defined by 1 × CTL[n].SRC_TR_WIDTH/DST_TR_WIDTH.

■ **Burst transaction:**

- The transaction with a length of N defined by CTL[n].SRC_MSIZE/DEST_MSIZE, and a single transaction × N. If the flow controller is a peripheral, set the length of burst transaction to the same setting as the DMAC.

■ **Block:**

- This is the total amount of data transferred in one DMA execution, and is the same as the block transfer size set by the flow controller. It consists of a series of transactions, and when the transfer of the block size is completed, one DMA is completed. The block transfer size is the total number of single transactions transferred.

■ **Multi-block DMA transfer:**

- DMA transfer may consist of multiple DMA blocks. Multi-block DMA transfers are supported through block chaining (linked list pointers), auto-reloading channel registers, and contiguous blocks. The source and destination can independently select which method to use.

■ **Linked lists (block chaining):**

- Linked list pointer (LLP) points to the location in system memory where the next linked list item (LLI) exists. The LLI is a set of registers that describes the next block (block descriptor) and an LLP register. The DMAC fetches the LLI at the beginning of every block when block chaining is enabled.

■ **Auto-reloading:**

- The DMAC automatically reloads the channel registers to the value when the channel was first enabled at the end of each block transfer.

■ **Contiguous blocks:**

- Address between successive blocks is selected to be a continuation from the end of the previous block.

■ **Scatter:**

- Relevant to destination transfers within a block. The destination address is incremented or decremented by a programmed amount when a scatter boundary is reached.

■ **Gather:**

- Relevant to source transfers within a block. The source address is incremented or decremented by a programmed amount when a gather boundary is reached.

■ **FIFO mode:**

– Special mode to improve bus utilization. When enabled, the channel waits until the FIFO is less than half full to fetch the data from the source peripheral and waits until the FIFO is greater than or equal to half full to send data to the destination peripheral. Because of this behavior, the channel can transfer the data using bursts, which improve bus utilization ratio. When this mode is not enabled, the channel waits only until the FIFO can transmit or accept a single AHB transfer.

11.4 Register Map

11.4.1 Register Map of DMAC1

Table 11.1 Register Map of DMAC1

Address	Register Symbol	Register Name
4010 4000h + 58h × n	SAR[n] (n = 0..7)	Source Address Register for Channel [n]
4010 4008h + 58h × n	DAR[n] (n = 0..7)	Destination Address Register for Channel [n]
4010 4010h + 58h × n	LLP[n] (n = 0..7)	Linked List Pointer Register for Channel [n]
4010 4018h + 58h × n	CTL[n] (n = 0..7)	Control Register for Channel [n]
4010 4020h + 58h × n	SSTAT[n] (n = 0..7)	Source Status Register for Channel [n]
4010 4028h + 58h × n	DSTAT[n] (n = 0..7)	Destination Status Register for Channel [n]
4010 4030h + 58h × n	SSTATAR[n] (n = 0..7)	Source Status Address Register for Channel [n]
4010 4038h + 58h × n	DSTATAR[n] (n = 0..7)	Destination Status Address Register for Channel [n]
4010 4040h + 58h × n	CFG[n] (n = 0..7)	Configuration Register for Channel [n]
4010 4048h + 58h × n	SGR[n] (n = 0..7)	Source Gather Register for Channel [n]
4010 4050h + 58h × n	DSR[n] (n = 0..7)	Destination Scatter Register for Channel [n]
4010 42C0h	RawTfr	Raw Status for IntTfr Interrupt Register
4010 42C8h	RawBlock	Raw Status for IntBlock Interrupt Register
4010 42D0h	RawSrcTran	Raw Status for IntSrcTran Interrupt Register
4010 42D8h	RawDstTran	Raw Status for IntDstTran Interrupt Register
4010 42E0h	RawErr	Raw Status for IntErr Interrupt Register
4010 42E8h	StatusTfr	Status for IntTfr Interrupt Register
4010 42F0h	StatusBlock	Status for IntBlock Interrupt Register
4010 42F8h	StatusSrcTran	Status for IntSrcTran Interrupt Register
4010 4300h	StatusDstTran	Status for IntDstTran Interrupt Register
4010 4308h	StatusErr	Status for IntErr Interrupt Register
4010 4310h	MaskTfr	Mask for IntTfr Interrupt Register
4010 4318h	MaskBlock	Mask for IntBlock Interrupt Register
4010 4320h	MaskSrcTran	Mask for IntSrcTran Interrupt Register
4010 4328h	MaskDstTran	Mask for IntDstTran Interrupt Register
4010 4330h	MaskErr	Mask for IntErr Interrupt Register
4010 4338h	ClearTfr	Clear for IntTfr Interrupt Register
4010 4340h	ClearBlock	Clear for IntBlock Interrupt Register
4010 4348h	ClearSrcTran	Clear for IntSrcTran Interrupt Register
4010 4350h	ClearDstTran	Clear for IntDstTran Interrupt Register
4010 4358h	ClearErr	Clear for IntErr Interrupt Register
4010 4360h	StatusInt	Combined Interrupt Status Register
4010 4368h	ReqSrcReg	Source Software Transaction Request Register
4010 4370h	ReqDstReg	Destination Software Transaction Request Register
4010 4378h	SglRqSrcReg	Single Source Transaction Request Register
4010 4380h	SglRqDstReg	Single Destination Transaction Request Register
4010 4388h	LstSrcReg	Last Source Transaction Request Register
4010 4390h	LstDstReg	Last Destination Transaction Request Register
4010 4398h	DmaCfgReg	DMA Configuration Register
4010 43A0h	ChEnReg	DMA Controller Channel Enable Register
4010 43A8h	DmaIdReg	DMA ID Register
4010 43B0h	DmaTestReg	DMA Controller Test Register

11.4.2 Register Map of DMAC2

Table 11.2 Register Map of DMAC2

Address	Register Symbol	Register Name
4010 5000h + 58h × n	SAR[n] (n = 0..7)	Source Address Register for Channel [n]
4010 5008h + 58h × n	DAR[n] (n = 0..7)	Destination Address Register for Channel [n]
4010 5010h + 58h × n	LLP[n] (n = 0..7)	Linked List Pointer Register for Channel [n]
4010 5018h + 58h × n	CTL[n] (n = 0..7)	Control Register for Channel [n]
4010 5020h + 58h × n	SSTAT[n] (n = 0..7)	Source Status Register for Channel [n]
4010 5028h + 58h × n	DSTAT[n] (n = 0..7)	Destination Status Register for Channel [n]
4010 5030h + 58h × n	SSTATAR[n] (n = 0..7)	Source Status Address Register for Channel [n]
4010 5038h + 58h × n	DSTATAR[n] (n = 0..7)	Destination Status Address Register for Channel [n]
4010 5040h + 58h × n	CFG[n] (n = 0..7)	Configuration Register for Channel [n]
4010 5048h + 58h × n	SGR[n] (n = 0..7)	Source Gather Register for Channel [n]
4010 5050h + 58h × n	DSR[n] (n = 0..7)	Destination Scatter Register for Channel [n]
4010 52C0h	RawTfr	Raw Status for IntTfr Interrupt Register
4010 52C8h	RawBlock	Raw Status for IntBlock Interrupt Register
4010 52D0h	RawSrcTran	Raw Status for IntSrcTran Interrupt Register
4010 52D8h	RawDstTran	Raw Status for IntDstTran Interrupt Register
4010 52E0h	RawErr	Raw Status for IntErr Interrupt Register
4010 52E8h	StatusTfr	Status for IntTfr Interrupt Register
4010 52F0h	StatusBlock	Status for IntBlock Interrupt Register
4010 52F8h	StatusSrcTran	Status for IntSrcTran Interrupt Register
4010 5300h	StatusDstTran	Status for IntDstTran Interrupt Register
4010 5308h	StatusErr	Status for IntErr Interrupt Register
4010 5310h	MaskTfr	Mask for IntTfr Interrupt Register
4010 5318h	MaskBlock	Mask for IntBlock Interrupt Register
4010 5320h	MaskSrcTran	Mask for IntSrcTran Interrupt Register
4010 5328h	MaskDstTran	Mask for IntDstTran Interrupt Register
4010 5330h	MaskErr	Mask for IntErr Interrupt Register
4010 5338h	ClearTfr	Clear for IntTfr Interrupt Register
4010 5340h	ClearBlock	Clear for IntBlock Interrupt Register
4010 5348h	ClearSrcTran	Clear for IntSrcTran Interrupt Register
4010 5350h	ClearDstTran	Clear for IntDstTran Interrupt Register
4010 5358h	ClearErr	Clear for IntErr Interrupt Register
4010 5360h	StatusInt	Combined Interrupt Status Register
4010 5368h	ReqSrcReg	Source Software Transaction Request Register
4010 5370h	ReqDstReg	Destination Software Transaction Request Register
4010 5378h	SglRqSrcReg	Single Source Transaction Request Register
4010 5380h	SglRqDstReg	Single Destination Transaction Request Register
4010 5388h	LstSrcReg	Last Source Transaction Request Register
4010 5390h	LstDstReg	Last Destination Transaction Request Register
4010 5398h	DmaCfgReg	DMA Configuration Register
4010 53A0h	ChEnReg	DMA Controller Channel Enable Register
4010 53A8h	DmaIdReg	DMA ID Register
4010 53B0h	DmaTestReg	DMA Controller Test Register

11.5 Register Description

11.5.1 SAR[n] — Source Address Register for Channel [n] (n = 0..7)

The starting source address is programmed by software before the DMA channel is enabled, or by an LLI update before the start of the DMA transfer. While the DMA transfer is in progress, this register is updated to reflect the source address of the current transfer.

Address: 4010 4000h + 58h × n (DMAC1)
4010 5000h + 58h × n (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SAR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SAR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.3 SAR[n] Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b32	Reserved		R
b31 to b0	SAR	Current Source Address of DMA transfer. Updated after each source transfer. The SINC field in the CTL[n] register determines whether the address increments, decrements, or is left unchanged on every source transfer throughout the block transfer.	R/W

11.5.2 DAR[n] — Destination Address Register for Channel [n] (n = 0..7)

The starting destination address is programmed by software before the DMA channel is enabled, or by an LLI update before the start of the DMA transfer. While the DMA transfer is in progress, this register is updated to reflect the destination address of the current transfer.

Address: 4010 4008h + 58h × n (DMAC1)
4010 5008h + 58h × n (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DAR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DAR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.4 DAR[n] Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b32	Reserved		R
b31 to b0	DAR	Current Destination address of DMA transfer. Updated after each destination transfer. The DINC field in the CTL[n] register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer.	R/W

11.5.3 LLP[n] — Linked List Pointer Register for Channel [n] (n = 0..7)

This register has to be programmed to point to the first Linked List Item (LLI) in memory prior to enabling the channel if block chaining is enabled.

Address: 4010 4010h + 58h × n (DMAC1)
4010 5010h + 58h × n (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	LOC															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	LOC														—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	X

Table 11.5 LLP[n] Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b32	Reserved		R
b31 to b2	LOC	Starting Address In Memory of next LLI if block chaining is enabled. Note that the two LSBs of the starting address are not stored because the address is assumed to be aligned to a 32-bit boundary. LLI accesses are always 32-bit accesses aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit.	R/W
b1, b0	Reserved		R

11.5.4 CTL[n] — Control Register for Channel [n] (n = 0..7)

This register contains fields that control the DMA transfer. The CTL[n] register is part of the block descriptor (linked list item—LLI) when block chaining is enabled. It can be varied on a block-by-block basis within a DMA transfer when block chaining is enabled. This register has to be programmed prior to enabling the channel.

Address: 4010 4018h + 58h × n (DMAC1)
4010 5018h + 58h × n (DMAC2)

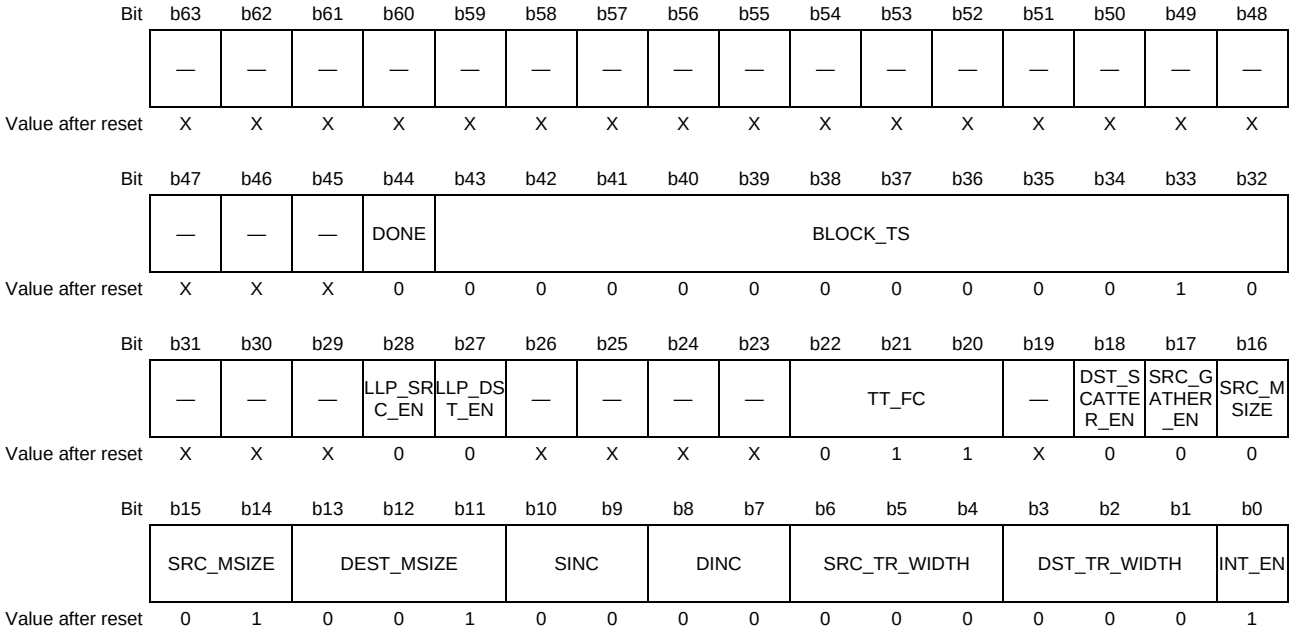


Table 11.6 CTL[n] Register Contents (1/3)

Bit Position	Bit Name	Function	R/W
b63 to b45	Reserved		R
b44	DONE	Done bit. If status write-back is enabled, the upper word of the control register, CTL[n][63:32], is written to the control register location of the Linked List Item (LLI) in system memory at the end of the block transfer with the done bit set. Software can poll the LLI CTL[n].DONE bit to see when a block transfer is complete. The LLI CTL[n].DONE bit should be cleared when the linked lists are set up in memory prior to enabling the channel. LLI accesses are always 32-bit accesses aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit.	R/W
b43 to b32	BLOCK_TS	Block Transfer Size. When DMAC is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size. The block transfer size is the total number of single transactions of the source transferred. Once the transfer starts, it is the number of data read from the source, regardless of what is the flow controller. When the source or destination peripheral is assigned as the flow controller, then the maximum block size that can be read back saturates at 4095, but the actual block size can be greater.	R/W
b31 to b29	Reserved		R
b28	LLP_SRC_EN	Block chaining is enabled on the source side only if the LLP_SRC_EN field is high and LLP[n].LOC is non-zero.	R/W

Table 11.6 CTL[n] Register Contents (2/3)

Bit Position	Bit Name	Function	R/W
b27	LLP_DST_EN	Block chaining is enabled on the destination side only if the LLP_DST_EN field is high and LLP[n].LOC is non-zero.	R/W
b26 to b23	Reserved		R
b22 to b20	TT_FC	Transfer Type and Flow Control. Flow Control can be assigned to the DMAC, the source peripheral, or the destination peripheral. 000b: Memory to Memory, DMAC as flow controller 001b: Memory to Peripheral, DMAC as flow controller 010b: Peripheral to Memory, DMAC as flow controller 011b: Reserved 100b: Peripheral to Memory, Peripheral as flow controller 101b: Reserved 110b: Memory to Peripheral, Peripheral as flow controller 111b: Reserved For multi-block transfers using linked list operation, TT_FC must be constant for all blocks of this multi-block transfer.	R/W
b19	Reserved		R
b18	DST_SCATTER_EN	Destination scatter enable bit: 0: Scatter disabled 1: Scatter enabled Scatter on the destination side is applicable only when the CTL[n].DINC bit indicates an incrementing or decrementing address control.	R/W
b17	SRC_GATHER_EN	Source gather enable bit: 0: Gather disabled 1: Gather enabled Gather on the source side is applicable only when the CTL[n].SINC bit indicates an incrementing or decrementing address control.	R/W
b16 to b14	SRC_MSIZE	Source Burst Transaction Length. The length of transaction read by one source request. Invalid for “Memory” setting. 000b: 1 001b: 4 010b: 8 011b: 16 100b: 32 101b: 64 110b: 128 111b: 256	R/W
b13 to b11	DEST_MSIZE	Destination Burst Transaction Length. The length of transaction read by one destination request. The meaning of the setting is the same as SRC_MSIZE. Invalid for “Memory” setting.	R/W
b10, b9	SINC	Source Address Increment. Indicates whether to increment or decrement the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to “No change”. 00b: Increment 01b: Decrement 1xb: No change ***Note*)** Incrementing or decrementing is done for alignment to the next CTL[n].SRC_TR_WIDTH boundary.	R/W

Table 11.6 CTL[n] Register Contents (3/3)

Bit Position	Bit Name	Function	R/W
b8, b7	DINC	Destination Address Increment. Indicates whether to increment or decrement the destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to “No change”. 00b: Increment 01b: Decrement 1xb: No change ***Note*)** Incrementing or decrementing is done for alignment to the next CTL[n].DST_TR_WIDTH boundary.	R/W
b6 to b4	SRC_TR_WIDTH	Source Transfer Width. 000b: 8 bits 001b: 16 bits 010b: 32 bits 011b: 64 bits Other: Reserved For a non-memory peripheral, typically the peripheral (source) FIFO width.	R/W
b3 to b1	DST_TR_WIDTH	Destination transfer width. Same as SRC_TR_WIDTH. For a non-memory peripheral, typically the peripheral (destination) FIFO width.	R/W
b0	INT_EN	Interrupt Enable Bit. If set, then all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel.	R/W

11.5.5 SSTAT[n] — Source Status Register for Channel [n] (n = 0..7)

This register is a temporary placeholder for the source status information on its way to the SSTAT[n] register location of the LLI. The source status information should be retrieved by software from the SSTAT[n] register location of the LLI, and not by a read of this register over the DMAC slave interface.

Address: 4010 4020h + 58h × n (DMAC1)
4010 5020h + 58h × n (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SSTAT															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SSTAT															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.7 SSTAT[n] Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b32	Reserved		R
b31 to b0	SSTAT	Source status information retrieved by hardware from the address pointed to by the contents of the SSTATAR[n] register.	R/W

11.5.6 DSTAT[n] — Destination Status Register for Channel [n] (n = 0..7)

This register is a temporary placeholder for the destination status information on its way to the DSTAT[n] register location of the LLI. The destination status information should be retrieved by software from the DSTAT[n] register location of the LLI and not by a read of this register over the DMAC slave interface.

Address: 4010 4028h + 58h × n (DMAC1)
4010 5028h + 58h × n (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DSTAT															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DSTAT															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.8 DSTAT[n] Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b32	Reserved		R
b31 to b0	DSTAT	Destination status information retrieved by hardware from the address pointed to by the contents of the DSTATAR[n] register.	R/W

11.5.7 SSTATAR[n] — Source Status Address Register for Channel [n] (n = 0..7)

After completion of each block transfer, hardware can retrieve the source status information from the user-defined address to which the contents of the SSTATAR[n] register point. The user can select any location in system memory that would provide a 32-bit value to indicate the status of the source transfer.

Address: 4010 4030h + 58h × n (DMAC1)
4010 5030h + 58h × n (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SSTATAR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SSTATAR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.9 SSTATAR[n] Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b32	Reserved		R
b31 to b0	SSTATAR	Pointer from where hardware can fetch the source status information, which is registered in the SSTAT[n] register and written out to the SSTAT[n] register location of the LLI before the start of the next block.	R/W

11.5.8 DSTATAR[n] — Destination Status Address Register for Channel [n] (n = 0..7)

After completion of each block transfer, hardware can retrieve the destination status information from the user-defined address to which the contents of the DSTATAR[n] register point. The user can select any location in system memory that would provide a 32-bit value to indicate the status of the destination transfer.

Address: 4010 4038h + 58h × n (DMAC1)
4010 5038h + 58h × n (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DSTATAR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DSTATAR															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.10 DSTATAR[n] Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b32	Reserved		R
b31 to b0	DSTATAR	Pointer from where hardware can fetch the destination status information, which is registered in the DSTAT[n] register and written out to the DSTAT[n] register location of the LLI before the start of the next block.	R/W

11.5.9 CFG[n] — Configuration Register for Channel [n] (n = 0..7)

This register contains fields that configure the DMA transfer. The channel configuration register remains fixed for all blocks of a multi-block transfer.

Address: 4010 4040h + 58h × n (DMAC1)
4010 5040h + 58h × n (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	DEST_PER				SRC_PER				SS_UPD_EN	DS_UPD_EN	—	—	—	FIFO_MODE	FCMODE
Value after reset	X	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	RELOAD_DST	RELOAD_SRC	MAX_ABRST										SRC_HS_POL	DST_HS_POL	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	HS_SEL_SRC	HS_SEL_DST	FIFO_EMPTY	CH_SUSP	CH_PRIOR			—	—	—	—	—
Value after reset	X	X	X	X	1	1	1	0	X	X	X	X	X	X	X	X

Table 11.11 CFG[n] Register Contents (1/3)

Bit Position	Bit Name	Function	R/W
b63 to b47	Reserved		R
b46 to b43	DEST_PER	Assigns a hardware request interface (0 to 15) to the destination of channel [n]. When software request interface is used, this field is ignored. The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware request interface. For correct DMAC operation, only one peripheral (source or destination) should be assigned to the same request interface.	R/W
b42 to b39	SRC_PER	Assigns a hardware request interface (0 to 15) to the source of channel [n]. When software request interface is used, this field is ignored. The channel can then communicate with the source peripheral connected to that interface through the assigned hardware request interface. For correct DMAC operation, only one peripheral (source or destination) should be assigned to the same request interface.	R/W
b38	SS_UPD_EN	Source Status Update Enable. Source status information is fetched only from the location pointed to by the SSTATAR[n] register, stored in the SSTAT[n] register and written out to the SSTAT[n] location of the LLI if SS_UPD_EN is high.	R/W
b37	DS_UPD_EN	Destination Status Update Enable. Destination status information is fetched only from the location pointed to by the DSTATAR[n] register, stored in the DSTAT[n] register and written out to the DSTAT[n] location of the LLI if DS_UPD_EN is high.	R/W
b36 to b34	Reserved	Keep the initial value.	R/W

Table 11.11 CFG[n] Register Contents (2/3)

Bit Position	Bit Name	Function	R/W
b33	FIFO_MODE	FIFO Mode Select. This mode improves system bus use efficiency. 0: The FIFO accesses the bus if single transfer is possible. 1: The FIFO accesses the bus by burst transfers whenever possible. Improves bus use efficiency.	R/W
b32	FCMODE	Flow Control Mode. Determines when source transaction requests are serviced when the Destination Peripheral is the flow controller. 0: Source transaction requests are serviced when they occur. 1: Source transaction requests are not serviced until a destination transaction request occurs. In this mode, the amount of data transferred from the source is limited so that it is guaranteed to be transferred to the destination prior to block termination by the destination.	R/W
b31	RELOAD_DST	Automatic Destination Reload. The DAR[n] register can be automatically reloaded from its initial setting value at the end of every block for multi-block transfers. A new block transfer is then initiated. For conditions under which this occurs, refer to **Table 11.46, Programming of Transfer Method and Channel Register Update Method**.	R/W
b30	RELOAD_SRC	Automatic Source Reload. The SAR[n] register can be automatically reloaded from its initial setting value at the end of every block for multi-block transfers. A new block transfer is then initiated. For conditions under which this occurs, refer to **Table 11.46, Programming of Transfer Method and Channel Register Update Method**.	R/W
b29 to b20	MAX_ABRST	Maximum Burst Length. Set the maximum burst length of the channel used for DMA transfers so that the DMAC does not occupy the bus. If this field is 0, the DMA transfer of the channel is not limited to the MAX_ABRST.	R/W
b19	SRC_HS_POL	Source Request Interface Polarity. In this LSI, only active high is valid and keep the initial value. 0: Active high 1: Active low	R/W
b18	DST_HS_POL	Destination Request Interface Polarity. In this LSI, only active high is valid and keep the initial value. 0: Active high 1: Active low	R/W
b17 to b12	Reserved		R
b11	HS_SEL_SRC	Source Software or Hardware Request Interface Select. This register selects which of the request interfaces — hardware or software — is active for source requests on this channel. 0: Hardware request interface. Software-initiated transaction requests are ignored. 1: Software request interface. Hardware-initiated transaction requests are ignored. If the source is “memory” that does not require a request interface, then this bit is ignored.	R/W
b10	HS_SEL_DST	Destination Software or Hardware Request Interface Select. This register selects which of the request interfaces — hardware or software — is active for destination requests on this channel. 0: Hardware request interface. Software-initiated transaction requests are ignored. 1: Software request interface. Hardware-initiated transaction requests are ignored. If the destination is “memory” that does not require a request interface, then this bit is ignored.	R/W

Table 11.11 CFG[n] Register Contents (3/3)

Bit Position	Bit Name	Function	R/W
b9	FIFO_EMPTY	Indicates if there is data left in the channel FIFO. Can be used in conjunction with CFG[n].CH_SUSP to disable a channel. 0: Channel FIFO not empty 1: Channel FIFO empty	R
b8	CH_SUSP	Channel Suspend. Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. Can also be used in conjunction with CFG[n].FIFO_EMPTY to disable a channel. 0: Not suspended. 1: Suspend DMA transfer from the source.	R/W
b7 to b5	CH_PRIOR	Channel priority. A priority of 7 is the highest priority, and 0 is the lowest. Reset value: 0 for Channel0… 7 for Channel7 The highest priority request is always executed. However, if a request of the same priority occurs, the lower numbered channel runs.	R/W
b4 to b0	Reserved		R

11.5.10 SGR[n] — Source Gather Register for Channel [n] (n = 0..7)

The Source Gather register contains two fields:

- Source gather count field (SGR[n].SGC) — Specifies the number of contiguous source single transactions between gather boundaries.
- Source gather interval field (SGR[n].SGI) — Specifies the source address increment/decrement in multiples of CTL[n].SRC_TR_WIDTH on a gather boundary when gather mode is enabled for the source transfer.

Address: 4010 4048h + 58h × n (DMAC1)
4010 5048h + 58h × n (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	SGC												SGI			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SGI															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.12 SGR[n] Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b32	Reserved		R
b31 to b20	SGC	Source gather count. Source contiguous transfer count between successive gather boundaries.	R/W
b19 to b0	SGI	Source gather interval.	R/W

11.5.11 DSR[n] — Destination Scatter Register for Channel [n] (n = 0..7)

The Destination Scatter register contains two fields:

- Destination scatter count field (DSR[n].DSC)—Specifies the number of contiguous destination single transactions between scatter boundaries.
- Destination scatter interval field (DSR[n].DSI)—Specifies the destination address increment/decrement in multiples of CTL[n].DST_TR_WIDTH on a scatter boundary when scatter mode is enabled for the destination transfer.

Address: 4010 4050h + 58h × n (DMAC1)
4010 5050h + 58h × n (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DSC												DSI			
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DSI															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.13 DSR[n] Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b32	Reserved		R
b31 to b20	DSC	Destination scatter count. Destination contiguous transfer count between successive scatter boundaries.	R/W
b19 to b0	DSI	Destination scatter interval.	R/W

11.5.12 RawTfr — Raw Status for IntTfr Interrupt Register

This interrupt is generated when the DMA transfer of all blocks is completed. This register has a bit allocated per channel. Each bit in this register is cleared by writing a 1 to the corresponding location in the ClearTfr. Write access is available to this register for software testing purposes only. Under normal operation, writes to this register is not recommended.

Address: 4010 42C0h (DMAC1)
4010 52C0h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	RAW							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.14 RawTfr Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	RAW	Raw Interrupt Status 0: No interrupt requested 1: Interrupt requested	R/W

11.5.13 RawBlock — Raw Status for IntBlock Interrupt Register

This interrupt is generated on DMA block transfer completion. This register has a bit allocated per channel. Each bit in this register is cleared by writing a 1 to the corresponding location in the ClearBlock. Write access is available to this register for software testing purposes only. Under normal operation, writes to this register is not recommended.

Address: 4010 42C8h (DMAC1)
4010 52C8h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	RAW							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.15 RawBlock Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	RAW	Raw Interrupt Status 0: No interrupt requested 1: Interrupt requested	R/W

11.5.14 RawSrcTran — Raw Status for IntSrcTran Interrupt Register

This interrupt is generated after completion of the last AHB transfer of the requested single/burst transaction from the request interface on the source side. This register has a bit allocated per channel. Each bit in this register is cleared by writing a 1 to the corresponding location in the ClearSrcTran. Write access is available to this register for software testing purposes only. Under normal operation, writes to this register is not recommended.

Address: 4010 42D0h (DMAC1)
4010 52D0h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	RAW							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.16 RawSrcTran Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	RAW	Raw Interrupt Status 0: No interrupt requested 1: Interrupt requested	R/W

11.5.15 RawDstTran — Raw Status for IntDstTran Interrupt Register

This interrupt is generated after completion of the last AHB transfer of the requested single/burst transaction from the request interface on the destination side. This register has a bit allocated per channel. Each bit in this register is cleared by writing a 1 to the corresponding location in the ClearDstTran. Write access is available to this register for software testing purposes only. Under normal operation, writes to this register is not recommended.

Address: 4010 42D8h (DMAC1)
4010 52D8h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	RAW							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.17 RawDstTran Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	RAW	Raw Interrupt Status 0: No interrupt requested 1: Interrupt requested	R/W

11.5.16 RawErr — Raw Status for IntErr Interrupt Register

This interrupt is generated when an ERROR response is received from an AHB slave during a DMA transfer. In addition, the DMA transfer is cancelled and the channel is disabled. This register has a bit allocated per channel. Each bit in this register is cleared by writing a 1 to the corresponding location in the ClearErr. Write access is available to this register for software testing purposes only. Under normal operation, writes to this register is not recommended.

Address: 4010 42E0h (DMAC1)
4010 52E0h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	RAW							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.18 RawErr Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	RAW	Raw Interrupt Status 0: No interrupt requested 1: Interrupt requested	R/W

11.5.17 StatusTfr — Status for IntTfr Interrupt Register

This register has a bit allocated per channel after masking. The content of this register is used to generate the interrupt signals leaving the DMA.

Address: 4010 42E8h (DMAC1)
4010 52E8h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	STATUS							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.19 StatusTfr Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	STATUS	Interrupt Status 0: No Interrupt 1: Interrupt generated	R

11.5.18 StatusBlock — Status for IntBlock Interrupt Register

This register has a bit allocated per channel after masking. The content of this register is used to generate the interrupt signals leaving the DMA.

Address: 4010 42F0h (DMAC1)
4010 52F0h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	STATUS							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.20 StatusBlock Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	STATUS	Interrupt Status 0: No Interrupt 1: Interrupt generated	R

11.5.19 StatusSrcTran — Status for IntSrcTran Interrupt Register

This register has a bit allocated per channel after masking. The content of this register is used to generate the interrupt signals leaving the DMA.

Address: 4010 42F8h (DMAC1)
4010 52F8h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	STATUS							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.21 StatusSrcTran Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	STATUS	Interrupt Status 0: No Interrupt 1: Interrupt generated	R

11.5.20 StatusDstTran — Status for IntDstTran Interrupt Register

This register has a bit allocated per channel after masking. The content of this register is used to generate the interrupt signals leaving the DMA.

Address: 4010 4300h (DMAC1)
4010 5300h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	STATUS							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.22 StatusDstTran Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	STATUS	Interrupt Status 0: No Interrupt 1: Interrupt generated	R

11.5.21 StatusErr — Status for IntErr Interrupt Register

This register has a bit allocated per channel after masking. The content of this register is used to generate the interrupt signals leaving the DMA.

Address: 4010 4308h (DMAC1)
4010 5308h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	STATUS							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.23 StatusErr Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	STATUS	Interrupt Status 0: No Interrupt 1: Interrupt generated	R

11.5.22 MaskTfr — Mask for IntTfr Interrupt Register

The content of RawTfr register is masked with the content of the MaskTfr register.

The INT_MASK[7:0] bits will be written only if the corresponding the INT_MASK_WE[7:0] bit is 1. This allows software to set a mask bit without performing a read-modified write operation.

Address: 4010 4310h (DMAC1)
4010 5310h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	INT_MASK_WE								INT_MASK							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.24 MaskTfr Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b16	Reserved		R
b15 to b8	INT_MASK_WE	Interrupt Mask Write-Enable 0: Write disabled 1: Write enabled	W
b7 to b0	INT_MASK	Interrupt Mask 0: Masked 1: Unmasked	R/W

11.5.23 MaskBlock — Mask for IntBlock Interrupt Register

The content of RawBlock register is masked with the content of the MaskBlock register.

The INT_MASK[7:0] bits will be written only if the corresponding the INT_MASK_WE[7:0] bit is 1. This allows software to set a mask bit without performing a read-modified write operation.

Address: 4010 4318h (DMAC1)
4010 5318h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	INT_MASK_WE								INT_MASK							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.25 MaskBlock Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b16	Reserved		R
b15 to b8	INT_MASK_WE	Interrupt Mask Write-Enable 0: Write disabled 1: Write enabled	W
b7 to b0	INT_MASK	Interrupt Mask 0: Masked 1: Unmasked	R/W

11.5.24 MaskSrcTran — Mask for IntSrcTran Interrupt Register

The content of RawSrcTran register is masked with the content of the MaskSrcTran register.

The INT_MASK[7:0] bits will be written only if the corresponding the INT_MASK_WE[7:0] bit is 1. This allows software to set a mask bit without performing a read-modified write operation.

Address: 4010 4320h (DMAC1)
4010 5320h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	INT_MASK_WE								INT_MASK							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.26 MaskSrcTran Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b16	Reserved		R
b15 to b8	INT_MASK_WE	Interrupt Mask Write-Enable 0: Write disabled 1: Write enabled	W
b7 to b0	INT_MASK	Interrupt Mask 0: Masked 1: Unmasked	R/W

11.5.25 MaskDstTran — Mask for IntDstTran Interrupt Register

The content of RawDstTran raw is masked with the content of the MaskDstTran register.

The INT_MASK[7:0] bits will be written only if the corresponding the INT_MASK_WE[7:0] bit is 1. This allows software to set a mask bit without performing a read-modified write operation.

Address: 4010 4328h (DMAC1)
4010 5328h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	INT_MASK_WE								INT_MASK							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.27 MaskDstTran Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b16	Reserved		R
b15 to b8	INT_MASK_WE	Interrupt Mask Write-Enable 0: Write disabled 1: Write enabled	W
b7 to b0	INT_MASK	Interrupt Mask 0: Masked 1: Unmasked	R/W

11.5.26 MaskErr — Mask for IntErr Interrupt Register

The content of RawErr register is masked with the content of the MaskErr register.

The INT_MASK[7:0] bits will be written only if the corresponding the INT_MASK_WE[7:0] bit is 1. This allows software to set a mask bit without performing a read-modified write operation.

Address: 4010 4330h (DMAC1)
4010 5330h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	INT_MASK_WE								INT_MASK							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.28 MaskErr Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b16	Reserved		R
b15 to b8	INT_MASK_WE	Interrupt Mask Write-Enable 0: Write disabled 1: Write enabled	W
b7 to b0	INT_MASK	Interrupt Mask 0: Masked 1: Unmasked	R/W

11.5.27 ClearTfr — Clear for IntTfr Interrupt Register

Each bit in the RawTfr and StatusTfr registers is cleared on the same cycle by writing a 1 to the corresponding location in the ClearTfr register. This register has a bit allocated per channel.

This register is not readable.

Address: 4010 4338h (DMAC1)
4010 5338h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	CLEAR							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.29 ClearTfr Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	CLEAR	Interrupt clear. 0: No effect 1: Clear interrupt	W

11.5.28 ClearBlock — Clear for IntBlock Interrupt Register

Each bit in the RawBlock and StatusBlock registers is cleared on the same cycle by writing a 1 to the corresponding location in the ClearBlock register. This register has a bit allocated per channel.

This register is not readable.

Address: 4010 4340h (DMAC1)
4010 5340h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	CLEAR							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.30 ClearBlock Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	CLEAR	Interrupt clear. 0: No effect 1: Clear interrupt	W

11.5.29 ClearSrcTran — Clear for IntSrcTran Interrupt Register

Each bit in the RawSrcTran and StatusSrcTran registers is cleared on the same cycle by writing a 1 to the corresponding location in the ClearSrcTran register. This register has a bit allocated per channel.

This register is not readable.

Address: 4010 4348h (DMAC1)
4010 5348h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	CLEAR							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.31 ClearSrcTran Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	CLEAR	Interrupt clear. 0: No effect 1: Clear interrupt	W

11.5.30 ClearDstTran — Clear for IntDstTran Interrupt Register

Each bit in the RawDstTran and StatusDstTran registers is cleared on the same cycle by writing a 1 to the corresponding location in the ClearDstTran register. This register has a bit allocated per channel.

This register is not readable

Address: 4010 4350h (DMAC1)
4010 5350h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	CLEAR							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.32 ClearDstTran Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	CLEAR	Interrupt clear. 0: No effect 1: Clear interrupt	W

11.5.31 ClearErr — Clear for IntErr Interrupt Register

Each bit in the RawErr and StatusErr registers is cleared on the same cycle by writing a 1 to the corresponding location in the ClearErr register. This register has a bit allocated per channel.

This register is not readable.

Address: 4010 4358h (DMAC1)
4010 5358h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	CLEAR							
Value after reset	X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

Table 11.33 ClearErr Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b8	Reserved		R
b7 to b0	CLEAR	Interrupt clear. 0: No effect 1: Clear interrupt	W

11.5.32 StatusInt — Combined Interrupt Status Register

Status for each Interrupt type. The contents of each of the five Status registers — StatusTfr, StatusBlock, StatusSrcTran, StatusDstTran, StatusErr — is ORed to produce a single bit for each interrupt type in the Combined Interrupt Status register (StatusInt).

This register is read-only.

Address: 4010 4360h (DMAC1)
4010 5360h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	ERR	DSTT	SRCT	BLOCK	TFR
Value after reset	X	X	X	X	X	X	X	X	X	X	X	0	0	0	0	0

Table 11.34 StatusInt Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b5	Reserved		R
b4	ERR	OR of the contents of StatusErr register	R
b3	DSTT	OR of the contents of StatusDstTran register	R
b2	SRCT	OR of the contents of StatusSrcTran register	R
b1	BLOCK	OR of the contents of StatusBlock register	R
b0	TFR	OR of the contents of StatusTfr register	R

11.5.33 ReqSrcReg — Source Software Transaction Request Register

A bit is assigned for each channel in this register. ReqSrcReg[n] is ignored when software request interface is not enabled for the source of channel n. The SRC_REQ[7:0] bits will be written only if the corresponding the SRC_REQ_WE[7:0] bit is 1, and if the channel is enabled in the ChEnReg register. This allows software to set a bit in the ReqSrcReg register without performing a read-modified write operation.

Address: 4010 4368h (DMAC1)
4010 5368h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SRC_REQ_WE								SRC_REQ							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.35 ReqSrcReg Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b16	Reserved		R
b15 to b8	SRC_REQ_WE	Source request Write-Enable 0: Write disabled 1: Write enabled	W
b7 to b0	SRC_REQ	Source request This bit is cleared after the transaction request completed. • When the DMAC is a flow controller Set single or burst setting before software request. 0: Single transaction 1: Burst transaction	R/W

11.5.34 ReqDstReg — Destination Software Transaction Request Register

A bit is assigned for each channel in this register. ReqDstReg[n] is ignored when software request interface is not enabled for the source of channel n. The DST_REQ[7:0] bits will be written only if the corresponding the DST_REQ_WE[7:0] bit is 1, and if the channel is enabled in the ChEnReg register.

Address: 4010 4370h (DMAC1)
4010 5370h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DST_REQ_WE								DST_REQ							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.36 ReqDstReg Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b16	Reserved		R
b15 to b8	DST_REQ_WE	Destination request Write-Enable 0: Write disabled 1: Write enabled	W
b7 to b0	DST_REQ	Destination request This bit is cleared after the transaction request completed. • When the DMAC is a flow controller Set single or burst setting before software request. 0: Single transaction 1: Burst transaction	R/W

11.5.35 SglRqSrcReg — Single Source Transaction Request Register

A bit is assigned for each channel in this register. SglRqSrcReg[n] is ignored when software request interface is not enabled for the source of channel n. The SRC_SGLREQ[7:0] bits will be written only if the corresponding the SRC_SGLREQ_WE[7:0] bit is 1, and if the channel is enabled in the ChEnReg register.

Address: 4010 4378h (DMAC1)
4010 5378h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	SRC_SGLREQ_WE								SRC_SGLREQ							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.37 SglRqSrcReg Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b16	Reserved		R
b15 to b8	SRC_SGLREQ_WE	Source single or burst request Write-Enable 0: Write disabled 1: Write enabled	W
b7 to b0	SRC_SGLREQ	Source single or burst request • When the DMAC is a flow controller Transaction request 0: Transaction is not requested 1: Transaction is requested This bit is cleared after the transaction request completed. Set transaction setting (single or burst) to the Source Software Transaction Request Register.	R/W

11.5.36 SglRqDstReg — Single Destination Transaction Request Register

A bit is assigned for each channel in this register. SglRqDstReg[n] is ignored when software request interface is not enabled for the destination of channel n. The DST_SGLREQ[7:0] bits will be written only if the corresponding the DST_SGLREQ_WE[7:0] bit is 1, and if the channel is enabled in the ChEnReg register.

Address: 4010 4380h (DMAC1)
4010 5380h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DST_SGLREQ_WE								DST_SGLREQ							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.38 SglRqDstReg Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b16	Reserved		R
b15 to b8	DST_SGLREQ_WE	Destination single or burst request Write-Enable 0: Write disabled 1: Write enabled	W
b7 to b0	DST_SGLREQ	Destination single or burst request • When the DMAC is a flow controller Transaction request 0: Transaction is not requested 1: Transaction is requested This bit is cleared after the transaction request completed. Set transaction setting (single or burst) to the Destination Software Transaction Request Register.	R/W

11.5.37 LstSrcReg — Last Source Transaction Request Register

A bit is assigned for each channel in this register. LstSrcReg[n] is ignored when software request interface is not enabled for the source of channel n, or when the source of channel n is not a flow controller. The LSTSRC[7:0] bits will be written only if the corresponding the LSTSRC_WE[7:0] bit is 1, and if the channel is enabled in the ChEnReg register.

Address: 4010 4388h (DMAC1)
4010 5388h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	LSTSRC_WE								LSTSRC							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.39 LstSrcReg Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b16	Reserved		R
b15 to b8	LSTSRC_WE	Source last transaction request Write-Enable 0: Write disabled 1: Write enabled	W
b7 to b0	LSTSRC	Source last transaction request 0: Not last transaction in current block 1: Last transaction in current block	R/W

11.5.38 LstDstReg — Last Destination Transaction Request Register

A bit is assigned for each channel in this register. LstDstReg[n] is ignored when software request interface is not enabled for the destination of channel n or when the destination of channel n is not a flow controller. The LSTDST[7:0] bits will be written only if the corresponding the LSTDST_WE[7:0] bit is 1, and if the channel is enabled in the ChEnReg register.

Address: 4010 4390h (DMAC1)
4010 5390h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	LSTDST_WE								LSTDST							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.40 LstDstReg Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b16	Reserved		R
b15 to b8	LSTDST_WE	Destination last transaction request Write-Enable 0: Write disabled 1: Write enabled	W
b7 to b0	LSTDST	Destination last transaction request 0: Not last transaction in current block 1: Last transaction in current block	R/W

11.5.39 DmaCfgReg — DMA Configuration Register

This register is used to enable the DMAC, which must be done before any channel activity can begin. If this bit is cleared while any channel is still active, then DmaCfgReg.DMA_EN still returns 1 to indicate that there are channels still active until hardware has terminated all activity on all channels, at which point the DmaCfgReg.DMA_EN bit returns 0.

Address: 4010 4398h (DMAC1)
4010 5398h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	DMA_EN
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

Table 11.41 DmaCfgReg Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b1	Reserved		R
b0	DMA_EN	DMAC Enable bit. 0: DMAC Disabled 1: DMAC Enabled.	R/W

11.5.40 ChEnReg — DMA Controller Channel Enable Register

This is the DMAC Channel Enable Register. If software needs to set up a new channel, then it can read this register in order to find out which channels are currently inactive; it can then enable an inactive channel with the required priority. The CH_EN[7:0] bits will be written only if the corresponding the CH_EN_WE[7:0] is 1. A channel CH_EN bit is cleared to 0 when the DmaCfgReg.DMA_EN is 0. When the DmaCfgReg.DMA_EN is 0, then a write to the ChEnReg register is ignored and a read will always read back 0.

Address: 4010 43A0h (DMAC1)
4010 53A0h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	CH_EN_WE								CH_EN							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.42 ChEnReg Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b16	Reserved		R
b15 to b8	CH_EN_WE	Channel enable Write-Enable 0: Write disabled 1: Write enabled	W
b7 to b0	CH_EN	Enables/Disables the channel. Setting this bit enables a channel; clearing this bit disables the channel. 0: Disable the Channel 1: Enable the Channel This bit is automatically cleared by hardware to disable the channel after the last AHB transfer of the DMA transfer to the destination has completed. Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.	R/W

11.5.41 DmaIdReg — DMA ID Register

A 32-bit value that is hardwired and read back by a read to the DMAC ID Register.

Address: 4010 43A8h (DMAC1)
4010 53A8h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DMA_ID															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DMA_ID															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0/1	1/0

Table 11.43 DmaIdReg Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b32	Reserved		R
b31 to b0	DMA_ID	DMA ID register DMAC1 → 0x1 DMAC2 → 0x2	R

11.5.42 DmaTestReg — DMA Controller Test Register

This register is used to put the AHB slave interface into test mode, during which the readback value of the writable registers match the value written, assuming the DMA configuration has not optimized the same registers.

Address: 4010 43B0h (DMAC1)
4010 53B0h (DMAC2)

Bit	b63	b62	b61	b60	b59	b58	b57	b56	b55	b54	b53	b52	b51	b50	b49	b48
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b47	b46	b45	b44	b43	b42	b41	b40	b39	b38	b37	b36	b35	b34	b33	b32
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	TEST_SLV_IF
Value after reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0

Table 11.44 DmaTestReg Register Contents

Bit Position	Bit Name	Function	R/W
b63 to b1	Reserved		R
b0	TEST_SLV_IF	Puts the AHB slave interface into test mode. In this mode, the readback value of the writable registers always matches the value written. This bit does not allow writing to read-only registers. 0: Normal mode 1: Test mode	R/W

11.6 Operation

11.6.1 DMA Transfer Mode

This chapter describes the functional details of DMA coupling between DMAC and peripherals:

The following peripheral modules are available as request interface (DMA request). A DMA transfer transaction is performed once by the source or destination request interface. Since “memory” does not require the request interface, enabling the channel will process the DMA transfer immediately.

- UARTs
- SPIs
- MSEBI Master
- Timer block
- ADC
- ETHERCAT
- SERCOSIII
- GMAC1

The main DMA features used are:

- DMAC or Peripheral flow controller
- Peripheral to Memory and Memory to Peripheral
- Single and burst transaction management
- Handshaking by DMAC signals (Peripheral or DMAC flow controller mode)

CAUTION

If the channel is disabled by software during the DMA transfer process, the data is not guaranteed. And, if a transfer is processing, it will not be disabled immediately, so check with the ChEnReg.CH_EN bit.

11.6.1.1 Flow Controller and Transfer Type

The device that controls the block transfer size is known as the flow controller. Either the DMAC, the source peripheral, or the destination peripheral must be assigned as the flow controller.

- If the peripheral (source of the request interface) cannot control the block transfer size, then the DMAC should be programmed as the flow controller. The block size should be programmed into the DMAC:CTL[n].BLOCK_TS field.

  This mode is used by following peripherals:

  - Timer blocks
  - ADC
  - ETHERCAT
  - SERCOSIII
  - GMAC1

- If the peripheral (source of the request interface) can control the block transfer size, either the source or destination peripheral must be the flow controller. The block size should be programmed into following fields:

  UART:DEST_BLOCK_SIZE/UART:SRC_BLOCK_SIZE
  SPI:DEST_BLOCK_SIZE/SPI:SRC_BLOCK_SIZE
  MSEBI:DEST_BLOCK_SIZE/MSEBI:SRC_BLOCK_SIZE

  This mode is used by following peripherals:

  - UARTs
  - SPIs
  - MSEBI Master

The DMAC:CTL[n].TT_FC field indicates the transfer type and flow controller for that channel. The table below lists valid transfer types and flow controller combinations.

Table 11.45 DMA Transfer Type and Flow Control Combinations

Transfer Type (TT_FC)	Flow Controller	Peripheral hardware request interface
Memory to Memory	DMAC	None
Memory to Peripheral	DMAC	Destination DMA request (TIMER, ADC, ETHERCAT, SERCOSIII, GMAC1)
Peripheral to Memory	DMAC	Source DMA request (TIMER, ADC, ETHERCAT, SERCOSIII, GMAC1)
Memory to Peripheral	Peripheral	Destination DMA request (UART, SPI, MSEBI)
Peripheral to Memory	Peripheral	Source DMA request (UART, SPI, MSEBI)

When using the request interface (DMA request), specify “Peripheral” as the transfer type. When you do not use the request interface, specify “Memory” as the transfer type.

CAUTION

Please refer to Bus Connection Map in Section 2, Network-On-Chip for DMAC configurable transfers.

11.6.1.2 Block Chaining Using Linked Lists

The DMA transfer can be executed by the DMAC register setting or linked lists.

The DMA transfer by linked lists reprograms the channel registers prior to the start of each block by fetching the block descriptor for that block from system memory. This is known as an LLI update.

DMAC block chaining uses a Linked List Pointer register (LLP[n]) that stores the address in memory of the next linked list item. Each LLI contains the corresponding block descriptors:

1. SAR[n]
2. DAR[n]
3. LLP[n]
4. CTL[n]
5. SSTAT[n]
6. DSTAT[n]

To set up block chaining, you program a sequence of Linked Lists in memory.

LLI accesses are always 32-bit accesses aligned to 32-bit boundaries.

The SAR[n], DAR[n], LLP[n], and CTL[n] registers are fetched from system memory on an LLI update. The updated contents of the CTL[n], SSTAT[n], and DSTAT[n] registers are written back to memory on block completion. Following Figure shows how you use chained linked lists in memory to define multi-block transfers using block chaining.

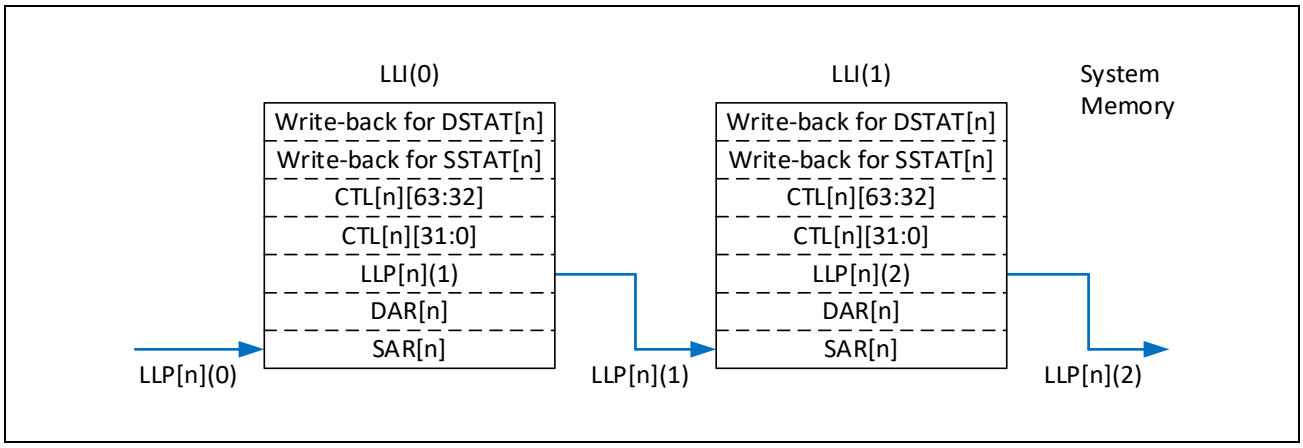


Figure 11.2 Multi-Block Transfer Using Linked Lists

NOTE

In this chapter, the SAR[n], DAR[n], LLP[n], CTL[n], SSTAT[n], and DSTAT[n] uses the following notation to distinguish those from the corresponding DMAC registers.

LLI.SAR[n], LLI.DAR[n], LLI.LLP[n], LLI.CTL[n], LLI.SSTAT[n], and LLI.DSTAT[n].

Table 11.46 Programming of Transfer Method and Channel Register Update Method

	Transfer Method	LLP.LOC = 0	LLP_SRC_EN (CTL[n])	RELOAD_SRC (CFG[n])	LLP_DST_EN (CTL[n])	RELOAD_DST (CFG[n])	CTL[n], LLP[n] Update Method	SAR[n] Update Method	DAR[n] Update Method	Write Back
1.	Single-block or last transfer of multi-block.	Yes	0	0	0	0	None, user reprograms	None (single)	None (single)	No
2.	Auto-reload multi-block transfer with contiguous SAR	Yes	0	0	0	1	CTL[n], LLP[n] are reloaded from initial values.	Contiguous	Auto-Reload	No
3.	Auto-reload multi-block transfer with contiguous DAR.	Yes	0	1	0	0	CTL[n], LLP[n] are reloaded from initial values	Auto-Reload	Contiguous	No
4.	Auto-reload multi-block transfer	Yes	0	1	0	1	CTL[n], LLP[n] are reloaded from initial values	Auto-Reload	Auto-Reload	No
5.	Single-block or last transfer of multi-block.	No	0	0	0	0	None, user reprograms	None (single)	None (single)	Yes
6.	Linked list multi-block transfer with contiguous SAR	No	0	0	1	0	CTL[n], LLP[n] loaded from next Linked List item.	Contiguous	Linked List	Yes
7.	Linked list multi-block transfer with auto-reload SAR	No	0	1	1	0	CTL[n], LLP[n] loaded from next Linked List item.	Auto-Reload	Linked List	Yes
8.	Linked list multi-block transfer with contiguous DAR	No	1	0	0	0	CTL[n], LLP[n] loaded from next Linked List item.	Linked List	Contiguous	Yes
9.	Linked list multi-block transfer with auto-reload DAR	No	1	0	0	1	CTL[n], LLP[n] loaded from next Linked List item.	Linked List	Auto-Reload	Yes
10.	Linked list multi-block transfer	No	1	0	1	0	CTL[n], LLP[n] loaded from next Linked List item.	Linked List	Linked List	Yes

Transfer Method 6 to 10 of above table shows the required values of LLP[n], CTL[n], and CFG[n] for multi-block DMA transfers using block chaining.

NOTE

For Transfer Method 6 to 10, the LLI.CTL[n], LLI.LLP[n], LLI.SAR[n], and LLI.DAR[n] register locations of the LLI are always affected at the start of every block transfer. The LLI.LLP[n] and LLI.CTL[n] locations are always used to reprogram the DMAC LLP[n] and CTL[n] registers. However, the LLI.SAR[n]/LLI.DAR[n] is used to update the DMAC SAR[n]/DAR[n] only when using a linked list.

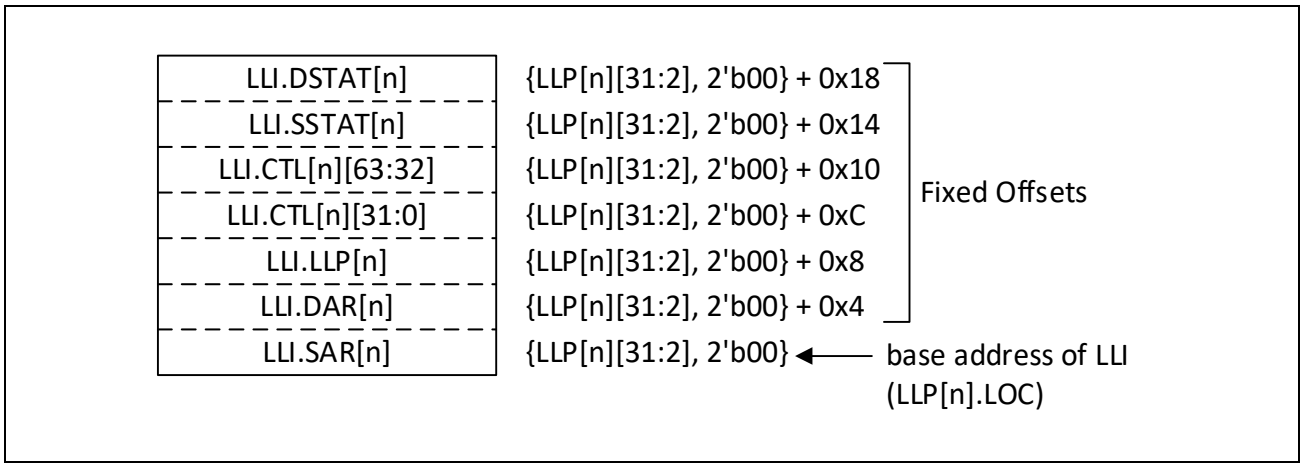


Figure 11.3 Mapping of Block Descriptor (LLI) in Memory to Channel Registers

(1) Auto-Reloading of Channel Registers

During auto-reloading, the channel registers are reloaded with their initial values at the completion of each block and the new values used for the new block. Depending on Transfer Method, some or all of the SAR[n], DAR[n], and CTL[n] channel registers are reloaded from their initial value at the start of a block transfer.

(2) Contiguous Address Between Blocks

In this case, the address between successive blocks is selected as a continuation from the end of the previous block.

Without automatic reloading and updating from the linked list, successive source or destination addresses will be set according to CTL [n].SINC/CTL [n].DINC.

For source address: CTL[n].LLP_SRC_EN = 0 and CFG[n].RELOAD_SRC = 0

For destination address: CTL[n].LLP_DST_EN = 0 and CFG[n].RELOAD_DST = 0

(3) Suspension of Transfers Between Blocks

For Transfer Method 6, 8 and 10 of **Table 11.46**, the DMA transfer does not stall between block transfers. For example, at the end-of-block N, the DMAC automatically proceeds to block N + 1.

For Transfer Method 2, 3, 4, 7 and 9 of **Table 11.46** (SAR[n] and/or DAR[n] auto-reloaded between block transfers), the DMA transfer automatically stalls after the end-of-block interrupt is asserted.

At the end of every block transfer, an end-of-block interrupt is asserted if:

1. Interrupts are enabled, CTL[n].INT_EN = 1.
2. The channel block interrupt is unmasked, MaskBlock.INT_MASK[n] = 1.

The DMAC does not proceed to the next block transfer until a write to the ClearBlock[n] register, done by software to clear the channel block-complete interrupt, is detected by hardware.

(4) Ending Multi-Block Transfers

All multi-block transfers must end as shown in either Transfer Method 1 or 5 of **Table 11.46**. At the end of every block transfer, the DMAC samples the Transfer Method, and if the DMAC is in the Transfer Method 1 or 5 state, then the previous block transferred was the last block and the DMA transfer is terminated.

When to use automatic reload with Transfer Method 2, 3 and 4 of **Table 11.46** (LLP[n].LOC = 0 and CFG[n].RELOAD_SRC and/or CFG[n].RELOAD_DST is set), automatic reload must be disabled before the last block transfer. When the end-of-block interrupt of the previous block of the last block is generated, CFG [n].RELOAD_SRC and CFG [n].RELOAD_DST should be programmed to 0 to hold the block transfer. This puts the DMAC into the Transfer Method 1 state.

For Transfer Method 6, 8 and 10 of **Table 11.46** (both CFG[n].RELOAD_SRC and CFG[n].RELOAD_DST cleared), the user must set up the last block descriptor in memory so that both LLI.CTL[n].LLP_SRC_EN and LLI.CTL[n].LLP_DST_EN are 0.

The sampling of the LLP[n].LOC takes place exclusively at the beginning of the transfer when the channel is enabled. This determines whether writeback is enabled throughout the complete transfer, and changing the value in subsequent blocks does not have any effect.

11.6.1.3 Basic Interface Definitions

The following equations below, refer to the decoded values of the parameters.
For example:

CTL[n].SRC_MSIZE: Burst Transaction Length 1–256
CTL[n].SRC_TR_WIDTH: Transfer Width 8/16/32/64 bits

- PERIPHERAL:SRC_BURST_SIZE, refers PERIPHERAL register SRC_BURST_SIZE
  (Initialization required only for peripheral flow controller like UART, SPI, MSEBI)

The following definitions are used in this chapter:

- Source single transaction size in bytes

  (1) src_single_size_bytes = DMAC:CTL[n].SRC_TR_WIDTH/8

- Source burst transaction size in bytes

  (2) src_burst_size_bytes = DMAC:CTL[n].SRC_MSIZE × src_single_size_bytes
  = PERIPHERAL:SRC_BURST_SIZE × src_single_size_bytes

- Destination single transaction size in bytes

  (3) dst_single_size_bytes = DMAC:CTL[n].DST_TR_WIDTH/8

- Destination burst transaction size in bytes

  (4) dst_burst_size_bytes = DMAC:CTL[n].DEST_MSIZE × dst_single_size_bytes
  = PERIPHERAL:DEST_BURST_SIZE × dst_single_size_bytes

- Burst size in bytes:

  With the DMAC as the flow controller

  – The length of a DMA burst transaction is programmed into the DMAC:CTL[n].SRC_MSIZE/DMAC:CTL[n].DEST_MSIZE.

  With the Peripheral as the flow controller

  – The length of a DMA burst transaction is programmed into the DMAC:CTL[n].SRC_MSIZE/DMAC:CTL[n].DEST_MSIZE.

  – The length of a Peripheral burst transaction is programmed into the PERIPHERAL:DEST_BURST_SIZE/PERIPHERAL:SRC_BURST_SIZE field of each peripheral used.

  – The peripheral registers are
  UART:DEST_BURST_SIZE/UART:SRC_BURST_SIZE,
  SPI:DEST_BURST_SIZE/SPI:SRC_BURST_SIZE,
  MSEBI:DEST_BURST_SIZE/MSEBI:SRC_BURST_SIZE.

- Block size in bytes:

  With the DMAC as the flow controller:

  – The processor programs the DMAC with the number of data items (block size) of source transfer width (DMAC:CTL[n].SRC_TR_WIDTH) to be transferred by the DMAC in a block transfer, this is programmed into the DMAC:CTL[n].BLOCK_TS field.

  Therefore, the total number of bytes to be transferred in a block is:

  (5a) blk_size_bytes_dma = DMAC:CTL[n].BLOCK_TS × src_single_size_bytes

  With the Peripheral as the flow controller:

  – The block size registers of the peripheral are
  UART:DEST_BLOCK_SIZE/UART:SRC_BLOCK_SIZE,
  SPI:DEST_BLOCK_SIZE/SPI:SRC_BLOCK_SIZE,
  MSEBI:DEST_BLOCK_SIZE/MSEBI:SRC_BLOCK_SIZE.

  Therefore, the total number of bytes to be transferred in a block is:

  When source peripheral is the flow controller:

  (5b) blk_size_bytes_dma = PERIPHERAL:SRC_BLOCK_SIZE × src_single_size_bytes

  When destination peripheral is the flow controller:

  (5b) blk_size_bytes_dma = PERIPHERAL:DEST_BLOCK_SIZE × dst_single_size_bytes

- Block of DMA data.

  – The amount of which is the block transfer size and is determined by the flow controller. For transfers between the DMAC and memory or peripheral, a block is broken directly into a sequence of bursts and single transactions.

- Transaction

Length of a transaction is programmed into the DMAC:CTL[n].SRC_MSIZE/DMAC:CTL[n].DEST_MSIZE and also in the PERIPHERAL:DEST_BURST_SIZE/PERIPHERAL:SRC_BURST_SIZE (Peripheral flow controller only). The burst transaction is converted into a sequence of bursts and single AHB transfers.

DMAC executes each burst transfer no longer than the maximum burst size set (SRC_MSIZE/DEST_MSIZE and MAX_ABRST). The burst transaction length is under program control. Normally, set an appropriate value according to the DMAC FIFO size and the FIFO watermark level of the peripheral to be transferred.

CAUTION

For peripheral flow controller, the burst size transaction must have the same values on DMAC and Peripherals (UART, SPI or MSEBI only).

11.6.1.4 Transaction Examples

(1) DMA in Burst Transaction Mode Only

The device that controls the block transfer size is known as the flow controller.

- If the flow controller is DMAC, the block transfer size should be programmed into the DMAC:CTL[n].BLOCK_TS field. If the flow controller is peripheral, it should be programmed into the PERIPHERAL:SRC_BLOCK_SIZE/PERIPHERAL:DST_BLOCK_SIZE.
- The DMAC:CTL[n].TT_FC field indicates the transfer type and flow controller for that channel. **Table 11.45** lists valid transfer types and flow controller combinations.

As an example, when the DMAC is the flow controller, the DMAC tries to transfer the data using burst transactions and, when possible, fill or empty the channel FIFO in single bursts, as shown below.

A total of 48 bytes are transferred in the block; that is, blk_size_bytes_dma = 48. As shown below, this block transfer consists of three bursts of length 4 from the source, interleaved with three bursts, again of length 4, to the destination.

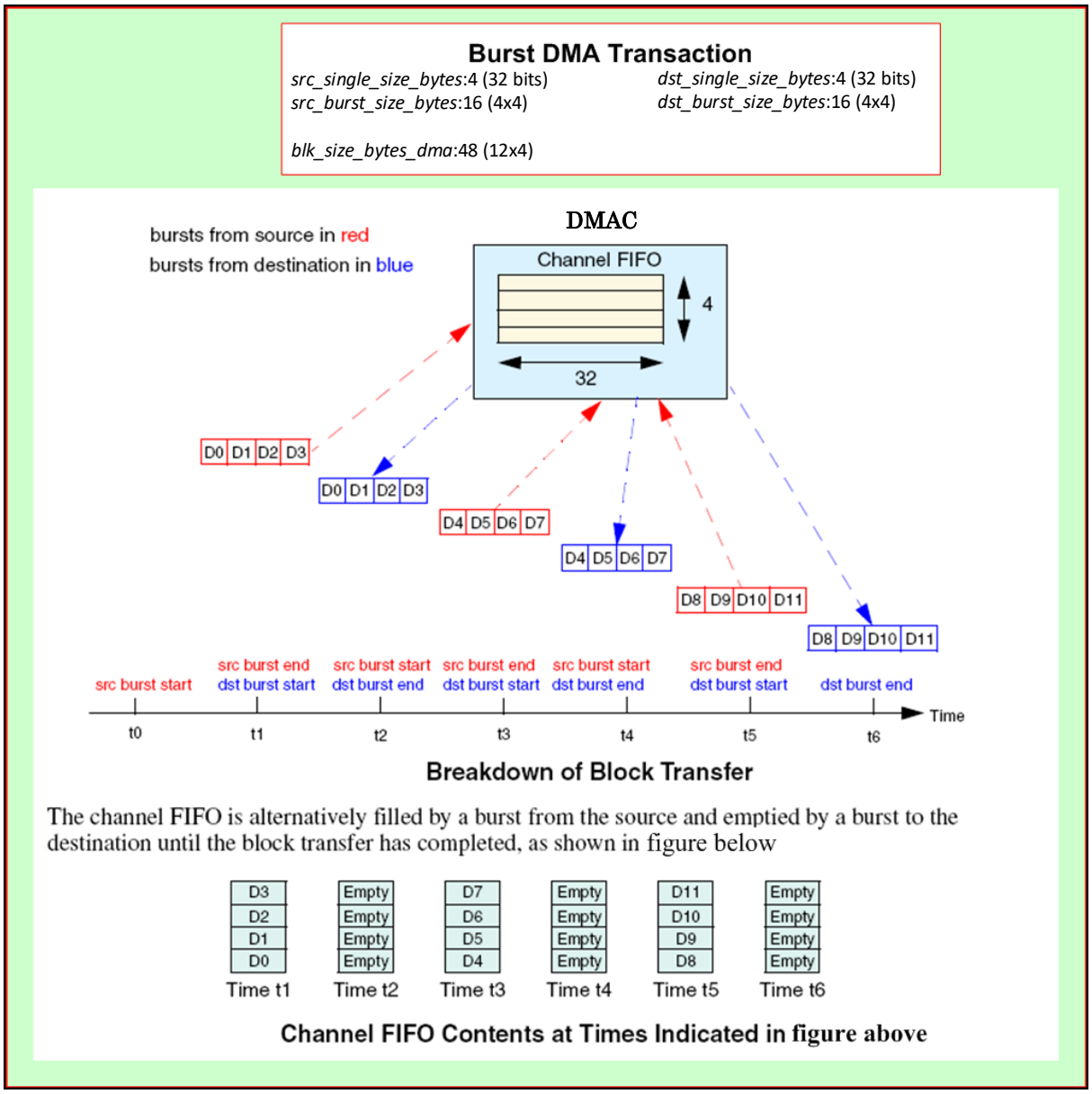


Figure 11.4 Burst DMA Transaction

(2) DMA in Burst and Single Transaction Mode at the Same Time

There are cases where a DMA block transfer cannot be completed using only burst transactions. Typically this occurs when the block size is not a multiple of the burst transaction length.

In these cases, the transfer uses burst transactions up to the point where the amount of data left to complete the block is less than the amount of data in a burst transaction. At this point, the Peripheral (UART, SPI or MSEBI) controls DMA and completes the block transfer using single transaction request.

The Peripheral asserts the single transaction request to indicate to the DMAC that there is enough data or space to complete a single transaction from or to the source/destination peripheral.

The Single Transaction Region is the time interval where the Peripheral (UART, SPI or MSEBI) uses single transactions to complete the block transfer, burst transactions are exclusively used outside this region.

The precise definition of when this region is entered:

- The source peripheral enters the Single Transaction Region when the number of bytes left to complete in the source block transfer is less than src_burst_size_bytes.
  - If blk_size_bytes_dma/src_burst_size_bytes = integer
    then the source never enters this region, and the source block uses only burst transactions.
- The destination peripheral enters the Single Transaction Region when the number of bytes left to complete in the destination block transfer is less than dst_burst_size_bytes.
  - If blk_size_bytes_dma/dst_burst_size_bytes = integer
    then the destination never enters this region, and the destination block uses only burst transactions.

The example below demonstrates how a block from the source can be completed using a series of single transactions. It also demonstrates how the watermark level that triggers a burst request in the source peripheral.

Consider the case where the watermark level that triggers a source burst request in the source peripheral is equal to DMAC:CTL[n].SRC_MSIZE = PERIPHERAL:SRC_BURST_SIZE = 8, that is, eight entries or more need to be in the source peripheral FIFO in order to trigger a burst request. The figure shows how this block transfer is broken into burst and single transactions.

When the amount of data left to transfer to complete the source block is less than src_burst_size_bytes, the source Peripheral (UART, SPI, MSEBI), knowing the block transfer size, requests single transactions regardless the watermark level. This region, where the amount of data left to transfer in the source block is less than src_burst_size_bytes, is known as the Single Transaction Region.

In the Single Transaction Region, the source Peripheral requests single transaction to the DMAC until the source block transfer has completed.

In this example, the source Peripheral completes the source block transfer using four single transactions to the DMAC.

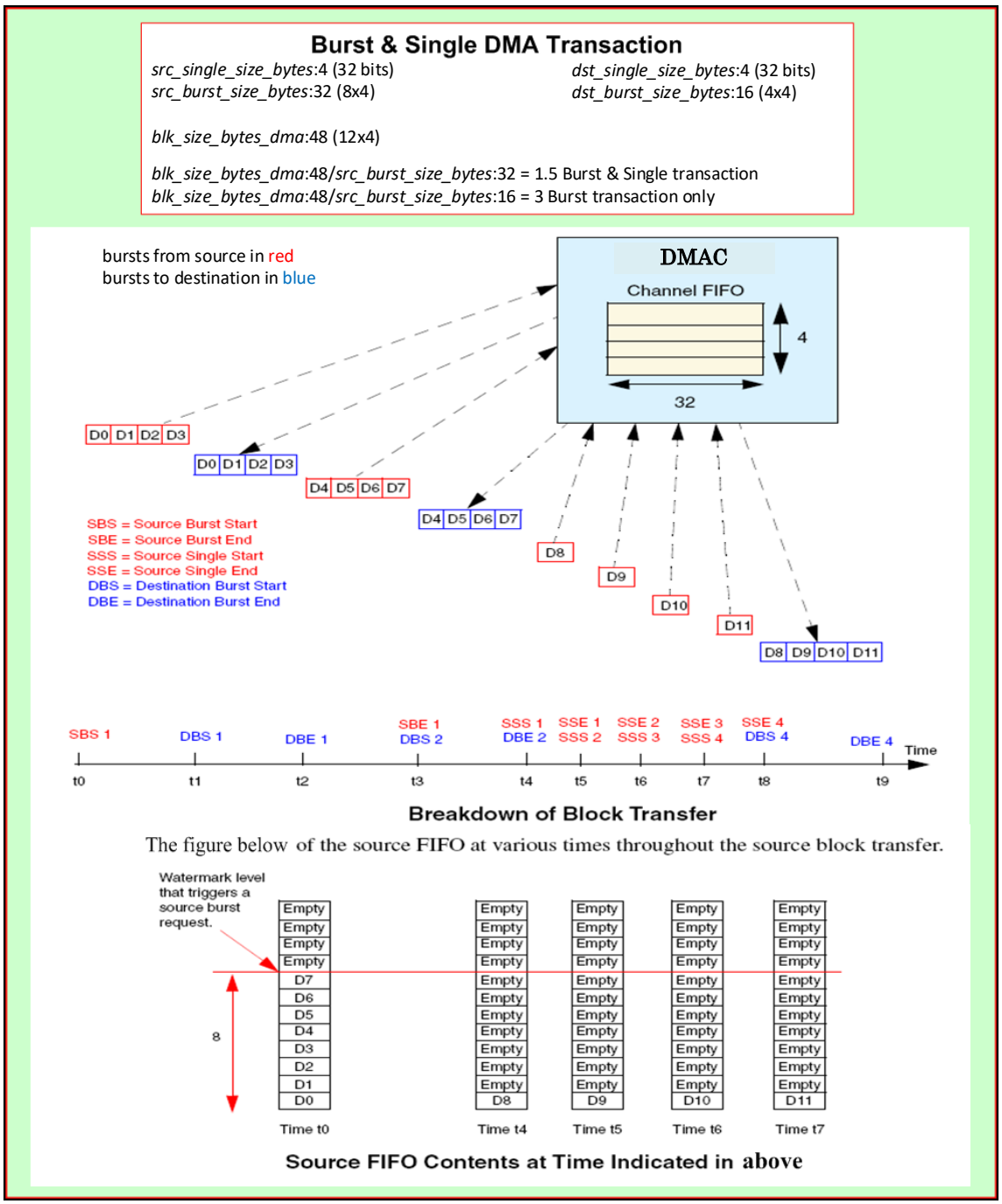


Figure 11.5 Burst and Single DMA Transaction

11.6.1.5 DMAC Programing Example

The flow diagram shows an overview of DMA programming example.

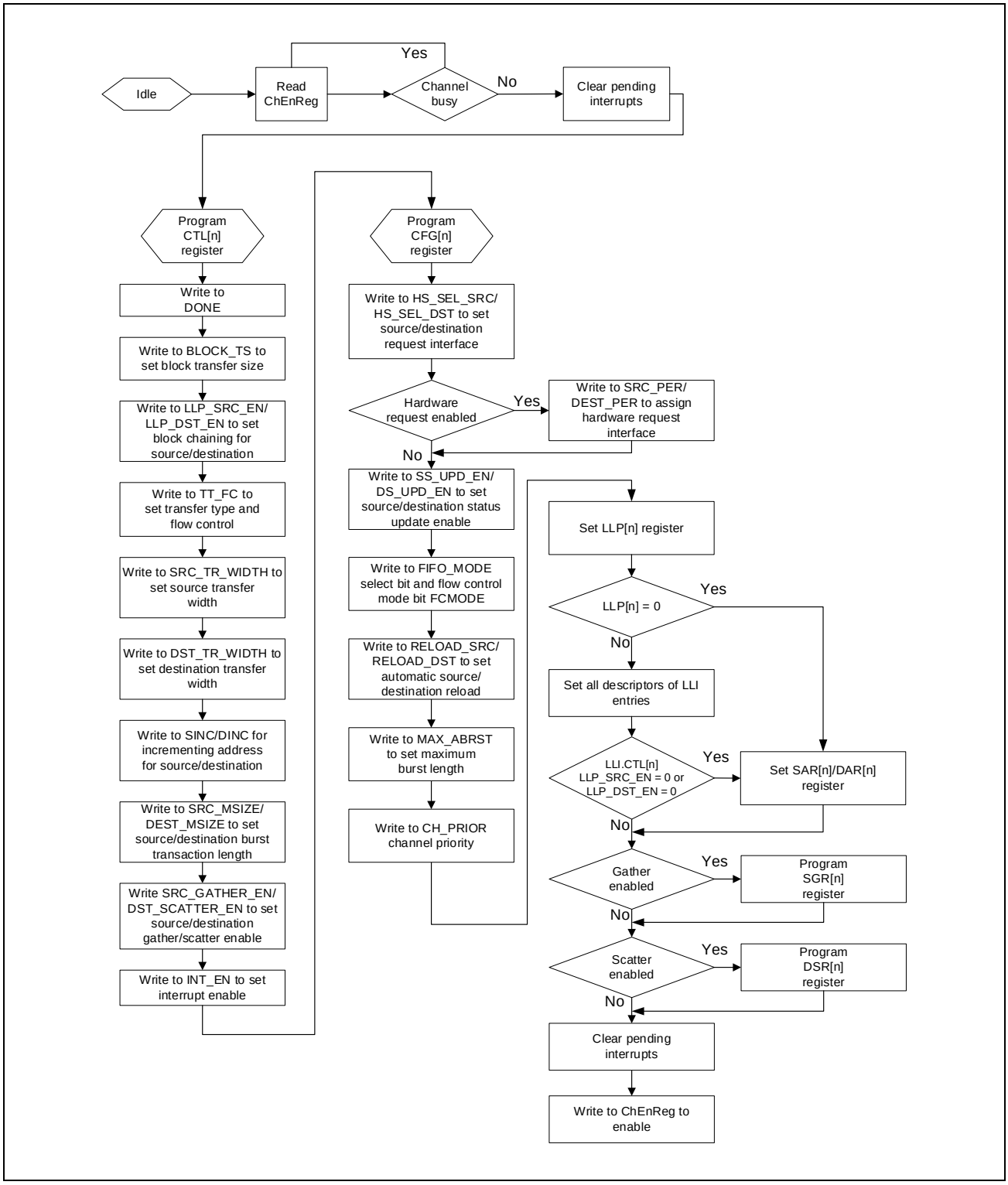


Figure 11.6 Flowchart for DMA Programming Example

11.6.2 DMA Request Allocation

Each DMAC1 and DMAC2 provides 16 peripheral hardware request interface. There are 64 DMA request in the system. To connect total 32 hardware request interface to 64 DMA request, the additional multiplexers exist. It is managed by dedicated CFG_DMAMUX registers in System Controller. The table below describes the details.

Table 11.47 DMA Multiplexing Channel Request Signals

Hardware Request Interface	Peripheral DMA Request	
DMAC1	CFG_DMAMUX[15:0] in System Controller, dedicated multiplexing bit to manage	
Register bit:	1'b1:	1'b0:
Request interface0	UART4 RX	CAT_SYNC0 or SERCOS3_Int[0]*1
Request interface1	UART4 TX	CAT_SYNC1 or SERCOS3_Int[1]*1
Request interface2	UART5 RX	MAC_PPS[0]*1
Request interface3	UART5 TX	MAC_PPS[1]*1
Request interface4	UART6 RX	MAC_TRIG[1]*1
Request interface5	UART6 TX	S3_CONCLK*1
Request interface6	UART7 RX	S3_DIVCLK*1
Request interface7	UART7 TX	TIMER1_SubTimer6
Request interface8	SPI1 RX	CAT_SYNC0 or SERCOS3_Int[0]*1
Request interface9	SPI1 TX	CAT_SYNC1 or SERCOS3_Int[1]*1
Request interface10	SPI2 RX	MAC_PPS[0]*1
Request interface11	SPI2 TX	MAC_PPS[1]*1
Request interface12	SPI3 RX	MAC_TRIG[1]*1
Request interface13	SPI3 TX	S3_CONCLK*1
Request interface14	SPI4 RX	S3_DIVCLK*1
Request interface15	SPI4 TX	TIMER1_SubTimer7
DMAC2	CFG_DMAMUX[31:16] in System Controller, dedicated multiplexing bit to manage	
Register bit:	1'b1:	1'b0:
Request interface0	SPI5 RX	CAT_SYNC0 or SERCOS3_Int[0]*1
Request interface1	SPI5 TX	CAT_SYNC1 or SERCOS3_Int[1]*1
Request interface2	SPI6 RX	MAC_PPS[0]*1
Request interface3	SPI6 TX	MAC_PPS[1]*1
Request interface4	UART8 RX	MAC_TRIG[1]*1
Request interface5	UART8 TX	S3_CONCLK*1
Request interface6	Not used	S3_DIVCLK*1
Request interface7	Not used	TIMER2_SubTimer6
Request interface8	Not used	CAT_SYNC0 or SERCOS3_Int[0]*1
Request interface9	Not used	CAT_SYNC1 or SERCOS3_Int[1]*1
Request interface10	MSEBIM0 (RX on CS0_N)	MAC_PPS[0]*1
Request interface11	MSEBIM1 (TX on CS0_N)	MAC_PPS[1]*1
Request interface12	MSEBIM2 (RX on CS1_N)	MAC_TRIG[1]*1
Request interface13	MSEBIM3 (TX on CS1_N)	S3_CONCLK*1
Request interface14	ADC ch0	S3_DIVCLK*1
Request interface15	ADC ch1	TIMER2_SubTimer7

Note 1. CAT_SYNC0 or SERCOS3_Int[0] is selected by DMACTRL in Ethernet Accessory registers.
CAT_SYNC1 or SERCOS3_Int[1] is selected by DMACTRL in Ethernet Accessory registers.
7 DMA requests from Ethernet Peripherals are assigned to the request interface 4 times, only one request should be used at the same time.

11.6.3 Illegal Register Access

An illegal access can be any of the following:

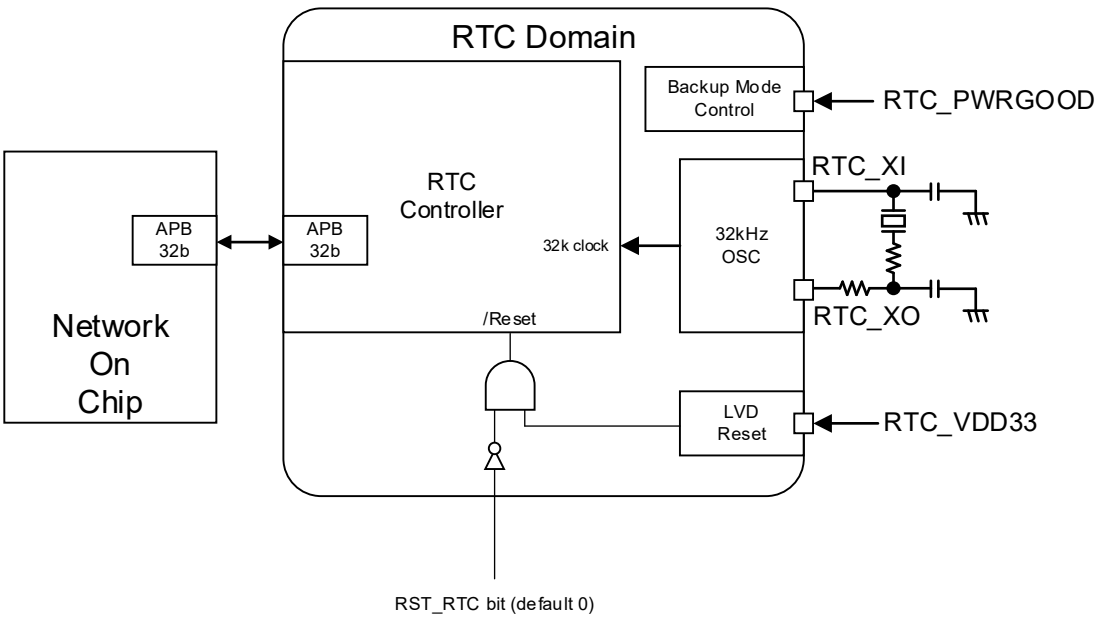
- A write to the SAR[n], DAR[n], LLP[n], CTL[n], SSTAT[n], DSTAT[n], SSTATAR[n], DSTATAR[n], SGR[n] or DSR[n] registers occur when the channel is enabled.
- A read from the ClearBlock, ClearDstTran, ClearErr, ClearSrcTran, ClearTfr is attempted.
- A write to the StatusBlock, StatusDstTran, StatusErr, StatusSrcTran, StatusTfr is attempted.
- A write to the StatusInt register is attempted.
- A write to the DmaIdReg register is attempted.

An illegal access (read/write) returns an error response on AHB bus.

Section 12 RTC

12.1 Overview

The RTC keeps track of the real time of day. It also functions as an alarm and a calendar. The time is displayed in 24-hour format, and time/calendar values are stored in binary-coded decimal format.



Note: LVD reset outputs 0 when RTC_VDD33 becomes low voltage

Figure 12.1 RTC Interfaces and Connections

- Time-of-day clock in 24-hour mode
- Calendar
- Alarm capability
- Interrupt capability
  - 1second
  - Fixed interval
  - Alarm
- XTAL 32 kHz
- Separate and isolated power supply for backup mode
  - Self-isolation mode, which allows RTC to work even if power is not supplied to the rest of the device.
  - RTC is reset to the default value when it detects a backup voltage failure

12.2 Signal Interfaces

Signal Name	Input Output	Description
RTC_PCLK	Input net	Clock for RTC APB interface and NoC interconnect
Reset		
RSTN_FW_RTC	--	Software reset to the interconnect for RTC
RST_RTC	Input	Active high reset to RTC module
Interrupt		
RTCATINTAL_Int	Output	Alarm interrupt, level sensitive, Active High*1
RTCATINTR_Int	Output	Fixed interval interrupt, level sensitive, Active High*1
RTCATINT1S_Int	Output	1second interrupt, level sensitive, Active High*1
External Signal		
RTC_PWRGOOD	input	RTC backup mode control (dedicated pin)

Note: RTC_PCLK, RSTN_FW_RTC and RST_RTC are controlled by PWRCTRL_RTC register of System Control.

Note 1. Interrupt output for one clock of 32k clock

12.3 Register Map

Table 12.1 Register Map of RTC

Address	Register Symbol	Register Name
4000 6000h	RTCA0CTL0	RTC Control Register 0
4000 6004h	RTCA0CTL1	RTC Control Register 1
4000 6008h	RTCA0CTL2	RTC Control Register 2
4000 600Ch	RTCA0SUBC	RTC Sub Count Register
4000 6010h	RTCA0SRBU	RTC Sub Count Register Read Buffer
4000 6014h	RTCA0SEC	RTC Sec Count Buffer Register
4000 6018h	RTCA0MIN	RTC Min Count Buffer Register
4000 601Ch	RTCA0HOUR	RTC Hour Count Buffer Register
4000 6020h	RTCA0WEEK	RTC Week Count Buffer Register
4000 6024h	RTCA0DAY	RTC Day Count Buffer Register
4000 6028h	RTCA0MONTH	RTC Month Count Buffer Register
4000 602Ch	RTCA0YEAR	RTC Year Count Buffer Register
4000 6030h	RTCA0TIME	RTC Time Set Register
4000 6034h	RTCA0CAL	RTC Calendar Set Register
4000 6038h	RTCA0SUBU	RTC Clock Error Correction Register
4000 603Ch	RTCA0SCMP	RTC Sub Count Compare Register
4000 6040h	RTCA0ALM	RTC Alarm Min Set Register
4000 6044h	RTCA0ALH	RTC Alarm Hour Set Register
4000 6048h	RTCA0ALW	RTC Alarm Week Set Register
4000 604Ch	RTCA0SECC	RTC Second Count Register
4000 6050h	RTCA0MINC	RTC Minute Count Register
4000 6054h	RTCA0HOURC	RTC Hour Count Register
4000 6058h	RTCA0WEEKC	RTC Week Count Register
4000 605Ch	RTCA0DAYC	RTC Day Count Register
4000 6060h	RTCA0MONC	RTC Month Count Register
4000 6064h	RTCA0YEARC	RTC Year Count Register
4000 6068h	RTCA0TIMEC	RTC Time Count Register
4000 606Ch	RTCA0CALC	RTC calendar Count Register
4000 6070h	RTCA0TCR	RTC Test Register

12.4 Register Description

12.4.1 RTCA0CTL0 — RTC Control Register 0

Address: 4000 6000h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	RTCA0CE	RTCA0CEST	RTCA0AMPM	RTCA0SLSB	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.2 RTCA0CTL0 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7	RTCA0CE	RTC Controller enable bit 0: Clock counter operation stop 1: Clock counter operation enable. The clock counter counts up. Setting this bit to “0” initializes the internal circuit of RTC Controller other than the control registers*1. Moreover, interrupts (RTCATINTR_Int, RTCATINT1S_Int and RTCATINTAL_Int) are cleared.	R/W
b6	RTCA0CEST	RTC Controller enable status 0: Clock counter operation stopped status 1: Clock counter operation enabled status (The clock counter counts up.)	R
b5	RTCA0AMPM	RTCA0HOUR, RTCA0ALH display format selection bit 0: 12 hour display 1: 24 hour display	R/W
b4	RTCA0SLSB	RTCA0SUBU and RTCA0SCMP enable/disable setting 0: RTCA0SUBU setting enabled RTCA0SCMP setting disabled 1: RTCA0SUBU setting disabled RTCA0SCMP setting enabled When this bit is set to “0”, RTC Controller operates on the assumption that a 32.768 kHz clock is input to 32k clock. At this time, the error of the input clock can be corrected in the range of 32.76180000 kHz to 32.77420000 kHz using the RTCA0SUBU register. When this bit is set to “1”, refer to the RTCA0SCMP register description. ***Caution*)** When setting this bit, be sure to follow the flow described in **Section 12.5.1.1, Initial Setting**. It is prohibited to change the setting of this bit while the clock counter operation is enabled (RTCA0CE = 1). If the setting of this bit is changed during clock counter operation, the operation of RTC Controller cannot be guaranteed. When changing the setting of this bit, first perform initialization according to **Section 12.5.1.5, Initialization of RTC While Clock Counter Operation is Enabled**, and then perform the setting of this bit again according to the flow described in **Section 12.5.1.1, Initial Setting**.	R/W
b3 to b0	Reserved		R

Note 1. Status flags RTCA0WST, RTCA0RSST, RTCA0WSST, and RTCA0WUST, which cannot be written by the user, are cleared.

12.4.2 RTCA0CTL1 — RTC Control Register 1

Address: 4000 6004h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	RTCA0 ALME	RTCA0 1SE	RTCA0CT		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.3 RTCA0CTL1 Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b6	Reserved		R
b5	Reserved	Should be 0.	R/W
b4	RTCA0ALME	Alarm interrupt (RTCATINTAL_Int) output enable bit. 0: Disable 1: Enable	R/W
b3	RTCA01SE	1 second interrupt (RTCATINT1S_Int) output enable bit 0: Stops interrupt output at every 1 second 1: Enables interrupt output at every 1 second	R/W
b2 to b0	RTCA0CT	Fixed interval interrupt (RTCATINTR_Int) output setting bit RTCA0CT[2]/RTCA0CT[1]/RTCA0CT[0] 000b: Fixed interval interrupt output stop 001b: Interrupt once every 0.25 seconds (in synchronization with second count-up) 010b: Interrupt once every 0.5 seconds (in synchronization with second count-up) 011b: Interrupt once every 1 second (in synchronization with second count-up) 100b: Interrupt once every 1 minute (every 1 minute 00 seconds) 101b: Interrupt once every 1 hour (every 1 hour 00 minutes 00 seconds) 110b: Interrupt once every 1 day (every 1 day 00 hours 00 minutes 00 seconds) 111b: Interrupt once every 1 month (every 1 month first day 00 hours 00 minutes 00 seconds a.m.) Refer to **Section 12.5.1.8, Changing the Setting of Fixed Interval Interrupt During Clock Counter Operation** for changing this field setting.	R/W

12.4.3 RTCA0CTL2 — RTC Control Register 2

Address: 4000 6008h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	RTCA0 WUST	RTCA0 WSST	RTCA0 RSST	RTCA0 RSUB	RTCA0 WST	RTCA0 WAIT
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.4 RTCA0CTL2 Register Contents (1/2)

Bit Position	Bit Name	Function	R/W
b31 to b6	Reserved		R
b5	RTCA0WUST	RTCA0SUBU write status 0: RTCA0SUBU write completed 1: RTCA0SUBU write in progress This bit is set (to “1”) when RTCA0SUBU is written to while the clock counter operation is enabled (RTCA0CE = 1). It is then cleared (to “0”) upon completion of RTCA0SUBU write (RTCA0SUBC overflow). ***Caution*)** Do not write to RTCA0SUBU while the value of this bit is “1” because write to RTCA0SUBU is in progress.	R
b4	RTCA0WSST	RTCA0SCMP write status 0: RTCA0SCMP write completed 1: RTCA0SCMP write in progress This bit is set (to “1”) when RTCA0SCMP is written to while the clock counter operation is enabled (RTCA0CE = 1). It is then cleared (to “0”) upon completion of RTCA0SCMP write (RTCA0SUBC overflow). ***Caution*)** Do not write to RTCA0SCMP while the value of this bit is “1” because write to RTCA0SCMP is in progress.	R
b3	RTCA0RSST	RTCA0SRBU transfer status 0: RTCA0SRBU data hold status 1: Transfer of the value of RTCA0SUBC to RTCA0SRBU completed When “1” has been written to RTCA0RSUB, the value of this bit becomes “1” when the value of RTCA0SUBC is loaded to RTCA0SRBU. Before reading RTCA0SRBU during clock counter operation (RTCA0CE = 1), check that the value of this register is “1”.	R
b2	RTCA0RSUB	RTCA0SUBC data transfer control 0: RTCA0SRBU data hold 1: Transfer of value of RTCA0SUBC to RTCA0SRBU This is a control bit for transferring the value of RTCA0SUBC of RTCA0SRBU. Use it when reading the value of RTCA0SRBU while the clock counter operation is enabled (RTCA0CE = 1). When “1” is written to this bit, the value of RTCA0SUBC is loaded to RTCA0SRBU in synchronization with 32k clock. For the usage method, refer to **Section 12.5.1.4, Reading RTCA0SRBU While Clock Counter Operation is Enabled** (RTCA0CE = 1).	R/W

Table 12.4 RTCA0CTL2 Register Contents (2/2)

Bit Position	Bit Name	Function	R/W
b1	RTCA0WST	RTC Controller counter wait status 0: Count-up status of RTCA0SECC, RTCA0MINC, RTCA0HOURC, RTCA0WEEKC, RTCA0DAYC, RTCA0MONC, RTCA0YEARC 1: Count-up stopped status of RTCA0SECC, RTCA0MINC, RTCA0HOURC, RTCA0WEEKC, RTCA0DAYC, RTCA0MONC, RTCA0YEARC. (Count-up operation of second, minute, hour, week, day, month, and year count registers is stopped.) When “1” has been written to RTCA0WAIT, the value of this bit becomes “1” when the count-up operation of RTCA0SEC, RTCA0MIN, RTCA0HOUR, RTCA0WEEK, RTCA0DAY, RTCA0MONTH, RTCA0YEAR has stopped completely. Before reading and writing clock counters (RTCA0SEC, RTCA0MIN, RTCA0HOUR, RTCA0WEEK, RTCA0DAY, RTCA0MONTH, RTCA0YEAR) during clock counter operation (RTCA0CE = 1), check that the value of this register is “1”. Also, when write to a clock counter (RTCA0SEC, RTCA0MIN, RTCA0HOUR, RTCA0WEEK, RTCA0DAY, RTCA0MONTH, RTCA0YEAR) has occurred, the value written to RTCA0WAIT is reflected only once the write operation has been completed (status hold).	R
b0	RTCA0WAIT	RTC Controller counter wait control 0: Counter operation 1: Count-up operation of RTCA0SEC, RTCA0MIN, RTCA0HOUR, RTCA0WEEK, RTCA0DAY, RTCA0MONTH, RTCA0YEAR is stopped. (Count-up operation of second, minute, hour, week, day, month, and year count registers is stopped.) This bit controls the count-up operation of the counters. Be sure to write “1” to this bit when reading and writing counter values while the clock counter operation is enabled (RTCA0CE = 1). When the value of this bit is “1”, upon occurrence of RTCA0SUBC overflow, the overflow information is held internally, and after “0” has been written to this bit, RTCA0SEC is counted up. However, when the value of the second counter is written while RTCA0WAIT = 1, the overflow information that was held is discarded.	R/W

12.4.4 RTCA0SUBC — RTC Sub Count Register

Address: 4000 600Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	RTCA0SUBC					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RTCA0SUBC															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.5 RTCA0SUBC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b22	Reserved		R
b21 to b0	RTCA0SUBC	Register that counts the 1 second reference time This register is cleared when reset or upon write to the RTCA0SEC (RTC Sec Count Buffer register) or RTCA0TIME (RTC Time Set register), and arbitrary values cannot be written to this register. Count-up occurs upon clock input when the clock counter operation is enabled (RTCA0CE = 1). When this register is read while the clock counter operation is enabled (RTCA0CE = 1), the register value, which changes in synchronization with 32k clock, is read asynchronously, so that the read value is not guaranteed. To check the value of this register while the clock counter operation is enabled (RTCA0CE = 1), use the RTCA0SRBU register. The count-up operation of this register functions according to the setting of RTCA0SLSB as follows. **<RTCA0SLSB = 0>** When the clock error correction function is not used, count-up operation starts from 0h, and when 7FFFh is reached, an overflow signal is output to RTCA0SEC (Second Count Buffer register), and this register is cleared. When clock error correction is performed, overflow occurs in the range of 7F83h to 807Bh due to the setting of RTCA0SUBU. The setting of RTCA0SCMP (Sub Count Compare register) is ignored. **<RTCA0SLSB = 1>** The count-up operation starts from 0h, and upon matching the value set to RTCA0SCMP, an overflow signal is output to RTCA0SEC (Second Count Buffer register), and this register is cleared. The setting of RTCA0SUBU (Clock Error Correction) is ignored. **[*Example*]** Operation using 32 kHz clock “32000 − 1 = 31999 (decimal code) = 7CFFh” is set to RTCA0SCMP. Counting up is performed each time a 32 kHz clock is input, taking on values from 0h to 7CFFh. 1 second is counted when the value changes from 7CFFh to 0h.	R

12.4.5 RTCA0SRBU — RTC Sub Count Register Read Buffer

Address: 4000 6010h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	RTCA0SRBU					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RTCA0SRBU															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.6 RTCA0SRBU Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b22	Reserved		R
b21 to b0	RTCA0SRBU	Read buffer register of RTCA0SUBC. This register is cleared when reset, at which time arbitrary values cannot be written. When “1” is written to the RTCA0RSUB bit while the clock counter operation is enabled (RTCA0CE = 1), the value of RTCA0SUBC is loaded to this register in synchronization with 32k clock. ***Caution*)** Be sure to perform RTCA0SRBU read according to the flow described in **Section 12.5.1.4, Reading RTCA0SRBU While Clock Counter Operation is Enabled**.	R

12.4.6 RTCA0SEC — RTC Sec Count Buffer Register

Address: 4000 6014h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	RTCA0SEC						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.7 RTCA0SEC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b7	Reserved		R
b6 to b0	RTCA0SEC	Buffer register to read/write RTC Second Count register (RTCA0SECC). This register is used when Second Count register is read and written. When setting this register, must set a decimal value of 00 to 59 in BCD code.	R/W

12.4.7 RTCA0MIN — RTC Min Count Buffer Register

Address: 4000 6018h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	RTCA0MIN						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.8 RTCA0MIN Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b7	Reserved		R
b6 to b0	RTCA0MIN	Buffer register to read/write RTC Minute Count register (RTCA0MINC). This register is used when Minute Count register is read and written. When setting this register, must set a decimal value of 00 to 59 in BCD code.	R/W

12.4.8 RTCA0HOUR — RTC Hour Count Buffer Register

Address: 4000 601Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	RTCA0HOUR					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0

Table 12.9 RTCA0HOUR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b6	Reserved		R
b5 to b0	RTCA0HOUR	Buffer register to read/write RTC Hour Count register (RTCA0HOURC). This register is used when Hour Count register is read and written. When setting this register, must set a decimal value of 01 to 12 or 21 to 32 when RTCA0AMPM = 0, or a decimal value of 0 to 23 when RTCA0AMPM = 1, in BCD code in both cases. Display of RTCA0HOUR is 12-hour display when the RTCA0AMPM bit is set to “0”, and 24-hour display when it is “1”. In the case of 12-hour display, am/pm is indicated by the 5th bit of RTCA0HOUR. During am, RTCA0HOUR[5] = 0, and during pm, RTCA0HOUR[5] = 1.	R/W

12-hour display	
Time	RTCA0HOUR Display
0 am	12h
1 am	01h
...	
9 am	09h
10 am	10h
11 am	11h
0 pm	32h
1 pm	21h
...	
9 pm	29h
10 pm	30h
11 pm	31h

24-hour display	
Time	RTCA0HOUR Display
0	00h
1	01h
...	
9	09h
10	10h
11	11h
12	12h
13	13h
...	
21	21h
22	22h
23	23h

12.4.9 RTCA0WEEK — RTC Week Count Buffer Register

Address: 4000 6020h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	RTCA0WEEK		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.10 RTCA0WEEK Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved		R
b2 to b0	RTCA0WEEK	Buffer register to read/write RTC Week Count register (RTCA0WEEKC). This register is used when Week Count register is read and written. When setting this register, must set a decimal value of 00 to 06 in BCD code. There is no particular correspondence between the value of RTCA0WEEK and the day of the week. Set the correspondence according to the application to be used. [Example] 0 → Sunday, 1 → Monday, …, 6 → Saturday	R/W

12.4.10 RTCA0DAY — RTC Day Count Buffer Register

Address: 4000 6024h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	RTCA0DAY					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 12.11 RTCA0DAY Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b6	Reserved		R
b5 to b0	RTCA0DAY	Buffer register to read/write RTC Day Count register (RTCA0DAYC). This register is used when Day Count register is read and written. When setting this register, must set a decimal value of 01 to 31 in BCD code. • 01 to 31 (January, March, May, July, August, October, December) • 01 to 30 (April, June, September, November) • 01 to 29 (February, leap year) • 01 to 28 (February, non-leap year)	R/W

12.4.11 RTCA0MONTH — RTC Month Count Buffer Register

Address: 4000 6028h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	RTCA0MONTH				
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 12.12 RTCA0MONTH Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b5	Reserved		R
b4 to b0	RTCA0MONTH	Buffer register to read/write RTC Month Count register (RTCA0MONC). This register is used when Month Count register is read and written. When setting this register, must set a decimal value of 01 to 12 in BCD code.	R/W

12.4.12 RTCA0YEAR — RTC Year Count Buffer Register

Address: 4000 602Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	RTCA0YEAR							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.13 RTCA0YEAR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7 to b0	RTCA0YEAR	Buffer register to read/write RTC Year Count register (RTCA0YEARC). This register is used when Year Count register is read and written. When setting this register, must set a decimal value of 00 to 99 in BCD code.	R/W

12.4.13 RTCA0TIME — RTC Time Set Register

RTCA0TIME is a register for accessing the RTCA0HOUR, RTCA0MIN, and RTCA0SEC registers simultaneously. By using this register, the RTCA0HOUR, RTCA0MIN, and RTCA0SEC registers can be read or written simultaneously.

Address: 4000 6030h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	RTCA0HOUR							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RTCA0MIN								RTCA0SEC							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.14 RTCA0TIME Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved		R
b23 to b16	RTCA0HOUR	Refer to RTCA0HOUR register	R/W
b15 to b8	RTCA0MIN	Refer to RTCA0MIN register	R/W
b7 to b0	RTCA0SEC	Refer to RTCA0SEC register	R/W

12.4.14 RTCA0CAL — RTC Calendar Set Register

RTCA0CAL is a register for accessing the RTCA0YEAR, RTCA0MONTH, RTCA0DAY, and RTCA0WEEK registers simultaneously. By using this register, the RTCA0YEAR, RTCA0MONTH, RTCA0DAY, and RTCA0WEEK registers can be read or written simultaneously.

Address: 4000 6034h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	RTCA0YEAR								RTCA0MONTH							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RTCA0DAY								RTCA0WEEK							
Value after reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Table 12.15 RTCA0CAL Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	RTCA0YEAR	Refer to RTCA0YEAR register	R/W
b23 to b16	RTCA0MONTH	Refer to RTCA0MONTH register	R/W
b15 to b8	RTCA0DAY	Refer to RTCA0DAY register	R/W
b7 to b0	RTCA0WEEK	Refer to RTCA0WEEK register	R/W

12.4.15 RTCA0SUBU — RTC Clock Error Correction Register

Address: 4000 6038h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	RTCA0DEV	RTCA0F6	RTCA0F5	RTCA0F4	RTCA0F3	RTCA0F2	RTCA0F1	RTCA0F0
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.16 RTCA0SUBU Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7	RTCA0DEV	Setting of clock error correction timing 0: Clock error correction when RTCA0SEC (second counter) is 00, 20, and 40 seconds 1: Clock error correction when RTCA0SEC (second counter) is 00 seconds	R/W
b6	RTCA0F6	Clock error correction value Bit6 0: Count value of RTCA0SUBC is incremented (+ correction) by the amount set with RTCA0F5 to 0. Increment value calculation formula: (RTCA0F[5:0] setting value − 1) × 2 1: Count value of RTCA0SUBC is decremented (− correction) by the amount set with RTCA0F5 to 0. Decrement value calculation formula: (Inverted data of setting value of RTCA0F[5:0] + 1) × 2	R/W
b5	RTCA0F5	Clock error correction value Bit5	R/W
b4	RTCA0F4	Clock error correction value Bit4	R/W
b3	RTCA0F3	Clock error correction value Bit3	R/W
b2	RTCA0F2	Clock error correction value Bit2	R/W
b1	RTCA0F1	Clock error correction value Bit1	R/W
b0	RTCA0F0	Clock error correction value Bit0	R/W

12.4.16 RTCA0SCMP — RTC Sub Count Compare Register

Address: 4000 603Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	RTCA0SCMP					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RTCA0SCMP															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.17 RTCA0SCMP Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b22	Reserved		R
b21 to b0	RTCA0SCMP	Register that sets the compare value of RTCA0SUBC (sub-counter). The setting of this register is effective when RTCA0SLSB = 1. When RTCA0SLSB = 1 is set, RTCA0SUBC starts counting up from 0, and when the value set to this register is matched, an overflow signal is output to RTCA0SEC (Second Count Buffer register), and the register is cleared. Set the value for this register according to the frequency of the selected input clock. Accurate one second counting can be performed to 32k clock through the value set to this register. **[Example]** When input clock = 32 kHz Setting of 32,000 (decimal) − 1 = 31,999 (decimal) = 7CFFh (hexadecimal) to RTCA0SCMP When performing write to this register during clock counter operation (RTCA0CE = 1), be sure to check that RTCA0WSST = 0. When writing to this register during clock counter operation (RTCA0CE = 1), write ends at the timing of RTCA0SUBC overflow. ***Caution*)** When RTCA0SLSB is set to “1”, be sure to set a value of 31999 or higher (32 kHz or higher) to this register. When RTCA0SLSB is set to “1”, the operation of RTC Controller cannot be guaranteed if a value of 31998 or lower has been set to this register. Be sure to perform RTCA0SCMP write according to the flows described in **Section 12.5.1.1, Initial Setting** and **Section 12.5.1.7, Writing to RTCA0SCMP During Clock Counter Operation**.	R/W

12.4.17 RTCA0ALM — RTC Alarm Min Set Register

Address: 4000 6040h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	RTCA0ALM						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.18 RTCA0ALM Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b7	Reserved		R
b6 to b0	RTCA0ALM	RTCA0ALM is a register that performs the minute setting for the alarm interrupt. Set a decimal value of 00 to 59 in BCD code. (Alarm detection is not performed if an out-of-range value is set.)	R/W

12.4.18 RTCA0ALH — RTC Alarm Hour Set Register

Address: 4000 6044h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	RTCA0ALH					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0

Table 12.19 RTCA0ALH Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b6	Reserved		R
b5 to b0	RTCA0ALH	RTCA0ALH is a register that performs the hour setting for the alarm interrupt. When RTCA0AMPM = 0, set a decimal value of 01 to 12 or 21 to 32, and when RTCA0AMPM = 1, set a decimal value of 00 to 23, in BCD code in both cases. (Alarm detection is not performed if an out-of-range value is set.)	R/W

12.4.19 RTCA0ALW — RTC Alarm Week Set Register

This register enables the alarm corresponding to RTCA0WEEK value.

[Example]

If RTCA0WEEK = 0 is Sunday, RTCA0ALW0 bit functions as Sunday alarm. If RTCA0WEEK = 6 is Saturday, RTCA0ALW6 bit functions as Saturday alarm.

Address: 4000 6048h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	RTCA0 ALW6	RTCA0 ALW5	RTCA0 ALW4	RTCA0 ALW3	RTCA0 ALW2	RTCA0 ALW1	RTCA0 ALW0
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.20 RTCA0ALW Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b7	Reserved		R
b6	RTCA0ALW6	Alarm interrupt day of the week setting bit 6 0: Disable alarm 1: Enable alarm	R/W
b5	RTCA0ALW5	Alarm interrupt day of the week setting bit 5 0: Disable alarm 1: Enable alarm	R/W
b4	RTCA0ALW4	Alarm interrupt day of the week setting bit 4 0: Disable alarm 1: Enable alarm	R/W
b3	RTCA0ALW3	Alarm interrupt day of the week setting bit 3 0: Disable alarm 1: Enable alarm	R/W
b2	RTCA0ALW2	Alarm interrupt day of the week setting bit 2 0: Disable alarm 1: Enable alarm	R/W
b1	RTCA0ALW1	Alarm interrupt day of the week setting bit 1 0: Disable alarm 1: Enable alarm	R/W
b0	RTCA0ALW0	Alarm interrupt day of the week setting bit 0 0: Disable alarm 1: Enable alarm	R/W

12.4.20 RTCA0SECC — RTC Second Count Register

Address: 4000 604Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	RTCA0SECC						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.21 RTCA0SECC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b7	Reserved		R
b6 to b0	RTCA0SECC	Counts Up the Seconds This register counts from 00 to 59, and when its value changes from 59 to 00 in BCD code, an overflow signal is output to RTCA0MINC.	R

12.4.21 RTCA0MINC — RTC Minute Count Register

Address: 4000 6050h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	RTCA0MINC						
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.22 RTCA0MINC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b7	Reserved		R
b6 to b0	RTCA0MINC	Counts Up the Minutes This register counts from 00 to 59, and when its value changes from 59 to 00 in BCD code, an overflow signal is output to RTCA0HOURC.	R

12.4.22 RTCA0HOURC — RTC Hour Count Register

Address: 4000 6054h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	RTCA0HOURC					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0

Table 12.23 RTCA0HOURC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b6	Reserved		R
b5 to b0	RTCA0HOURC	Counts Up the Hours This register counts from 00 to 23, or 01 to 12, or 21 to 32 in BCD code depending on the setting of RTCA0AMPM, and when its value changes from 23 to 00 or from 32 to 01, an overflow signal is output to RTCA0DAY and RTCA0WEEK.	R

12.4.23 RTCA0WEEKC — RTC Week Count Register

Address: 4000 6058h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	RTCA0WEEKC		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.24 RTCA0WEEKC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved		R
b2 to b0	RTCA0WEEKC	Counts Up the Weeks This register counts up upon occurrence of RTCA0HOURC overflow when the clock counter operation is enabled (RTCA0CE = 1).	R

12.4.24 RTCA0DAYC — RTC Day Count Register

Address: 4000 605Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	RTCA0DAYC					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 12.25 RTCA0DAYC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b6	Reserved		R
b5 to b0	RTCA0DAYC	Counts Up the Days This register counts up upon occurrence of RTCA0HOURC overflow when the clock counter operation is enabled (RTCA0CE = 1). This register counts according to the value of RTCA0MONC, and upon occurrence of an overflow, an overflow signal is output to RTCA0MONC.	R

12.4.25 RTCA0MONC — RTC Month Count Register

Address: 4000 6060h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	RTCA0MONC				
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 12.26 RTCA0MONC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b5	Reserved		R
b4 to b0	RTCA0MONC	Counts Up the Months This register counts up upon occurrence of RTCA0DAYC overflow while the clock counter operation is enabled (RTCA0CE = 1). This register counts from 01 to 12 in BCD code, and when its value changes from 12 to 01, an overflow signal is output to RTCA0YEARC.	R

12.4.26 RTCA0YEARC — RTC Year Count Register

Address: 4000 6064h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	RTCA0YEARC							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.27 RTCA0YEARC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b8	Reserved		R
b7 to b0	RTCA0YEARC	Counts Up the Years This register counts up upon occurrence of RTCA0MONC overflow while the clock counter operation is enabled (RTCA0CE = 1). This register counts from 00 to 99 in BCD code.	R

12.4.27 RTCA0TIMEC — RTC Time Count Register

This register is for reading the RTCA0HOURC, RTCA0MINC, and RTCA0SECC registers simultaneously.

Address: 4000 6068h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	RTCA0HOURC							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RTCA0MINC								RTCA0SECC							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.28 RTCA0TIMEC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	Reserved		R
b23 to b16	RTCA0HOURC	Refer to RTCA0HOURC register	R
b15 to b8	RTCA0MINC	Refer to RTCA0MINC register	R
b7 to b0	RTCA0SECC	Refer to RTCA0SECC register	R

12.4.28 RTCA0CALC — RTC Calendar Count Register

This register is for reading the RTCA0YEARC, RTCA0MONC, RTCA0DAYC, and RTCA0WEEKC simultaneously.

Address: 4000 606Ch

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	RTCA0YEARC								RTCA0MONC							
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RTCA0DAYC								RTCA0WEEKC							
Value after reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Table 12.29 RTCA0CALC Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b24	RTCA0YEARC	Refer to RTCA0YEARC register	R
b23 to b16	RTCA0MONC	Refer to RTCA0MONC register	R
b15 to b8	RTCA0DAYC	Refer to RTCA0DAYC register	R
b7 to b0	RTCA0WEEKC	Refer to RTCA0WEEKC register	R

12.4.29 RTCA0TCR — RTC Test Register

Not available in this LSI.

Address: 4000 6070h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	RTCA0OSE	—	—	—	—	—	—	—	—	—	—	—	RTCA0OS3	RTCA0OS2	RTCA0OS1	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 12.30 RTCA0TCR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved		R
b15	RTCA0OSE	0: Disable interrupt substitution latch. 1: Enable interrupt substitution latch.	R/W
b14 to b4	Reserved		R
b3	RTCA0OS3	Value of RTCATINTAL_Int substitution latch.	R/W
b2	RTCA0OS2	Value of RTCATINT1S_Int substitution latch.	R/W
b1	RTCA0OS1	Value of RTCATINTR_Int substitution latch.	R/W
b0	Reserved	Should be 0.	R/W

12.5 Operation

12.5.1 Programming RTC

12.5.1.1 Initial Setting

Perform initial setting of RTC according to the following flow.

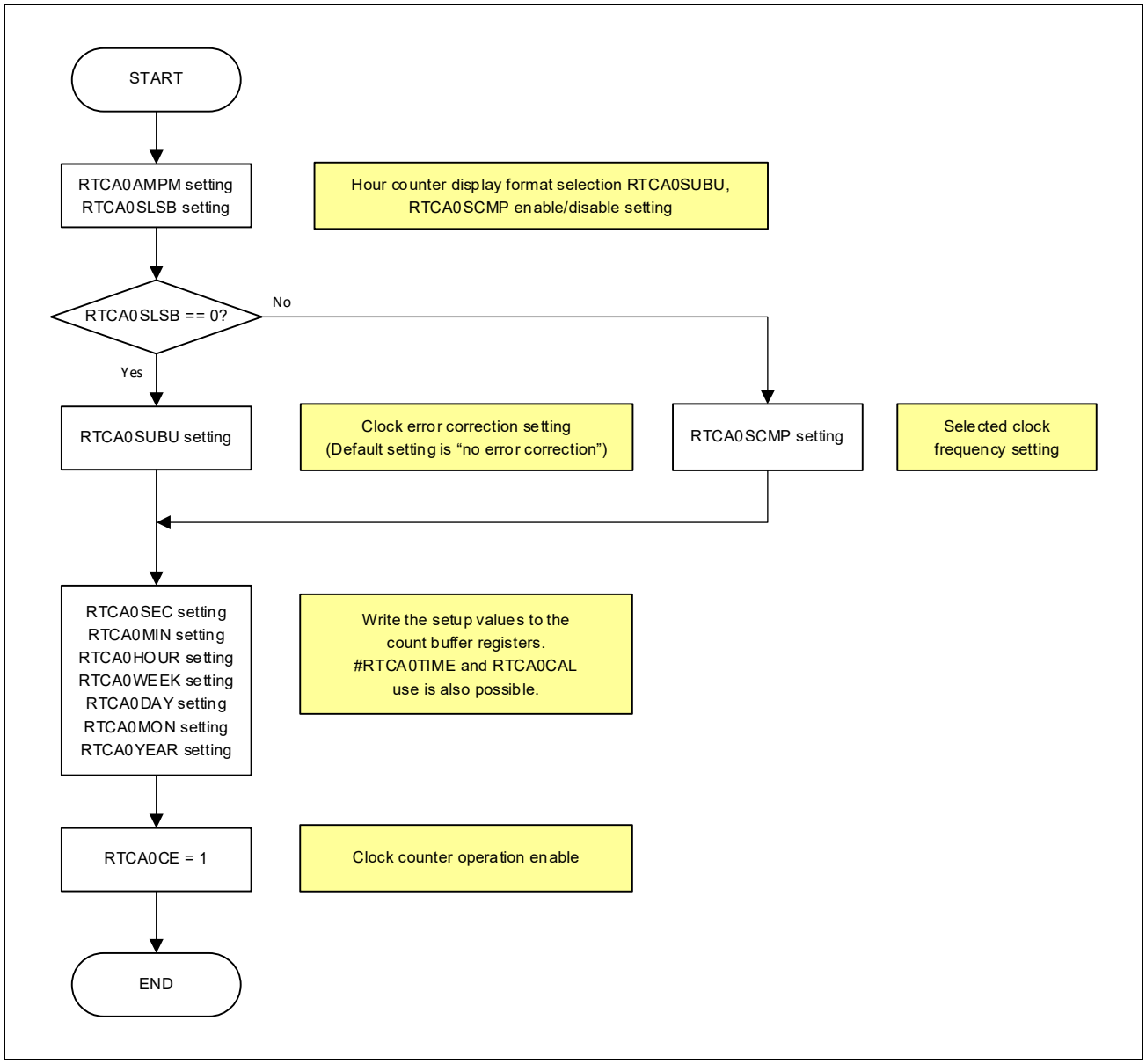


Figure 12.2 Setting RTC When Clock Counter is Disabled

The internal clock counter operates in synchronization with 32k clock, and two 32k clocks are required until completion of the above-described initial value setting procedure. Therefore, RTC_PCLK must be continuously supplied until completion of the initial value setting procedure. Therefore, to stop RTC_PCLK supply after the initial value setting procedure, check first that RTCA0CEST = 1 (clock counter operation enabled status).

12.5.1.2 Writing to Clock Counters While Clock Counter Operation is Enabled

Be sure to write to the RTCA0SEC, RTCA0MIN, RTCA0HOUR, RTCA0WEEK, RTCA0DAY, RTCA0MONTH and RTCA0YEAR counters while the clock counter operation is enabled (RTCA0CE = 1) according to the following flow

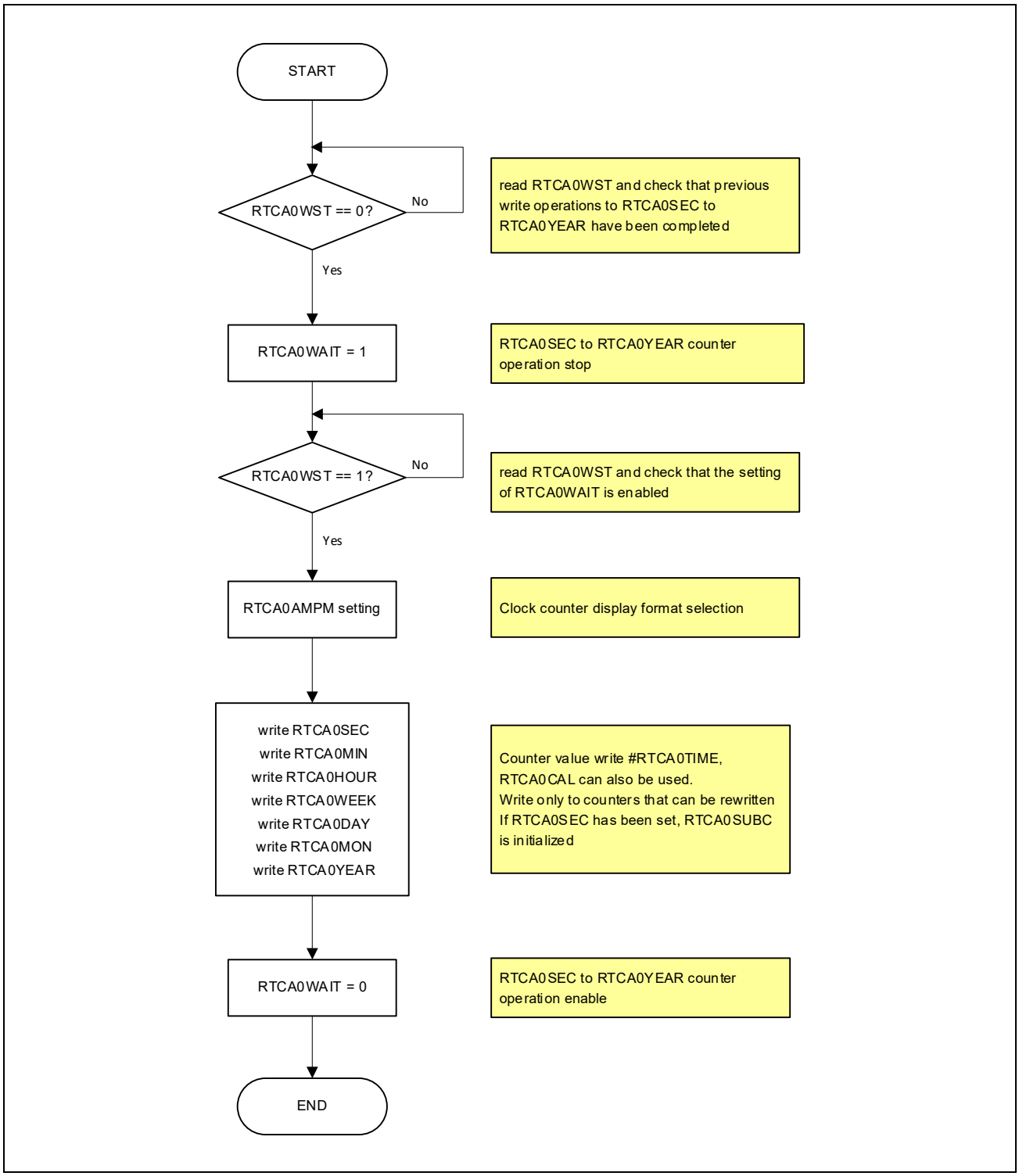


Figure 12.3 Setting RTC When Clock Counter Is Enabled

It is prohibited to write two or more times to the same register in the above-described flow. For example, it is prohibited to write twice to the second counter. End the programming flow within 1 second.

When RTCA0WAIT = 1 is set, the operation of RTCA0SEC (second counter buffer) stops. If RTCA0SUBC overflow occurs while RTCA0WAIT = 1, one overflow is held internally. However, if overflow occurs two or more times, the overflow count cannot be held.

The internal clock counter operates in synchronization with 32k clock. Two 32k clock cycles are required until completion of the above-described initial value setting procedure. Therefore, RTC_PCLK must be continuously supplied until completion of the initial value setting procedure.

To stop RTC_PCLK supply after the initial value setting procedure, check first that RTCA0WST = 0 (clock counter count-up status).

12.5.1.3 Reading Clock Counters While Clock Counter Operation is Enabled

There are two methods to read clock counters while clock counter operation is enabled (RTCA0CE = 1):

(1) Using Count Buffer Registers

This is how to read clock counter value by using by using buffer registers. The value of clock count registers are transferred to count buffer registers by writing 1 in RTCA0WAIT. The count buffer registers are read after that. In this method, program wait*[1] occurs from setting RTCA0WAIT=1 to completion of data transfer.

(2) Reading Count Registers

This is how to read clock counter immediately. RTCA0SECC (sec count register) is read twice in the beginning and the end. This is because it confirms whether overflow of sub-counter does not happen while counter is read. After that, the first read value is compared with the second read value. If the first read value is same as the second read value, it is judged that overflow of sub-counter didn't occur while counter read flow. If the first read value is not same as the second read value, it is judged that overflow*[2] of sub-counter occurred while counter read flow, and It begins to read clock counter again.

The merit and demerit of the two methods are mentioned in the next table.

		Merit	Demerit
(1)	Using count buffer registers	It is unnecessary to read clock counter several times like the method of (2), because clock counters are read synchronously.	Program wait*[1] occurs from setting RTCA0WAIT = 1 to completion of data transfer
(2)	Reading count registers	Program wait doesn't occur.	When the reading of counter and overflow of RTCA0SUBC competed, read clock counters several times.

Note 1. The maximum of program wait is 3 clock cycles of RTC_PCLK + 2 clock cycles of 32k clock.

Note 2. When reading of counter and overflow of sub-counter competed with each other, it has the possibility that unknown value is read.

Reading clock counters while clock counter operation is enabled (RTCA0CE = 1) (using count buffer registers)

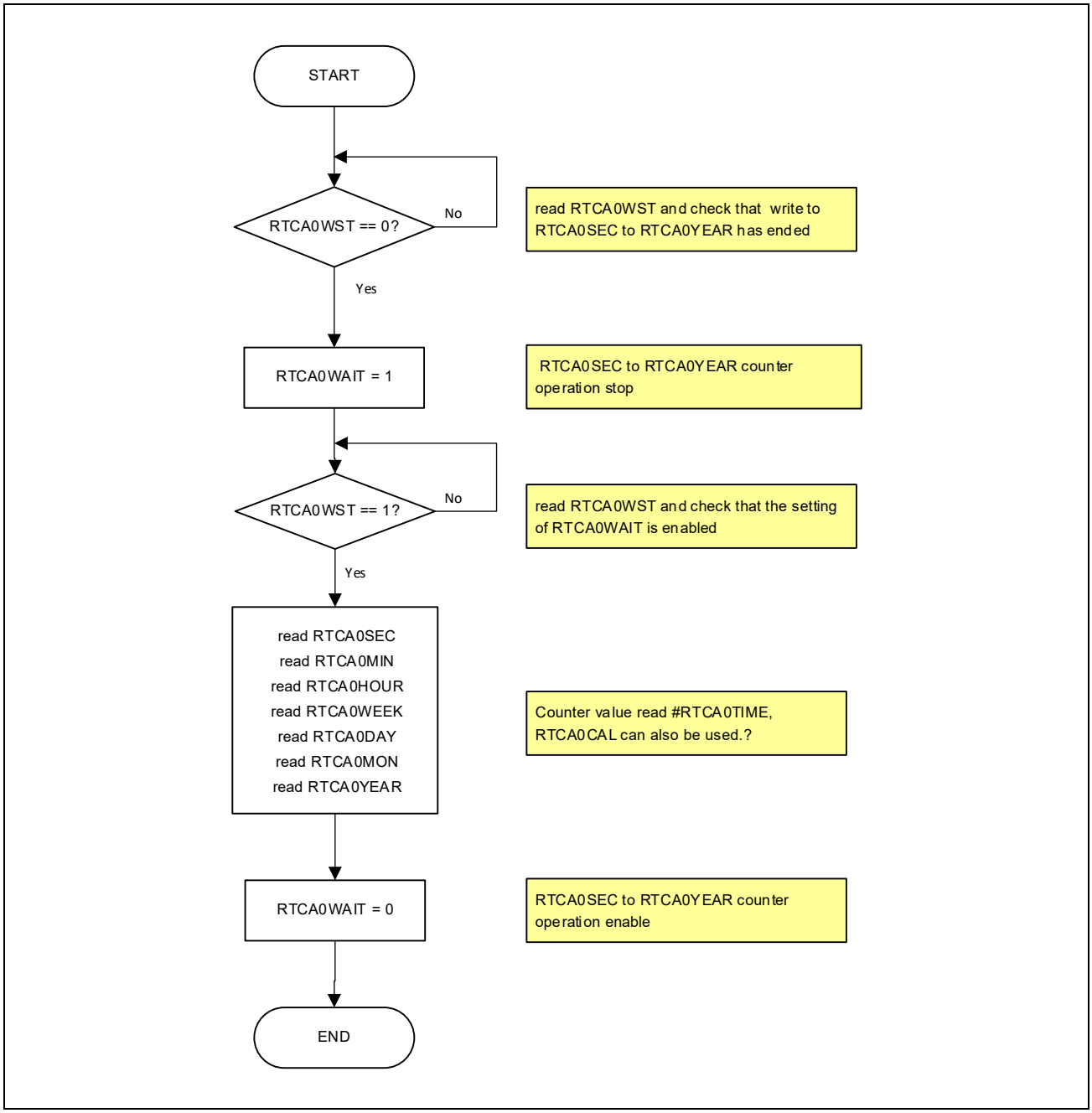


Figure 12.4 Reading RTC Using the Buffer Registers

End this flow within 1 second.

When RTCA0WAIT = 1 is set, the operation of RTCA0SEC (second counter buffer) stops. If RTCA0SUBC overflow occurs while RTCA0WAIT = 1, one overflow is held internally. However, if overflow occurs two or more times, the overflow count cannot be held.

Reading clock counters while clock counter operation is enabled (RTCA0CE = 1) (reading count registers)

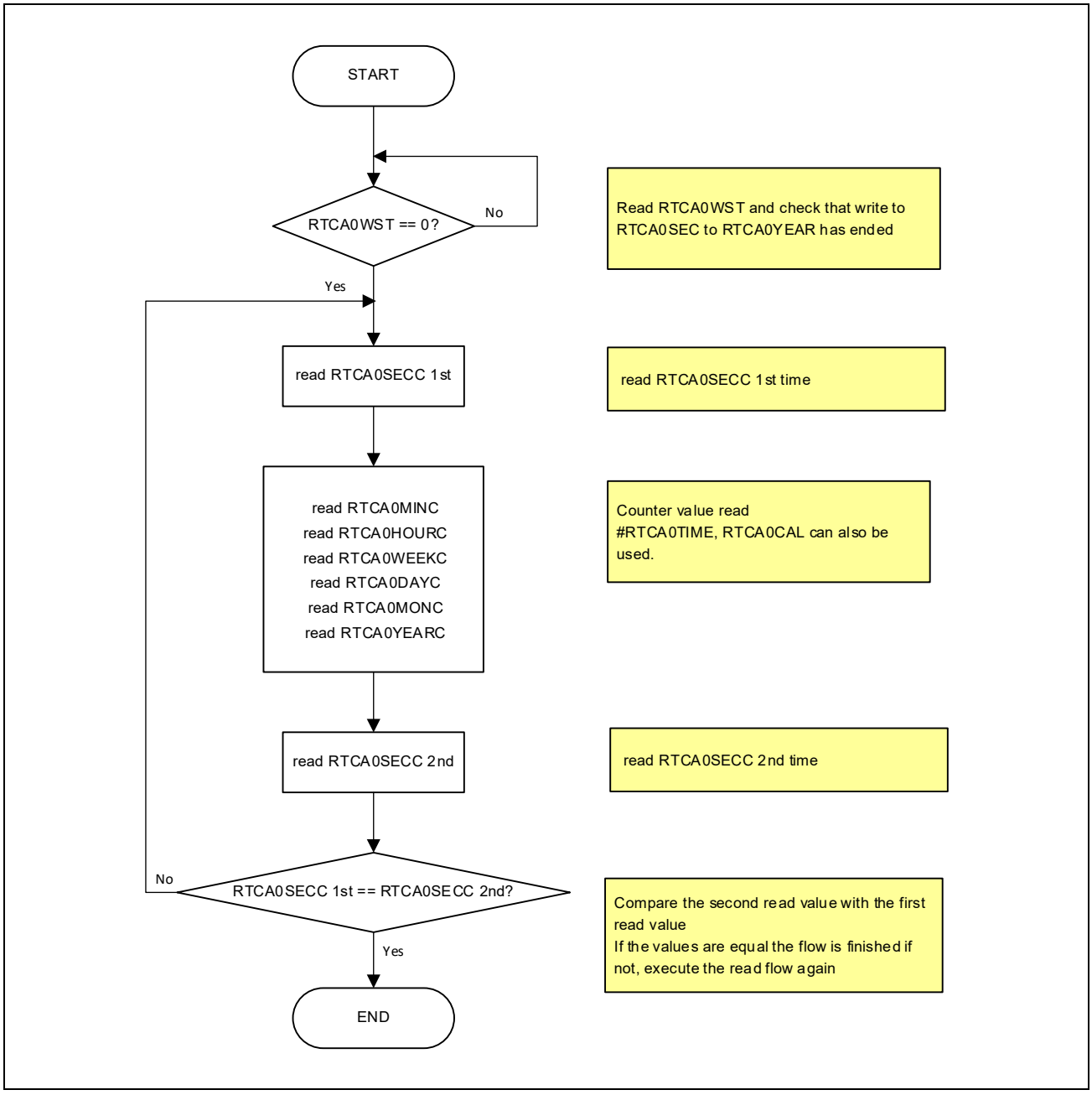


Figure 12.5 Reading RTC Using the Counting Registers

End this flow within 1 second.

When it isn't finished within one second, read clock counter value can't be assured.

12.5.1.4 Reading RTCA0SRBU While Clock Counter Operation is Enabled

Be sure to read RTCA0SRBU while the clock counter operation is enabled (RTCA0CE = 1) according to the following flow.

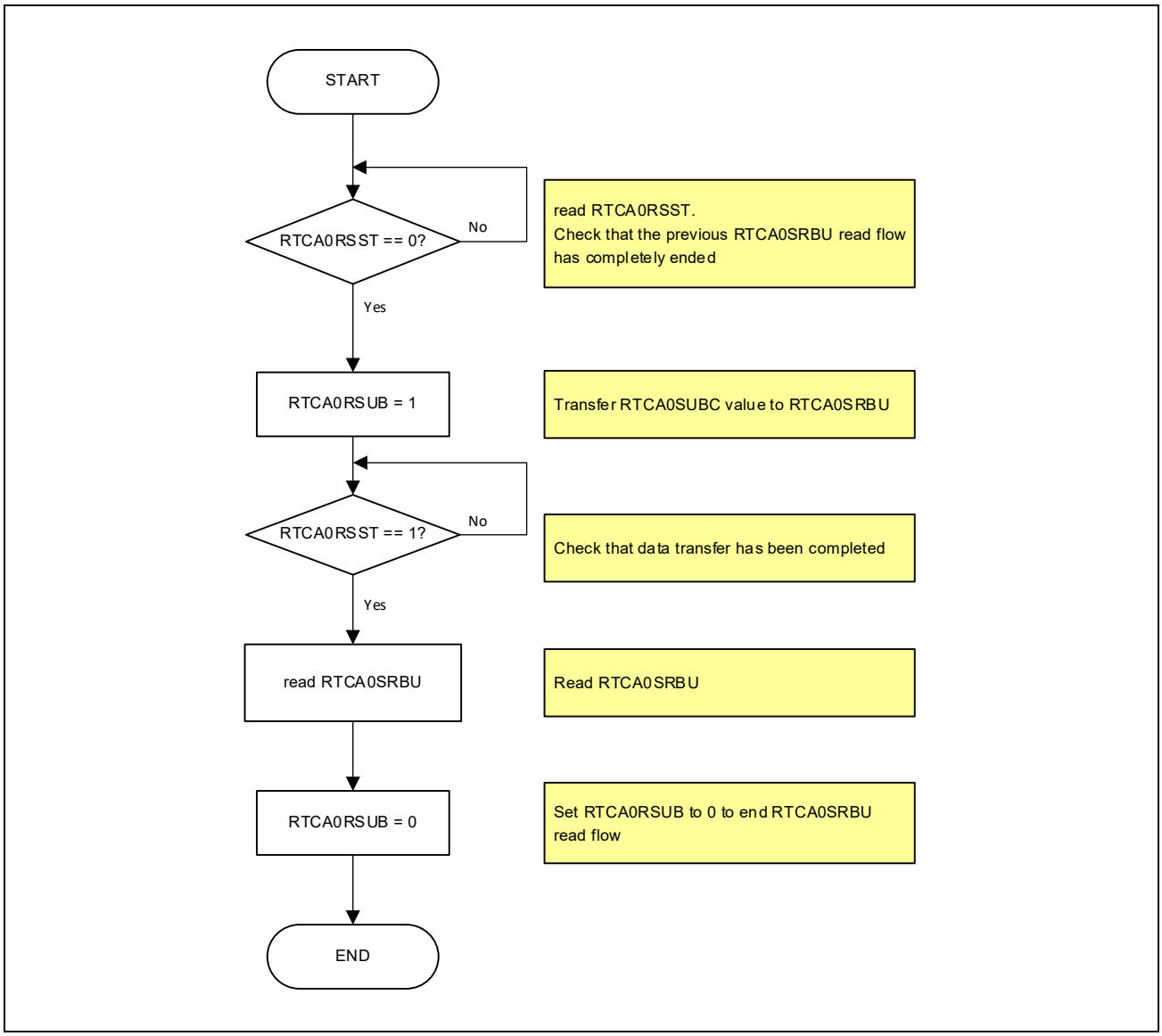


Figure 12.6 Reading RTCA0SRBU

12.5.1.5 Initialization of RTC While Clock Counter Operation is Enabled

Be sure to perform initialization of RTC controller while the clock counter operation is enabled (RTCA0CE = 1) according to the following flow. To restart the RTC controller operation at the end of the flow, implement the flow described in **Section 12.5.1.1, Initial Setting**.

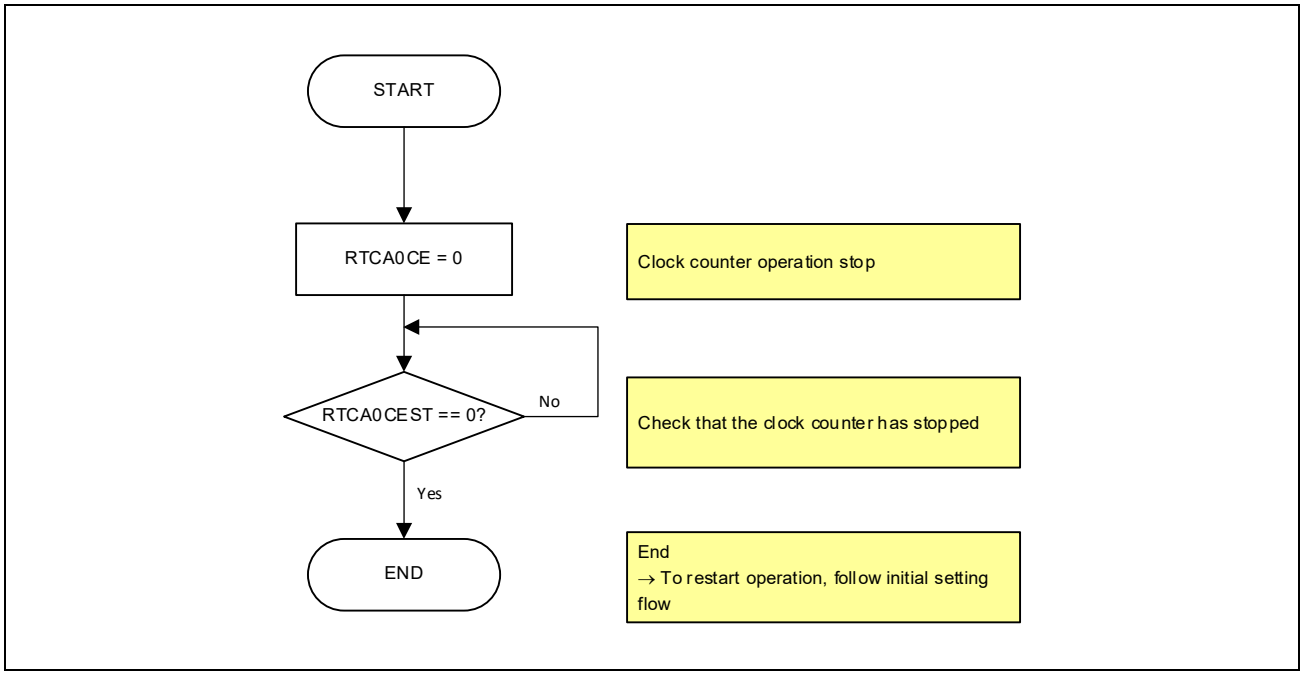


Figure 12.7 Initialization of RTC While Clock Counter Operation is Enabled

12.5.1.6 Writing to RTCA0SUBU Clock Counter Operation

Be sure to write to RTCA0SUBU while the clock counter operation is enabled (RTCA0CE = 1) according to the flow described below.

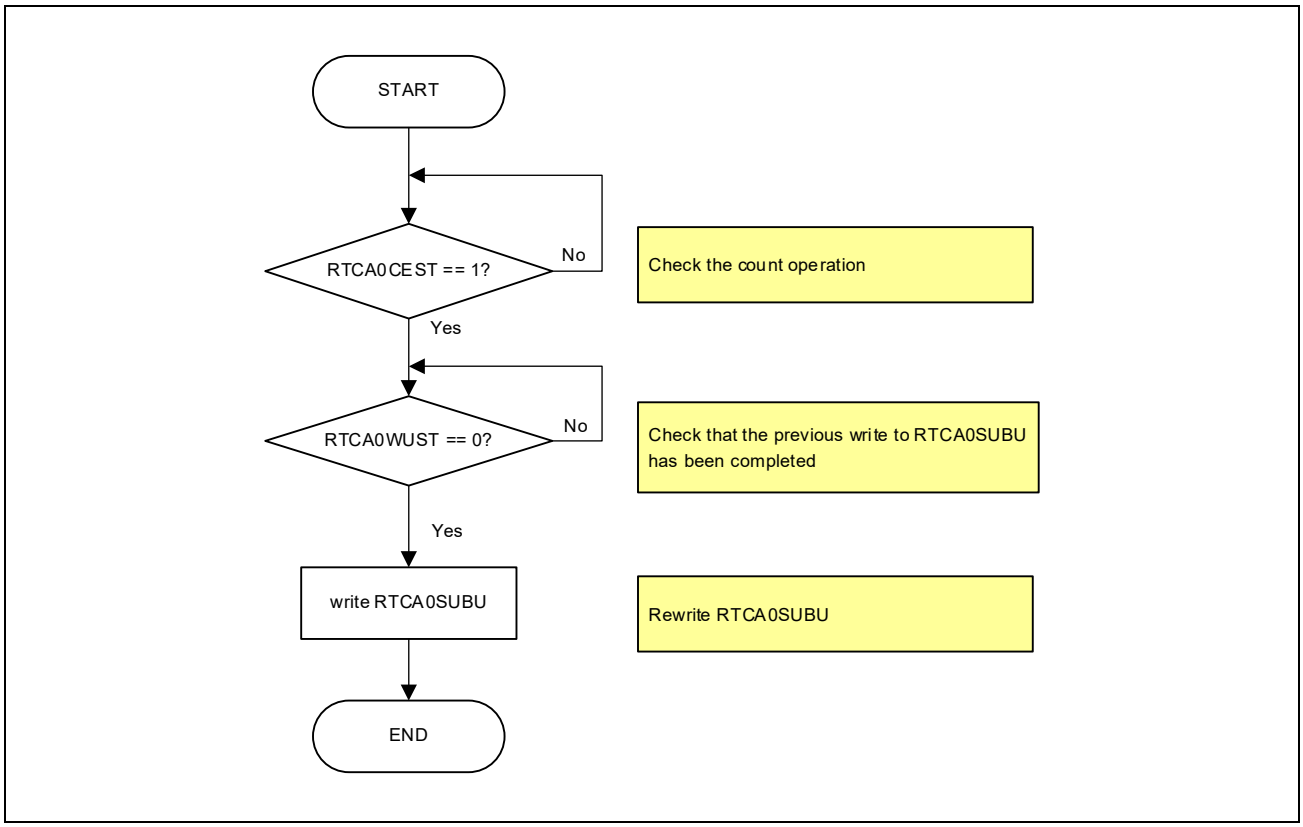


Figure 12.8 Writing to RTCA0SUBU

RTCA0WUST is set to “1” when RTCA0SUBU is written to while the clock counter operation is enabled (RTCA0CE = 1), and upon completion of RTCA0SUBU write (RTCA0SUBC overflow), it is cleared to “0”. Because RTCA0WUST displays “1” for up to 1 second for this reason, caution is required when polling RTCA0WUST.

12.5.1.7 Writing to RTCA0SCMP During Clock Counter Operation

Be sure to write to RTCA0SCMP while the clock counter operation is enabled (RTCA0CE = 1) according to the flow described below.

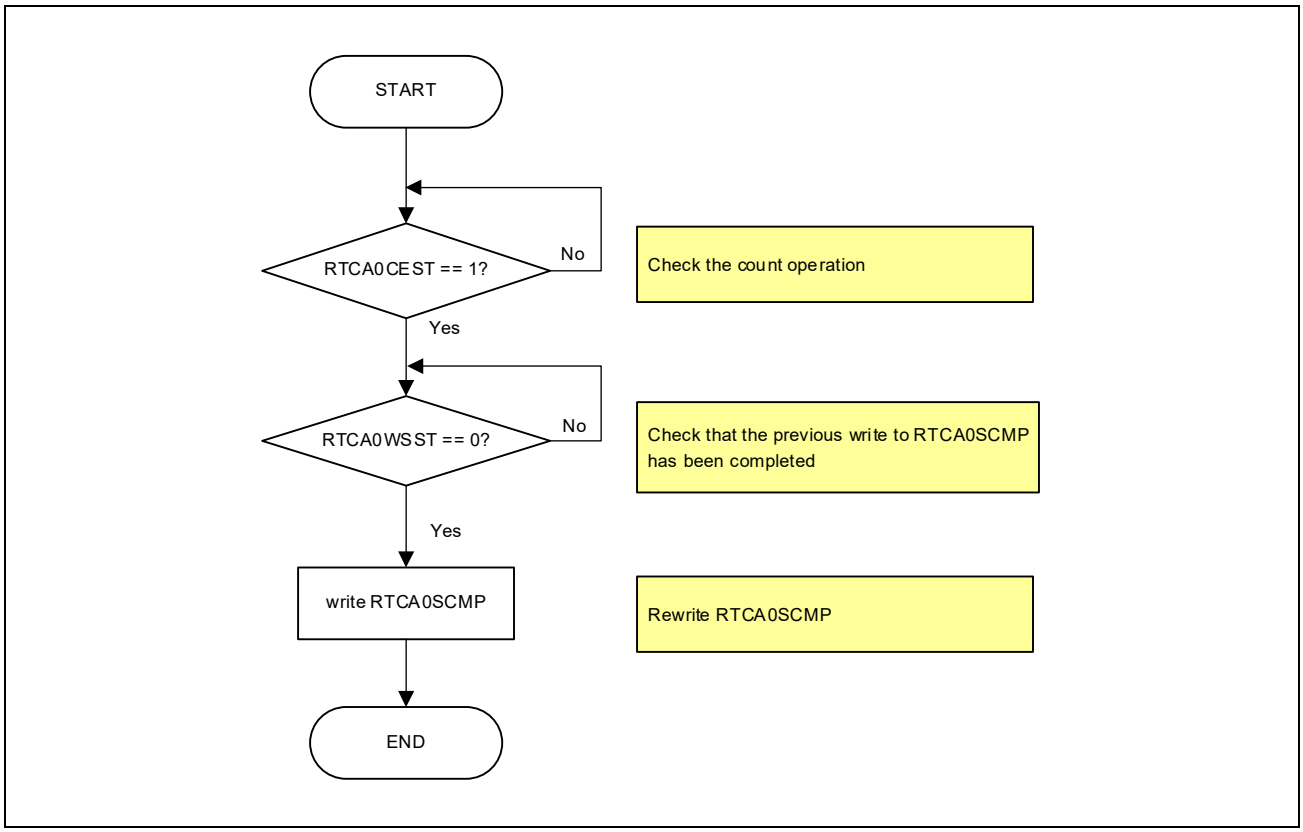


Figure 12.9 Writing to RTCA0SCMP

RTCA0WSST is set to “1” when RTCA0SCMP is written to while the clock counter operation is enabled (RTCA0CE = 1), and upon completion of RTCA0SCMP write (RTCA0SUBC overflow), it is cleared to “0”. Because RTCA0WSST displays “1” for up to 1 second for this reason, caution is required when polling RTCA0WSST.

12.5.1.8 Changing the Setting of Fixed Interval Interrupt During Clock Counter Operation

Be sure to change the fixed interval interrupt setting while the clock counter operation is enabled (RTCA0CE = 1) according to the following flow.

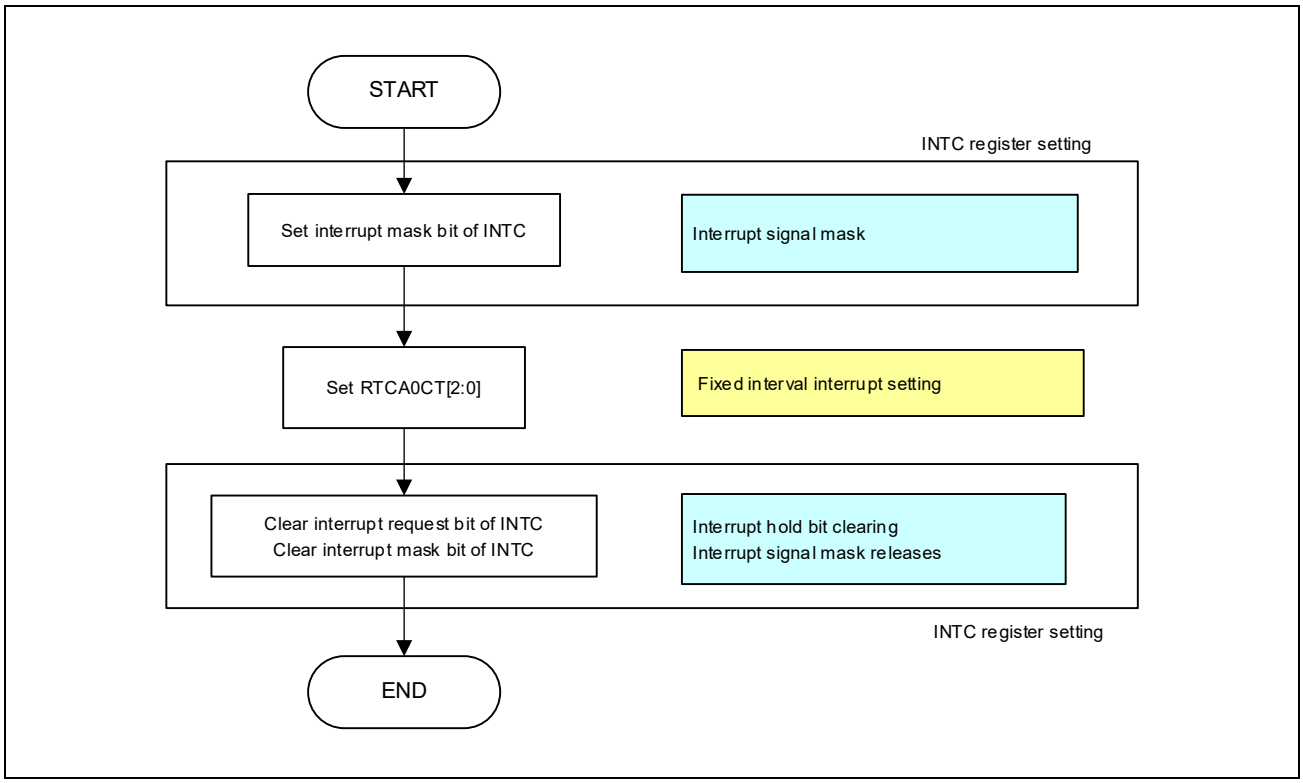


Figure 12.10 Setting of Fixed Interval Interrupt During Clock Counter Operation

12.5.1.9 Changing Alarm Setting During Clock Counter Operation

Be sure to change the alarm interrupt setting while the clock counter operation is enabled (RTCA0CE = 1) according to the following flow.

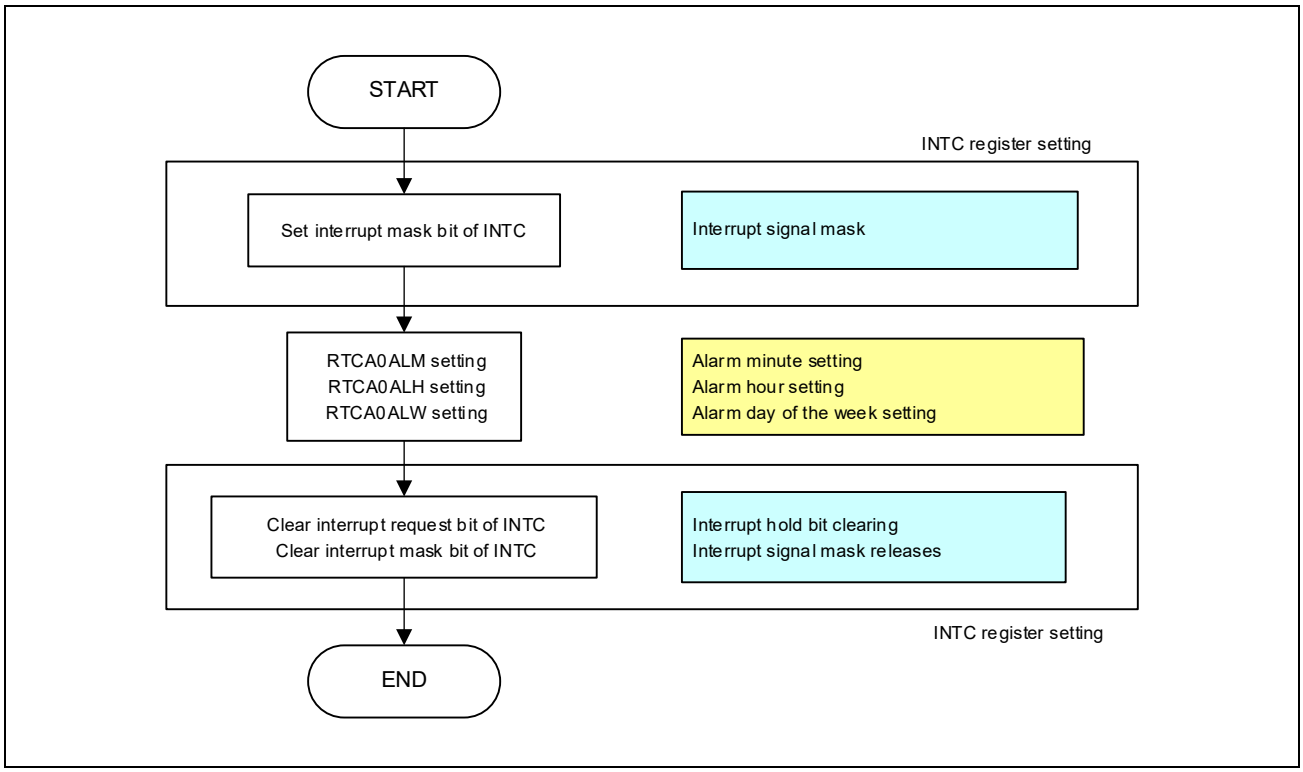


Figure 12.11 Changing the Alarm Setting During Clock Counter Operation

12.5.2 RTC Backup Mode

Power supply of RTC domain is separated from other domain. It can keep counting RTC controller while rest of domain is not powered.

RTC_PWRGOOD pin should be controlled properly when it enters or exits backup mode. Refer to power up/down sequence in Electrical Characteristics.

LVD reset circuits in RTC domain detects a low voltage, the circuit asserts reset and reset RTC controller. The circuit also works during cold power up of system as well.

CAUTION

PWRCTRL_OPPDIV of System Control register values should be changed to initial value in case of entering RTC backup mode.

12.5.3 Clock Error Correction

The clock error correction function of RTC controller is a function that corrects the oscillation frequency error of the connected resonator.

The correction value of RTCA0SUBC is determined by the RTCA0F6-RTCA0F0 bits of the RTCA0SUBU register. Whether RTCA0SUBC is incremented or decremented is decided by bit RTCA0F6, and the value is decided by bits RTCA0F5 to RTCA0F0.

- If RTCA0F6 = 0, the count value of RTCA0SUBC is incremented by the amount set with RTCA0F5 to RTCA0F0.
  Increment value calculation formula: (Setting value of RTCA0F5 to RTCA0F0 − 1) × 2
- If RTCA0F6 = 1, the count value of RTCA0SUBC is decremented by the amount set with RTCA0F5 to RTCA0F0.
  Decrement value calculation formula: (Inverted data of setting value of RTCA0F5 to RTCA0F0 + 1) × 2

[Example]

(1) Setting of RTCA0F6 = 0, RTCA0F5 to RTCA0F0 = 15h
(15h − 1) × 2 = 40
Increment count value of RTCA0SUBC by 40
Count value of RTCA0SUBC = 32768 + 40 = 32808

(2) Setting of RTCA0F6 = 1, RTCA0F5 to RTCA0F0 = 15h
Inverted data of “15h” = 2Ah
(2Ah + 1) × 2 = 86
Count value of RTCA0SUBC decremented by 86
Count value of RTCA0SUBC = 32768 − 86 = 32682

Bit RTCA0DEV determines the timing at which the setting of RTCA0F6 to RTCA0F0 above becomes effective. The value set with RTCA0F6 to RTCA0F0 is not reflected to the count value of RTCA0SUBC every time, but it is reflected to the count value of RTCA0SUBC when RTCA0SEC = 00, 20, 40 seconds if RTCA0DEV = 0, and when RTCA0SEC = 00 seconds if RTCA0DEV = 1.

[Example]

- Setting of 0010101b to RTCA0F6 to RTCA0F0
  When RTCA0DEV = 0, the count value of RTCA0SUBC is 32808 at 00, 20, and 40 seconds.
  At all other times, the count value is 32768.
  When RTCA0DEV = 1, the count value of RTCA0SUBC is 32808 at 00 seconds.
  At all other times, the count value is 32768.
- Correcting the count value of RTCA0SUBC every 20 seconds and 60 seconds instead of every second in this manner is so to allow adjustment to the deviation width of the resonator. The frequency range of the resonator that can actually be corrected is as follows.
  When RTCA0DEV = 0
  32.76180000 to 32.77420000 kHz
  When RTCA0DEV = 1
  32.76593333 to 32.77006667 kHz

The correctable frequency range when RTCA0DEV = 0 is three times as wide as that when RTCA0DEV = 1. However, three times higher accuracy can be set for RTCA0DEV = 1.

The setting values of bits RTCA0DEV and RTCA0F6 through RTCA0F0, as well as the frequencies that can be corrected at this time, are listed on the next page.

[Correctable Frequency Range when RTCA0DEV = 0]

RTCA0F6	RTCA0F5 to 0	RTCA0SUBC Correction Value	Connected Clock Frequency
0	000000b	No correction	—
0	000001b	No correction	—
0	000010b	Once every 20 s, RTCA0SUBC count value + 2	32.76810000 kHz
0	000011b	Once every 20 s, RTCA0SUBC count value + 4	32.76820000 kHz
0	000100b	Once every 20 s, RTCA0SUBC count value + 6	32.76830000 kHz
...	...	...	...
0	111011b	Once every 20 s, RTCA0SUBC count value + 120	32.77400000 kHz
0	111110b	Once every 20 s, RTCA0SUBC count value + 122	32.77410000 kHz
0	111111b	Once every 20 s, RTCA0SUBC count value + 124	32.77420000 kHz (upper limit)
1	000000b	No correction	—
1	000001b	No correction	—
1	000010b	Once every 20 s, RTCA0SUBC count value − 124	32.76180000 kHz (lower limit)
1	000011b	Once every 20 s, RTCA0SUBC count value − 122	32.76190000 kHz
1	000100b	Once every 20 s, RTCA0SUBC count value − 120	32.76200000 kHz
...	...	...	...
1	111011b	Once every 20 s, RTCA0SUBC count value − 6	32.76770000 kHz
1	111110b	Once every 20 s, RTCA0SUBC count value − 4	32.76780000 kHz
1	111111b	Once every 20 s, RTCA0SUBC count value − 2	32.76790000 kHz

[Correctable frequency range when RTCA0DEV = 1]

RTCA0F6	RTCA0F5 to 0	RTCA0SUBC Correction Value	Connected Clock Frequency
0	000000b	No correction	—
0	000001b	No correction	—
0	000010b	Once every 60 s, RTCA0SUBC count value + 2	32.76803333 kHz
0	000011b	Once every 60 s, RTCA0SUBC count value + 4	32.76806667 kHz
0	000100b	Once every 60 s, RTCA0SUBC count value + 6	32.76810000 kHz
...	...	...	...
0	111011b	Once every 60 s, RTCA0SUBC count value + 120	32.77000000 kHz
0	111110b	Once every 60 s, RTCA0SUBC count value + 122	32.77003333 kHz
0	111111b	Once every 60 s, RTCA0SUBC count value + 124	32.77006667 kHz (upper limit)
1	000000b	No correction	—
1	000001b	No correction	—
1	000010b	Once every 60 s, RTCA0SUBC count value − 124	32.76593333 kHz (lower limit)
1	000011b	Once every 60 s, RTCA0SUBC count value − 122	32.76596667 kHz
1	000100b	Once every 60 s, RTCA0SUBC count value − 120	32.76600000 kHz
...	...	...	...
1	111011b	Once every 60 s, RTCA0SUBC count value − 6	32.76790000 kHz
1	111110b	Once every 60 s, RTCA0SUBC count value − 4	32.76793333 kHz
1	111111b	Once every 60 s, RTCA0SUBC count value − 2	32.76796667 kHz

Section 13 Watchdog

13.1 Overview

The watchdog timer is based on a free running 12-bit decrementing counter with reload register: when the counter reaches 000h, the output can be used to activate a system reset or as an interrupt.

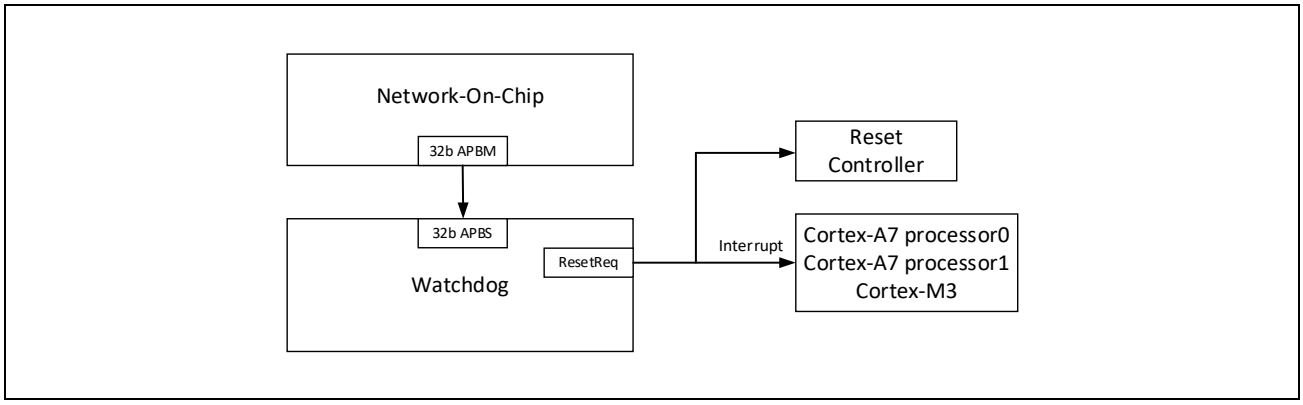


Figure 13.1 Watchdog Interfaces and Connections

The watchdog timers consist of the following:

- Free running 12-bit decrementing counters with reload register.
- Counters are retriggered by the CPUs (CA7 processor0, CA7 processor1, CM3)
- Clock source is the clock divided by a prescaler (2 or 2^{14})
- Output can be used to activate a system reset with the setting of RSTEN register in System Controller or can be used as an interrupt
- Stop of watchdog effect while CPU is being stopped by debugger (e.g. by breakpoint execution)

13.2 Signal Interfaces

Signal Name	Input Output	Description
Clock		
WDOGA7_PCLK WDOGM3_PCLK	Input	Internal bus clock (APB), no clock gating Half frequency of NoC clock.
Interrupt		
WDT_CA7_p0_reset_Int WDT_CA7_p1_reset_Int WDT_CM3_reset_Int	Output	Pulse interrupt, Active High

13.3 Register Map

13.3.1 Register Map CA7 Processor0 Watchdog

Table 13.1 Register Map of CA7 Processor0 Watchdog

Address	Register Symbol	Register Name
4000 8000h	CTRL_RETRIGGER	Control and Retrigger Register

13.3.2 Register Map CA7 Processsor1 Watchdog

Table 13.2 Register Map of CA7 Processor1 Watchdog

Address	Register Symbol	Register Name
4000 9000h	CTRL_RETRIGGER	Control and Retrigger Register

13.3.3 Register Map CM3 Watchdog

Table 13.3 Register Map of CM3 Watchdog

Address	Register Symbol	Register Name
4000 A000h	CTRL_RETRIGGER	Control and Retrigger Register

13.4 Register Description

13.4.1 CTRL_RETRIGGER — Control and Retrigger Register

This register can be set only once. After setting of the value, only retrigger is possible. Write this register with any value to retrigger counter reload with the value of WDRV.

Address: 4000 8000h (CA7 Processor0 Watchdog)
4000 9000h (CA7 Processor1 Watchdog)
4000 A000h (CM3 Watchdog)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	WDSI		WDE	PSF	WDRV											
Value after reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Table 13.4 CTRL_RETRIGGER Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b16	Reserved		R
b15, b14	WDSI	Watchdog timer secure ID 0x2: Allowed to write access to the lower 14 bits of this register	W
b13	WDE	Watchdog timer enable flag 0: Watchdog disabled 1: Enable Watchdog	R/W
b12	PSF	Prescaler factor 0: Prescaler is set to subdivide clock by 2 1: Prescaler is set to subdivide clock by 2^{14}	R/W
b11 to b0	WDRV	Watchdog timer reload value WDRV = ((fpclk × WDInterval) / PSF) − 1 fpclk: Half frequency of NoC clock [Hz] WDInterval: Interval time [s] PSF: 2^{14} or 2 (depends on PSF bit)	R/W

13.5 Operation

The reload value is loaded to the counter register each time the watchdog timer is retriggered by write with any value to CTRL_RETRIGGER register. The counter can be triggered by one of the internal CPU. If the counter register reaches 000h, the configurable output is activated as a system reset or an interrupt.

The watchdog timer interval is determined by the pre-scale factor and the value in the reload register. The pre-scale factor and reload value can be written just once. To protect unintentional write access, 3’b101 must be written to bit 15 to bit 13 together with the pre-scale factor and the reload value.

Section 14 Mailbox (IPCM)

14.1 Overview

The IPCM provides 3 mailboxes with control logic and interrupts generation to support inter-processor communication between Cortex-A7 and Cortex-M3. The IPCM has three interrupt output connected to every interrupt controller in the system, enabling any core to send a message to any other core; each mailbox contains seven data registers to hold the message.

- 3 mailboxes, each comprising seven 32-bit data registers to store the message.
- 3 sets of read-only interrupt status registers, one for each interrupt

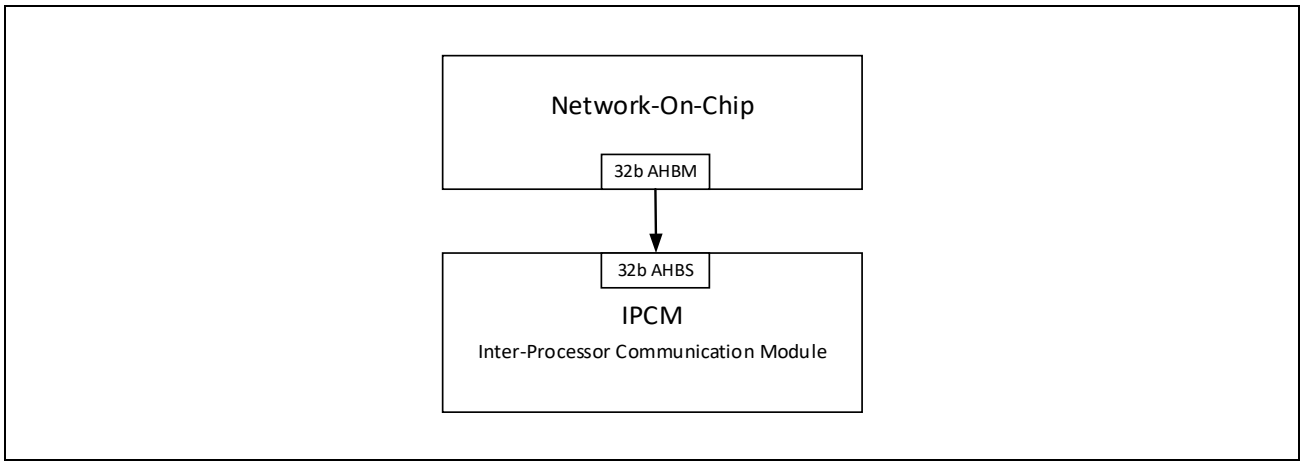


Figure 14.1 IPCM Interfaces and Connections

14.2 Signal Interfaces

Signal Name	Input Output	Description
Clock		
MBOX_HCLK	Input	Internal bus clock (AHB), no clock gating
Interrupt		
IPCM_Int[n]	Output	Level sensitive interrupt, Active High

Note: n = 0..2

14.3 Register Map

Table 14.1 IPCM Register Map

Address	Register Symbol	Register Name
4000 B000h + 40h × n	IPCM[n]SOURCE (n = 0..2)	Mailbox[n] Source Register
4000 B004h + 40h × n	IPCM[n]DSET (n = 0..2)	Mailbox[n] Destination Set Register
4000 B008h + 40h × n	IPCM[n]DCLEAR (n = 0..2)	Mailbox[n] Destination Clear Register
4000 B00Ch + 40h × n	IPCM[n]DSTATUS (n = 0..2)	Mailbox[n] Destination Status Register
4000 B010h + 40h × n	IPCM[n]MODE (n = 0..2)	Mailbox[n] Mode Register
4000 B014h + 40h × n	IPCM[n]MSET (n = 0..2)	Mailbox[n] Mask Set Register
4000 B018h + 40h × n	IPCM[n]MCLEAR (n = 0..2)	Mailbox[n] Mask Clear Register
4000 B01Ch + 40h × n	IPCM[n]MSTATUS (n = 0..2)	Mailbox[n] Mask Status Register
4000 B020h + 40h × n	IPCM[n]SEND (n = 0..2)	Mailbox[n] Send Register
4000 B024h + 40h × n + 4h × k	IPCM[n]DR[k] (n = 0..2) (k = 0..6)	Mailbox[n] Data Register [k]
4000 B800h + 8h × n	IPCMMIS[n] (n=0..2)	Masked Interrupt[n] Status Register
4000 B804h + 8h × n	IPCMRIS[n] (n=0..2)	Raw Interrupt[n] Status Register
4000 B900h	IPCMCFGSTAT	Configuration Status Register
4000 BF00h	IPCMTCR	Integration Test Control Register
4000 BF04h	IPCMTOR	Integration Test Output Register

14.4 Register Description

14.4.1 IPCM[n]SOURCE — Mailbox[n] Source Register (n = 0..2)

The read/write IPCM[n]SOURCE Register defines which core the message came from. The register is programmed with the Channel ID to identify which interrupt line to send the acknowledge interrupt through bit-wise encoding and can only be programmed to this value from 0x00000000. When the register is programmed, it must be cleared to 0x00000000 before it can be reprogrammed. The software must ensure that IPCM[n]SOURCE is only programmed to a one-hot encoded value.

Address: 4000 B000h + 40h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	SRC_SET		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.2 IPCM[n]SOURCE Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved		R
b2 to b0	SRC_SET	Used to define which core is the source and which interrupt line is asserted for the acknowledge interrupt	R/W

14.4.2 IPCM[n]DSET — Mailbox[n] Destination Set Register (n = 0..2)

The write-only IPCM[n]DSET Register sets bits in virtual Mailbox Destination Register. Virtual Mailbox Destination Register indicates the destination core and is reflected in the Mailbox Destination Status Register. It can only be written to after the Mailbox Source Register is defined.

Address: 4000 B004h + 40h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	DEST_SET		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.3 IPCM[n]DSET Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved		R
b2 to b0	DEST_SET	Used to set bits in the IPCM[n]DSTATUS Register by writing 1 to the corresponding bit.	W

14.4.3 IPCM[n]DCLEAR — Mailbox[n] Destination Clear Register (n = 0..2)

The write-only IPCM[n]DCLEAR Register clears bits in virtual Mailbox Destination Register. Virtual Mailbox Destination Register indicates the destination core and is reflected in the Mailbox Destination Status Register. It can only be written to after the Mailbox Source Register is defined.

Address: 4000 B008h + 40h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	DEST_CLR		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.4 IPCM[n]DCLEAR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved		R
b2 to b0	DEST_CLR	Used to clear bits in the Mailbox Destination Register by writing 1 to the corresponding bit.	W

14.4.4 IPCM[n]DSTATUS — Mailbox[n] Destination Status Register (n = 0..2)

The read-only IPCM[n]DSTATUS Register contains the current status of virtual Mailbox Destination Register. When set, the Mailbox Destination Registers determine which cores to send the message to through bit-wise encoding using the Channel ID for each core. For cores that use multiple Channel IDs, only a single Channel ID is used per message. The Mailbox Destination Registers are cleared in Auto Acknowledge Mode by destination cores to clear the mailbox interrupts to each core. When not in Auto Acknowledge mode, the Mailbox Destination Registers are only cleared by the source core when the mailbox is being reassigned. The Mailbox Destination Registers are cleared automatically by the mailbox regardless of which mode it is in when the Mailbox Source Register is cleared.

Address: 4000 B00Ch + 40h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	DEST_STAT		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.5 IPCM[n]DSTATUS Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved		R
b2 to b0	DEST_STAT	Gives the status of the Mailbox Destination Register. Defines which interrupt output to assert for the message.	R

14.4.5 IPCM[n]MODE — Mailbox[n] Mode Register (n = 0..2)

The read/write IPCM[n]MODE Register defines how the mailbox is used. The register can only be written to when the mailbox is assigned, indicated by a bit in the Mailbox Source Register being set.

Address: 4000 B010h + 40h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	AUTO_LINK	AUTO_ACK
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.6 IPCM[n]MODE Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b2	Reserved		R
b1	AUTO_LINK	Set to enable Auto Link.	R/W
b0	AUTO_ACK	Set to enable Auto Acknowledge.	R/W

14.4.6 IPCM[n]MSET — Mailbox[n] Mask Set Register (n = 0..2)

The write-only IPCM[n]MSET Register sets bits in virtual Mailbox Mask Register. Virtual Mailbox Mask Register indicates the mask state and is reflected in the Mailbox Mask Status Register. It can only be written to after the Mailbox Source Register is defined.

Address: 4000 B014h + 40h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	MASK_SET		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.7 IPCM[n]MSET Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved		R
b2 to b0	MASK_SET	Used to set bits in the Mailbox Mask Register by writing 1 to the corresponding bit.	W

14.4.7 IPCM[n]MCLEAR — Mailbox[n] Mask Clear Register (n = 0..2)

The write-only IPCM[n]MCLEAR Register clears bits in virtual Mailbox Mask Register. Virtual Mailbox Mask Register indicates the mask state and is reflected in the Mailbox Mask Status Register. It can only be written to after the Mailbox Source Register is defined.

Address: 4000 B018h + 40h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	MASK_CLR		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.8 IPCM[n]MCLEAR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved		R
b2 to b0	MASK_CLR	Used to clear bits in the Mailbox Mask Register by writing 1 to the corresponding bit.	W

14.4.8 IPCM[n]MSTATUS — Mailbox[n] Mask Status Register (n = 0..2)

The read-only IPCM[n]MSTATUS Register contains the current status of virtual Mailbox Mask Register. Each core is assigned its own bit. When set, the Mailbox Mask Registers enable the interrupts to each core through bit-wise encoding for each of the Channel IDs. These bits reset to 0, disabling the interrupts. When cleared, the Mailbox Mask Registers disable the interrupts, enabling the cores to use polling rather than interrupts for messaging. The Mailbox Mask Registers are all cleared when the Mailbox Source Register is cleared.

Address: 4000 B01Ch + 40h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	MASK_STAT		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.9 IPCM[n]MSTATUS Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved		R
b2 to b0	MASK_STAT	Gives the status of the Mailbox Mask Register For each bit position: 0: Mailbox interrupt disabled, polling used instead 1: Mailbox interrupt enabled	R

14.4.9 IPCM[n]SEND — Mailbox[n] Send Register (n = 0..2)

The read/write IPCM[n]SEND Register sends the message to either the source or destination cores. The Mailbox Send Register bits can only be written to after the Mailbox Source Register is defined:

- setting bit 0 generates an interrupt to the destination core(s)
- setting bit 1 generates an interrupt to the source core.

In Auto Acknowledge mode, when the Mailbox Destination Status Register changes from being non-zero to zero and the Mailbox Send Register currently contains 2'b01, the mailbox automatically changes the register to 2'b10, triggering the Auto Acknowledge interrupt back to the source core.

The Mailbox Send Registers are cleared when the Mailbox Source Register is cleared.

Address: 4000 B020h + 40h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	SEND	
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.10 IPCM[n]SEND Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b2	Reserved		R
b1, b0	SEND	Send message: 2'b00: Inactive 2'b01: Send message to destination core(s) 2'b10: Send message to source core 2'b11: Invalid, unpredictable behavior	R/W

14.4.10 IPCM[n]DR[k] — Mailbox[n] Data Register [k] (n = 0..2) (k = 0..6)

The read/write IPCM[n]DR[k] Register holds the message. The Mailbox Data Registers can only be written to after the Mailbox Source Register is defined and are cleared when the Mailbox Source Register is cleared.

Address: 4000 B024h + 40h × n + 4h × k

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	DATA															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	DATA															
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.11 IPCM[n]DR[k] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b0	DATA	Message data	R/W

14.4.11 IPCMMIS[n] — Masked Interrupt[n] Status Register (n = 0..2)

The read-only IPCMMIS[n] Register contains the current mailbox status for every interrupt identified by the address encoding. This enables each core to read a single register to determine which mailbox caused the interrupt.

Address: 4000 B800h + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	MASK_INT_STAT		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.12 IPCMMIS[n] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved		R
b2 to b0	MASK_INT_STAT	Masked interrupt status For each bit position: 0: No Interrupt 1: Interrupt generated	R

14.4.12 IPCMRIS[n] — Raw Interrupt[n] Status Register (n = 0..2)

The read-only IPCMRIS[n] Register indicates the unmasked interrupt status of each mailbox for each core.

Address: 4000 B804h + 8h × n

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	RAW_INT_STAT		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.13 IPCMRIS[n] Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved		R
b2 to b0	RAW_INT_STAT	Raw interrupt status For each bit position: 0: No interrupt requested 1: Interrupt requested	R

14.4.13 IPCMCFGSTAT — Configuration Status Register

The read-only IPCMCFGSTAT Register indicates the hardware configuration options chosen for implementation of the IPCM.

Address: 4000 B900h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	MAILBOXES					
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	INTERRUPTS						—	—	—	—	—	DATA_WORDS		
Value after reset	0	0	0	0	0	0	1	1	0	0	0	0	0	1	1	1

Table 14.14 IPCMCFGSTAT Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b22	Reserved		R
b21 to b16	MAILBOXES	Number of Mailboxes	R
b15, b14	Reserved		R
b13 to b8	INTERRUPTS	Number of Interrupts	R
b7 to b3	Reserved		R
b2 to b0	DATA_WORDS	Number of data registers	R

14.4.14 IPCMTCR — Integration Test Control Register

The read/write IPCMTCR Register controls the IPCM integration test mode. When ITEN = 1, the IPCM is placed in integration test mode.

Address: 4000 BF00h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	ITEN
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.15 IPCMTCR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b1	Reserved		R
b0	ITEN	Integration test enable: 0: Integration test mode disabled 1: Integration test mode enabled	R/W

14.4.15 IPCMTOR — Integration Test Output Register

The read/write IPCMTOR Register enables the output port signals of the IPCM to be driven directly rather than from their normal internal logic source when in integration test mode, that is, when ITEN = 1 in the IPCMTCR Register.

Address: 4000 BF04h

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	—	—	—	—	—	—	—	—	—	—	—	—	—	INTTEST		
Value after reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 14.16 IPCMTOR Register Contents

Bit Position	Bit Name	Function	R/W
b31 to b3	Reserved		R
b2 to b0	INTTEST	IPCM_Int[2:0] output	R/W

14.5 Operation

The IPCM generates interrupts under software control. These interrupts normally have data associated with them and can be directed to one or more interrupt outputs. Each interrupt output corresponds directly to a bit in every Mailbox Source, Mailbox Destination, and Mailbox Mask Register in every mailbox in the IPCM. Mailbox Destination and Mailbox Mask Registers are virtual. These registers therefore control which interrupt lines are asserted when messages are sent and acknowledged.

14.5.1 Channel ID

The Channel ID is defined as the one-hot encoded value that corresponds to a specific interrupt output from the IPCM. In the current configuration, the IPCM has 3 interrupt outputs with 3 corresponding Channel IDs. The Channel ID programs the Mailbox Source, Mailbox Destination, and Mailbox Mask Registers.

The following table shows the mapping of Channel ID to Interrupt output.

Table 14.17 Channel ID to Interrupt Output

Channel ID	Interrupt Output
0x00000001	IPCM_Int[0]
0x00000002	IPCM_Int[1]
0x00000004	IPCM_Int[2]

14.5.2 Defining Source Core

A core must obtain a mailbox to send a message. To do this the core writes one of its Channel IDs to the Mailbox Source Register and then reads the Mailbox Source Register back again to check whether the write was successful. The Mailbox Source Register must only contain a one-hot encoded value, that is, a single Channel ID. The software must ensure that only a one-hot encoded number is written to the Mailbox Source Register. You can only clear the Mailbox Source Register after it is programmed. Any writes other than 0x00000000 are ignored. This mechanism guarantees that only a single core has control of the mailbox at any one time.

A core gives up a mailbox, when it is no longer required, by clearing the Mailbox Source Register. Clearing the Mailbox Source Register also clears all the other registers in the mailbox. This guarantees that a mailbox is always cleared when it is newly allocated.

14.5.3 Defining Destination Core

The Mailbox Destination Register has separate Set and Clear write locations to enable you to set individual bits in the Mailbox Destination Register without using read-modify-write transfers. You can set a single bit in the Mailbox Destination Register by writing that bit to the Destination Set Register. This causes the hardware to OR that bit with the current Mailbox Destination Register value. Similarly, you can clear a single bit in the Mailbox Destination Register by writing that bit to the Destination Clear Register.

When the source core defines the mode of a mailbox, it defines which other cores are to receive the message by programming the OR of all the Channel IDs into the Mailbox Destination Register. If a core has more than one Channel ID only one is used per message. You can only write to the Mailbox Destination Register after the Mailbox Source Register is defined.

14.5.4 Using the Mailbox Mask Register

The Mailbox Mask Register uses separate Set and Clear registers for modification similar to the Mailbox Destination Register. The Mailbox Mask Register enables the interrupt outputs. To enable interrupts for a particular mailbox, a core writes its Channel ID to the Mask Set location. The interrupt for that mailbox can be masked out by writing the same Channel ID to the Mask Clear location. You can only write to the Mailbox Mask Register locations after the Mailbox Source Register is defined.

14.5.5 Using the Mailbox Send Register

A message is sent by setting bit 0 of the Mailbox Send Register. This triggers the interrupt to the destination core. Clearing this bit clears the interrupt to the destination core. The acknowledge message is sent to the source core by setting bit 1 of the Mailbox Send Register. Clearing this bit clears the interrupt to the source core. You can use one write to clear bit 0 and set bit 1 in the Mailbox Send Register, although this is not mandatory. You cannot set bit 1 then clear bit 0 because 2'b11 is an invalid value for the Mailbox Send Register. The Mailbox Send Register can only be written to after the Mailbox Source Register is defined.

14.5.6 Mailbox Data Registers

The Mailbox Data Registers are general-purpose 32-bit registers that contain the message and can only be written to after the Mailbox Source Register is defined. The Mailbox Data Registers are normally written to before sending the message.

14.5.7 Setting Mode

The Mailbox Mode Register controls how the acknowledge interrupt is sent back to the source core, and whether the current mailbox is linked to the next mailbox in the IPCM. The Mailbox Mode Register has two bits and you can only write to it after the Mailbox Source Register is defined.

Auto Acknowledge

In Auto Acknowledge mode, an acknowledge interrupt is automatically sent to the source core after the final destination core has cleared its interrupt. Destination cores must clear their interrupts by writing their Channel ID value to the Destination Clear location. This clears their Channel ID from the Mailbox Destination Register. When the Mailbox Destination Register finally reaches zero, indicating that all destination cores have cleared their interrupts, the mailbox automatically detects this, clears bit 0 and sets bit 1 of the Mailbox Send Register. The source core then receives the acknowledge interrupt. The data associated with an Auto Acknowledge is the same as that for the original message.

When Auto Acknowledge mode is disabled, the acknowledge interrupt is optional. The destination core must clear its interrupt by clearing bit 0 of the Mailbox Send Register. Only when the destination core sets bit 1 of the Mailbox Send Register does the source core obtain its acknowledge interrupt, indicating that the destination core has finished with the message. You can only disable Auto Acknowledge mode when there is only one destination core, where there is also a possibility of updating the message for the acknowledge.

Auto Link

Auto Link provides a mechanism to link mailboxes together so that when a message is acknowledged in one mailbox, the next message is sent from the linked mailbox instead of interrupting the source core. When Auto Link is enabled, the destination core clears bit 0 and sets bit 1 of the Mailbox Send Register in the usual way, but the acknowledge interrupt to the source core is masked out and Mailbox Send Register bit 0 is set in the next mailbox, sending that message.

In this mode, a source core can allocate multiple mailboxes to itself, link them together by setting the Auto Link bits and preload messages in all the mailboxes. When the first message is sent, it is not acknowledged until all the messages

have been sent. There is no restriction on the destinations of these messages or whether Auto Acknowledge is enabled when Auto Link is used. When Auto Link is disabled, the source core is interrupted if an acknowledge interrupt is sent that has no effect on any other mailbox.

14.5.8 Interrupts and Status Registers

When a core receives an IPCM interrupt, it determines which mailbox triggered it by reading the Masked Interrupt Status Register related to that interrupt line. Each Masked Interrupt Status Register contains 3 bits, each bit referring to a single mailbox.

If a core is using a mailbox in polled mode, it can use the Raw Interrupt Status Register to indicate which mailbox requires attention.

Each mailbox can generate up to 3 interrupts, one for each Channel ID. The number of interrupts defines the number of bits in the Mailbox Source Register, Mailbox Destination Register, and Mailbox Mask Register. In Figure Mailbox Interrupt mapping, the IPCM has 3 interrupt outputs. Mailbox0 generates bit 0 of the IPCMMIS0-2 buses, while Mailbox2 generates bit 2 of the IPCMMIS0-2 buses.

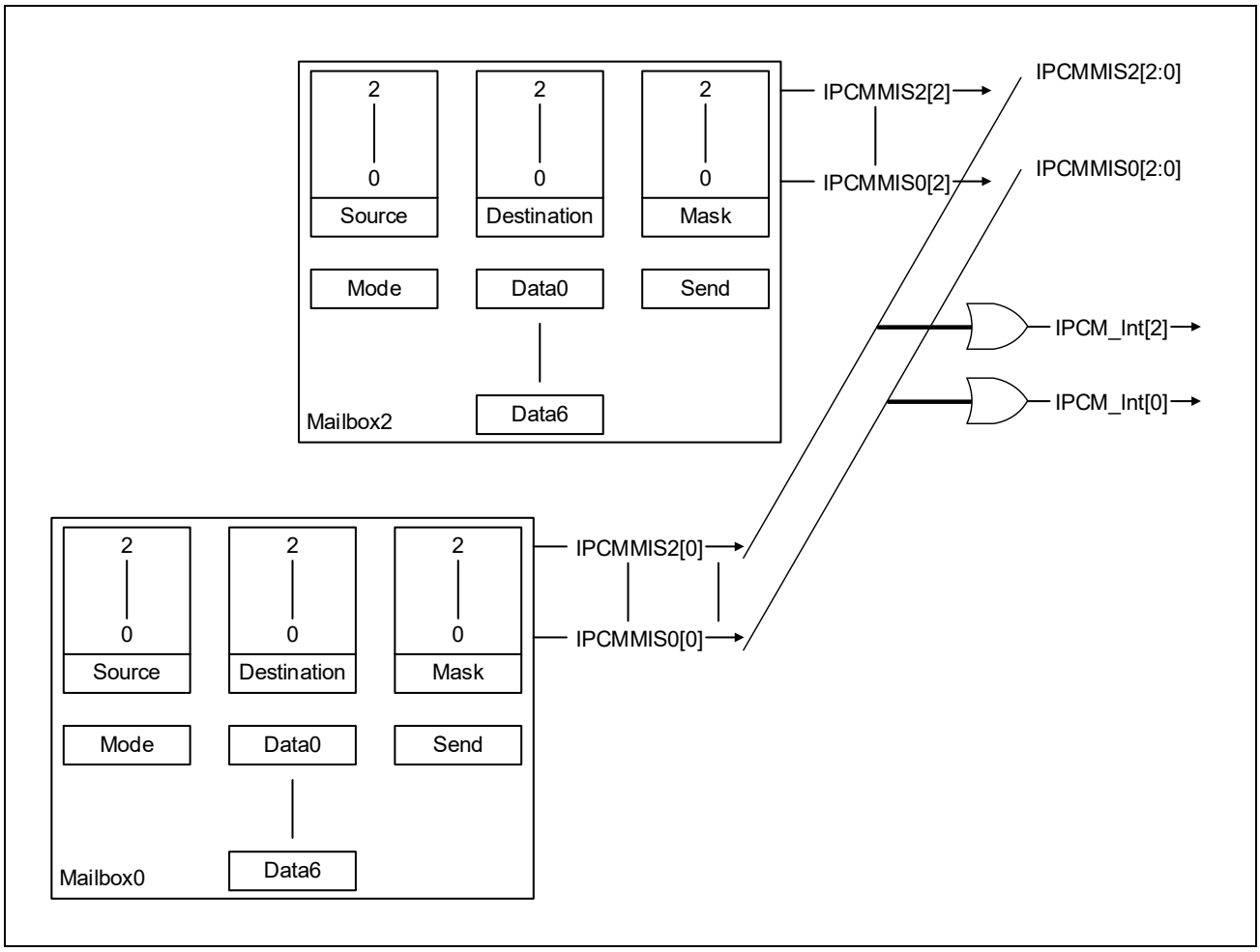


Figure 14.2 Mailbox Interrupt Mapping

Multiple mailboxes are grouped together as shown in the figure above to form the 3-bit IPCM interrupt bus, IPCM_Int[2:0]. All the interrupt bits from each mailbox relating to a single Channel ID are grouped together to form the masked interrupt status buses, IPCMMIS0[2:0] to IPCMMIS2[2:0]. The bits within these buses are then ORed together to form the IPCM interrupt bus, IPCM_Int[2:0].

Below figure shows how each mailbox status is generated.

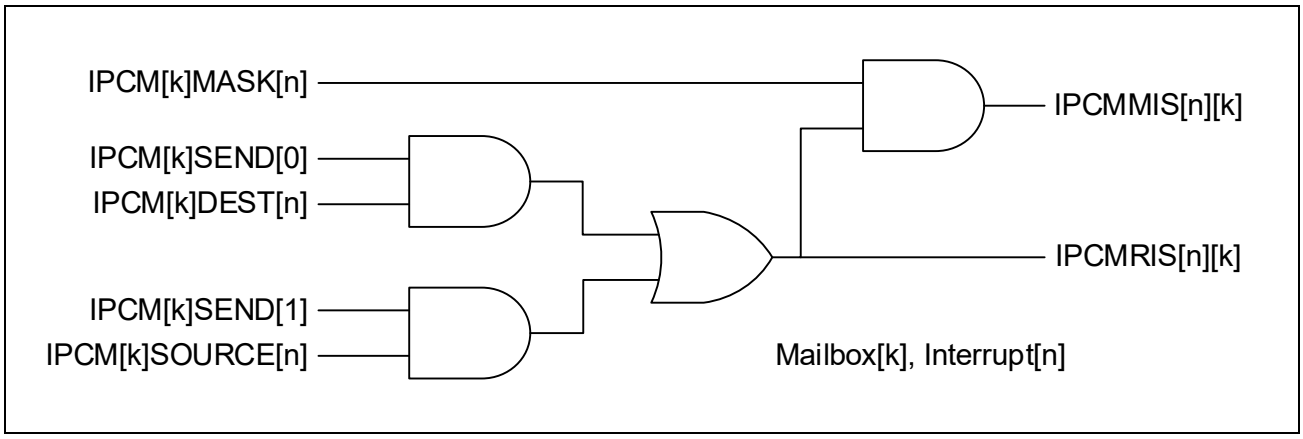


Figure 14.3 Interrupt Status of Mailbox[k], Interrupt[n]

14.5.9 Configuration Status Register

The selected configuration options (3 mailbox, 7 data registers per mailbox, 3 interrupts) define the read-only Configuration Status Register, enabling software to determine the IPCM configuration by reading this register. This enables a generic IPCM software driver to determine how to use each IPCM instance within a system.

14.5.10 Usage Constraints

There are several valid use models for a mailbox and some constraints under which they can be used. Messages can be sent to:

- Multiple cores
  If a message is sent to multiple cores, it must use the Auto Acknowledge feature and data must not be modified for the acknowledge. Destination cores must clear their interrupts by writing their Channel ID to the Destination Clear Register.
- Single core
  If there is only a single destination core, the Auto Acknowledge mode is optional. If it disables the Auto Acknowledge mode, the acknowledge is optional, although an acknowledge normally happens, and the Mailbox Data Register can optionally be updated. When Auto Acknowledge is disabled, the destination core must clear its interrupt by clearing bit 0 of the Mailbox Send Register. It can only use the Auto Link feature when there is an acknowledge.

It can use the Auto Link feature with either:

- Auto Acknowledge enabled
  The mailbox automatically sets the acknowledge when the final destination core clears its interrupt.
- Auto Acknowledge disabled
  The destination core must send the acknowledge.

REVISION HISTORY	RZ/N1D Group, RZ/N1S Group, RZ/N1L Group User's Manual: System Control and Peripheral

Rev.	Date	Description	
		Page	Summary
0.50	Jun 30, 2017	—	First Edition issued
0.80	Oct 31, 2017	26 to 28	Section 1, all chapter title, corrected
		28	1.2.2, CM_FCLK: no clock gating, deleted. CM3_STCLK: no clock gating, added. NMI_CORTEXM3: active high, added
		28	1.2.3, description, modified
		30, 31	2.2, Table 2.1 to 2.3, corrected and modified
		35	3.4.3, Table 3.5: b19 to b16 of Reserved, R → R/W, corrected
		54	5.1, Figure 5.1, revised
		57	5.2.3, Figure 5.5, modified
		60	6.1, description, modified
		62	6.2, DDR_HCLK: no clock gating, added
		219	7.4.1, CMD_SEQ → cmd_seq, corrected
		222	7.4.3, MEMx_ST → MEM[n]_ST, corrected
		251	7.4.34, lun_sel → mlun_sel, corrected
		273	7.5.3, corrected (DEL0 → DELAY0, DEL1 → DELAY1)
		299	7.6, corrected (ADDRx_AUTO_INCR → ADDR[n]_AUTO_INCR, DEVn_PTR → DEV[n]_PTR, DEVn_SIZE → DEV[n]_SIZE)
		300	7.6, Figure 7.31, corrected (DEV_PTRx → DEV[n]_PTR, DEV_SIZEx → DEV[n]_SIZE)
		301	7.6.1, corrected (MEMx_WP → MEM[n]_WP, MEMx_ST → MEM[n]_ST)
		302	7.6.2, MEMx_ST → MEM[n]_ST, corrected
		303	7.6.3, corrected (MEMx_WP → MEM[n]_WP, MEMx_ST → MEM[n]_ST)
		305	7.6.4 (6), corrected (MEMx_WP → MEM[n]_WP, the 0 address register → the address register 0)
		308	7.6.6, Figure 7.38: figure title, corrected
		313	7.6.9, corrected (ecc → ECC, MEMx_WP → MEM[n]_WP). Figure 7.43, corrected (write COMMAND 0x3000002A → write COMMAND 0x0010800C).
		317	7.8.1, corrected (ADDRx_COL → ADDR[n]_COL, ADDRx_ROW → ADDR[n]_ROW)
		319	7.8.4, newly added
		323, 325	8.4.1, Table 8.3: mstr_baud_div_fld: 1/2 (mstr_baud_div_fld = 0000b) is not available, corrected. b2, b1: SPI clock → QUAD_CLK, revised. enb_qspi_fld: spi_enable → QSPI Enable, corrected.
		334	8.4.12, Table 8.14: description of Function, modifed
		353	8.6.1, SPI clock → QUAD_CLK, revised
		396	9.4.26, b7 to b0: R → R/W, modified
		401	9.4.30, Table 9.32: Bit Position: b5 → b6, b5, corrected
		420	10.2, Description, modified
		421	10.3, PLL USB → USDBPLL, USB_CFG → CFG_USB, revised
		491	10.5.5.6, Table 10.75: PCI_WIN1_BASEADR: PCI_AHB_WIN_SIZE → PCI_AHB_WIN1_SIZE, corrected
		494	10.5.6.1, Table 10.79: PCI_WIN1_BASEADR: PCI_AHB_WIN_SIZE → PCI_AHB_WIN1_SIZE, corrected
		495	10.5.6.2, Table 10.80: PCI_AHB_WIN2_SIZE → PCI_AHB_WIN2_EN, corrected
		503	10.5.6.8, Table 10.86: PCICLK_MASK: description of Function, modified
		505	10.5.7.1, chapter title, curected

Rev.	Date	Description	
		Page	Summary
0.80	Oct 31, 2017	507	10.5.7.2, Table 10.89: USB address register → Frame Number & USB Address Register, corrected
		523	10.5.7.15, Table 10.102: EP[m]_ONAK: EP0 read register → EP[m] Read Register
		529	10.5.7.18, Table 10.105: EP[m]_DEND_SET and EP[m]_BURST_SET: description, revised
		540	10.5.8.8, Table 10.117: EP[m] length & DMA counter register → EP[m] Length & DMA Count Register, corrected
		541	10.5.8.9, Address: 4001 F104h + 10h × (m − 1) → 4001 F114h + 10h × (m − 1), corrected
		542	10.5.8.10, Address: 4001 F108h + 10h × (m − 1) → 4001 F118h + 10h × (m − 1), corrected. Table 10.118: EP[m]DCR1.EP[m]_REQEN → EP[m]_REQEN, revised.
		548	10.6.2, Table 10.121: (6): 800_0000h → 8000_0000h, corrected
		549	10.6.2.1, changed chapter number (10.6.3 → 10.6.2.1, 10.6.4 → 10.6.2.2)
		549	10.6.2.1, Table10.122, corrected
		553	10.6.4.1(6), Table 10.130: EP0/EPn → EP0/EP[m], corrected
		556	10.6.5.1(2), OCI1/OCI2 → USB_OCI1/OCI2, corrected
		557	10.6.5.2, corrected (OCI1/PPON1 → USB_OCI1/PPON1, OCI2/PPON2 → USB_OCI2/PPON2, PPON1/PPON2 → USB_PPON1/PPON2, PPON1 → USB_PPON1, PPON2 → USB_PPON2)
		557	10.6.5.2, Table 10.133: the right end of header line: PPON2 → USB_PPON1, corrected. description of PPON Output Pin behavior, modified.
		559	10.6.5.4, Figure 10.10, revised
		560	10.6.6.1, description, modified
		561	10.6.6.3, Figure 10.12: USB Contzrol Register Setting → USB Control Register Setting, corrected
		563 to 566	10.6.7.1, Figure 10.13, 10.14, 10.15 and 10.16: PCLK → USB_PCICLK, corrected
		567 to 569	10.6.7.2, Figure 10.17, 10.18 and 10.19, revised
		570, 572, 573	10.6.7.3, Figure 10.20, 10.22 and 10.23: USB macro → USB subsystem, corrected
		573	10.6.7.3, NOTE, USB macro → USB subsystem, corrected
		575	10.6.8.1, corrected (EPn_BASEAD → EP[m]_BASEAD, EPn max packet & base address register → EP[m] MaxPacket & BaseAddress Register, EPn_MPKT → EP[m]_MPKT)
		576	10.6.9.1, Figure 10.24, revised
		577, 578	10.6.9.2(1), Figure 10.25, revised. 10.6.9.2(1), Figure 10.25: figure title, corrected.
		579	10.6.9.2(2), Figure 10.26: EPn → EP[m], corrected
		581	10.6.9.4(1), Figure 10.27: corrected (EPn → EP[m], Data Lenght → Data Length)
		582	10.6.9.4(2), Figure 10.28: EPn → EP[m], corrected
		582, 583, 593, 596, 599, 600, 605 to 608	10.6.9.4, EPn → EP[m], corrected
		583	10.6.9.4(3), corrected (EPn length & DMA count register → EP[m]_LEN_DCNT register, USB_INTF0 → U2F_INT, USB_INTF1 → U2F_EPC_INT)
		584 to 586	10.6.9.4(3), Figure 10.29, corrected (EPn → EP[m], EPn_Burst_Set → EP[m]_BURST_SET, Behaviour when ENDB_EPn asserted → EP[m]_DEND auto set enable)
		587	10.6.9.4(4), Figure 10.30, corrected (EPn → EP[m], EPn_Burst_Set → EP[m]_BURST_SET, Behaviour when ENDB_EPn asserted → EP[m]_DEND auto set enable)
		588	10.6.9.4(5), Figure 10.31: EPn → EP[m], corrected
		589	10.6.9.4(6), Figure 10.32: EPn → EP[m], corrected
		594	10.6.9.4(8)(b), Table 10.139: the lower right corner: (1) → (2), corrected
		604	10.6.9.4(10), Figure 10.34, revised

Rev.	Date	Description	
		Page	Summary
0.80	Oct 31, 2017	605 to 608	10.6.9.4(11), Figure 10.35, corrected (EPn → EP[m], TEPn Write Register → EP[m] Write Register, ENDB_EPn auto transmit enable → EP[m]_DEND auto set enable)
		617	11.4.4, Table 11.6: SRC_MSIZE and DEST_MSIZE, description, modified
		651	11.4.35, SglReqSrcReg → SglRqSrcReg, corrected
		651	11.4.35, Table 11.37: SRC_SGLREQ_WE: Single write enable → Source single request Write-Enable, revised
		652	11.4.36, SglReqDstReg → SglRqDstReg, corrected
		652	11.4.36, Table 11.38: DST_SGLREQ_WE: Destination write enable → Destination single or burst request Write-Enable, revised
		671	11.5.2, Table11.47, corrected
		672	11.5.3, chapter title, corrected
		680	12.4.4, Table 12.5, corrected
		681	12.4.6 to 12.4.12, Table 12.7 to 12.13: description of Function, modified
		686	12.4.14, chapter title, corrected
		688	12.4.16, chapter title, corrected
		688	12.4.16, Table 12.17, corrected (second count register → Second Count Buffer register)
		708	12.5.3, description, modified
		712	13.4.1, Value after reset, corrected
		714	Section 14, chapter title, modified
		716 to 719, 721 to 722	14.4.1 to 14.4.8, 14.4.11 and 14.4.12, bit assignment, corrected
		722	14.4.13, Value after reset, corrected (0x00020301 → 0x00030307)
		722	14.4.13, Table 14.14: description of Function, modified
		723	14.4.15, bit assignment, corrected
		726	14.5.8, Figure 14.2, corrected
		727	14.5.8, Figure 14.3, added
0.90	Dec 28, 2017	all	All sections, corrected English spelling and syntax errors
		26, 28, 54	Section 1 and 5, add trademarks
		26, 28, 54, 56 to 59	Section 1 and 5, ARM → Arm, changed
		26 to 27	1.1, description, revised
		26	1.1.1, Cortex-A7 URL, changed
		28	1.2.1, description, revised. 1.2.1, Cortex-M3 URL, changed.
		29	2.2, description, revised
		32	3.2, R-IN Accessory register → R-IN Engine Accessory register, corrected
		37	3.4.5, Table 3.7: Function, revised
		43	3.6.4, CAUTION, modified. 3.6.5, description, revised.
		48	4.4.5, Table 4.7: Function, revised
		53	4.6.4, CAUTION, modified. 4.6.5, description, revised.
		59	5.3.3, description, modified
		68	6.4.1.3, Table 6.5: Function of ASYNC_CDC_STAGES, revised
		85	6.4.1.32, Table 6.34: Function of MRW_STATUS, revised
		101, 172, 196, 198, 201, 202	6.4.1.57, 6.5.1.3, 6.5.6.3(1), 6.5.6.5, 6.5.6.7, and 6.5.6.9, user word → 64-bit-defined-word, changed
		121 to 123	6.4.1.89 to 6.4.1.91, Function of *_FIFO_TYPE_REG, modified
		158	6.4.2.1, Table 6.143: Function of MASKSDLOFS, modified
		159	6.4.2.2, Table 6.144: Function of ASDLLOCK and MDLLOCK, revised

Rev.	Date	Description	
		Page	Summary
0.90	Dec 28, 2017	159	6.4.2.2, Table 6.144: Function of DDMODE, modified
		160	6.4.2.2, Table 6.144: Function of MFSL and MDLLSTBY, modified
		161	6.4.2.3, Table 6.145: Function of ZQCALFREQ and ZQCALMODE, modified.
		162	6.4.2.4, Table 6.146: Function of CAPHASE and FIFORPINIT, modified
		164	6.4.2.5, Table 6.147: Function, modified
		166	6.4.2.7, Table 6.149: Function, revised
		167	6.4.2.8, Table 6.150: Function, Hi-Z → Floating, DQSB → DDR_DQS_N, MRESETB → DDR_RESET_N, MCKE → DDR_CLKEN, MODT → DDR_ODT, changed.
		168	6.4.2.9, Table 6.151: Function of WLVEND and WLSTR, revised
		168	6.4.2.9, Table 6.151: Function of WL2OFS and WL1OFS, modified (deleted “The setting that is smaller than the minimum SDLY delay will be a minimum delay.”) and corrected (condition of [6]=1)
		169	6.4.2.10. Table 6.152: Function of DQCAL2OFS and DQCAL1OFS, revised
		185	6.5.2,9, chapter title, changed. 6.5.2,9, description, modified
		186, 187	6.5.3, description, modified
		188	6.5.4, chapter title, changed
		196	6.5.6.3 (1), description, modified
		204	6.5.7.1 (2), CAUTION, corrected (deleted DDR1)
		205	6.5.7.1 (4), CAUTION, corrected (deleted DDR1)
		206	6.5.7.2 (1), CAUTION, corrected (deleted DDR1)
		212	6.6.2 (2), Request idle state for AXI interconnection and wait for acknowledge, modified (deleted “{wait} PWRSTAT_DDRC.MIRACK_A = 1”)
		213	6.6.2 (2), Request active state for AXI interconnection and wait for acknowledge, modified (DDRC.MIRACK → DDRC.MISTAT_A)
		213	6.6.2 (3), ZQCALCTRL register setting, modified
		213	6.6.2 (3), Enable output for CKE, ODT and RESET → Enable output for DDR_CLKEN, DDR_ODT and DDR_RESET_N, changed
		213	6.6.2 (3), Configuration for ZQ calibration, modified
		214	6.6.2 (3), Wait more than 200us, or wait until MDLL locked and initial ZQ calibration completed → Wait more than 200us, or wait until MDLL locked and ZQ calibration completed, changed
		221	7.4.3, Table 7.4: Function of datasize_error_st, revised
		227	7.4.7, Table 7.8: Function of err_threshold, revised
		236	7.4.21, Table 7.22: Function of dma_burst, modifed
		240 to 245, 253	7.4.25 to 7.4.30 and 7.4.39, description, revised
		258	7.5.1, description, modifed
		260	7.5.2.2, Table 7.45: colum of COL_A0, corrected
		275	7.5.4.1, Table 7.46: SYNCH_RESET and LUN_RESET, deleted
		275	7.5.4.1, Table 7.46: VOLUME_SELECT → VOLUME SELECT, corrected
		277	7.5.4, LUN RESET Command and SYNCH RESET Command sub-section, deleted
		285	7.5.4.22, description, modified
		317	7.8.3 and 7.8.4, description, revised
		318	8.1, description, modified
		321	8.4.1, Table 8.3 (1/3): Function of mstr_baud_div_fld and enter_xip_mode_imm_fld, revised
		322	8.4.1, Table 8.3 (2/3): Function of enter_xip_mode_fld and enb_dir_acc_ctlr_fld, revised
		323	8.4.1, Table 8.3 (3/3): Function of enb_qspi_fld, revised
		324	8.4.2, Table 8.4: Function of addr_xfer_type_std_mode_fld, revised
		326	8.4.4, Table 8.6: Function, revised

Rev.	Date	Description	
		Page	Summary
0.90	Dec 28, 2017	330	8.4.9 and 8.4.10, Function of level_fld, revised
		333, 334	8.4.13, Table 8.15: Function of b11 to b8 and b1, revised
		335	8.4.14, Table 8.16: Function of b11 to b8, revised
		336	8.4.16, Table 8.18: Function, revised
		338	8.4.18, Table 8.20: Function of cmd_opcode_fld, revised
		340	8.4.20, Table 8.22: Function, revised
		342	8.4.24, Table 8.26: Function, revised
		344	8.5.1.2, A block can be between 1 and 65K bytes → A block can be between 1 and 64K bytes, corrected
		345	8.5.3 and 8.5.4, description, modified
		346	8.5.6, description, modified
		347	8.5.7, description, modified
		348	8.5.8, description, revised
		349	8.5.9, description, revised
		350	8.5.9.1 (9), description, modified
		351, 352	8.6 (including sub-section), description, modified and corrected
		354	8.6.5 (including sub-section), description, modified and revised
		355	8.6.7, description, revised
		356	9.1, description, modified
		374	9.4.12, Table 9.14: Function of hostctrl1_extdatawidth, revised
		408	9.5.1, Table 9.42: ARM processor → CPU, changed
		415	10.1, descripton, revised
		419	10.3, description, revised
		432	10.5.1.6, Table 10.14: Function of b6 to b0, revised
		445	10.5.1.22, Table 10.30: Function of R_PRS__W_SPR, modified
		453	10.5.2.6, Table 10.36: HCHALTED, corrected
		460	10.5.2.13, Table 10.43: Function of PP, modified
		461	10.5.2.13, Table 10.43: Function of FORCE_PORT_RESUME
		472	10.5.3.13, Table 10.56: Function of b18 to b16, corrected
		488 to 490	10.5.5.5 to 10.5.5.7, Function of TYPE, revised
		498	10.5.6.6, Table 10.84: Function of RESERR_INT, modified
		503	10.5.7.1, Table 10.88: Function of FORCEFS, modified
		520	10.5.7.15, Table 10.102: Function of EP[m]_AUTO, Ex2) 32 bit CPU MaxPacketSize = 512 → Ex2) 32 bit CPU MaxPacketSize = 511, corrected
		533	10.5.8.2, Table 10.111: Function of ARBITER_CTR, corrected
		543	10.6.2, Tabe 10.120: (1), Description, corrected (window area burst address → window area base address)
		568	10.6.7.3 (1), description, modified
		568	10.6.7.3 (1), Figure 10.20, revised
		570, 571	10.6.7.3 (2), NOTE, modified. Figure 10.22 and 10.23, revised
		580	10.6.9.4 (2), moved Example 1 and 2 to under Note 1 of Figure 10.28
		599	10.6.9.4 (9) (a), Table 10.158: bcdUSB, Description, revised. 10.6.9.4 (9) (b), Table 10.159: bcdUSB, Description, revised
		607	11.1, Unidirectional transfer supported, deleted
		614	11.4.4, Table 11.6: Function of BLOCK_TS, modified
		622	11.4.9, Table 11.11: Function of MAX_ABRST, modified

Rev.	Date	Description	
		Page	Summary
0.90	Dec 28, 2017	646	11.4.32, description, corrected (Combined Status register → Combined Interrupt Status register)
		658	11.5.1.1, description, revised (following functions → following peripherals)
		672	12.2, chapter title, changed (Module Interface Signals → Signal Interfaces)
		674	12.4.1, Table 12.2: Function of RTCA0CE, revised
		675	12.4.2, Table 12.3: Function, revised
		678	12.4.4, Table 12.5: Function of RTCA0SUBC, RTCA0PE → RTCA0CE, corrected
		679	12.4.5, Table 12.6: Function of RTCA0SRBU, CAUTION, corrected
		685	12.4.15, Table 12.16: Function of RTCA0DEV, revised
		692	12.4.27, description, RTCA0TIME → RTCA0TIMEC, corrected
		693	12.4.28, description, RTCA0CAL → RTCA0CALC, corrected
		695	12.5.1.2, RTCA0SEC (second counter) → RTCA0SEC (second counter buffer), corrected
		697	12.5.1.3, RTCA0SEC (second counter) → RTCA0SEC (second counter buffer), corrected
		708	13.1, software reset → system reset, corrected
		710	13.4.1, Table 13.4: Function of WDSI, revised
		711	13.5, description. revised
0.95	Oct 19, 2018	—	All sections, spelling, syntax errors and appearances are corrected, and expressions are modified properly
		4	How to Use This Manual, 1. Objective and Target Users, Documents related to RZ/N1 (R18DS0026 → R01DS0323), table modified
		6	How to Use This Manual, 3. List of Abbreviations and Acronyms, INTC, OTP, description modified
		26	1.1.1 Overview, Figure 1.1 Arm Cortex-A7 Core Interfaces and Connections (16K → 16KB), figure modified
		26, 27	1.1.1 Overview (add: Generic timer, add: Configuration Base Address Register (CBAR) value, others), description modified
		27	1.1.2 Usage Notes, description modified
		28	1.2.1 Overview, description modified
		28	1.2.3.1 Restriction, expression modified
		29	2.2 Features, description modified
		30	2.2 Features, Table 2.1 RZ/N1D Bus Connection Map, Note1,2, description added
		31	2.2 Features, Table 2.2 RZ/N1S Bus Connection Map, Note1,2, description added
		32	2.2 Features, Table 2.3 RZ/N1L Bus Connection Map, Note1,2, description added
		—	Overall of Section 3 2MB SRAM (1-bit/2-bit → single/double), term modified
		33	3.2 Signal Interfaces, RINBUS_HCLK (N/A → Input), description modified
		43	3.6.2 ECC Decoder Config Register (RAMEDC) (or → and), description modified
		44	3.6.5 Double Bit ECC Error Address Register (RAMDBEAD), description modified
		—	Overall of Section 4 4MB SRAM (1-bit/2-bit → single/double), term modified
		53	4.6.2 SRAM 4MB ECC Decoder Config Register (SR4EDC) (or → and), description modified
		54	4.6.5 SRAM 4MB Double Bit ECC Error Address Register (SR4DBEAD), description modified
		59	5.3.1 RZ/N1 Reset Signals, description modified
		59	5.3.2 Debugger Reset Signals, description modified
		68	6.4.1.1 DDR_CTL_00 — DDR-Controller Status & Control 00, DRAM_CLASS, description added
		69	6.4.1.2 DDR_CTL_01 — DDR-Controller Status & Control 01, Value after reset, modified 6,4,1,3..7, 6.4.1.138 Same as 6.4.1.2 DDR_CTL_01

Rev.	Date	Description	
		Page	Summary
0.95	Oct 19, 2018	74	6.4.1.12 DDR_CTL_11 — DDR-Controller Status & Control 11, CASLAT_LIN, description modified
		83	6.4.1.27 DDR_CTL_26 — DDR-Controller Status & Control 26, LOWPOWER_REFRESH_ENABLE, description modified
		84	6.4.1.28 DDR_CTL_27 — DDR-Controller Status & Control 27, LP_STATE, description modified
		85	6.4.1.30 DDR_CTL_29 — DDR-Controller Status & Control 29, expression modified
		95	6.4.1.49 DDR_CTL_48 — DDR-Controller Status & Control 48, ROW_DIFF, BANK_DIFF, description added
		96	6.4.1.50 DDR_CTL_49 — DDR-Controller Status & Control 49, COL_DIFF, description modified
		102	6.4.1.57 DDR_CTL_56 — DDR-Controller Status & Control 56, INT_STATUS (MC → memory controller), expression modified 6.4.1.72, 6.4.1.117, 6.4.1.133..134 Same as 6.4.1.57 DDR_CTL_56
		103	6.4.1.58 DDR_CTL_57 — DDR-Controller Status & Control 57, INT_ACK (mask → bit), description modified
		110	6.4.1.69 DDR_CTL_68 — DDR-Controller Status & Control 68, OCD_ADJUST_PUP_CS_0, OCD_ADJUST_PDN_CS_0, description modified
		130	6.4.1.102 DDR_CTL_[k] — Port0 Range[n] Protect Setting Register2 (n = 0..14) (k = 220 + n × 2), Bit name(AXI0_RANGE_PROT_BITS[n+1] → AXI0_RANGE_PROT_BITS_[n+1]), modified 6.4.1.105, 6.4.1.108, 6.4.1.111 Same as 6.4.1.102 DDR_CTL_[k]
		140	6.4.1.112 DDR_CTL_346 — Port3 Range15 Protect Setting Register2, AXI3_RANGE_WID_CHECK_BITS_ID_LOOKUP_15, AXI3_RANGE_RID_CHECK_BITS_ID_LOOKUP_15 ([n] → 15), modified
		143	6.4.1.116 DDR_CTL_350 — DDR-Controller Status & Control 350, DLL_RST_ADJ_DLY, DLL_RST_DELAY, description modified
		156	6.4.1.138 DDR_CTL_372 — DDR-Controller Status & Control 372, OPTIMAL_RMODW_EN, description modified
		171	6.5.1 Address Mapping, description deleted
		171	6.5.1.1 DDR SDRAM Address Mapping Options, description modified
		173	6.5.1.3 Memory Mapping to Address Space, description modified
		173	6.5.1.3 Memory Mapping to Address Space, CAUTION (address 0x2, the Datapath bits must be defined as 3'b010 → address 0x1, the Datapath bits must be defined as 1'b1), value modified
		176	6.5.2.4 Understanding Port Bandwidth (DDR Controller initialization → DDR Controller), description modified
		185	6.5.2.8 Arbitration Examples, Table 6.157 Priority Round-Robin with Bandwidth Consideration, table axis (Arbitration Winner → Bandwidth Held Off?), description modified
		187	6.5.3 Port Protection Option (AXI0_RANGE → AXIY_RANGE), description modified
		190	6.5.4.1 Rules of the Placement Algorithm, Table 6.159 Simple Command Queue Example, title modified
		190	6.5.4.1 Rules of the Placement Algorithm PQ, term deleted
		197	6.5.6.3 ECC Control (a 2 bit parameter → a parameter), description modified
		198	6.5.6.4 Syndromes, caution added
		200	6.5.6.5 Command Processing when ECC is Enabled, Table 6.162 ECC Functionality on Various Transaction Types (2/2), Masked Write, description modified
		205	6.5.7.1 Low Power States, (1) Active Power-Down, (3) Pre-Charge Power-Down, description modified
		207	6.5.7.2 Management of the Low Power Control Module, Table 6.164 Low Power State Management, Automatic (No → Yes), table modified
		209	6.5.7.3 Software Programmable Interface, (2) Software Programmable Interface Commands, reference modified

Rev.	Date	Description	
		Page	Summary
0.95	Oct 19, 2018	209	6.5.7.3 Software Programmable Interface, (2) Software Programmable Interface Commands (bit → bit (LP_CMD[4:2])), description modified
		211	6.5.7.4 Automatic Interface, (1) Automatic Entry, CAUTION (four power → power), description modified
		211	6.5.7.4 Automatic Interface, (4) Refresh Masking, description deleted
		215	7.1 Overview, Figure 7.1 NAND Flash Controller Interfaces and Connections (FNAND_RYBY_N → FNAND_RY/BY_N), figure modified
		220	7.4.2 CONTROL — CONTROL Register, int_en (Global Interrupt → Interrupt), expression modified
		226	7.4.6 INT_STATUS — INT_STATUS Register (‘1’ value or the rising edge →‘1’ value), description modified
		227	7.4.7 ECC_CTRL — ECC Control Register (present → available), description modified
		232	7.4.14 PROTECT — Protect Register, description deleted
		233	7.4.15 FIFO_DATA — FIFO Data Register (external CPU → CPU), description modified
		235	7.4.20 DMA_CNT — DMA Counter Register, cnt_init (FFFF_FFFDh → FFFF_FFFCh), value modified
		242	7.4.27 TIME_SEQ_1 — Command Sequence Timing Register 1, trr, twb, description modified
		244	7.4.29 TIME_GEN_SEQ_1 — Generic Command Sequence Register 1, t0_d7, t0_d6, t0_d5, t0_d4 (commands → address, others), description modified
		245	7.4.30 TIME_GEN_SEQ_2 — Generic Command Sequence Register 2, t0_d11, t0_d10, t0_d9, t0_d8, description modified
		248	7.4.33 GEN_SEQ_CTRL — Generic Sequence Register (delay → DELAY, others), description modified
		250	7.4.34 MLUN — LUN Configuration Register, mlun_sel, description modified
		257	7.4.42 PARAM_REG — PARAMETER Register, toggle_mode_implement (toggle work mode → toggle mode), description modified
		275	7.5.4.1 Instruction Set, Table 7.46 Instruction Set (1/2), Instruction (PROGRAM PAGE IMD → PROGRAM PAGE IMMEDIATE, PROGRAM PAGE DEL → PROGRAM PAGE DELAYED), term modified
		276	7.5.4.3 RESET Command, description deleted
		277	7.5.4.4 READ ID Command, description deleted
		278	7.5.4.7 GET FEATURES Command, description deleted
		281	7.5.4.13 LUN STATUS Command (SELECT LUN WITH STATUS → LUN STATUS), term modified
		286	7.5.4.25 PROGRAM PAGE IMMEDIATE Command (Then five address cycles containing the column address and row address → Then three address cycles containing the row address), description modified
		286	7.5.4.25 PROGRAM PAGE IMMEDIATE Command, Table 7.69 PROGRAM PAGE IMMEDIATE Instruction Encoding (PROGRAM PAGE IMD → PROGRAM PAGE IMMEDIATE), title modified
		287	7.5.4.26 PROGRAM PAGE DELAYED Command (The PROGRAM PAGE DELAY instruction uses the SEQ_12 → The PROGRAM PAGE DELAYED instruction uses the SEQ_23), description modified
		287	7.5.4.26 PROGRAM PAGE DELAYED Command, Table 7.70 PROGRAM PAGE DELAYED Instruction Encoding (DELAY → DELAYED), title modified
		293	7.5.4.41 PAGE READ OTP Command (SEQ_9 → SEQ_10), description modified
		294	7.5.5 Multi LUN Work Mode (LUN_ → MLUN _, LUN STATUS 0 → LUN_STATUS_0), expression modified
		297	7.6 Setup and Configuration (Global Interrupt → Interrupt, others), description modified
		301	7.6.3 Send Data to NAND Flash via Master Interface (Using DMA) (global interrupt → interrupt), description modified

Rev.	Date	Description	
		Page	Summary
0.95	Oct 19, 2018	303	7.6.4 Fast Writing and Reading of Several Pages from the Memory Using DMA (global interrupt → interrupt, (10): section 8 → section 7, (18): CHCHE SEQUENTIAL → CHCHE, (19): section 18 → section 16), description modified
		316	7.8.2 Protect Register (PROTECT), description deleted
		318	8.1 Overview (Transmit and receive FIFOs are 16 bytes), description added
		323	8.4.1 config_reg — QSPI Configuration Register, sel_clk_phase_fld, sel_clk_pol_fld, description modified
		324	8.4.2 dev_instr_rd_config_reg — Device Read Instruction Configuration Register, Function (Standard SPI → SPI), description modified
		325	8.4.3 dev_instr_wr_config_reg — Device Write Instruction Configuration Register, Function (Standard SPI → SPI), description modified
		328	8.4.6 dev_size_config_reg — Device Size Configuration Register, Function, description added
		335	8.4.14 irq_mask_reg — Interrupt Mask Register, Function (add: mask), description modified
		338	8.4.18 flash_cmd_ctrl_reg — Flash Command Control Register, cmd_opcode_fld, description modified
		348	8.5.7 Selecting the Flash Instruction Type, Table 8.29 Number of Lanes for WRITE Instructions Supported by the N25Q128 Device, note added
		349	8.5.9.1 Example of an 8 byte Read Transfer, (8) (26 cycles → 16 cycles), description modified
		352	8.6.2 Configuring the QuadSPI Controller for optimal use, (6) (d_nss field → d_nss_fld), description modified
		353	8.6.4 Using SPI Legacy Mode (RX-FIFO are of limited depth → RX-FIFO are of limited depth (8 bytes) in legacy mode), description added
		354	8.6.5.3 Exiting XIP mode (enb_dir_acc_ctlr_fld → enter_xip_mode_fld), description modified
		371 to 373	9.4.11 reg_presentstate — Present State Register, Function, description added
		374	9.4.12 reg_hostcontrol1 — Host Control 1 Register, hostctrl1_cdsigselect (detection → selection), description modified
		376	9.4.13 reg_powercontrol — Power Control Register, emmc_hwreset, description modified
		383	9.4.18 reg_softwarereset — Software Reset Register, description added
		383	9.4.18 reg_softwarereset — Software Reset Register, swreset_for_all, description modified
		384	9.4.19 reg_normalintrsts — Normal Interrupt Status Register (normalintrsts_cardintsts, normalintrsts_cardremsts), description modified
		388	9.4.20 reg_errorintrsts — Error Interrupt Status Register, errorintrsts_cmdcrcerror (CRT → CRC), description modified
		391	9.4.23 reg_normalintrsigena — Normal Interrupt Signal Enable Register (sample → same), description modified
		392	9.4.24 reg_errorintrsigena — Error Interrupt Signal Enable Register (sample → same), description modified
		395	9.4.27 reg_capabilities — Capabilities Register, Value after reset, modified
		395	9.4.27 reg_capabilities — Capabilities Register (corecfg_1p8voltsupport, corecfg_3p0voltsupport), description modified
		398	9.4.29 reg_maxcurrentcap — Maximum Current Capabilities Register, Function, description modified
		402	9.4.33 reg_admasysaddr0 — ADMA System Address Register Low, adma_sysaddress0 (lower → system address), description modified
		409	9.5.2 DMA Transaction, Figure 9.3 Data Transfer Using DAT Line Sequence (Using DMA) (8: Complete → Clear), figure modified
		410	9.5.2 DMA Transaction, Table 9.43 DMA Transaction, 11 (Step(4) → Step(14)), description modified

Rev.	Date	Description	
		Page	Summary
0.95	Oct 19, 2018	410	9.5.2 DMA Transaction, DMA Transaction Sequence (512 → 512 bytes), description modified
		—	Overall of Section 10 USB 2.0 HS Host/Function Controller (USBh/USBf) (OCI → USB_OC, PPON → USB_PPON), term modified
		415	10.1 Overview, description modified
		419	10.3 USBPLL Features, description modified
		445	10.5.1.22 HCRHPORTSTATUS1/ HCRHPORTSTATUS2 − HcRhPortStatus1/ HcRhPortStatus2 Register, R_PPS__W_SPP (turn off → turn on), description modified
		452	10.5.2.5 USBCMD — USBCMD Register, PERIODIC_SCHEDULE_ENABLE, description modified
		455	10.5.2.7 USBINTR — USBINTR Register, Function, description modified
		512	10.5.7.9 EP0_CONTROL — EP0 Control Register, EP0_STL (set to 1b → cleared to 0b), description modified
		519	10.5.7.15 EP[m]_CONTROL — EP[m] Control Register (m = 1..15), Bit (b1: RESERVED → –), expression modified
		521	10.5.7.15 EP[m]_CONTROL — EP[m] Control Register (m = 1..15), EP[m]_ISTL (Return → Not Return), description modified
		522 to 524	10.5.7.16 EP[m]_STATUS — EP[m] Status Register (m = 1..15) (EP[m]_ISO_PIDERR, EP[m]_OUT_NAK_ERR_INT, EP[m]_IN_NAK_ERR_INT, EP[m]_IN_INT) , description modified
		554	10.6.5.1 Overcurrent Control, Figure 10.8 USB_OC1 and USB_PPON1 Assert, De-assert Timing Chart (SPP → R_PPS__W_SPP, SGP → R_LPSC__W_SGP), figure modified
		555	10.6.5.2 VBUS Control, Table 10.133 Relationship between Register Settings and USB_PPON1/PPON2, table modified
		556	10.6.5.3 Overcurrent Detection Using PPON, Figure 10.9 Overcurrent Detection Using PPON (POCI → R_POCI__W_CSS, SPP → R_PPS__W_SPP), figure modified
		557	10.6.5.4 PPON Setting Flow, Figure 10.10 PPON Setting Flow (SPP → R_PPS__W_SPP), figure modified
		559	10.6.6.3 VBUS Detection Flow, Figure 10.12 VBUS Detection Flow(NOCP → VBUS_INT), figure modified
		561	10.6.7.1 Power Management in the Host Controller, (1), Figure 10.13 Host Controller Power-Down Flow (When Using OHCI) (PS → R_PSS__W_SPS, SRWE → R_DRWE__W_SRWE), figure modified
		564	10.6.7.1 Power Management in the Host Controller, (2), Figure 10.16 Host Controller Power-Up Flow (When Using EHCI) (1. Connect/Disconnect: PORTSC1 Register), figure modified
		567	10.6.7.2 Power Management in the Function Controller, (2), Figure 10.19 Power Up by Detecting the Resume Signal (RemoteWakeUp) (PLL_RESUME=0b → PLL_RESUME=1b), figure modified
		568	10.6.7.3 Direct Power-Down Feature, (1), Figure 10.20 Entering the Direct Power-Down Mode (EPCRST → EPC_RST), figure modified
		571	10.6.7.3 Direct Power-Down Feature, (2), Figure 10.23 Exiting the Direct Power-Down Mode (Function Controller) (bit8 DIRPD=0b or → bit8 DIRPD=0b), figure modified
		573	10.6.8.1 Specifying the Base Address (or the case shown in the table before the table above → example), description modified
		573	10.6.8.1 Specifying the Base Address, Table 10.135 BaseAddress settings example, title modified
		574	10.6.9.1 Reset Sequence, Figure 10.24 Reset Sequence (PLL reset is (description): AHB reset → Release USB reset) , figure modified
		575	10.6.9.2 Initial Setup Sequence, (1), Figure 10.25 Host Controller Initial Setup Sequence (1/2) (PREFETCH_EN → PREFETCH, PCI_ARBITERCTR → PCI_ARBITER_CTR), figure modified
		576	10.6.9.2 Initial Setup Sequence, (1), Figure 10.25 Host Controller Initial Setup Sequence (2/2) (PCIAHB_WIN1_CTR → AHBPCI_WIN1_CTR, others), figure modified

Rev.	Date	Description	
		Page	Summary
0.95	Oct 19, 2018	577	10.6.9.2 Initial Setup Sequence, (2), Figure 10.26 Function Controller Initial Setup Sequence (PCI Configuration Registers for OHCI/EHCI initial setting → EPC initial setting), figure modified
		582	10.6.9.4 Function Transfer Overview, (3), Figure 10.29 DMA OUT Transfer Overview (1/3) (EP[m] Status → EP[m] Length & DMA Count, EP[m] Length & DMA Count → EP[m] Status), figure modified
		584	10.6.9.4 Function Transfer Overview, (3), Figure 10.29 DMA OUT Transfer Overview (3/3) (Clear this interrupt (write 0b) → (Clear this interrupt (write 1b)), figure modified
		585	10.6.9.4 Function Transfer Overview, (4), Figure 10.30 DMA IN Transfer Overview (EP[m]_OUT_END_INT → EP[m]_IN_END_INT), figure modified
		587	10.6.9.4 Function Transfer Overview, (6), Figure 10.32 Control Transfer Overview (1/3) (SETUP_INT → EP0_STALL_INT), figure modified
		588	10.6.9.4 Function Transfer Overview, (6), Figure 10.32 Control Transfer Overview (2/3) (Routine for the Request → Specific Routine for the Request), figure modified
		589	10.6.9.4 Function Transfer Overview, (6), Figure 10.32 Control Transfer Overview (3/3) (Routine for the Request → Request data transfer), figure modified
		598	10.6.9.4 Function Transfer Overview, (8), Table 10.156 Set Interface Request Processing (BaseAddress register → BaseAddress register etc), description modified
		599	10.6.9.4 Function Transfer Overview, (9), Table 10.158 Device Descriptor (bMaxPacktetSize0 → bMaxPacketSize0), term modified
		599	10.6.9.4 Function Transfer Overview, (9), Table 10.159 Device Qualifier Descriptor (bMaxPacktetSize0→ bMaxPacketSize0, others), description modified
		600	10.6.9.4 Function Transfer Overview, (9), Table 10.160 Configuration Descriptor and Other Speed Configuration Descriptor (bNumInterface → bNumInterfaces, MaxPower → bMaxPower) , term modified
		603	10.6.9.4 Function Transfer Overview, (11) (EP1 → EP[m]), description modified
		604	10.6.9.4 Function Transfer Overview, (11), Figure 10.35 EP[m] Data Loopback (2/4) (EP[m]REQEN → EP[m]_REQEN, others), figure modified
		—	Overall of Section 11 DMA Controller (handshaking → request, handshaking → request interface), term modified
		607	11.1 Overview (sources → sources (request interfaces)), description added
		609 to 611	11.3 Basic Definitions, subsection added
		624, 625	11.5.9 CFG[n] — Configuration Register for Channel [n] (n = 0..7), DEST_PER, SS_UPD_EN, RELOAD_DST, RELOAD_SRC, SRC_HS_POL, DST_HS_POL, description modified
		633	11.5.16 RawErr — Raw Status for IntErr Interrupt Register (slave on the HRESP bus → slave), description modified
		649	11.5.32 StatusInt — Combined Interrupt Status Register, DSTT (StatusDst → StatusDstTran), term modified
		657	11.5.40 ChEnReg — DMA Controller Channel Enable Register, CH_EN, description added
		660	11.6.1 DMA Transfer Mode, description modified
		661	11.6.1.1 Flow Controller and Transfer Type, Table 11.45 DMA Transfer Type and Flow Control Combination, description modified
		661	11.6.1.1 Flow Controller and Transfer Type, caution added
		666	11.6.1.3 Basic Interface Definitions, Transactions (decodes to 32 → is decoded to 32), description modified
		669	11.6.1.4 Transaction Examples, Figure 11.4 Burst DMA Transaction (AMBA bursts from → bursts from), figure modified
		670	11.6.1.4 Transaction Examples, (2) DMA in Burst and Single Transaction Mode at the Same Time (blk_size_bytes → blk_size_bytes_dma), description modified
		671	11.6.1.4 Transaction Examples, (2), Figure 11.5 Burst and Single DMA Transaction (AMBA bursts from → bursts from), figure modified

Rev.	Date	Description	
		Page	Summary
0.95	Oct 19, 2018	672	11.6.2 DMA Request Allocation (DMA Channel Allocation → DMA Request Allocation), title modified
		672	11.6.2 DMA Request Allocation, description modified
		672	11.6.2 DMA Request Allocation, Table 11.47 DMA Multiplexing Channel Request Signals (Request ch → Request interface, others), description modified
		672	11.6.2 DMA Request Allocation, Table 11.47 DMA Multiplexing Channel Request Signals, Note1, description modified
		675	12.2 Signal Interfaces, Note1, description added
		694	12.4.24 RTCA0DAYC — RTC Day Count Register, RTCA0DAYC, description modified
		696	12.4.28 RTCA0CALC — RTC Calendar Count Register, description modified
		696	12.4.29 RTCA0TCR — RTC Test Register, description added
		702	12.5.1.4 Reading RTCA0SRBU While Clock Counter Operation is Enabled, Figure 12.6 Reading RTCA0SRBU (Read RTCA0SUB → Read RTCA0RSUB), figure modified
		704	12.5.1.6 Writing to RTCA0SUBU Clock Counter Operation, Figure 12.8 Writing to RTCA0SUBU (RTCA0SRBU → RTCA0SUBU), title modified
		708	12.5.2 RTC Backup Mode, CAUTION (PWRCTRL_OPPDIV → PWRCTRL_OPPDIV of System Control), description modified
		711	13.1 Overview, description modified
		713	13.4.1 CTRL_RETRIGGER — Control and Retrigger Register, description added
		713	13.4.1 CTRL_RETRIGGER — Control and Retrigger Register, Value after reset, modified
		713	13.4.1 CTRL_RETRIGGER — Control and Retrigger Register, WDRV, description modified
		714	13.5 Operation, description modified
		714	13.6 Usage Notes, subsection deleted
		727	14.5.8 Interrupts and Status Registers (up to 32 bits → 3 bits), description modified
1.00	Mar 29, 2019	—	All sections, spelling, syntax errors and appearances are corrected, and expressions are modified properly
		27	1.1.1 Overview, TrustZone, description added
		28	1.2.1 Overview, MPU, description added
		33	3.1 Overview, Table 3.1 2MB SRAM Summary, description deleted
		33	3.2 Signal Interfaces, ECC_2MB_Int, description modified
		35 to 39	3.4.2 RAMEDC — RAM_SYS ECC Decoder Config Register, As same, 3.4.3..4, description modified
		43	3.5.4 Self-Testing of the ECC Circuit, Figure 3.2 Example of ECC Error-Injection Setting Procedure, description modified
		—	Overall of Section 4 4MB SRAM, (way → area), term modified
		47	4.1 Overview, Table 4.1 4MB SRAM Summary, description deleted
		47	4.2 Signal Interfaces, ECC_4MB_Int, description modified
		48 to 50	4.4.2 SR4EDC — SRAM 4MB ECC Decoder Config Register, As same, 4.4.3..4, description modified
		53	4.5.4 Self-Testing of the ECC Circuit, description modified
		53	4.5.4 Self-Testing of the ECC Circuit, Figure 4.2 Example of ECC Error-Injection Setting Procedure, description modified
		65	6.2 Signal Interfaces, DDR_ADDR, description modified
		100 to 146	6.4.1.53 DDR_CTL_52 — DDR-Controller Status & Control 52, As same, 6.4.1.54..55, 6.4.1.58, 6.4.1.65, 6.4.1.115..116, description modified
		104	6.4.1.57 DDR_CTL_56 — DDR-Controller Status & Control 56, INT_STATUS, description modified
		105	6.4.1.59 DDR_CTL_58 — DDR-Controller Status & Control 58, INT_MASK, description modified

Rev.	Date	Description	
		Page	Summary
1.00	Mar 29, 2019	112 to 158	6.4.1.69 DDR_CTL_68 — DDR-Controller Status & Control 68, As same, 6.4.1.70..88, 6.4.1.124..137, description modified
		163	6.4.2.2 DLLCTRL — MDLL Control Register, MDLLOCK, description modified
		165	6.4.2.3 ZQCALCTRL — ZQ Calibration Control Register, Value after reset, value modified
		173	6.4.2.10 DQCALOFS1 — DQS Offset Setting, B2RSSAT, B1RSSAT, description modified
		224	7.4.3 STATUS — STATUS Register, Value after reset, value modified
		226	7.4.4 STATUS_MASK — STATUS_MASK Register, error_mask, state_mask, description modified
		233 to 256	7.4.10 ADDR0_COL — Column Address 0 Register, As same, 7.4.11..14, 7.4.25, 7.4.27..30, 7.4.33, 7.4.39, description modified
		238	7.4.19 DMA_ADDR — DMA Address Register, description modified
		241	7.4.23 MEM_CTRL — Memory Devices Control Register, mem3_wp, mem2_wp, mem1_wp, mem0_wp, description modified
		245	7.4.27 TIME_SEQ_1 — Command Sequence Timing Register 1, tww, description modified
		250	7.4.32 FIFO_STATE — FIFO Status Register, description modified
		263	7.5.2.2 Command Sequence Encoding, Table 7.45 Command Sequence Encoding, SEQ_20, value modified
		270	7.5.2.18 Sequence SEQ_15, description modified
		272	7.5.2.24 Sequence SEQ_22, description modified
		273	7.5.2.28 Sequence SEQ_26, description deleted
		288	7.5.4.22 QUEUE PAGE READ Command, Table 7.66 QUEUE PAGE READ Instruction Encoding, description modified
		297	7.5.5 Multi LUN Work Mode, description modified
		302	7.6.1 Send Data to NAND Flash via Slave Interface, description modified
		304	7.6.3 Send Data to NAND Flash via Master Interface (Using DMA), description modified
		306	7.6.4 Fast Writing and Reading of Several Pages from the Memory Using DMA, description modified
		308	7.6.5 Writing of Data into Two NAND Flash Memory Devices, Figure 7.35 Writing Data into Two NAND Flash Memory Devices 1, figure modified
		310	7.6.7 Writing Data into Four NAND Flash Memory Devices, Figure 7.38 Writing Data into Four NAND Flash Memory Devices 1, figure modified
		312	7.6.8 Reading Data from Four NAND Flash Memory Devices, Figure 7.40 Reading Data from Four NAND Flash Memory Devices 1, figure modified
		314	7.6.9 Writing Partial Pages, description modified
		318	7.8.1 ADDR[n]_COL and ADDR[n]_ROW Registers, Table 7.87 Address Cycles, expression modified
		321	8.1 Overview, description modified
		322	8.2 Signal interfaces, caution added
		325 to 326	8.4.1 config_reg — QSPI Configuration Register, description modified
		328	8.4.2 dev_instr_rd_config_reg — Device Read Instruction Configuration Register, rd_opcode_non_xip_fld, term modified
		329	8.4.3 dev_instr_wr_config_reg — Device Write Instruction Configuration Register, wel_dis_fld, description modified
		332	8.4.6 dev_size_config_reg — Device Size Configuration Register, bytes_per_device_page_fld, description deleted
		332	8.4.6 dev_size_config_reg — Device Size Configuration Register, num_addr_bytes_fld, description modified
		335	8.4.11 write_completion_ctrl_reg — Write Completion Control Register, disable_polling_fld, description modified
		338	8.4.13 irq_status_reg — Interrupt Status Register, illegal_access_det_fld, prot_wr_attempt_fld, description modified

Rev.	Date	Description	
		Page	Summary
1.00	Mar 29, 2019	342 to 343	8.4.18 flash_cmd_ctrl_reg — Flash Command Control Register, description modified
		349	8.5.4 Servicing a STIG request, description modified
		350	8.5.6 SPI Command Translation, term modified
		351	8.5.7 Selecting the Flash Instruction Type, Table 8.28 Number of Lanes for READ Instructions Supported by the N25Q128 Device, expression modified
		352	8.5.7 Selecting the Flash Instruction Type, Table 8.29 Number of Lanes for WRITE Instructions Supported by the N25Q128 Device, expression modified
		358	8.6.5 Entering and Exiting NoCMD mode, description modified
		375 to 376	9.4.11 reg_presentstate — Present State Register, description modified
		384	9.4.16 reg_clockcontrol — Clock Control Register, clkctrl_sdclkfreqsel, description modified
		386	9.4.17 reg_timeoutcontrol — Timeout Control Register, timeout_ctrvalue, description modified
		388	9.4.19 reg_normalintrsts — Normal Interrupt Status Register, description modified
		391 to 392	9.4.20 reg_errorintrsts — Error Interrupt Status Register, description modified
		393	9.4.21 reg_normalintrstsena — Normal Interrupt Status Enable Register, description modified
		394	9.4.22 reg_errorintrstsena — Error Interrupt Status Enable Register, description modified
		395	9.4.23 reg_normalintrsigena — Normal Interrupt Signal Enable Register, description modified
		396	9.4.24 reg_errorintrsigena — Error Interrupt Signal Enable Register, description modified
		400	9.4.27 reg_capabilities — Capabilities Register, corecfg_maxblklength, corecfg_baseclkfreq, corecfg_timeoutclkunit, description modified
		403	9.4.30 reg_ForceEventforAUTOCMDErrorStatus — Force Event for Auto CMD Error Status Register, description modified
		404	9.4.31 reg_forceeventforerrintsts — Force Event for Error Interrupt Status Register, description modified
		411	9.5.1 Non-DMA Transaction, Figure 9.2 Data Transfer Using DAT Line Sequence (Not Using DMA), description modified
		413	9.5.2 DMA Transaction, Figure 9.3 Data Transfer Using DAT Line Sequence (Using DMA), description modified
		415	9.5.3 ADMA Transactions, Figure 9.4 ADMA Transaction Flow, description modified
		417	9.5.4.1 Synchronous Abort, Figure 9.5 Synchronous Abort, description modified
		433 to 434	10.5.1.4 HCINTERRUPTSTATUS — HcInterruptStatus Register, description modified
		435	10.5.1.5 HCINTERRUPTENABLE — HcInterruptEnable Register, description modified
		436	10.5.1.6 HCINTERRUPTDISABLE — HcInterruptDisable Register, description modified
		442	10.5.1.15 HCFMREMAINING — Hc Frame Remaining Register, FRT, expression modified
		444	10.5.1.19 HCRHDESCRIPTORA — HcRhDescriptorA Register, PSM, description modified
		446 to 447	10.5.1.21 HCRHSTATUS — HcRhStatus Register, R_LPSC__W_SGP, R_LPS__W_CGP, description modified
		449	10.5.1.22 HCRHPORTSTATUS1/ HCRHPORTSTATUS2 — HcRhPortStatus1/HcRhPortStatus2 Register, R_POCI__W_CSS, description deleted
		458	10.5.2.6 USBSTS — USBSTS Register, USBINT, description modified
		464 to 465	10.5.2.13 PORTSC1/PORTSC2 — PORTSC1/PORTSC2 Register, description modified
		487	10.5.4.14 EXT2 — EXT2 Register (EHCI), description added
		506	10.5.6.9 PCI_ARBITER_CTR — PCI Arbiter Control Register, PCIBP_MODE, PCIREQ1, PCIREQ0, expression modified
		507	10.5.7.1 USB_CONTROL — USB Control Register, USBTESTMODE, description modified
		514	10.5.7.8 USB_INT_ENA — USB Interrupt Enable Register, description added
		515 to 516	10.5.7.9 EP0_CONTROL — EP0 Control Register, description modified

Rev.	Date	Description	
		Page	Summary
1.00	Mar 29, 2019	517 to 518	10.5.7.10 EP0_STATUS — EP0 Status Register, description modified
		520 to 521	10.5.7.11 EP0_INT_ENA — EP0 Interrupt Enable Register, description modified
		525 to 526	10.5.7.15 EP[m]_CONTROL — EP[m] Control Register (m = 1..15), description modified
		527, 529	10.5.7.16 EP[m]_STATUS — EP[m] Status Register (m = 1..15), description modified
		531	10.5.7.17 EP[m]_INT_ENA — EP[m] Interrupt Enable Register (m = 1..15), description modified
		540	10.5.8.4 AHBBINTEN — AHB-EPC Bridge Interrupt Enable Register, description modified
		553	10.6.3.1 Reset Configuration, description modified
		557 to 558	10.6.4.3 Interrupt Signal Clear Time, value modified
		561	10.6.5.2 VBUS Control, Table 10.133 Relationship between Register Settings and USB_PPON1/PPON2, expression modified
		562	10.6.5.3 Overcurrent Detection Using PPON, Figure 10.9 Overcurrent Detection Using PPON, figure modified
		563	10.6.5.4 PPON Setting Flow, Figure 10.10 PPON Setting Flow, figure modified
		565	10.6.6.3 VBUS Detection Flow, Figure 10.12 VBUS Detection Flow, figure modified
		567	10.6.7.1 Power Management in the Host Controller, Figure 10.13 Host Controller Power-Down Flow (When Using OHCI), figure modified
		568	10.6.7.1 Power Management in the Host Controller, Figure 10.14 Host Controller Power-Down Flow (When Using EHCI), figure modified
		569	10.6.7.1 Power Management in the Host Controller, Figure 10.15 Host Controller Power-Up Flow (When Using OHCI), figure modified
		570	10.6.7.1 Power Management in the Host Controller, Figure 10.16 Host Controller Power-Up Flow (When Using EHCI), figure modified
		578	10.6.7.4 Notes on Suspend state transition, subsection added
		582	10.6.9.2 Initial Setup Sequence, Figure 10.25 Host Controller Initial Setup Sequence (1/2), figure modified
		584	10.6.9.2 Initial Setup Sequence, Figure 10.26 Function Controller Initial Setup Sequence, figure modified
		590	10.6.9.4 Function Transfer Overview, Figure 10.29 DMA OUT Transfer Overview (2/3), figure modified
		592	10.6.9.4 Function Transfer Overview, Figure 10.30 DMA IN Transfer Overview, figure modified
		593	10.6.9.4 Function Transfer Overview, Figure 10.31 Stopping a DMA Transfer, figure modified
		597	10.6.9.4 Function Transfer Overview, Figure 10.33 Protocol Error NAK Processing Overview, figure modified
		602	10.6.9.4 Function Transfer Overview, (g) Set Configuration, Table 10.149, title modified
		605	10.6.9.4 Function Transfer Overview, (k) Sync Frame, Table 10.157, title modified
		612	10.6.9.4 Function Transfer Overview, Figure 10.35 EP[m] Data Loopback (3/4), figure modified
		615	11.2 Signal Interfaces, DMA[m]_HCLK, expression modified
		631	11.5.9 CFG[n] — Configuration Register for Channel [n] (n = 0..7), description modified
		635 to 639	11.5.12 RawTfr — Raw Status for IntTfr Interrupt Register, As same, 11.5.13..16, description modified
		640 to 644	11.5.17 StatusTfr — Status for IntTfr Interrupt Register Register, As same, 11.5.18..21, description modified
		656	11.5.33 ReqSrcReg — Source Software Transaction Request Register, SRC_REQ, description modified
		657	11.5.34 ReqDstReg — Destination Software Transaction Request Register, DST_REQ, description modified

Rev.	Date	Description	
		Page	Summary
1.00	Mar 29, 2019	658	11.5.35 SglRqSrcReg — Single Source Transaction Request Register, SRC_SGLREQ_WE, SRC_SGLREQ, description modified
		659	11.5.36 SglRqDstReg — Single Destination Transaction Request Register, DST_SGLREQ, description modified
		663	11.5.40 ChEnReg — DMA Controller Channel Enable Register, CH_EN_WE, description modified
		674	11.6.1.4 Transaction Examples, (1) DMA in Burst Transaction Mode Only, description modified
		680	12.1 Overview, Figure 12.1 RTC Interfaces and Connections, figure modified
		681	12.2 Signal Interfaces, term modified
		691	12.4.9 RTCA0WEEK — RTC Week Count Buffer Register, description modified
		695	12.4.16 RTCA0SCMP — RTC Sub Count Compare Register, RTCA0SCMP, description modified
		697	12.4.19 RTCA0ALW — RTC Alarm Week Set Register, description modified
		706	12.5.1.3 Reading Clock Counters While Clock Counter Operation is Enabled, description deleted
		708	12.5.1.4 Reading RTCA0SRBU While Clock Counter Operation is Enabled, Figure 12.6 Reading RTCA0SRBU, figure modified
		721	14.1 Overview, description modified
		721	14.2 Signal Interfaces, IPCM_Int[n], expression modified
		724	14.4.2 IPCM[n]DSET — Mailbox[n] Destination Set Register (n = 0..2), description modified
		725	14.4.3 IPCM[n]DCLEAR — Mailbox[n] Destination Clear Register (n = 0..2), description modified
		726	14.4.4 IPCM[n]DSTATUS — Mailbox[n] Destination Status Register (n = 0..2), description modified
		727 to 728	14.4.6 IPCM[n]MSET — Mailbox[n] Mask Set Register (n = 0..2), As same, 14.4.6..8, description modified
		730	14.4.11 IPCMMIS[n] — Masked Interrupt[n] Status Register (n = 0..2), MASK_INT_STAT, description modified
		731	14.4.12 IPCMRIS[n] — Raw Interrupt[n] Status Register (n = 0..2), RAW_INT_STAT, description modified
		733	14.5 Operation, description modified
1.10	Jun 30, 2020	—	All sections, spelling, syntax errors and appearances are corrected, and expressions are modified properly
		91	6.4.1.37 DDR_CTL_36 — DDR-Controller Status & Control 36, ECC_EN, description added
		102	6.4.1.55 DDR_CTL_54 — DDR-Controller Status & Control 54, REDUC, description added
		200	6.5.6.3 ECC Control, description added
		321	8.1 Overview, description added
		325	8.4.1 config_reg — QSPI Configuration Register, enable_ahb_decoder_fld, description modified
		326	8.4.1 config_reg — QSPI Configuration Register, periph_cs_lines_fld, periph_sel_dec_fld, enb_legacy_ip_mode_fld, description modified
		328	8.4.2 dev_instr_rd_config_reg — Device Read Instruction Configuration Register, data_xfer_type_ext_mode_fld, addr_xfer_type_std_mode_fld, expression modified
		329	8.4.3 dev_instr_wr_config_reg — Device Write Instruction Configuration Register, data_xfer_type_ext_mode_fld, addr_xfer_type_std_mode_fld, expression modified
		330	8.4.4 dev_delay_reg — QSPI Device Delay Register, d_nss_fld, description modified
		332	8.4.6 dev_size_config_reg — Device Size Configuration Register, mem_size_on_cs3_fld, description modified
		335	8.4.11 write_completion_ctrl_reg — Write Completion Control Register, poll_rep_delay_fld, expression modified

Rev.	Date	Description	
		Page	Summary
1.10	Jun 30, 2020	336	8.4.12 no_of_polls_bef_exp_reg — Polling Expiration Register, no_of_polls_bef_exp_fld, expression modified
		348	8.5.1.1 Remapping the AHB address, description modified
		349	8.5.2 Direct Access Controller (DAC), description modified
		357	8.6.4 Using SPI Legacy Mode, expression modified
		358	8.6.5.2 Exiting NoCMD mode, description modified
		359	8.6.6 Remapping AHB addresses, description deleted
		359	8.7 Usage Notes, description added
		380	9.4.13 reg_powercontrol — Power Control Register, pwrctrl_sdbuspower, description added
		384	9.4.16 reg_clockcontrol — Clock Control Register, clkctrl_sdclkena, description added
		399	9.4.27 reg_capabilities — Capabilities Register, corecfg_slottype, description modified
		579	10.6.8 USB Function Endpoints Configuration, description added
		614	11.1 Overview, description modified
		616	11.3 Basic Definitions, Peripheral:, description added
		616	11.3 Basic Definitions, Channel:, Flow controller:, description modified
		617	11.3 Basic Definitions, Transaction:, Single transaction:, Burst transaction:, description modified
		617	11.3 Basic Definitions, Block:, description added
		624	11.5.4 CTL[n] — Control Register for Channel [n] (n = 0..7), BLOCK_TS, description modified
		625	11.5.4 CTL[n] — Control Register for Channel [n] (n = 0..7), TT_FC, SRC_MSIZE, DEST_MSIZE, description modified
		632	11.5.9 CFG[n] — Configuration Register for Channel [n] (n = 0..7), FIFO_MODE, FCMODE, MAX_ABRST, description modified
		633	11.5.9 CFG[n] — Configuration Register for Channel [n] (n = 0..7), FIFO_EMPTY, CH_SUSP, CH_PRIOR, description modified
		634	11.5.10 SGR[n] — Source Gather Register for Channel [n] (n = 0..7), expression modified
		635	11.5.11 DSR[n] — Destination Scatter Register for Channel [n] (n = 0..7), expression modified
		657	11.5.33 ReqSrcReg — Source Software Transaction Request Register, SRC_REQ, description modified
		658	11.5.34 ReqDstReg — Destination Software Transaction Request Register, DST_REQ, description modified
		667	11.6.1 DMA Transfer Mode, description modified
		668	11.6.1.1 Flow Controller and Transfer Type, description modified
		668	11.6.1.1 Flow Controller and Transfer Type, Table 11.45 DMA Transfer Type and Flow Control Combinations, description modified
		669	11.6.1.2 Block Chaining Using Linked Lists, description added
		671	11.6.1.2 Block Chaining Using Linked Lists, (2) Contiguous Address Between Blocks, description modified
		671	11.6.1.2 Block Chaining Using Linked Lists, (3) Suspension of Transfers Between Blocks, description modified
		672	11.6.1.2 Block Chaining Using Linked Lists, (4) Ending Multi-Block Transfers, description modified
		673	11.6.1.3 Basic Interface Definitions, description modified
		674	11.6.1.3 Basic Interface Definitions, Transaction, description modified
		675	11.6.1.4 Transaction Examples, (1) DMA in Burst Transaction Mode Only, description modified

Rev.	Date	Description	
		Page	Summary
1.10	Jun 30, 2020	678	11.6.1.4 Transaction Examples, Figure 11.5 Burst and Single DMA Transaction, figure modified
		679	11.6.1.5 DMAC Programing Example, subsection added
1.20	Dec 29, 2021	—	All sections, spelling, syntax errors and appearances are corrected, and expressions are modified properly
		74	6.4.1.9 DDR_CTL_08 — DDR-Controller Status & Control 08, TRST_PWRON, expression modified
		78	6.4.1.17 DDR_CTL_16 — DDR-Controller Status & Control 16, TCKESR, TCKE, expression modified
		84	6.4.1.26 DDR_CTL_25 — DDR-Controller Status & Control 25, CKE_DELAY, expression modified
		85	6.4.1.27 DDR_CTL_26 — DDR-Controller Status & Control 26, CKSRX, CKSRE, expression modified
		87	6.4.1.29 DDR_CTL_28 — DDR-Controller Status & Control 28, LP_AUTO_PD_IDLE, expression modified
		98	6.4.1.50 DDR_CTL_49 — DDR-Controller Status & Control 49, COMMAND_AGE_COUNT, AGE_COUNT, expression modified
		108	6.4.1.64 DDR_CTL_63 — DDR-Controller Status & Control 63, TODTH_WR, TODTL_2CMD, expression modified
		109	6.4.1.65 DDR_CTL_64 — DDR-Controller Status & Control 64, RD_TO_ODTH, WR_TO_ODTH, TODTH_RD, expression modified
		110	6.4.1.67 DDR_CTL_66 — DDR-Controller Status & Control 66, W2W_DIFFCS_DLY, W2R_DIFFCS_DLY, R2W_DIFFCS_DLY, R2R_DIFFCS_DLY, expression modified
		111	6.4.1.68 DDR_CTL_67 — DDR-Controller Status & Control 67, W2W_SAMECS_DLY, W2R_SAMECS_DLY, R2W_SAMECS_DLY, R2R_SAMECS_DLY, expression modified
		151	6.4.1.123 DDR_CTL_357 — DDR-Controller Status & Control 357, TDFI_DRAM_CLK_DISABLE, TDFI_CTRL_DELAY, WRLAT_ADJ, RDLAT_ADJ, expression modified
		172	6.4.2.9 WLCTRL1 — Write Leveling Control Register 1, WL2OFS, WL1OFS, description modified
		173	6.4.2.10 DQCALOFS1 — DQS Offset Setting, DQCAL2OFS, DQCAL1OFS, description modified
		616	11.3 Basic Definitions, description added
		632	11.5.9 CFG[n] — Configuration Register for Channel [n] (n = 0..7), HS_SEL_SRC, HS_SEL_DST, description modified
		636	11.5.12 RawTfr — Raw Status for IntTfr Interrupt Register, description modified
		637	11.5.13 RawBlock — Raw Status for IntBlock Interrupt Reg, description modified
		668	11.6.1.1 Flow Controller and Transfer Type, description modified
		668	11.6.1.1 Flow Controller and Transfer Type, Table 11.45 DMA Transfer Type and Flow Control Combinations, description modified

RZ/N1D Group, RZ/N1S Group, RZ/N1L Group
User's Manual: System Control and Peripheral

Publication Date: Rev.0.50 Jun 30, 2017
Rev.1.20 Dec 29, 2021

Published by: Renesas Electronics Corporation

RZ/N1D Group, RZ/N1S Group, RZ/N1L Group

Renesas Electronics Corporation

R01UH0751EJ0120