FTNANDC024

NAND FLASH CONTROLLER WITH AHB INTERFACE

Block Data Sheet
Rev.: 1.8
Issue Date: August 2013

REVISION HISTORY

FTNANDC024 Block Data Sheet

Date	Rev.	From	To
Jun. 2011	1.0	-	Original
Aug. 2011	1.1	-	Modified Table 2-1, Table 3-1, Figure 3-1, Section 3.2, and Section 5.1
Oct. 2011	1.2	-	• Updated the IP version to (1.2.0) • Modified Table 4-53 Table 4-
Nov. 2011	1.3	-	Fixed the cmp_thres error by modifying Table 2-3, Table 4-21, Section 4.2.39, and Section 5.2
Jan. 2012	1.4	-	Added the pre-fetch description in Table 4-49
Mar. 2012	1.5	-	• Updated the IP version to (1.4.0) • Modified Table 2-1, Table 4-19, and Table 4-53 • Added note in Section 4.2.32 and Section 5.2
Dec. 2012	1.6	-	• Updated the IP version to (2.0.0) • Updated Sections 1.2 through 1.5, 3.2, 4.1, 4.2.15, 4.2.20, 4.2.21, 4.2.44through 4.2.46, 5.1 through 5.3 • Updated Figure 2-1 • Updated Table 2-1, Table 2-2, Table 2-3, Table 2-5, Table 3-1, Table 4-4 through Table 4-8, Table 4-14 through Table 4-32, Table 4-37, Table 4-48 and Table 4-49 Table 4- • Updated Chapter 6 • Added Table 2-6 • Added Sections 5.6 through 5.8, 5.10, 5.11, 5.13, and 5.14
Mar. 2013	1.7	-	• Updated the IP version to (2.1.0) • Supported the eD3 flow
Aug. 2013	1.8	-	Updated Table 2-2 and Table 2-5

© Copyright Faraday Technology, 2013

All Rights Reserved.

Printed in Taiwan 2013

Faraday and the Faraday Logo are trademarks of Faraday Technology Corporation in Taiwan and/or other countries.
Other company, product and service names may be trademarks or service marks of others.

All information contained in this document is subject to change without notice. The products described in this document are NOT intended for use in implantation or other life support application where malfunction may result in injury or death to persons. The information contained in this document does not affect or change Faraday's product specification or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of Faraday or third parties. All information contained in this document was obtained in specific environments, and is presented as an illustration. The results obtained in other operating environments may vary.

THE INFORMATION CONTAINED IN THIS DOCUMENT IS PROVIDED ON AN “AS IS” BASIS. In no event will Faraday be liable for damages arising directly or indirectly from any use of the information contained in this document.

Faraday Technology Corporation
No. 5, Li-Hsin Road III, Hsinchu Science Park, Hsinchu City, Taiwan 300, R.O.C.

Faraday's home page can be found at:
http://www.faraday-tech.com

TABLE OF CONTENTS

Chapter 1 Introduction 1
1.1 Version of the IP 2
1.2 Terminology 2
1.3 Features 3
1.4 Block Diagram 4
1.5 Overview 5
Chapter 2 External Signal Description 7
2.1 Signal Connection 8
2.2 Pin Descriptions 9
Chapter 3 Configuration Parameters 13
3.1 Configurable Items 14
3.2 Relationship of Two Clocks 15
Chapter 4 Memory Map/Register Definition 17
4.1 Summary of Control Registers 18
4.2 Register Definitions 22
4.2.1 ECC Status Register 0 and Register 1 (Offset = 0x0000 and Offset = 0x0004) 22
4.2.2 ECC Control Register (Offset = 0x0008) 23
4.2.3 ECC Interrupt Enable Register (Offset = 0x0020) 26
4.2.4 ECC Interrupt Status Register (Offset = 0x0024) 26
4.2.5 ECC Status Clear Register (Offset = 0x0028) 28
4.2.6 ECC Status of Spare Region Registers 0 and Register 1 (Offset = 0x002C and Offset = 0x0030) 28
4.2.7 Spare Region ECC Control Register (Offset = 0x0034 ~ 0x0040) 29
4.2.8 Device Busy/Ready Status Register (Offset = 0x0100) 31
4.2.9 NANDC General Setting Register (Offset = 0x0104) 32
4.2.10 Memory Attribute Setting Register 1 (Offset = 0x0108) 33
4.2.11 Memory Attribute Setting Register 2 (Offset = 0x010C) 34
4.2.12 Flash AC Timing Register 0 of Channel *x* (Offset = 0x0110 + *x**0x8, *x* = 0 ~ 7)

.... 36
4.2.13 .Flash AC Timing Register 1 of Channel *x* (Offset = 0x0114 + *x* * 0x8, *x* = 0 ~ 7) 38
4.2.14 .Flash AC Timing Register 2 of Channel *x* (Offset = 0x0190 + *x* * 0x8, *x* = 0 ~ 7) 39
4.2.15 .Flash AC Timing Register 3 of Channel *x* (Offset = 0x0194 + *x* * 0x8, *x* = 0 ~ 7) 40
4.2.16 ..NANDC Interrupt Enable Register (Offset = 0x0150) 46
4.2.17 ..NANDC Interrupt Status Register (Offset = 0x0154) 46
4.2.18 ..Current Access Row Address Register (Offset = 0x0158 ~ 0x0174) 47
4.2.19 ..Read Status Register 0 and Register 1 (Offset = 0x0178 ~ 0x017C) 47
4.2.20 ..Address Toggle Bit Location Register (Offset = 0x0180) 48
4.2.21 ..NANDC Software Reset Register (Offset = 0x0184) 48
4.2.22 ..NANDC Auto-compare Pattern Register (Offset = 0x018C) 49
4.2.23 ..Variable Address Register (Offset = 0x01D0) 49
4.2.24 ..Command Queue Status Register (Offset = 0x0200) 50
4.2.25 ..Command Queue Flush Register (Offset = 0x0204) 50
4.2.26 ..Command Complete Counter Register (Offset = 0x0208) 51
4.2.27 ..Command Complete Counter Reset Register (Offset = 0x020C) 52
4.2.28 ..General Command Queue Register (Offset = 0x0280 ~ 0x0294) 52
4.2.29 ..Command Queue Register (Offset = 0x0300 + *n* * 20 ~ 0x030C + *n* * 0x20, *n* = 0 ~ 7) 52
4.2.30 ..BMC Region Status Register (Offset = 0x0400) 57
4.2.31 ..Region *n* User Mode Pointer Adjustment Register (Offset = 0x0404 + *n* * 0x4, *n* = 0 ~ 7) 58
4.2.32 ..DMA Mode Write Data Fill/Read Data Pop Register (Offset = 0x0424) 58
4.2.33 ..Region Software Reset Register (Offset = 0x0428) 59
4.2.34 ..Force Region Fill Read Data Register (Offset = 0x042C) 59
4.2.35 ..Region *x* Remaining Sector Count of Read Data Register (Offset = 0x0430 + *x* * 0x4, *x* = 0 ~ 7) 60
4.2.36 ..Revision Number Register (Offset = 0x0500) 60
4.2.37 ..Feature 1 Register (Offset = 0x0504) 60
4.2.38 ..AHB Slave Memory Space Range Register (Offset = 0x0508) 62
4.2.39 ..Global Software Reset Register (Offset = 0x050C) 64
4.2.40 ..AHB Data Slave Reset Register (Offset = 0x0510) 65

4.2.41 ..ECC Correction Capability Register 1 (Offset = 0x0514) 65
4.2.42 ..ECC Correction Capability Register 2 (Offset = 0x0518) 66
4.2.43 ..ECC Correction Capability Register 3 (Offset = 0x051C) 67
4.2.44 ..Programmable OPCODE Register (Offset = 0x0700 ~ 0x0704) 68
4.2.45 ..dqs_in_delay Register (Offset = 0x0520) 68
4.2.46 ..Spare Access Register (Offset = 0x1000 ~ 0x10FF) 69
4.2.47 ..Micro-Code SRAM Access Register (Offset = 0x2000 ~ 0x20FF) 69
4.2.48 ..Data SRAM Access Register (Offset = 0x2_0000 ~ 0x3_FFFF) 69
Chapter 5 Function Description 71
5.1 AHB Data Slave Port 72
5.2 Buffer Management Controller 73
5.3 AHB Register Slave Port 75
5.4 ECC Correction Error Handling 76
5.5 Auto Compare Error Handling 77
5.6 Auto-compare Check Mechanism 77
5.7 Scramble and Data Inverter 77
5.8 Performance 78
5.9 Abort Sequence 78
5.10 EBI (External Bus Interface) Operation 78
5.11 ODT/Warm-up Cycles Function 80
5.12 SRAM Behavior 80
5.13 Host Controller Data Format 81
5.13.1 ..Spare Register Organization 82
5.13.2 ..User Data and ECC Usages 83
5.14 Description of NANDC MicroCode 84
5.14.1 ..MicroCode Description 86
5.14.2 ..Flow Rules 90
5.14.3 ..Command Register Setting for Fixed Flow Command 1 92
5.14.4 ..Fixed Flow Command 1 (CTD) and Usage 94
5.14.5 ..Command Register Setting for Fixed Flow Command 2 104
5.14.6 ..Fixed Flow Command 2 (MTD) and Usage 106
Chapter 6 Initialization/Application 117
6.1 Initial Steps 118

6.2 Command Queue Access Method 118
6.3 Selection Restriction of BMC Region 118

LIST OF TABLES

Table 2-1. Global Signals 9
Table 2-2. AHB Data Slave Port Signals (*x* = 0 ~ 3 for Data Slave Ports 0 ~ 3) 10
Table 2-3. AHB Register Slave Port Signals 10
Table 2-4. Normal-speed NAND Flash Signals (*x* = 0 ~ 7 for NAND Flash Channel Controllers 0 ~ 7) 11
Table 2-5. High-speed NAND Flash Signals (*x* = 0 ~ 7 for NAND Flash Channel Controllers 0 ~ 7) .. 11
Table 2-6. EBI Signals (*x* = 0 ~ 7 for NAND Flash Channel Controllers 0 ~ 7) 11
Table 3-1. Configurable Items 14
Table 4-1. Summary of Control Registers 18
Table 4-2. ECC Status Register 0 (Offset = 0x00) 22
Table 4-3. ECC Status Register 1 (Offset = 0x04) 22
Table 4-4. ECC Control Register (Offset = 0x0008) 23
Table 4-5. Threshold Number of ECC Error Bits Register 0 (Offset = 0x0010) 24
Table 4-6. Threshold Number of ECC Error Bits Register 1 (Offset = 0x0014) 24
Table 4-7. Number of ECC Correction Capability Bits Register 0 (Offset = 0x0018) 25
Table 4-8. Number of ECC Correction Capability Bits Register 1 (Offset = 0x001C) 25
Table 4-9. ECC Interrupt Enable Register (Offset = 0x0020) 26
Table 4-10. ECC Interrupt Status Register (Offset = 0x0024) 26
Table 4-11. ECC Status Clear Register (Offset = 0x0028) 28
Table 4-12. ECC Status Register 0 (Offset = 0x002C) 28
Table 4-13. ECC Status Register 1 (Offset = 0x0030) 29
Table 4-14. Threshold Number of Spare Region ECC Error Bits Register 0 (Offset = 0x0034) 29
Table 4-15. Threshold Number of Spare Region ECC Error Bits Register 1 (Offset = 0x0038) 30
Table 4-16. Number of Spare Region ECC Correction Capability Bits Register 0 (Offset = 0x003C) ... 30
Table 4-17. Number of ECC Correction Capability Bits Register 1 (Offset = 0x0040) 31
Table 4-18. Device Busy/Ready Status Register (Offset = 0x0100) 31
Table 4-19. NANDC General Setting Register (Offset = 0x0104) 32
Table 4-20. Memory Attribute Setting Register 1 (Offset = 0x0108) 33
Table 4-21. Memory Attribute Setting Register 2 (Offset = 0x010C) 34

Table 4-22. Flash AC Timing Register 0 (Offset = 0x0110 + *x* * 0x8, *x* = 0 ~ 7) 36
Table 4-23. Flash AC Timing Register 1 of Channel *x* (Offset = 0x0114 + *x* *0x8, *x* = 0 ~ 7) 38
Table 4-24. Flash AC Timing Register 2 of Channel *x* (Offset = 0x0190 + *x* * 0x8, *x* = 0 ~ 7) 39
Table 4-25. Flash AC Timing Register 3 of Channel *x* (Offset = 0x0194 + *x* * 0x8, *x* = 0 ~ 7) 40
Table 4-26. NANDC Interrupt Enable Register (Offset = 0x0150) 46
Table 4-27. NANDC Interrupt Status Register (Offset = 0x0154) 46
Table 4-28. Current Access Row Address Register (Offset = 0x0158 ~ 0x0174) 47
Table 4-29. Read Status Register 0 (Offset = 0x0178) 47
Table 4-30. Read Status Register 1 (Offset = 0x017C) 48
Table 4-31. Address Toggle Bit Location Register (Offset = 0x0180) 48
Table 4-32. NANDC Software Reset Register (Offset = 0x0184) 48
Table 4-33. NANDC Auto-compare Pattern Register (Offset = 0x018C) 49
Table 4-34. Variable Address Register (Offset = 0x01D0) 49
Table 4-35. Command Queue Status Register (Offset = 0x0200) 50
Table 4-36. Command Queue Flush Register (Offset = 0x0204) 50
Table 4-37. Command Complete Counter Register (Offset = 0x0208) 51
Table 4-38. Command Complete Counter Reset Register (Offset = 0x020C) 52
Table 4-39. Command Queue First Word Register (Offset = 0x0300 + *n* * 0x20, *n* = 0 ~ 7) 52
Table 4-40. Command Queue Second Word Register (Offset = 0x0304 + *n* * 0x20, *n* = 0 ~ 7) 53
Table 4-41. Command Queue Third Word Register (Offset = 0x0308 + *n* * 0x20, *n* = 0 ~ 7) 53
Table 4-42. Command Queue Forth Word Register (Offset = 0x030C + *n* * 0x20, *n* = 0 ~ 7) 54
Table 4-44. Command Queue fifth Word Register (Offset = 0x0310 + *n* * 0x20, *n* = 0 ~ 7) 56
Table 4-45. Command Queue sixth Word Register (Offset = 0x0314 + *n* * 0x20, *n* = 0 ~ 7) 56
Table 4-46. BMC Region Status Register (Offset = 0x0400) 57
Table 4-47. Region *n* User Mode Pointer Adjustment Register (Offset = 0x0404 + *n* * 0x4, *n* = 0 ~ 7) 58
Table 4-48. DMA Mode Write Data Fill/Read Data Pop Register (Offset = 0x0424) 59
Table 4-49. Region Software Reset Register (Offset = 0x0428) 59
Table 4-50. Force Region Fill Read Data Register (Offset = 0x042C) 60
Table 4-51. Region *x* Remaining Sector Count of Read Data Register (Offset = 0x0430 + *x* * 0x4, *x* = 0 ~ 7) 60
Table 4-52. Revision Number Register (Offset = 0x0500) 60
Table 4-53. Feature 1 Register (Offset = 0x0504) 60
Table 4-54. AHB Slave Memory Space Range Register (Offset = 0x0508) 62

Table 4-55. Global Software Reset Register (Offset = 0x050C) 64
Table 4-56. AHB Data Slave Reset Register (Offset = 0x0510) 65
Table 4-57. ECC Correction Capability Register 1 (Offset = 0x0514) 65
Table 4-58. ECC Correction Capability Register 2 (Offset = 0x0518) 66
Table 4-59. ECC Correction Capability Register 3 (Offset = 0x051C) 67
Table 4-60. Programmable OPCODE Register (Offset = 0x0700) 68
Table 4-61. Programmable OPCODE Register (Offset = 0x0704) 68
Table 4-62. dqs_in_delay Register (Offset = 0x0520) 68
Table 5-1. Summary of MicroCode 84
Table 5-2. OPCODE 86
Table 5-3. Command Register Setting 92

LIST OF FIGURES

Figure 1-1. Block Diagram 4
Figure 1-2. Relationship among Block, Page, and Sector Sizes 5
Figure 2-1. Signal Connection 8
Figure 3-1. Relationship among mem_clk, core_clk, and clk_en 15
Figure 4-1. Warm-up 2 Cycles for RE_n Signal and DQS Signal 36
Figure 4-2. Timing Diagram of Address State 37
Figure 4-3. Timing Diagram of Command State 37
Figure 4-4. Timing Diagram of Write Data State 37
Figure 4-5. Timing Diagram of Read Data State 38
Figure 4-6. Timing Diagram of Busy State 39
Figure 4-7. Timing Diagram of Buffer1/Buffer2/Buffer3/Buffer4 State 40
Figure 4-8. Timing Diagram of Address State 41
Figure 4-9. Timing Diagram of Preamble Write State 41
Figure 4-10. Timing Diagram of Preamble Read State 42
Figure 4-11. Timing Diagram of Post-amble Write State 42
Figure 4-12. Timing Diagram of Post-amble Read State 43
Figure 4-13. Timing Diagram of Preamble Write State 43
Figure 4-14. Timing Diagram of Preamble Read State 44
Figure 4-15. Timing Diagram of Post-amble Write State 44
Figure 4-16. Timing Diagram of Post-amble Read State 45
Figure 4-17. Summary of AC Timing Usage 45
Figure 4-18. Data SRAM Address Offset Mapped to BMC Region 70
Figure 5-1. Mapping between AHB Slave and BMC Region 72
Figure 5-2. User Mode Operation 74
Figure 5-3. Scramble and Data Inverter 77
Figure 5-4. External Bus Interface Timing 78
Figure 5-5. Grant Stop Handshaking with EBI Arbiter 79
Figure 5-6. Example of EBI Interface Connection 79

Figure 5-7. Waveform for SRAM Behavior 80
Figure 5-8. Data Format at no_ecc_parity = ‘0’ 81
Figure 5-9. Spare Register Organization 82
Figure 5-10. User Data and ECC Usages 83
Figure 5-11. 2-plane Write 4 Pages Cross Block 91

Chapter 1

Introduction

This chapter contains the following sections:

- 1.1 Version of the IP
- 1.2 Terminology
- 1.3 Features
- 1.4 Block Diagram
- 1.5 Overview

1.1 Version of the IP

IP release version: 2.1.0

1.2 Terminology

BMC: Buffer Management Unit

NANDC: NAND Flash channel Controller (Except for used in term of "FTNANDC024")

NANDIF: Module between NANDC and flash

ECC: Error Check and Correction

F/W: Firmware

H/W: Hardware

MSC: MicroCode SRAM Controller

Spare Data: User-defined data

DDR: Double Data Rate Interface same to high-speed interface

eD3: Samsung 21-nm TLC Flash

1.3 Features

- Supports page sizes of 512, 2K, 4K, 8K and 16K bytes for NAND Flash
- Supports 8-bit data bus for NAND Flash
- Connects up to eight CEs for each channel (Flash chip enable)
- Supports programmable timing parameters for NAND Flash
- Supports programmable command flow for different NAND Flash flow
- Supports interleaving operation when two or more CEs connected to one channel
- Supports 74 bits of ECC correction capability for spare region
- Supports synchronous/asynchronous/same clock modes between core and Flash memory interface
- Supports asynchronous/same clock mode between core and AHB interface
- Supports AHB 32/64 bits data width
- Supports 74 bits for ECC correction capability of a 512-byte or 1K-byte sector
- Supports DMA handshake mode
- Supports scramble function
- Supports data inverse function
- Supports valid page of block function
- Provides AHB data slave port to support RETRY response without exploiting DMA handshake mode
- Supports flexible user-defined data length
- Supports warm-up function
- Supports ODT function
- Supports EBI function
- Supports ONFi 2.2 high-speed interface
- Supports Toggle 1.0/2.0 interface for Samsung NAND Flash
- Supports SLC, MLC, and TLC Flash
- Supports Samsung 21-nm TLC Flash

1.4 Block Diagram

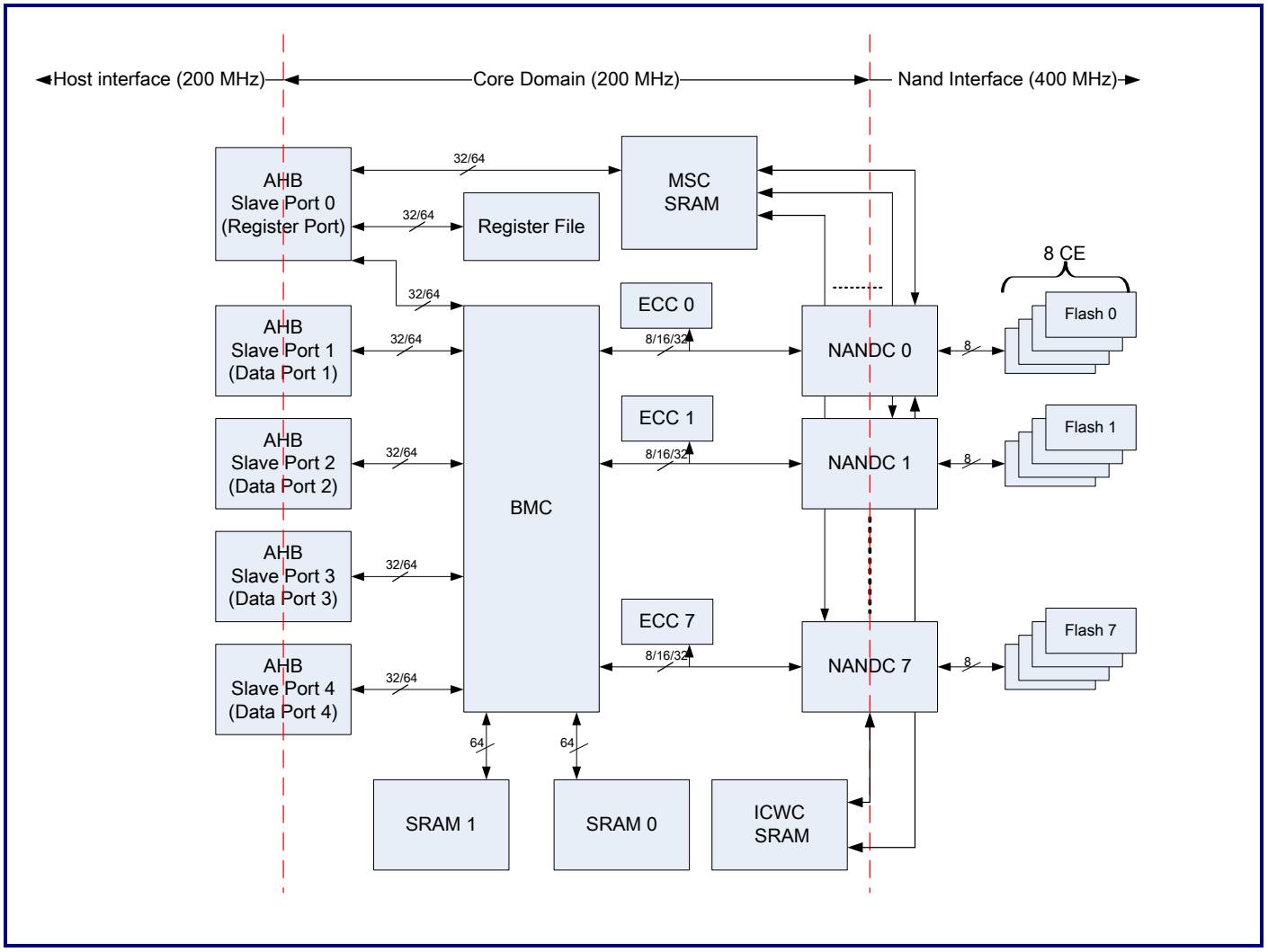


Figure 1-1. Block Diagram

Please refer to Figure 1-2 for the relationship among the Block, Page, and Sector sizes.

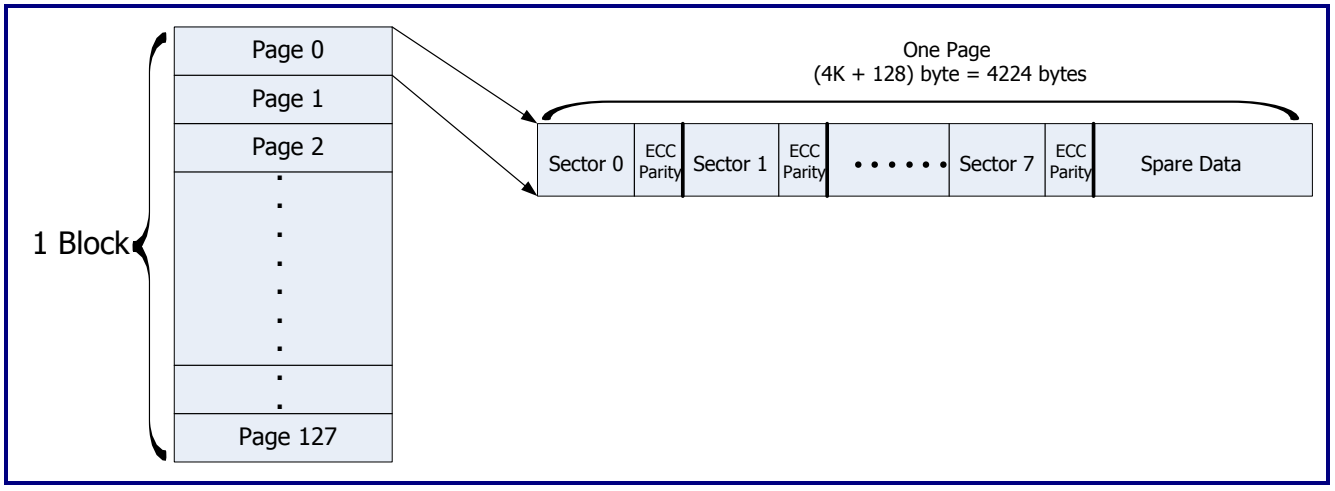


Figure 1-2. Relationship among Block, Page, and Sector Sizes

Notes:

1. The block and page sizes depend on the specification of Flash.
2. The sector size is programmable by users.

1.5 Overview

FTNANDC024 is a NAND Flash controller with AHB interface. This controller supports up to eight NAND channels and the data bus of each NAND channel is 8–bit wide. This controller also supports up to four AHB data slave ports for accessing the data buffer. The registers can be accessed through a standalone AHB slave. BMC is used to control the data storage space for data from the AHB data slave port to NANDC or from NANDC to the AHB data slave port. If an error occurs after the parity check, the ECC engine will be responsible for the ECC parity generation, ECC parity check, and data correction. The MSC SRAM module provides a dedicated space for storing the programming flow, which can be read from NANDC to execute flash flow or written from register port.

Chapter 2

External Signal Description

This chapter contains the following sections:

- 2.1 Signal Connection
- 2.2 Pin Descriptions

2.1 Signal Connection

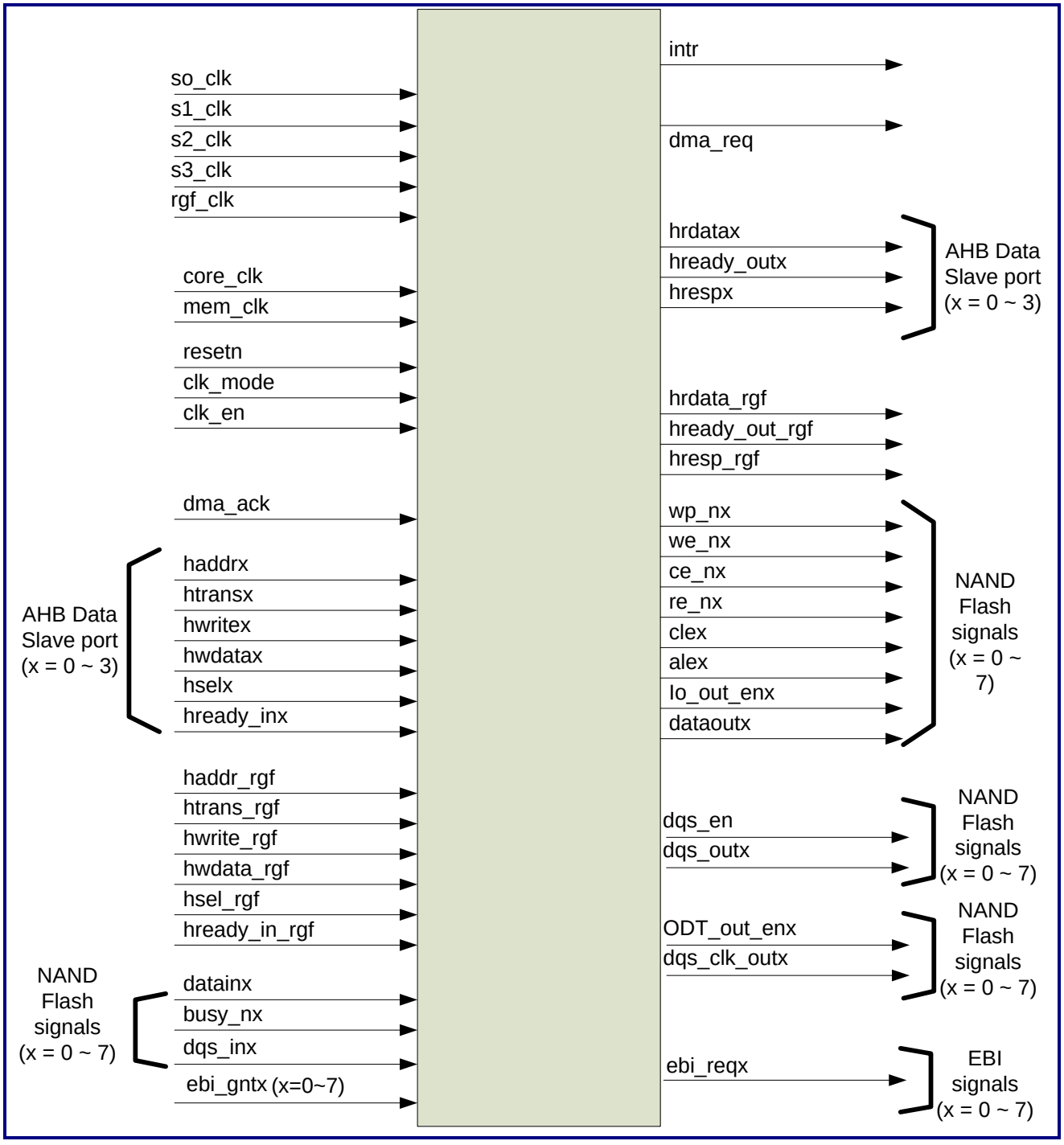


Figure 2-1. Signal Connection

2.2 Pin Descriptions

Table 2-1. Global Signals

Signal Name	I/O Type	Description
S0_hclk	Input	AHB slave0 clock If the slave port 0 is configured as the ASYNC mode, S0_hclk can be fully asynchronous with core_clk.
S1_hclk	Input	AHB slave1 clock If the slave port 1 is configured as the ASYNC mode, S1_hclk can be fully asynchronous with core_clk.
S2_hclk	Input	AHB slave2 clock If the slave port 2 is configured as the ASYNC mode, S2_hclk can be fully asynchronous with core_clk.
S3_hclk	Input	AHB slave3 clock If the slave port 3 is configured as the ASYNC mode, S3_hclk can be fully asynchronous with core_clk.
rgf_hclk	Input	AHB register clock
core_clk	Input	IP core clock
mem_clk	Input	NAND interface clock
resetn	Input	AHB hardware reset Active low
intr	Output	Hardware interrupt
clk_mode	Input	Clock mode selection between core_clk and mem_clk This signal will be active only when the clock mode configuration is BYPORT. 1: Asynchronous mode 0: Synchronous mode
clk_en	Input	Clock enable between core_clk and mem_clk in the synchronous mode
dma_req[7:0]	Output	DMA handshake request This signal will be asserted when the handshake mode of a NAND command is enabled and at least one sector will be read if it is a read command or at least one sector space will be available in BMC if it is a write command.
dma_ack[7:0]	Input	DMA handshake acknowledge This signal will be asserted by the DMA controller when a length of 512 bytes is transferred. The burst size of DMA can be set no larger than 512 bytes.

Table 2-2. AHB Data Slave Port Signals (*x* = 0 ~ 3 for Data Slave Ports 0 ~ 3)

Signal Name	I/O Type	Description
hmaster[3:0]	Input	AHB master number This signal will be active when the asynchronous AHB slave port is configured.
haddrx[18:0]	Input	AHB address bus
htransx[1:0]	Input	AHB transfer type
hburstx[2:0]	Input	AHB burst type
hwritex	Input	AHB transfer direction
hwdatax[*x*:0]	Input	AHB write data bus, x denotes 31 bits or 63 bits
hselx	Input	AHB slave select
hready_inx	Input	AHB transfer done input
hrdatax[31:0]	Output	AHB read data bus, x denotes 31 bits or 63 bits
hready_outx	Output	AHB transfer done output
hrespx[1:0]	Output	AHB transfer response The response can be OKAY or RETRY.
Hsplit[15:0]	Output	AHB split completion request

Note: HSIZE is not used. The data slave port can only support the accesses with HSIZE in WORD/DWORD.

Table 2-3. AHB Register Slave Port Signals

Signal Name	I/O Type	Description
haddr_rgf[18:0]	Input	AHB address bus
htrans_rgf[1:0]	Input	AHB transfer type
hwrite_rgf	Input	AHB transfer direction
hwdata_rgf[*x*:0]	Input	AHB write data bus, *x* denotes 31 bits or 63 bits
hsize_rgf[1:0]	Input	AHB transfer size
hsel_rgf	Input	AHB slave select
hready_in_rgf	Input	AHB transfer done input
hrdata_rgf[*x*:0]	Output	AHB read data bus, *x* denotes 31 bits or 63 bits
hready_out_rgf	Output	AHB transfer done output
hresp_rgf[1:0]	Output	AHB transfer response The response can only be OKAY.

Table 2-4. Normal-speed NAND Flash Signals (*x* = 0 ~ 7 for NAND Flash Channel Controllers 0 ~ 7)

Signal Name	I/O Type	Description
wp_nx	Output	Write protect
we_nx	Output	Write enable
ce_nx	Output	Chip enable
re_nx	Output	Read enable
clex	Output	Command latch enable
alex	Output	Address latch enable
io_out_enx[7:0]	Output	Data output enable
dataoutx[7:0]	Output	Data output
datainx[7:0]	Input	Data input
busy_nx	Input	Ready/Busy signal

Table 2-5. High-speed NAND Flash Signals (*x* = 0 ~ 7 for NAND Flash Channel Controllers 0 ~ 7)

Signal Name	I/O Type	Description
dgs_out[7:0]	Output	DQS output
dqs_in[7:0]	Input	DQS input
dqs_en[7:0]	Output	DQS enable
dqs_clk_outx	Output	DLL reference clock
ODT_out_enx	Output	ODT (On-Die Terminal) I/O enable Please refer to Toggle 2.0 for the detailed information.
DQS_c	Output	DQS complement output enable signal
RE_c	Output	RE_c complement output enable signal
ODT[3:0]	Output	ODT IO enable
Vref	Output	External reference voltage for the input and I/O signals

Table 2-6. EBI Signals (*x* = 0 ~ 7 for NAND Flash Channel Controllers 0 ~ 7)

Signal Name	I/O Type	Description
ebi_reqx	Output	EBI request signal
ebi_gntx	Input	EBI grant signal

Chapter 3

Configuration Parameters

This chapter contains the following sections:

- 3.1 Configurable Items
- 3.2 Relationship of Two Clocks

3.1 Configurable Items

The configurable items of FTNANDC024 are listed and described in Table 3-1.

Table 3-1. Configurable Items

Item Name	Possible Value	Description
FTNANDC024_CORE_MEM_x	SYNC/BYPORT/SAME	Clock mode between core_clk and mem_clk SYNC: Integer ratio between core_clk and mem_clk BYPORT: SYNC mode or ASYNC mode is selected by the input port. SAME: Only single clock is core_clk, which is used for both NANDC and NANDIF.
FTNANDC024_MEM_DDR_x	ON/OFF	Enable or disable the DDR Flash interface
FTNANDC024_AHBS	1 ~ 4	Number of the AHB data slave ports
FTNANDC024_RGF_x	SAME/ASYNC	AHB register slave port clock mode SAME: hclk of the register slave port is the same as the core clock. ASYNC: hclk of the register slave port has no relation with the core clock.
FTNANDC024_AHB_Sx_y	x: 0~3 y:SAME/ASYNC	AHB data slave port clock mode (x = 0 ~ 3) SAME: hclk of the data slave port is the same as core_clk. ASYNC: hclk of the data slave port has no relation with core_clk. Note: if the AHB Slave ports are connected to single AHB bus, the AHB data slave port clock mode must be all SAME or ASYNC. For example, if users configured the AHB Slave port 2, Slave0 and Slave1, and Slave0 and Slave1 must be configured as both SAME or ASYNC, S0 is ASYNC and S1 is SAME or S0 is SAME and S1 is ASYNC is prohibited.
FTNANDC024_AHB_DW_x	x: 32/64	AHB slave data port width
FTNANDC024_NANDCH_x	1, 2, 4, or 8	Number of the NANDC channels [Note]
FTNANDC024_CQDEPTH_x	1, 2, or 4	Depth of the NAND command queue
FTNANDC024_ECC_CS_NUM_x	1 ~ 8	Number of the Chien search entries
FTNANDC024_ECC_CS_UNFOLD_x	4, 8, 16, 32, 64	Chien search unfolding factor
FTNANDC024_ECC_CORR_BITx_y	x = 1 ~ 74 y = ON/OFF	1 ~ 74 ECC correction bit support
FTNANDC024_MSC_DEPTH	128/256	Depth of MicroCode SRAM Controller(MSC), 128/256 byte
FTNANDC024_CH_DATA_x	8/16/32	Channel data width, 8/16/32 bits

Item Name	Possible Value	Description
FTNANDC024_ECC_MBA_FFM_NUM_x	8/16/32	MBA multiplier number
FTNANDC024_EBI_x	ON/OFF	Support the EBI interface It is used for the SDR Interface. There is no DDR Interface.
FTNANDC024_ED3_x	ON/OFF	Enable the ED3 logic

Note: The number of regions is equal to the number of the NANDC channels.

3.2 Relationship of Two Clocks

If the clock mode is synchronous, the clock enable signals (clk_en) will be required for a faster clock to perform synchronization. Figure 3-1 depicts the behaviors of core_clk, mem_clk, and clk_en when the speed ratio between core_clk and mem_clk is 1:2.

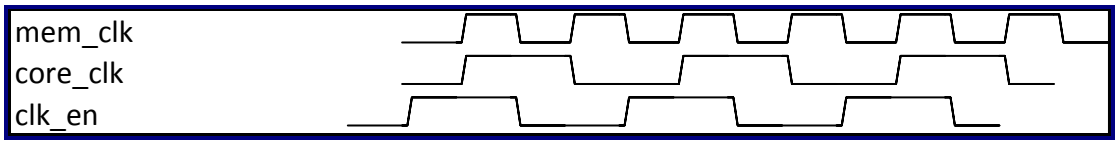


Figure 3-1. Relationship among mem_clk, core_clk, and clk_en

If the clock speed ratio is 1:1, clk_en should be tied to '1'.

Chapter 4

Memory Map/Register Definition

This chapter contains the following sections:

- 4.1 Summary of Control Registers
- 4.2 Register Definitions

4.1 Summary of Control Registers

The FTNANDC024 control registers are summarized in Table 4-1. The registers between offsets 0x0 to 0x204 and offsets from 0x400 to 0x1FFFF can be accessed with BYTE, HWORD, WORD or DWORD as HSIZE. The command queue input entry can only accept WORD/DWORD as HSIZE.

Table 4-1. Summary of Control Registers

Address Offset	Type	Description	Reset Value
ECC Control			
0x0_0000	R	ECC Status Register 0	0x0000_0000
0x0_0004	R	ECC Status Register 1	0x0000_0000
0x0_0008	R/W	ECC Control Register	0x0000_0000
0x0_0010	R/W	Threshold Number of ECC Error Bits Register 0	0x0000_0000
0x0_0014	R/W	Threshold Number of ECC Error Bits Register 1	0x0000_0000
0x0_0018	R/W	Number of ECC Correction Capability Bits Register 0	0x0000_0000
0x0_001C	R/W	Number of ECC Correction Capability Bits Register 1	0x0000_0000
0x0_0020	R/W	ECC Interrupt Enable Register	0x0000_0000
0x0_0024	R/W1C	ECC Interrupt Status Register	0x0000_0000
0x0_0028	W1C	ECC Status Clear Register	0x0000_0000
0x0_002C	R	ECC Status of Spare Region Register 0	0x0000_0000
0x0_0030	R	ECC Status of Spare Region Register 1	0x0000_0000
0x0_0034	R	Spare Region ECC Control Register 0	0x0000_0000
0x0_0038	R	Spare Region ECC Control Register 1	0x0000_0000
0x0_003C	R	Spare Region ECC Control Register 2	0x0000_0000
0x0_0040	R	Spare Region ECC Control Register 3	0x0000_0000
NANDC Control			
0x0_0100	R	Device Busy/Ready Status Register	0x0000_00FF
0x0_0104	R/W	NANDC General Setting Register	0x0200_6000
0x0_0108	R/W	Memory Attribute Setting Register1	0x0000_0000
0x0_010C	R/W	Memory Attribute Setting Register2	0x00FF_8000
0x0_0110	R/W	AC Timing Register 0 of NANDC 0	0x0F1F_0F1F_
0x0_0114	R/W	AC Timing Register 1 of NANDC 0	0x0000_7F7F
0x0_0118	R/W	AC Timing Register 0 of NANDC 1	0x0F1F_0F1F_
0x0_011C	R/W	AC Timing Register 1 of NANDC 1	0x0000_7F7F
0x0_0120	R/W	AC Timing Register 0 of NANDC 2	0x0F1F_0F1F_

Address Offset	Type	Description	Reset Value
0x0_0124	R/W	AC Timing Register 1 of NANDC 2	0x0000_7F7F
0x0_0128	R/W	AC Timing Register 0 of NANDC 3	0x0F1F_0F1F
0x0_012C	R/W	AC Timing Register 1 of NANDC 3	0x0000_7F7F
0x0_0130	R/W	AC Timing Register 0 of NANDC 4	0x0F1F_0F1F
0x0_0134	R/W	AC Timing Register 1 of NANDC 4	0x0000_7F7F
0x0_0138	R/W	AC Timing Register 0 of NANDC 5	0x0F1F_0F1F
0x0_013C	R/W	AC Timing Register 1 of NANDC 5	0x0000_7F7F
0x0_0140	R/W	AC Timing Register 0 of NANDC 6	0x0F1F_0F1F
0x0_0144	R/W	AC Timing Register 1 of NANDC 6	0x0000_7F7F
0x0_0148	R/W	AC Timing Register 0 of NANDC 7	0x0F1F_0F1F
0x0_014C	R/W	AC Timing Register 1 of NANDC 7	0x0000_7F7F
0x0_0150	R/W	NANDC Interrupt Enable Register	0x0000_0000
0x0_0154	R/W1C	NANDC Interrupt status Register	0x0000_0000
0x0_0158	R	Current Access Row Address of Channel 0	0x0000_0000
0x0_015C	R	Current Access Row Address of Channel 1	0x0000_0000
0x0_0160	R	Current Access Row Address of Channel 2	0x0000_0000
0x0_0164	R	Current Access Row Address of Channel 3	0x0000_0000
0x0_0168	R	Current Access Row Address of Channel 4	0x0000_0000
0x0_016C	R	Current Access Row Address of Channel 5	0x0000_0000
0x0_0170	R	Current Access Row Address of Channel 6	0x0000_0000
0x0_0174	R	Current Access Row Address of Channel 7	0x0000_0000
0x0_0178	R	Read Status Register 0	0x0000_0000
0x0_017C	R	Read Status Register 1	0x0000_0000
0x0_0180	R/W	Address Toggle Bit Location Register	0x0000_0000
0x0_0184	W	NANDC Software Reset Register	-
0x0_0188	R/W	NANDC Auto Compare Pattern Threshold Register	0x0000_0000
0x0_018C	R/W	NANDC Auto Compare Pattern Register	0x0000_0000
0x0_0190	-	AC Timing Register 2 of NANDC 0	0x7F7F_7F7F
0x0_0194	-	AC Timing Register 3 of NANDC 0	0xFF1F_001F
0x0_0198	-	AC Timing Register 2 of NANDC 1	0x7F7F_7F7F
0x0_019C	-	AC Timing Register 3 of NANDC 1	0xFF1F_001F
0x0_01A0	-	AC Timing Register 2 of NANDC 2	0x7F7F_7F7F
0x0_01A4	-	AC Timing Register 3 of NANDC 2	0xFF1F_001F
0x0_01A8	-	AC Timing Register 2 of NANDC 3	0x7F7F_7F7F

Address Offset	Type	Description	Reset Value
0x0_01AC	-	AC Timing Register 3 of NANDC 3	0xFF1F_001F
0x0_01B0	-	AC Timing Register 2 of NANDC 4	0x7F7F_7F7F
0x0_01B4	-	AC Timing Register 3 of NANDC 4	0xFF1F_001F
0x0_01B8	-	AC Timing Register 2 of NANDC 5	0x7F7F_7F7F
0x0_01BC	-	AC Timing Register 3 of NANDC 5	0xFF1F_001F
0x0_01C0	-	AC Timing Register 2 of NANDC 6	0x7F7F_7F7F
0x0_01C4	-	AC Timing Register 3 of NANDC 6	0xFF1F_001F
0x0_01C8	-	AC Timing Register 2 of NANDC 7	0x7F7F_7F7F
0x0_01CC	-	AC Timing Register 3 of NANDC 7	0xFF1F_001F
Command Queue Control			
0x0_0200	R	Command Queue Status Register	0x0000_00FF
0x0_0204	W	Command Queue Flush Register	-
0x0_0208	R	Command Complete Counter	0x0000_0000
0x0_020C	W	Command Complete Counter Reset Register	-
0x0_0280 ~ 0x0_028C	W	General Command Queue Access	-
NANDC0 Command Queue			
0x0_0300 ~ 0x0_0314	R/W	Command Queue 0	0x0000_0000 (All)
NANDC1 Command Queue			
0x0_0320 ~ 0x0_0334	R/W	Command Queue 1	0x0000_0000 (All)
NANDC2 Command Queue			
0x0_0340 ~ 0x0_0354	R/W	Command Queue 2	0x0000_0000 (All)
NANDC3 Command Queue			
0x0_0360 ~ 0x0_0374	R/W	Command Queue 3	0x0000_0000 (All)
NANDC4 Command Queue			
0x0_0380 ~ 0x0_0394	R/W	Command Queue 4	0x0000_0000 (All)
NANDC5 Command Queue			
0x0_03A0 ~ 0x0_03B4	R/W	Command Queue 5	0x0000_0000 (All)
NANDC6 Command Queue			
0x0_03C0 ~ 0x0_03D4	R/W	Command Queue 6	0x0000_0000 (All)
NANDC7 Command Queue			
0x0_03E0 ~ 0x0_03F4	R/W	Command Queue 7	0x0000_0000 (All)
BMC Control			
0x0_0400	R	Region Status Register	0xFF00_00FF
0x0_0404	W	Region 0 User Mode Pointer Adjustment Register	-

Address Offset	Type	Description	Reset Value
0x0_0408	W	Region 1 User Mode Pointer Adjustment Register	-
0x0_040C	W	Region 2 User Mode Pointer Adjustment Register	-
0x0_0410	W	Region 3 User Mode Pointer Adjustment Register	-
0x0_0414	W	Region 4 User Mode Pointer Adjustment Register	-
0x0_0418	W	Region 5 User Mode Pointer Adjustment Register	-
0x0_041C	W	Region 6 User Mode Pointer Adjustment Register	-
0x0_0420	W	Region 7 User Mode Pointer Adjustment Register	-
0x0_0424	W	DMA Mode Write Data Fill/Read Data Pop Register	-
0x0_0428	W	Region Software Reset Register	-
0x0_042C	R/W	Force Region Fill Read Data Register	0x0000_0000
0x0_0430	R	Region 0 remaining sector count of read data	0x0000_0000
0x0_0434	R	Region 1 remaining sector count of read data	0x0000_0000
0x0_0438	R	Region 2 remaining sector count of read data	0x0000_0000
0x0_043C	R	Region 3 remaining sector count of read data	0x0000_0000
0x0_0440	R	Region 4 remaining sector count of read data	0x0000_0000
0x0_0444	R	Region 5 remaining sector count of read data	0x0000_0000
0x0_0448	R	Region 6 remaining sector count of read data	0x0000_0000
0x0_044C	R	Region 7 remaining sector count of read data	0x0000_0000
Miscellaneous			
0x0_0500	R	Revision Number Register	TBD
0x0_0504	R	Feature Register	TBD
0x0_0508	R/W	AHB Slave Memory Space Range Register	0x0280_FF01
0x0_050C	W	Global Software Reset Register	-
0x0_0510	W	AHB data slave Reset Register	-
0x0_514	R	ECC Correction Capability Register1	-
0x0_518	R	ECC Correction Capability Register2	
0x0_518	R	ECC Correction Capability Register3	
Programmable OPCODE			
0x0_0700 ~ 0x0_0707	R/W	Programmable OP Code Register	0xxxxx_xxxx
Spare SRAM Access Port			
0x0_1000 ~ 0x0_10FF	-	The spare data are stored in .spare SRAM and can be accessed through this port.	-
Programmable Flow Control			
0x0_2000 ~ 0x0_20FF	R/W	Programmable Flow Control Register	0xxxxx_xxxx (All)

Address Offset	Type	Description	Reset Value
Data SRAM Access Port			
0x2_0000 ~ 0x3_FFFF	-	SRAM can be accessed directly through this register.	-

4.2 Register Definitions

The following subsections provide the detailed descriptions of the control registers

4.2.1 ECC Status Register 0 and Register 1 (Offset = 0x0000 and Offset = 0x0004)

The ECC status register reflects the number of ECC error bits. Only the maximum error bits value is recorded. To clear this register, write '1' to the ECC status clear register (Offset = 0x0028). When ECC fails, the ECC status register will not be valid and the ECC status information may not be correct.

Table 4-2. ECC Status Register 0 (Offset = 0x00)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	ecc_err_no_ch3	R	Number of the ECC error bits on Channel 3
23	-	-	Reserved
[22:16]	ecc_err_no_ch2	R	Number of the ECC error bits on Channel 2
15	-	-	Reserved
[14:8]	ecc_err_no_ch1	R	Number of the ECC error bits on Channel 1
7	-	-	Reserved
[6:0]	ecc_err_no_ch0	R	Number of the ECC error bits on Channel 0

Table 4-3. ECC Status Register 1 (Offset = 0x04)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	ecc_err_no_ch7	R	Number of ECC error bits on Channel 7
23	-	-	Reserved
[22:16]	ecc_err_no_ch6	R	Number of ECC error bits on Channel 6
15	-	-	Reserved
[14:8]	ecc_err_no_ch5	R	Number of ECC error bits on Channel 5
7	-	-	Reserved
[6:0]	ecc_err_no_ch4	R	Number of ECC error bits on Channel 4

4.2.2 ECC Control Register (Offset = 0x0008)

The ECC control register provides the following features:

- Use threshold of the ECC error bits to generate the ECC interrupt
- ECC correction capability bits. Users must set an accepted ECC correction capability at the data region and spare region whenever the ECC engine is enabled or disabled. For example, if the user hardware configuration can only be 8-bit, 24-bit, or 64-bit capability, user must set 8-bit, 24-bit, or 64-bit ECC correction capability at both data region and spare region.
- ECC base size (512 bytes or 1K bytes)
- ECC function enable/disable switch
- ECC error data blocking function mask (Sectors containing the uncorrectable data that can be read from the AHB data slave port without blocking.)

Table 4-4. ECC Control Register (Offset = 0x0008)

Bit	Name	Type	Description
[31:18]	-	-	Reserved
17	no_ecc_parity	R/W	No ECC parity 1: No spacing between the data sectors to accommodate ECC parity 0: Keep spacing between the data sectors to accommodate ECC parity This bit is set when using the Flash device with the embedded ECC circuit. Note: no_ecc_parity is enabled; ecc_en is not valid.
16	ecc_base	R/W	ECC base size 1: ECC parity is generated and checked based on the data size of 1K bytes. 0: ECC parity is generated and checked based on the data size of 512 bytes.
[15:8]	ecc_en*x*	R/W	ECC function enable/disable switch of channel 0 ~ 7 (*x* = 0 ~ 7) 1: Enable the ECC function 0: Disable the ECC function
[7:0]	ecc_err_mask*x*	R/W	ECC error data blocking mask of channel 0 ~ 7 (*x* = 0 ~ 7) 1: Mask the ECC error data blocking function 0: Keep the ECC error data blocking function Note: The ECC-related interrupts will remain asserting when this bit is set. It is only for the data region, which does not include the spare region.

Table 4-5. Threshold Number of ECC Error Bits Register 0 (Offset = 0x0010)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	ecc_thres_bits3	R/W	Threshold number of the ECC error bits of Channel 3 7'd0 ~ 7'd73: 1 bit ~ 74 bits
23	-	-	Reserved
[22:16]	ecc_thres_bits2	R/W	Threshold number of the ECC error bits of Channel 2 7'd0 ~ 7'd73: 1 bit ~ 74 bits
15	-	-	Reserved
[14:8]	ecc_thres_bits1	R/W	Threshold number of the ECC error bits of Channel 1 7'd0 ~ 7'd73: 1 bit ~ 74 bits
7	-	-	Reserved
[6:0]	ecc_thres_bits0	R/W	Threshold number of the ECC error bits of Channel 0 7'd0 ~ 7'd73: 1 bit ~ 74 bits

Table 4-6. Threshold Number of ECC Error Bits Register 1 (Offset = 0x0014)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	ecc_thres_bits7	R/W	Threshold number of the ECC error bits of Channel 7 7'd0 ~ 7'd73: 1 bit ~ 74 bits
23	-	-	Reserved
[22:16]	ecc_thres_bits6	R/W	Threshold number of the ECC error bits of Channel 6 7'd0 ~ 7'd73: 1 bit ~ 74 bits
15	-	-	Reserved
[14:8]	ecc_thres_bits5	R/W	Threshold number of the ECC error bits of Channel 5 7'd0 ~ 7'd73: 1 bit ~ 74 bits
7	-	-	Reserved
[6:0]	ecc_thres_bits4	R/W	Threshold number of the ECC error bits of Channel 4 7'd0 ~ 7'd73: 1 bit ~ 74 bits

The permitted numbers of the ECC correction capability bits are listed in Table 4-7 and Table 4-8.

Table 4-7. Number of ECC Correction Capability Bits Register 0 (Offset = 0x0018)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	ecc_corr_bits3	R/W	This field sets the number of the ECC correction capability bits of Channel 3. 7'd0 ~ 7'd73: 1 bit ~ 74 bits
23	-	-	Reserved
[22:16]	ecc_corr_bits2	R/W	This field sets the number of the ECC correction capability bits of Channel 2. 7'd0 ~ 7'd73: 1 bit ~ 74 bits
15	-	-	Reserved
[14:8]	ecc_corr_bits1	R/W	This field sets the number of the ECC correction capability bits of Channel 1. 7'd0 ~ 7'd73: 1 bit ~ 74 bits
7	-	-	Reserved
[6:0]	ecc_corr_bits0	R/W	This field sets the number of the ECC correction capability bits of Channel 0. 7'd0 ~ 7'd73: 1 bit ~ 74 bits

Table 4-8. Number of ECC Correction Capability Bits Register 1 (Offset = 0x001C)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	ecc_corr_bits7	R/W	This field sets the number of the ECC correction capability bits of Channel 7. 7'd0 ~ 7'd73: 1 bit ~ 74 bits
23	-	-	Reserved
[22:16]	ecc_corr_bits6	R/W	This field sets the number of the ECC correction capability bits of Channel 6. 7'd0 ~ 7'd73: 1 bit ~ 74 bits
15	-	-	Reserved
[14:8]	ecc_corr_bits5	R/W	This field sets the number of the ECC correction capability bits of Channel 5. 7'd0 ~ 7'd73: 1 bit ~ 74 bits
7	-	-	Reserved
[6:0]	ecc_corr_bits4	R/W	This field sets the number of the ECC correction capability bits of Channel 4. 7'd0 ~ 7'd73: 1 bit ~ 74 bits

4.2.3 ECC Interrupt Enable Register (Offset = 0x0020)

The ECC function may results in the following two kinds of interrupts:

- Number of the ECC error bits hit the threshold which is set in the ECC control register.
- Failure of the ECC correction

Table 4-9. ECC Interrupt Enable Register (Offset = 0x0020)

Bit	Name	Type	Description
[31:4]	-	-	Reserved
3	ecc_err_hit_thres_en_sp	R/W	ECC error bits hit the threshold number in the spare region interrupt enable. 1: Enable 0: Disable
2	ecc_corr_fail_en_sp	R/W	ECC correction fail of the spare region interrupt enable. 1: Enable 0: Disable
1	ecc_err_hit_thres_en	R/W	ECC error bits hit the threshold number interrupt enable. 1: Enable 0: Disable
0	ecc_corr_fail_en	R/W	ECC correction fail interrupt enable 1: Enable 0: Disable

4.2.4 ECC Interrupt Status Register (Offset = 0x0024)

The ECC interrupt status register indicates the failure of ECC correction fail or the error bits hitting the threshold of each channel. Write '1' to the specific bit to clear the status.

Table 4-10. ECC Interrupt Status Register (Offset = 0x0024)

Bit	Name	Type	Description
31	ecc_err_hit_thres_ch7_sp	R/W1C	ECC error bits hit the threshold of Channel 7 spare region.
30	ecc_err_hit_thres_ch6_sp	R/W1C	ECC error bits hit the threshold of Channel 6 spare region.
29	ecc_err_hit_thres_ch5_sp	R/W1C	ECC error bits hit the threshold of Channel 5 spare region.
28	ecc_err_hit_thres_ch4_sp	R/W1C	ECC error bits hit the threshold of Channel 4 spare region.
27	ecc_err_hit_thres_ch3_sp	R/W1C	ECC error bits hit the threshold of Channel 3 spare region.
26	ecc_err_hit_thres_ch2_sp	R/W1C	ECC error bits hit the threshold of Channel 2 spare region.

Bit	Name	Type	Description
25	ecc_err_hit_thres_ch1_sp	R/W1C	ECC error bits hit the threshold of Channel 1 spare region.
24	ecc_err_hit_thres_ch0_sp	R/W1C	ECC error bits hit the threshold of Channel 0 spare region.
23	ecc_err_fail_ch7_sp	R/W1C	ECC correction fail of the Channel 7 spare region
22	ecc_err_fail_ch6_sp	R/W1C	ECC correction fail of the Channel 6 spare region
21	ecc_err_fail_ch5_sp	R/W1C	ECC correction fail of the Channel 5 spare region
20	ecc_err_fail_ch4_sp	R/W1C	ECC correction fail of the Channel 4 spare region
19	ecc_err_fail_ch3_sp	R/W1C	ECC correction fail of the Channel 3 spare region
18	ecc_err_fail_ch2_sp	R/W1C	ECC correction fail of the Channel 2 spare region
17	ecc_err_fail_ch1_sp	R/W1C	ECC correction fail of the Channel 1 spare region
16	ecc_err_fail_ch0_sp	R/W1C	ECC correction fail of the Channel 0 spare region
15	ecc_err_hit_thres_ch7	R/W1C	ECC error bits hit the threshold of Channel 7
14	ecc_err_hit_thres_ch6	R/W1C	ECC error bits hit the threshold of Channel 6.
13	ecc_err_hit_thres_ch5	R/W1C	ECC error bits hit the threshold of Channel 5.
12	ecc_err_hit_thres_ch4	R/W1C	ECC error bits hit the threshold of Channel 4.
11	ecc_err_hit_thres_ch3	R/W1C	ECC error bits hit the threshold of Channel 3.
10	ecc_err_hit_thres_ch2	R/W1C	ECC error bits hit the threshold of Channel 2.
9	ecc_err_hit_thres_ch1	R/W1C	ECC error bits hit the threshold of Channel 1.
8	ecc_err_hit_thres_ch0	R/W1C	ECC error bits hit the threshold of Channel 0.
7	ecc_err_fail_ch7	R/W1C	ECC correction fail of Channel 7
6	ecc_err_fail_ch6	R/W1C	ECC correction fail of Channel 6
5	ecc_err_fail_ch5	R/W1C	ECC correction fail of Channel 5
4	ecc_err_fail_ch4	R/W1C	ECC correction fail of Channel 4
3	ecc_err_fail_ch3	R/W1C	ECC correction fail of Channel 3
2	ecc_err_fail_ch2	R/W1C	ECC correction fail of Channel 2
1	ecc_err_fail_ch1	R/W1C	ECC correction fail of Channel 1
0	ecc_err_fail_ch0	R/W1C	ECC correction fail of Channel 0

4.2.5 ECC Status Clear Register (Offset = 0x0028)

Table 4-11. ECC Status Clear Register (Offset = 0x0028)

Bit	Name	Type	Description
[31:16]	-	-	Reserved
[15:8]	ecc_err_cnt_sp_clr_chx	W	ECC error bits clear region of the channel x (x = 0 ~ 7) spare region 1: Clear the error bits recorded in the ECC status register of the spare region 0: No effect
[7:0]	ecc_err_cnt_clr_chx	W	ECC error bits clear register of Channel *x* (*x* = 0 ~ 7) 1: Clear the error bits recorded in the ECC status register 0: No effect

4.2.6 ECC Status of Spare Region Registers 0 and Register 1 (Offset = 0x002C and Offset = 0x0030)

The ECC status of the spare region register reflects the number of the ECC error bits happened in the spare region. Only the values of the maximum error bits are recorded. To clear this register, write '1' to the ECC status clear register bits[15:8] (Offset = 0x0028). When ECC for spare fails, the ECC status of the spare register will not be valid and the status of the spare region register may be incorrect.

Table 4-12. ECC Status Register 0 (Offset = 0x002C)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	ecc_err_no_ch3_sp	R	Number of the ECC error bits on Channel 3 spare region
23	-	-	Reserved
[22:16]	ecc_err_no_ch2_sp	R	Number of the ECC error bits on Channel 2 spare region
15	-	-	Reserved
[14:8]	ecc_err_no_ch1_sp	R	Number of the ECC error bits on Channel 1 spare region
7	-	-	Reserved
[6:0]	ecc_err_no_ch0_sp	R	Number of the ECC error bits on Channel 0 spare region

Table 4-13. ECC Status Register 1 (Offset = 0x0030)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	ecc_err_no_ch7_sp	R	Number of the ECC error bits on Channel 7 spare region
23	-	-	Reserved
[12:16]	ecc_err_no_ch6_sp	R	Number of the ECC error bits on Channel 6 spare region
15	-	-	Reserved
[14:8]	ecc_err_no_ch5_sp	R	Number of the ECC error bits on Channel 5 spare region
7	-	-	Reserved
[6:0]	ecc_err_no_ch4_sp	R	Number of the ECC error bits on Channel 4 spare region

4.2.7 Spare Region ECC Control Register (Offset = 0x0034 ~ 0x0040)

Table 4-14. Threshold Number of Spare Region ECC Error Bits Register 0 (Offset = 0x0034)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	ecc_thres_bits3_sp	R/W	Threshold number of the ECC error bits of Channel 3 spare region 7'd0 ~ 7'd73: 1 bit ~ 74 bits
23	-	-	Reserved
[22:16]	ecc_thres_bits2_sp	R/W	Threshold number of the ECC error bits of Channel 2 spare region 7'd0 ~ 7'd73: 1 bit ~ 74 bits
15	-	-	Reserved
[14:8]	ecc_thres_bits1_sp	R/W	Threshold number of the ECC error bits of Channel 1 spare region 7'd0 ~ 7'd73: 1 bit ~ 74 bits
7	-	-	Reserved
[6:0]	ecc_thres_bits0_sp	R/W	Threshold number of the ECC error bits of Channel 0 spare region 7'd0 ~ 7'd73: 1 bit ~ 74 bits

Table 4-15. Threshold Number of Spare Region ECC Error Bits Register 1 (Offset = 0x0038)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	ecc_thres_bits7_sp	R/W	Threshold number of the ECC error bits of Channel 7 spare region 7'd0 ~ 7'd73: 1 bit ~ 74 bits
23	-	-	Reserved
[22:16]	ecc_thres_bits6_sp	R/W	Threshold number of the ECC error bits of Channel 6 spare region 7'd0 ~ 7'd73: 1 bit ~ 74 bits
15	-	-	Reserved
[14:8]	ecc_thres_bits5_sp	R/W	Threshold number of the ECC error bits of Channel 5 spare region 7'd0 ~ 7'd73: 1 bit ~ 74 bits
7	-	-	Reserved
[6:0]	ecc_thres_bits4_sp	R/W	Threshold number of the ECC error bits of Channel 4 spare region 7'd0 ~ 7'd73: 1 bit ~ 74 bits

The permitted numbers of the spare region ECC correction capability bits are listed in Table 4-16 and Table 4-17.

Table 4-16. Number of Spare Region ECC Correction Capability Bits Register 0 (Offset = 0x003C)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	ecc_corr_bits3_sp	R/W	This field sets the number of the ECC correction capability bits of Channel 3 spare region. 7'd0 ~ 7'd73: 1 bit ~ 74 bits
23	-	-	Reserved
[22:16]	ecc_corr_bits2_sp	R/W	This field sets the number of the ECC correction capability bits of Channel 2 spare region. 7'd0 ~ 7'd73: 1 bit ~ 74 bits
15	-	-	Reserved
[14:8]	ecc_corr_bits1_sp	R/W	This field sets the number of the ECC correction capability bits of Channel 1 spare region. 7'd0 ~ 7'd73: 1 bit ~ 74 bits
7	-	-	Reserved
[6:0]	ecc_corr_bits0_sp	R/W	This field sets the number of the ECC correction capability bits of Channel 0 spare region. 7'd0 ~ 7'd73: 1 bit ~ 74 bits

Table 4-17. Number of ECC Correction Capability Bits Register 1 (Offset = 0x0040)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	ecc_corr_bits7_sp	R/W	This field sets the number of the ECC correction capability bits of Channel 7 spare region. 7'd0 ~ 7'd73: 1 bit ~ 74 bits
23	-	-	Reserved
[22:16]	ecc_corr_bits6_sp	R/W	This field sets the number of the ECC correction capability bits of Channel 6 spare region. 7'd0 ~ 7'd73: 1 bit ~ 74 bits
15	-	-	Reserved
[14:8]	ecc_corr_bits5_sp	R/W	This field sets the number of the ECC correction capability bits of Channel 5 spare region. 7'd0 ~ 7'd73: 1 bit ~ 74 bits
7	-	-	Reserved
[6:0]	ecc_corr_bits4_sp	R/W	This field sets the number of the ECC correction capability bits of Channel 4 spare region. 7'd0 ~ 7'd73: 1 bit ~ 74 bits

4.2.8 Device Busy/Ready Status Register (Offset = 0x0100)

The busy/ready pins of a Flash device can be monitored by reading this register.

Table 4-18. Device Busy/Ready Status Register (Offset = 0x0100)

Bit	Name	Type	Description
[31:8]	-	-	Reserved
[7:0]	dev_busy_rdyx	R	The busy/ready (busy_n) status of Channel *x* (*x* = 0 ~ 7) of the Flash device 1: Flash is ready. 0: Flash is busy.

4.2.9 NANDC General Setting Register (Offset = 0x0104)

This register provides the general settings for the NANDC module.

Table 4-19. NANDC General Setting Register (Offset = 0x0104)

Bit	Name	Type	Description
[31:26]	-	-	Reserved
[25:24]	ce_num	R/W	Number of CEs connected with Flash in a channel 00: 1 CE 01: 2 CEs 10: 4 CEs 11: 8 CEs Notes: • The number of Flashes used in a channel must be power of 2. • If two CEs are at the CE0 and CE2 positions, ce_num must be set to four CEs.
[23:15]	-	-	Reserved
[14:12]	busy_rdy_loc	R/W	Device busy/ready status bit location on the Flash data bus 3'b000 ~ 3'b111: Bit 0 ~ Bit 7
11	-	-	Reserved
[10:8]	cmd_sts_loc	R/W	Command pass/fail status bit location on the Flash data bus 3'b000 ~ 3'b111: Bit 0 ~ Bit 7
[7:3]	-	-	Reserved
2	wr_prot_en	R/W	Flash write protect pin control 1: Enable the write protect 0: Disable the write protect
1	data_inv_en	R/W	Data inverse control The inverted targets include the ECC parity and data. 1: Enable 0: Disable The inverse function is useful when reading data from an erased page in Flash. If the inverse function is not enabled, all the read data bits will be '1' and the ECC parity bits will also be '1', which will induce an ECC parity error. If the inverse function is enabled, all data will be '0' and the ECC parity bits will be '0'. All '0' data and parity can pass the ECC parity check without errors.

Bit	Name	Type	Description
0	scrambler_en	R/W	Data scrambler The scrambling operation only affects data. 1: Enable 0: Disable The scrambler adopts a formula to inverse data in random pseudo. The scramble seed depends on the sector number and row address executing to Flash.

4.2.10 Memory Attribute Setting Register 1 (Offset = 0x0108)

Table 4-20. Memory Attribute Setting Register 1 (Offset = 0x0108)

Bit	Name	Type	Description
[31:19]	-	-	Reserved
[18:16]	page_size	R/W	Page size of Flash 3'b000: 512 byte (Small page) 3'b001: 2K bytes 3'b010: 4K bytes 3'b011: 8K bytes 3'b100: 16K byte Other : Reserved Note: The small page fixed flow is only used for the small page setting.
[15]	-	-	Reserved
[14:13]	row_cyc	R/W	Flash row address cycles 2'b00: One cycle 2'b01: Two cycles 2'b10: Three cycles 2'b11: Reserved
12	col_cyc	R/W	Flash column address cycles 1'b0: One cycle 1'b1: Two cycles
[11:2]	block_size	R/W	Block size (Row address occupied by one block) 10'h0 ~ 10'h3FF: 1 page ~ 1024 pages. Note: block_size must be 2^ order and the block size must be more than or equal to valid_page_num (0x10C). The block size can only be 32, 64, 128, 256, 512, or 1024 pages.
[1:0]	-	-	Reserved

4.2.11 Memory Attribute Setting Register 2 (Offset = 0x010C)

Table 4-21. Memory Attribute Setting Register 2 (Offset = 0x010C)

Bit	Name	Type	Description
[31:26]	-	-	Reserved
[25:16]	valid_page_num	R/W	Number of the valid pages in a block 10'h0 ~ 10'h3FF: 1 page ~ 1024 pages For some TLC Flash, the number of the valid pages in a block will be 192 and the address occupied by one block will be 256 pages. The last 64 pages are blank and cannot be accessed. This register must be less than or equal to block_size (0x108). If valid_page_num is less than the block size (0x108), the start row address should not be more than valid_page_num. At SLC mode and TLC mode, this register may be different, user must care this. For example, 64 valid page at Samsung SLC mode but 192 valid page at Samsung TLC mode. If user execute SLC command and TLC command, must set this register before executing command. If user issue 3SLC copy to TLC mode command, this register is based on SLC mode.
15	dqs_clk_out_en	R/W	DLL reference clock enable bit 1: Enable (Default) 0: Disable
14	Vref	R/W	External voltage enable 1: Enable 0: Disable If this bit is set to '1', the external VREFQ will be used as a reference for the input and I/O signals. If this bit is set to '0', the internal VREFQ will be used as a reference for the input and I/O signals.
13	DQS_c	R/W	DQS complementary signal(DQS_c) enable 1: Enable 0: Disable If this bit is set to '1', the complementary DQS (DQS_c) signal will be enabled. If this bit is cleared to '0', the complementary DQS (DQS_c) signal will not be used.
12	RE_c	R/W	RE complementary signal (RE_c) enable 1: Enable 0: Disable If this bit is set to '1', the complementary RE_n (RE_c) signal will be enabled. If this bit is cleared to '0', the complementary RE_n (RE_c) signal will not be used.

Bit	Name	Type	Description
[11:8]	ODT	R/W	DQ/DQS/RE_n ODT Enable This field controls the on-die termination settings for the DQ[7:0], DQS_t, DQS_c, RE_t, and RE_c signals. The values are: 0h = ODT is disabled. 1h = ODT is enabled with Rtt of 150 Ω. 2h = ODT is enabled with Rtt of 100 Ω. 3h = ODT enabled with Rtt of 75 Ω. 4h = ODT enabled with Rtt of 50 Ω. 5h = ODT enabled with Rtt of 30 Ω (Optional). 6h-Fh Reserved Note: The settings of Rtt may be specified separately for DQ[7:0]/DQS and the RE_n signals. The DQ[7:0]/DQS may be specified separately for data input versus data output operation. Please refer to the definition of the ODT configure command in Section 5.25. If the values are specified with the ODT Configure command, this field will not be used. Features should return the previous value set in this field, regardless of the settings of Rtt specified using the ODT configuration.
[7:4]	warm_up_rd	R/W	Warm-up RE_n and DQS cycles This field indicates the number of warm-up cycles of RE_n and DQS provided for data output. These are the number of initial —dummy ‖ RE_t/RE_c cycles at the start of data output operations. There are corresponding —dummy ‖ DQS_t/DQS_c cycles to the —dummy ‖ RE_t/RE_c cycles that the host shall ignore. The values are: 0h = 0 cycle (Feature is disabled.) 1h = 1 warm-up cycle 2h = 2 warm-up cycles 3h = 4 warm-up cycles 4h ~ FFh = Reserved
[3:0]	warm_up_wr_	R/W	Warm-up DQS cycles This field indicates the number of warm-up cycles of DQS that are provided for data input. These are the number of initial —dummy ‖ DQS_t/DQS_c cycles at the start of data input operations. The values are: 0h = 0 cycle (Feature is disabled.) 1h = 1 warm-up cycle 2h = 2 warm-up cycles 3h = 4 warm-up cycles 4h ~ FFh = Reserved

The following figure depicts warm-up 2 cycles for the RE_n signal and DQS signal.

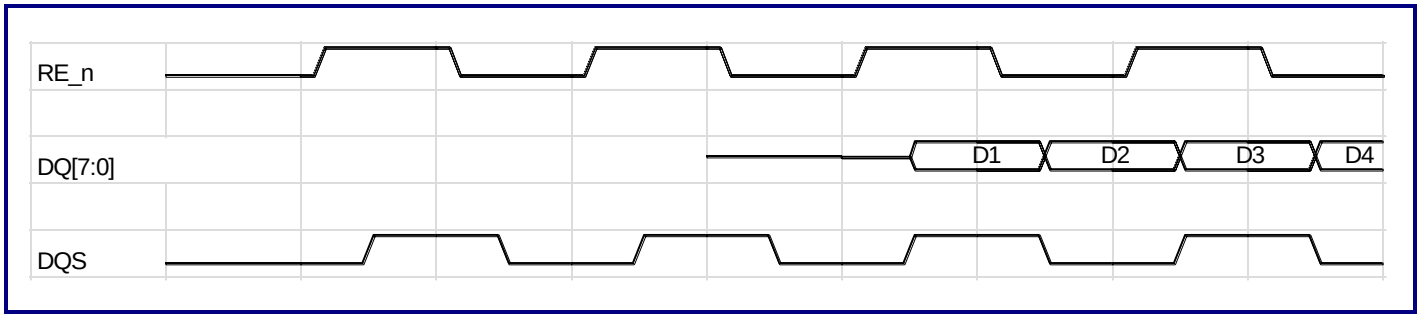


Figure 4-1. Warm-up 2 Cycles for RE_n Signal and DQS Signal

4.2.12 Flash AC Timing Register 0 of Channel *x* (Offset = 0x0110 + *x**0x8, *x* = 0 ~ 7)

The AC timing registers 0 and 1 are used to decide the clock cycles that will be issued by a specific signal pulse. The reference clock is mem_clk. Nand_clk/hs_clk is mem_clk as shown in Figure 4-2 through Figure 4-4.

Table 4-22. Flash AC Timing Register 0 (Offset = 0x0110 + *x* * 0x8, *x* = 0 ~ 7)

Bit	Name	Type	Description
[31:28]	-	-	Reserved
[27:24]	t_{WH}	R/W	Reference summary of AC timing usage
[23:21]	-	-	Reserved
[20:16]	t_{WP}	R/W	Reference summary of AC timing usage
[15:12]	-	-	Reserved
[11:8]	t_{REH}	R/W	Reference summary of AC timing usage
[7:5]	-	-	Reserved
[4:0]	t_{RES}	R/W	Reference summary of AC timing usage

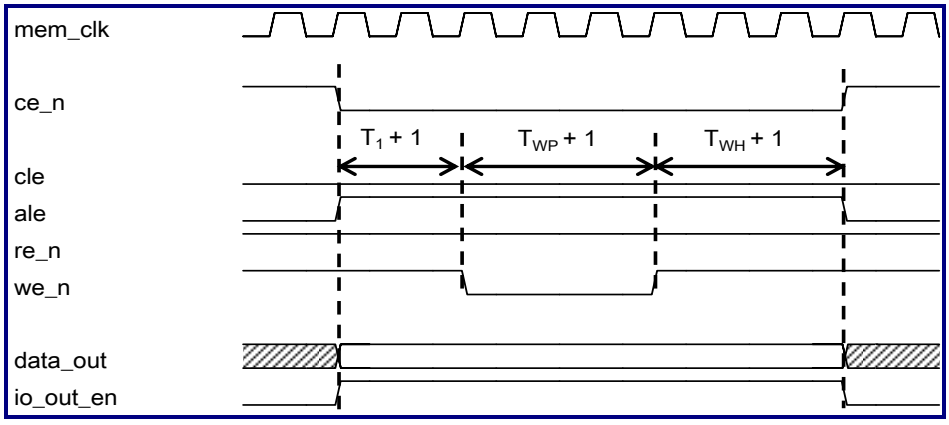


Figure 4-2. Timing Diagram of Address State

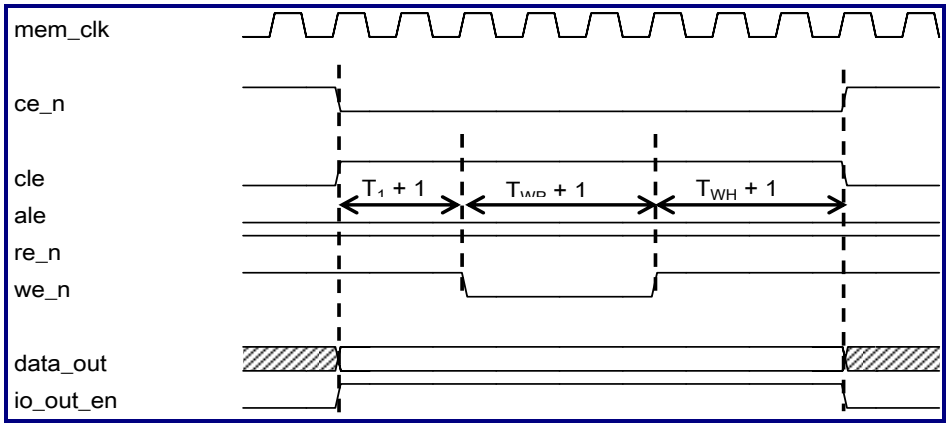


Figure 4-3. Timing Diagram of Command State

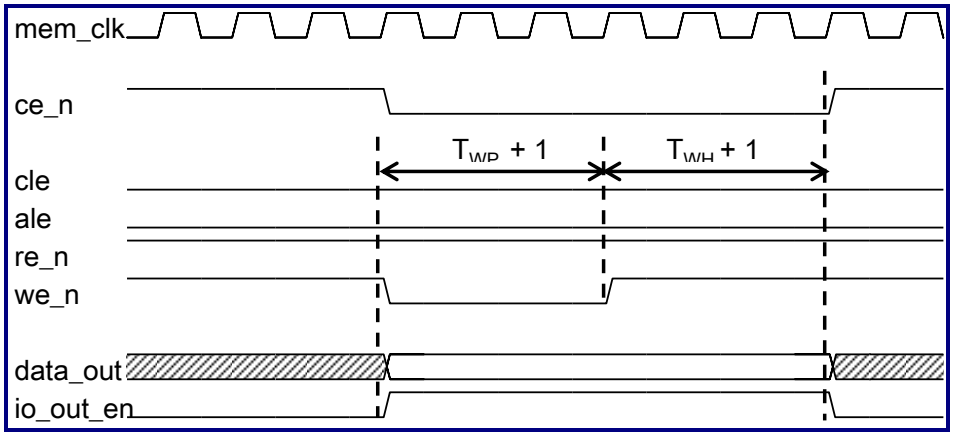


Figure 4-4. Timing Diagram of Write Data State

4.2.13 Flash AC Timing Register 1 of Channel *x* (Offset = 0x0114 + *x* * 0x8, *x* = 0 ~ 7)

Table 4-23. Flash AC Timing Register 1 of Channel *x* (Offset = 0x0114 + *x* *0x8, *x* = 0 ~ 7)

Bit	Name	Type	Description
[31:22]	-	-	Reserved
[21:16]	t_{RLAT}	R/W	Reference summary of AC timing usage
15	-	-	Reserved
[14:8]	t_{BSY}	R/W	Reference summary of AC timing usage
7	-	-	Reserved
[6:0]	t1	R/W	Reference summary of AC timing usage tCWAW for Samsung toggle

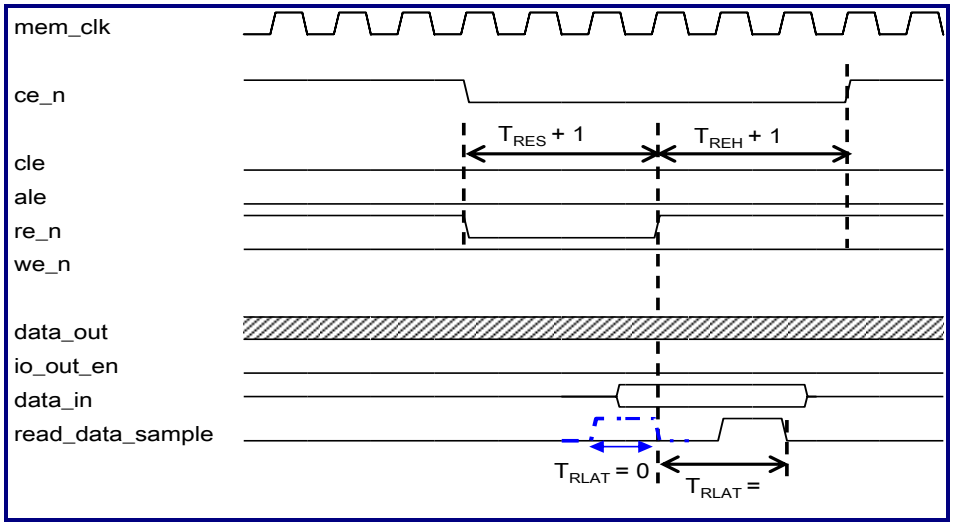


Figure 4-5. Timing Diagram of Read Data State

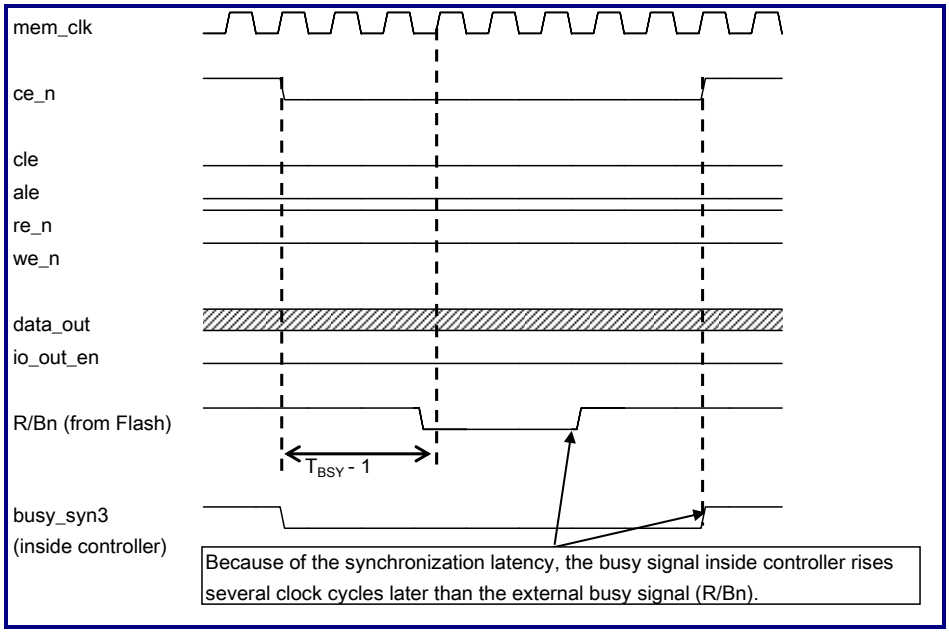


Figure 4-6. Timing Diagram of Busy State

4.2.14 Flash AC Timing Register 2 of Channel *x* (Offset = 0x0190 + *x* * 0x8, *x* = 0 ~ 7)

Table 4-24. Flash AC Timing Register 2 of Channel *x* (Offset = 0x0190 + *x* * 0x8, *x* = 0 ~ 7)

Bit	Name	Type	Description
31	-	-	Reserved
[30:24]	t_{BUF4}	R/W	Reference summary of AC timing usage
23	-	-	Reserved
[22:16]	t_{BUF3}	R/W	Reference summary of AC timing usage
15	-	-	Reserved
[14:8]	t_{BUF2}	R/W	Reference summary of AC timing usage
7	-	-	Reserved
[6:0]	t_{BUF1}	R/W	Reference summary of AC timing usage

Notes:

- t_{BUF4}, t_{BUF3}, t_{BUF2}, t_{BUF1} at the fixed flow AC timing are described as below:
- t_{BUF4} is WE high to RE low timing.
- t_{BUF3} is Max. (RE high to WE low, RE high to output Hi-Z, W/R# high to DQS/DQ tri-state by device)
- t_{BUF2} is busy high to RE low timing.
- t_{BUF1} is command phase to WE high phase.

t_{BUF4}	Max. (tWHR, tWHR2)
t_{BUF3}	Max. (tRHW, tRHZ, tDQSHZ)
t_{BUF2}	Max. (tAR, tRR, tCLR, tCDQSS, tCRES, tCALS, tCALS2, tDBS)
t_{BUF1}	Max. (tADL, tCCS, tCWAW)

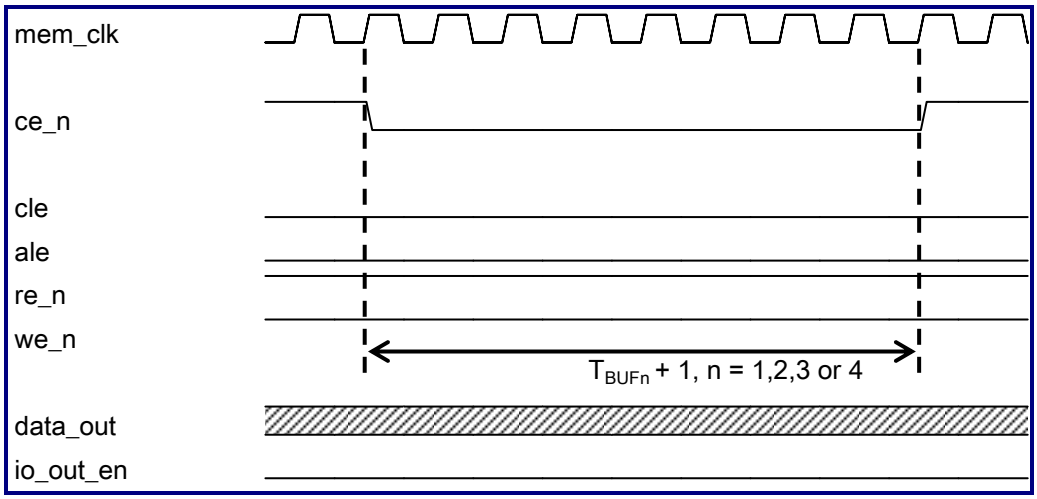


Figure 4-7. Timing Diagram of Buffer1/Buffer2/Buffer3/Buffer4 State

4.2.15 Flash AC Timing Register 3 of Channel *x* (Offset = 0x0194 + *x* * 0x8, *x* = 0 ~ 7)

Table 4-25. Flash AC Timing Register 3 of Channel *x* (Offset = 0x0194 + *x* * 0x8, *x* = 0 ~ 7)

Bit	Name	Type	Description
[31:28]	t_{PRE}	R/W	Minimum value is 1 and the detailed reference summary of AC timing usage
[27:24]	t_{PST}	R/W	Reference summary of AC timing usage
[23:21]	-	-	Reserved
[20:16]	t_{PSTH}	R/W	Reference summary of AC timing usage
[15:5]	-	-	Reserved
[4:0]	t_{WRCK}	R/W	Reference summary of AC timing usage

a. Timing Diagram of Micron ONFi

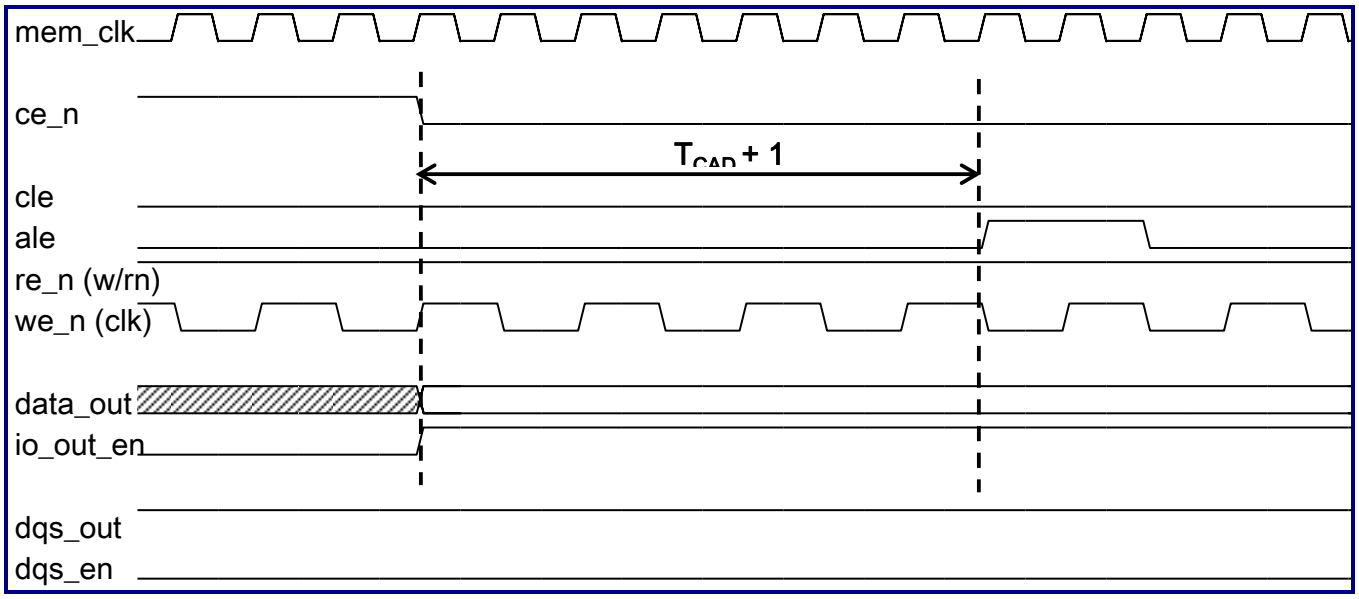


Figure 4-8. Timing Diagram of Address State

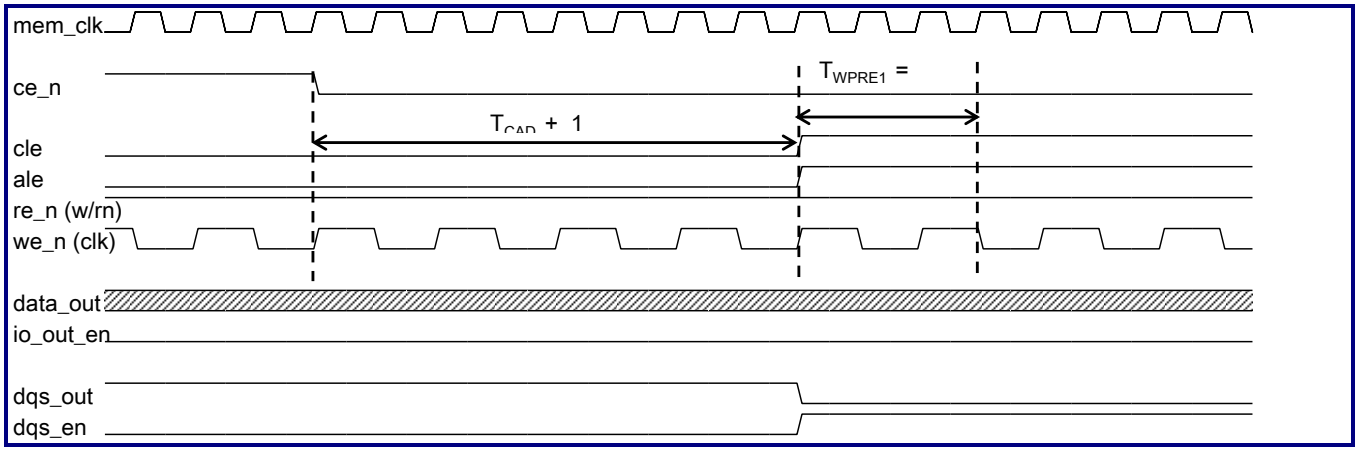


Figure 4-9. Timing Diagram of Preamble Write State

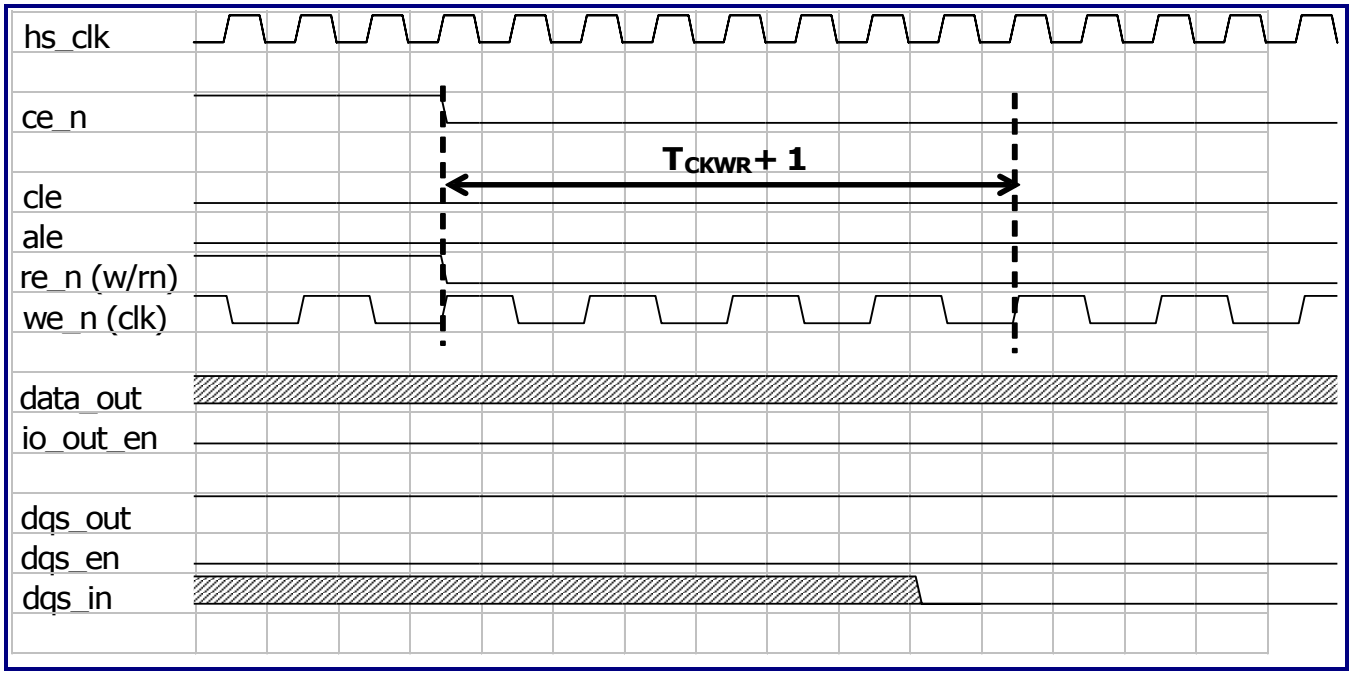


Figure 4-10. Timing Diagram of Preamble Read State

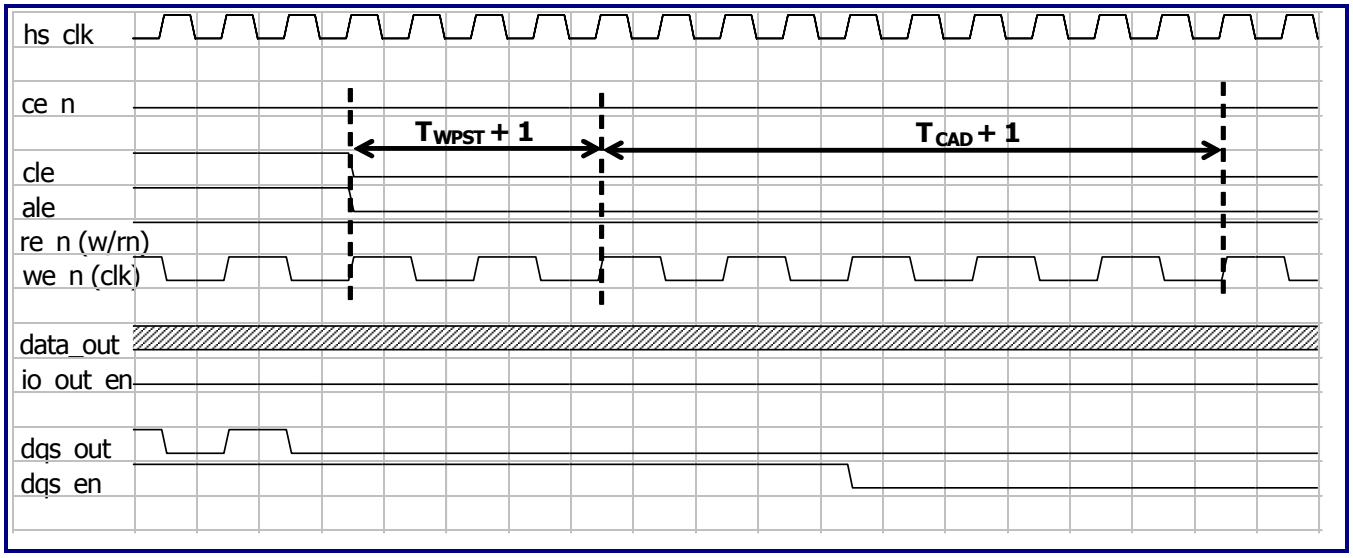


Figure 4-11. Timing Diagram of Post-amble Write State

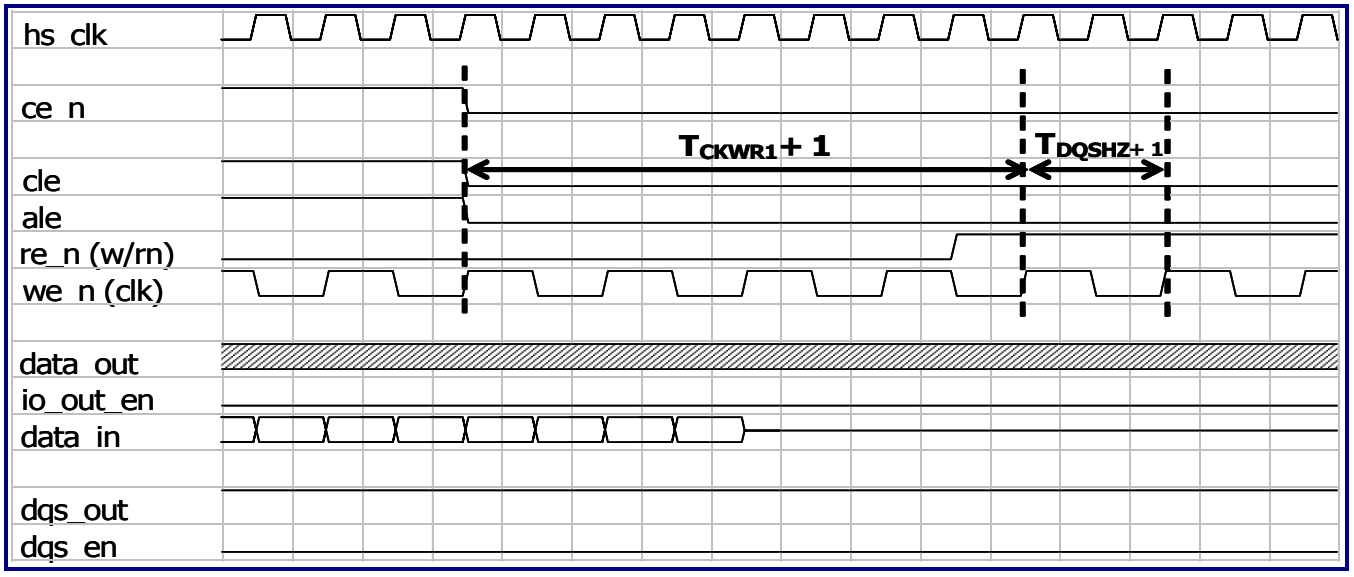


Figure 4-12. Timing Diagram of Post-amble Read State

b. Timing Diagram of Samsung Toggle

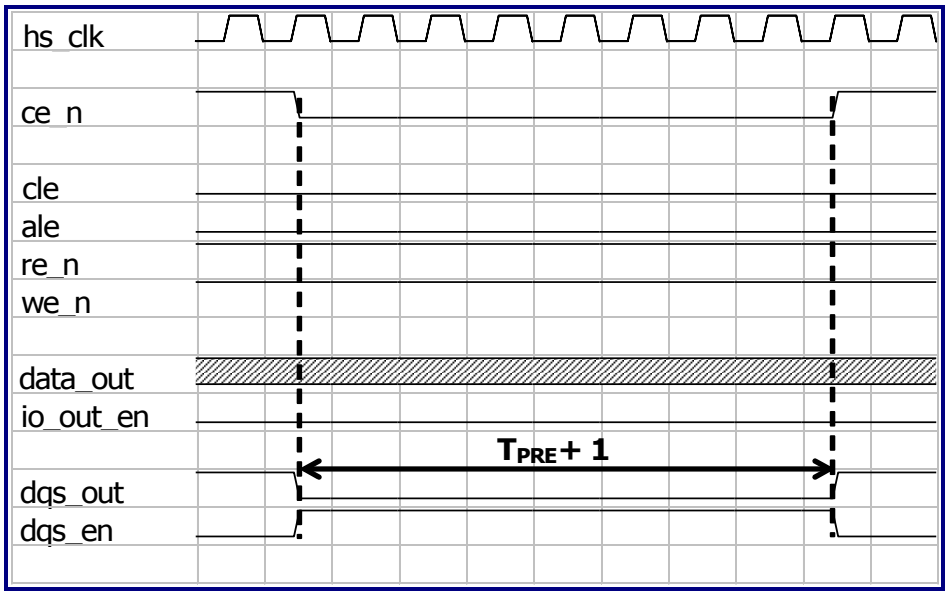


Figure 4-13. Timing Diagram of Preamble Write State

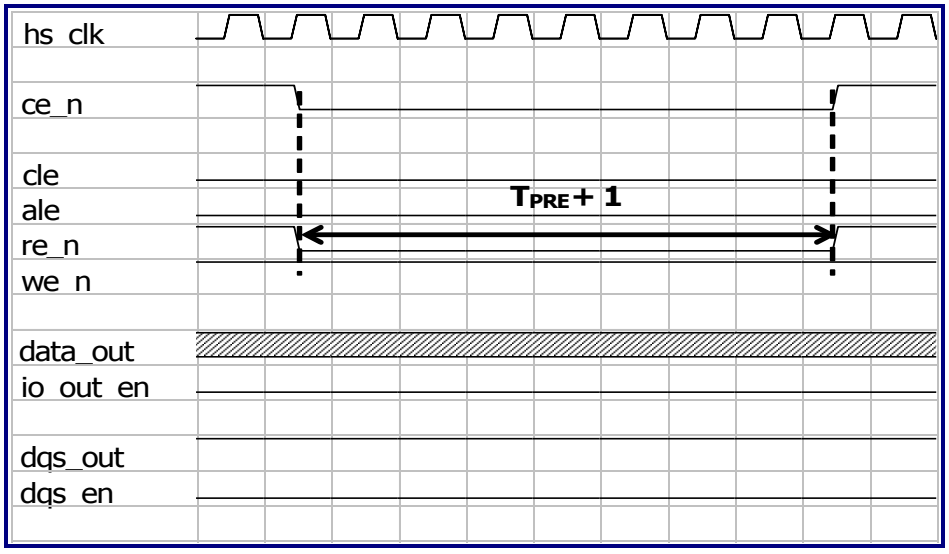


Figure 4-14. Timing Diagram of Preamble Read State

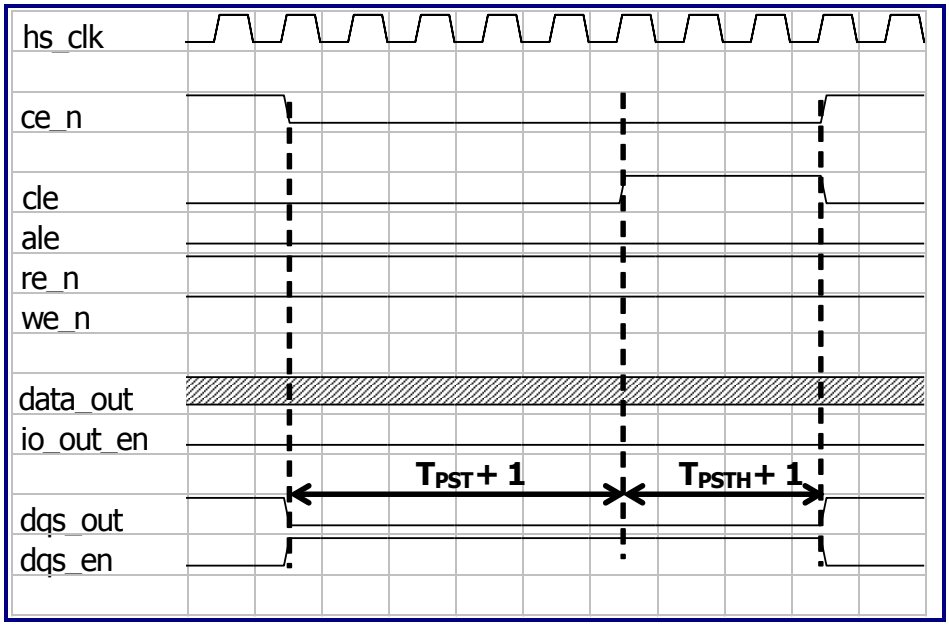


Figure 4-15. Timing Diagram of Post-amble Write State

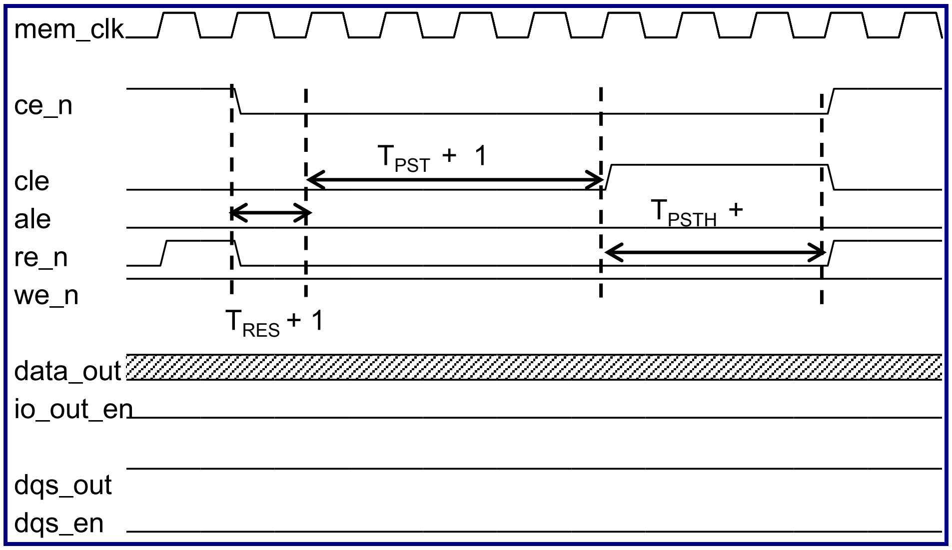


Figure 4-16. Timing Diagram of Post-amble Read State

Summary of AC Timing Usage

Item	Asynchronous IF	Toggle 1.0 IF	NV-DDR	Toggle 2.0	NV-DDR2
t_{WH}	$\max(t_{WH},t_{CH},t_{CLH},t_{ALH})$	$\max(t_{WH},t_{CH},t_{CALH})$	—	$\max(t_{CH},t_{CALH})$	$\max(t_{CH},t_{CALH})$
t_{WP}	t_{WP}	t_{WP}	t_{CAD}	t_{WP}	t_{WP}
t_{REH}	t_{REH}	t_{CR}	—	—	—
t_{RES}	$\max(t_{REA},t_{RP},t_{RSTO},t_{REAID})$ tRP when EDO mode	$\max(t_{RP},t_{REH})$	N (F_{tCK} = mem_clk/2(N+1)) [3]	$\max(t_{RP},t_{REH})$	$\max(t_{RP},t_{REH})$
t_{BSY}	t_{WB},t_{RB} The min. value is 1	t_{WB} The min. value is 1	t_{WB} The min. value is 1	t_{WB} The min. value is 1	t_{WB} The min. value is 1
t_{BUF1} [1] t_{BUF2} [2] t_{BUF3} t_{BUF4}	$\max(t_{WHR},t_{WHR2},t_{RHW},$ $t_{RR},t_{AR},t_{DBS},t_{CALS},t_{CALS2},$ $t_{ADL},t_{RHZ},t_{CDQSS},t_{CCS},t_{CRES})$ The min. value is 1	$\max(t_{WHR},t_{WHR2},t_{RHW},$ $t_{RR},t_{AR},t_{DBS},t_{CALS},t_{CALS2},$ $t_{ADL},t_{RHZ},t_{CDQSS},t_{CCS},t_{CRES})$ The min. value is 1	$\max(t_{WHR},t_{WHR2},t_{RHW},$ $t_{RR},t_{AR},t_{DBS},t_{CALS},t_{CALS2},$ $t_{ADL},t_{RHZ},t_{CDQSS},t_{CCS},t_{CRES})$ The min. value is 1	$\max(t_{WHR},t_{WHR2},t_{RHW},$ $t_{RR},t_{AR},t_{DBS},t_{CALS},t_{CALS2},$ $t_{ADL},t_{RHZ},t_{CDQSS},t_{CCS},t_{CRES})$ The min. value is 1	$\max(t_{WHR},t_{WHR2},t_{RHW},$ $t_{RR},t_{AR},t_{DBS},t_{CALS},t_{CALS2},$ $t_{ADL},t_{RHZ},t_{CDQSS},t_{CCS},t_{CRES})$ The min. value is 1
t_{RLAT}	internal latch timing	—	—	—	—
t_1	$\max(t_{CS},t_{CLS},t_{ALS})$ $- t_{WP}$	$\max(t_{CALS2} -$ $t_{WP},t_{CWAW})$	t_{CS} The min. value is 1	$\max(t_{CALS2} -$ $t_{WP},t_{CWAW})$	$\max(t_{CALS2} -$ $t_{WP},t_{CWAW})$
t_{PRE}	—	$\max(t_{WPRE},2t_{RPRE},$ $t_{RPRE}+t_{DQSRE}) + 1$	t_{CKWR} [4]	$\max(t_{WPRE},t_{WPRE2},$ $t_{RPRE},t_{RPRE2},t_{CS},t_{CS2}) + 1$	$\max(t_{WPRE},t_{WPRE2},$ $t_{RPRE},t_{RPRE2},t_{CS1},t_{CS2}) + 1$
t_{PST}	—	$\max(t_{WPST},t_{RPST}) + 1$	t_{WPST} [4]	$\max(t_{WPST},t_{RPST}) + 1$	$\max(t_{WPST},t_{RPST},t_{CH},t_{CALH}) +$
t_{PSTH}	EBI setup time $\max(t_{CS},t_{CLS},t_{ALS})$	$\max(t_{WPSTH},t_{RPSTH})$	Roundup{t_{DQSHZ}/tck} [4]	$\max(t_{WPSTH},t_{RPSTH})$	$\max(t_{WPSTH},t_{RPSTH},t_{CEH})$
t_{WRCK}	EBI hold time $\max(t_{RHZ},t_{REH})$	$\max(t_{DSL},t_{DSH},$ $t_{DQSL},t_{DQSH})$	t_{WRCK}	$\max(t_{DSL},t_{DSH},$ $t_{DQSL},t_{DQSH})$	$\max(t_{DSL},t_{DSH},$ $t_{DQSL},t_{DQSH})$

Figure 4-17. Summary of AC Timing Usage

1. When the error handling/abort occurs, tBUF1 will be used to idle I/Fs to avoid the signal conflict.
2. For the Toggle 2 operation, tBUF2 will be used for the WPRE/RPRE timing.
3. The N factor decides the operating frequency of FtCK and tCK = 1/FtCK
4. The unit is tCK; tCK is decided by the tRES field

Notes:
1. The ONFi 2.0 protocol includes the asynchronous IF and synchronous IF.
2. The "legacy protocol" means "asynchronous IF"

4.2.16 NANDC Interrupt Enable Register (Offset = 0x0150)

The interrupt of FTNANDC024 will be asserted when the interrupt enable register is set and corresponding event happens. The event status can be checked by reading the NANDC Interrupt Status Register (Offset = 0x0154)

Table 4-26. NANDC Interrupt Enable Register (Offset = 0x0150)

Bit	Name	Type	Description
[31:8]	-	-	Reserved
[7:0]	sts_fail_int_en	R/W	Flash status check fail interrupt enable 1: Enable 0: Disable (Bit 7 ~ Bit 0: Channel 7 ~ Channel 0)

4.2.17 NANDC Interrupt Status Register (Offset = 0x0154)

The FTNANDC024 interrupt can be asserted by using one of the following three NANDC events:

- Auto-pattern compare failed
- NANDC command completed
- Status check failed

Table 4-27. NANDC Interrupt Status Register (Offset = 0x0154)

Bit	Name	Type	Description
[31:24]	auto_cmp_pat_fail Note1	R/W1C	Auto pattern compare fail (Bit 31 ~ Bit 24: Channel 7 ~ Channel 0)

Bit	Name	Type	Description
[23:16]	nandc_cmd_cmplt Note2	R/W1C	NANDC command complete (Bit 23 ~ Bit 16: Channel 7 ~ Channel 0)
[15:8]	-	-	Reserved
[7:0]	sts_fail	R/W1C	Status check fail (Bit 7 ~ Bit 0: Channel 7 ~ Channel 0)

Notes:

1. The failure of the auto pattern compare will always trigger the global interrupt.
2. The NANDC command complete interrupt enable is set in the command queue.
3. When auto_cmp_pat_fail occurs, nandc_cmd_cmplt will not be asserted.

4.2.18 Current Access Row Address Register (Offset = 0x0158 ~ 0x0174)

The current access row address represents the row address that is issued by NANDC. These registers will be updated once NANDC issues a new row address.

Table 4-28. Current Access Row Address Register (Offset = 0x0158 ~ 0x0174)

Bit	Name	Type	Description
[31:0]	row_addr_ch*x**	R	Current access of the row address of channel *x* Note

Note: *x* = 0 ~ 7 (Corresponding to offsets 0x0158 ~ 0x0174)

4.2.19 Read Status Register 0 and Register 1 (Offset = 0x0178 ~ 0x017C)

NANDC stores the status returned from Flash in the register.

Table 4-29. Read Status Register 0 (Offset = 0x0178)

Bit	Name	Type	Description
[31:24]	read_sts_ch3	R	Flash status of Channel 3
[23:16]	read_sts_ch2	R	Flash status of Channel 2
[15:8]	read_sts_ch1	R	Flash status of Channel 1
[7:0]	read_sts_ch0	R	Flash status of Channel 0

Table 4-30. Read Status Register 1 (Offset = 0x017C)

Bit	Name	Type	Description
[31:24]	read_sts_ch7	R	Flash status of Channel 7
[23:16]	read_sts_ch6	R	Flash status of Channel 6
[15:8]	read_sts_ch5	R	Flash status of Channel 5
[7:0]	read_sts_ch4	R	Flash status of Channel 4

4.2.20 Address Toggle Bit Location Register (Offset = 0x0180)

The row address may need to be toggled for one-bit in order to perform the Flash two-plane access. This register can be set to determine which bit is toggled during the MicroCode of tog_1st_addr, tog_2nd_addr, tog_3rd_addr, and tog_targ_1st_addr.

Table 4-31. Address Toggle Bit Location Register (Offset = 0x0180)

Bit	Name	Type	Description
[31:5]	-	-	Reserved
[4:0]	toggle_bit_loc	R/W	Toggle bit location 0 ~ 23: Bit 0 ~ Bit 23 of the row addresses Others: Reserved

4.2.21 NANDC Software Reset Register (Offset = 0x0184)

Each NANDC and ECC can be reset by setting the NANDC software reset register. NANDC will return to the idle state when performing reset. The corresponding command queue will not pop and will be re-executed immediately after reset. If users issue this register to reset NANDC, the operation of flash will be suddenly stopped by the controller and may result unexpected behavior to flash. Therefore, to abort command, please follow the abort sequence (Please refer to Chapter 5) to complete the abort flow. No register value will be reset during issuing the software reset to NANDC.

Table 4-32. NANDC Software Reset Register (Offset = 0x0184)

Bit	Name	Type	Description
[31:8]	-	-	Reserved

Bit	Name	Type	Description
[7:0]	nandc_sw_rst	R/W	NANDC software reset register This reset will be cleared when the corresponding NANDC finishes the reset operation. This field must be checked to be '0' before issuing the new NAND commands. Write 1: Reset Write 0: No effect (Bit 7 ~ Bit 0: Channel 7 ~ Channel 0)

4.2.22 NANDC Auto-compare Pattern Register (Offset = 0x018C)

NANDC can perform a write command with specific patterns and read commands to be compared with the specific pattern recorded in this register. When FTNANDC024_CH_DATA_8 is defined by hardware configuration, only bits, auto_cmp_pat[7:0], will be used. The data region threshold for the blanking check located offset (0x0010 ~ 0x0014) for different channels. The spare region threshold for the blanking check located offset (0x0034 ~ 0x0038) for different channels.

The data after scrambler is the blanking write data when scrambler enable (Offset 0x104), and the scrambled seed is the row address[13:0]. When the seed is zero, the seed will be 14'b2AAA;

Table 4-33. NANDC Auto-compare Pattern Register (Offset = 0x018C)

Bit	Name	Type	Description
[31:0]	auto_cmp_pat	R/W	Auto-compare pattern

4.2.23 Variable Address Register (Offset = 0x01D0)

This register is used for microcode var_addr1, var_addr2, var_addr3 and var_addr4, respectively.

Table 4-34. Variable Address Register (Offset = 0x01D0)

Bit	Name	Type	Description
[31:24]	var_B4	R/W	Variable 4th byte
[23:16]	var_B3	R/W	Variable 3rd byte
[15:8]	var_B2	R/W	Variable 2nd byte
[7:0]	var_B1	R/W	Variable 1st byte

4.2.24 Command Queue Status Register (Offset = 0x0200)

Each NANDC has its own command queue. The command queue status includes:

- Command queue is full.
- Command queue is empty.

Before pushing a command into queue, please check the queue to make sure that it is not full.

Table 4-35. Command Queue Status Register (Offset = 0x0200)

Bit	Name	Type	Description
[31:16]	-	-	Reserved
[15:8]	cmdq_full	R	Command queue full status 1: Full 0: Not full (Bit 15 ~ Bit 8: Channel 7 ~ Channel 0)
[7:0]	cmdq_empty	R	Command queue empty status 1: Empty 0: Not empty (Bit 7 ~ Bit 0: Channel 7 ~ Channel 0)

4.2.25 Command Queue Flush Register (Offset = 0x0204)

The command queue can be flushed by writing '1' to the specific flush register. The corresponded NANDC and ECC are also reset during setting the command queue flush.

Table 4-36. Command Queue Flush Register (Offset = 0x0204)

Bit	Name	Type	Description
[31:8]	-	-	Reserved
[7:0]	cmdq_flush	W	Command queue flush 1: Write 1 to flush 0: Write 0 has no effect. (Bit 7 ~ Bit 0: Channel 7 ~ Channel 0)

4.2.26 Command Complete Counter Register (Offset = 0x0208)

When one command completes, the counter will increase by 1. By checking this register, the number of the command completion can be known. Please note that cmd_cmplt_cnt contains a maximum of seven cmd_cmplt. If cmd_cmplt is more than seven cmd_cmplt and cmd_cmplt is not cleared, cmd_cmplt_cnt will overflow.

Table 4-37. Command Complete Counter Register (Offset = 0x0208)

Bit	Name	Type	Description
31	-	-	Reserved
[30:28]	cmd_cmplt_cnt_ch7	R	Command complete counter of Channel 7
27	-	-	Reserved
[26:24]	cmd_cmplt_cnt_ch6	R	Command complete counter of Channel 6
23	-	-	Reserved
[22:20]	cmd_cmplt_cnt_ch5	R	Command complete counter of Channel 5
19	-	-	Reserved
[18:16]	cmd_cmplt_cnt_ch4	R	Command complete counter of Channel 4
15	-	-	Reserved
[14:12]	cmd_cmplt_cnt_ch3	R	Command complete counter of Channel 3
11	-	-	Reserved
[10:8]	cmd_cmplt_cnt_ch2	R	Command complete counter of Channel 2
7	-	-	Reserved
[6:4]	cmd_cmplt_cnt_ch1	R	Command complete counter of Channel 1
3	-	-	Reserved
[2:0]	cmd_cmplt_cnt_ch0	R	Command complete counter of Channel 0

4.2.27 Command Complete Counter Reset Register (Offset = 0x020C)

The command completion counter can be reset by writing '1' to this register.

Table 4-38. Command Complete Counter Reset Register (Offset = 0x020C)

Bit	Name	Type	Description
[31:8]	-	-	Reserved
[7:0]	cmd_cmplt_cnt_rst	W	Command complete counter reset Write 1: Clear the counter of corresponded channel Write 0: No effect

4.2.28 General Command Queue Register (Offset = 0x0280 ~ 0x0294)

Writing to this command register will push the command to all the command queues. Reading from this command register will get all zero returns. Please note that this register cannot be set if one or more command queues are full. The general command queue can only be pushed with a command that contains no data transfer from or to BMC (For example: When block erase, reset, and spare read/write are acceptable).

4.2.29 Command Queue Register (Offset = 0x0300 + *n* * 20 ~ 0x030C + *n* * 0x20, *n* = 0 ~ 7)

The operation of NANDC is decided by the command pushed into the command queue. One command contains four words for basic flow and two words for eD3 flow. Fifth and sixth command queue word exist at FTNANDC024_ED3_ON. Each command queue may have one, two or four entries of space for keeping the command. A command will be pushed into the command queue by writing the forth word of a command and will pop from the command queue when it is completely executed by a NAND channel controller.

During reading the command queue register, the returned data will be the commands that are currently processed By NANDC) but will not be the current command to be written to the command queue.

Table 4-39. Command Queue First Word Register (Offset = 0x0300 + *n* * 0x20, *n* = 0 ~ 7)

Bit	Name	Type	Description
[31:24]	-	-	Reserved

Bit	Name	Type	Description
[23:0]	row_addr_1st	R/W	The first row address/source row address (Page index) The first row address decides the source address in a copy-back or a blanking check flow. When eD3 is configurable and issue eD3 copy back; this register can be 1st SLC source address(4th, 5th row cycle address).

Table 4-40. Command Queue Second Word Register (Offset = 0x0304 + *n* * 0x20, *n* = 0 ~ 7)

Bit	Name	Type	Description
[31:24]	-	-	Reserved
[23:0]	row_addr_2nd	R/W	The second row address/source row address (Page index) The second row address decides the second chip row address in an interleaving flow, or decides the destination address in a copy-back or blanking check flow. When eD3 is configurable and issue eD3 copy back; this register can be 2nd SLC source address(4th, 5th row cycle address).

Table 4-41. Command Queue Third Word Register (Offset = 0x0308 + *n* * 0x20, *n* = 0 ~ 7)

Bit	Name	Type	Description
[31:16]	counter	R/W	Counter (For repeated operations) If the counter is used in a data transfer flow, the scale will show as below: If the ECC base is set to 1 kB, each count will represent 1 kB of data. If the ECC base is set to 512 bytes, each count will represent 512 bytes of data. The counter unit can be sector, page, or block. Please refer to the “Command Register Setting”(Table 5-3). (0 is inhibited).
[15:8]	sec_offset_2nd	R/W	The 2nd sector offset In the byte mode, this register decides the second column address (Issued with the first row address). Note: This value is only effective in the byte mode.
[7:0]	sec_offset_1st	R/W	The 1st sector offset This register decides the sector offset and hardware decode of this register to decide the first column address. In the byte mode, this register decides the first column address (Issued with the first row address).

Table 4-42. Command Queue Forth Word Register (Offset = 0x030C + *n* * 0x20, *n* = 0 ~ 7)

Bit	Name	Type	Description
[31:29]	start_ce	R/W	Command starting CE This register decides the Flash from which the current command starts. 3'b000: CE0 3'b001: CE1 3'b010: CE2 3'b011: CE3 3'b100: CE4 3'b101: CE5 3'b110: CE6 3'b111: CE7
28	byte_mode	R/W	Byte mode In the byte more, the spare value is updated from the spare register by channel. 1: Enable 0: Disable When executing the byte mode flow, data will not be protected by the ECC engine. The scramble and data inverse will not affect the byte mode flow.
27	bmc_ignore	R/W	Ignore the BMC region status of full/empty (User mode) 1: Ignore 0: Do not ignore Please refer to Chapter 5 for more detailed information.
[26:24]	bmc_region_sel	R/W	BMC region selection 3'b000 ~ 3'b111: Region 0 ~ Region 7 Note: NAND channels 0 ~ 3 can only select regions 0 ~ 3 and NAND channels 4 ~ 7 can only select regions 4 ~ 7. Please refer to Chapter 5 for more detailed information.
[23:19]	spare_num (0 ~ 31)	R/W	Number of spare date byte Constraints: • In the byte mode: Users can program spare_num 0 ~ 31, which is 1 byte ~ 32 byte, respectively. • Not in the byte mode: Users can only program spare_num 3, 7, 15, 31, which is 4, 8, 16, 32 bytes for the ECC engine protect spare data. • At ONFI or TOGGLE Flash, spare_num must be set to even number.

Bit	Name	Type	Description
[18:8]	cmd_index (11 bits)	R/W	Command index selection NANDC execute programming flow when bit[18] is set to '1'; otherwise, execute the fixed flow Bits[17:8] decide the starting position of the control flow. Please refer to Table 5-3 for more detailed information.
[7:5]	flash_type	R/W	Support flash operation type 3'b000: NV-SDR (Legacy Flash) 3;b001: NV-DDR1 (ONFI 2.0 flash) 3'b010: NV-DDR2 3'b011: Toggle 1.0 Flash 3'b100: Toggle 2.0 Flash
4	cmd_hsk_mode	R/W	Command handshake mode (Use the DMA handshake mode with the source DMA) This bit must be set '0' in the user mode. 1: Enable 0: Disable
[3:2]	cmd_scale	R/W	Command incremental scale 2'b11: Reserved 2'b10: By two block 2'b01: By one block 2'b00: By page Notes: • cmd_scale must set to 2'b00 in the Cache operation. • In the block erase flow, inc_by_page/blk must be set to in_by_blk. If the 2P block is erased or if the I2 block is erased, inc)by_page/blk must be set to two block. • In a blanking check flow, in_by_page/blk must be set to in_by_page.
1	-	-	Reserved
0	cmplt_intr_en	R/W	Command complete interrupt and status enable 1: Enable 0: Disable Note: When this bit is set, users must clear the command complete interrupt, and the host controller will execute the next command at the command queue. If the cmplt_intr_en bit is disabled and no command complete interrupt occurs, the host will automatically execute the next command. To know which command to be executed, please poll the command complete counter.

Table 4-44. Command Queue fifth Word Register (Offset = 0x0310 + *n* * 0x20, *n* = 0 ~ 7)

Bit	Name	Type	Description
[31:16]	row_addr_3rd	R/W	3rd SLC Source Address (4th 5th row cycle address) It is used for eD3 Flash
[15:0]	targ_row_addr	R/W	TLC Target Block Address (4th, 5th row cycle address) It is used for eD3 Flash

Table 4-45. Command Queue sixth Word Register (Offset = 0x0314 + *n* * 0x20, *n* = 0 ~ 7)

Bit	Name	Type	Description
[31:15]	-	-	Reserved
[14:12]	TLC_read_mod	R/W	Toshiba TLC read command register 3'b000: No issue Toshiba TLC read command 3'b001: Issue Toshiba TLC read lower command 3'b010: Issue Toshiba TLC read middle command 3'b011: Issue Toshiba TLC read upper command 3'b100: Issue Samsung TLC read command When executing Toshiba TLC read command including Lower/Middle/Upper TLC read command or Samsung TLC read command, this register must be set, otherwise this register must be set to '0'.
11	S_3SLC_TLC_en	R/W	Samsung 3SLC copy to TLC When executing Samsung 3SLC copy to TLC command, this bit must be set to '1', otherwise this bit must be set to '0',
10	T_3SLC_TLC_en	R/W	Toshiba 3SLC copy to TLC When executing Toshiba 3SLC copy to TLC command, this bit must be set to '1', otherwise this bit must be set to '0'. Note: Three registers of bit[10/11/12] are not allowed at the same time when two or three resisters are set to '1'.
[9:0]	Copy_back_index	R/W	eD3 Copy Back Index Index must be start from 0, this register is valid when bit[10] or bit[11] is set to '1'. For example, if TLC copy 5 times from 0 at one command, this register is set to '0' and counter is set to '5'. The next command must be from index 5, etc.

4.2.30 BMC Region Status Register (Offset = 0x0400)

This register provides the full or empty status of each region. The region halt status can also be read from this register (Region halt occurs when an ECC uncorrectable error is encountered and this region is not used in the user mode). Once a region is halted, the region software reset will be required for solving the halt state. Except the region FIFO, each region contains a small buffer to temporarily keep the write data. Please note that the region buffer empty status must be '1' and the region full status must be '0' before setting the DMA Mode Write Data Fill Register (Offset = 0x0424).

Table 4-46. BMC Region Status Register (Offset = 0x0400)

Bit	Name	Type	Description
[31:24]	region_buf_empty	R	Region buffer empty status 1: Region buffer is empty. 0: Region buffer is not empty. (Bit 31 ~ Bit 24: Region 7 ~ Region 0)
[23:16]	region_halt	R	Region halt status 1: Region is halted. 0: Region is not halted. (Bit 23 ~ Bit 16: Region 7 ~ Region 0)
[15:8]	region_full	R	Region full status 1: Region is full. 0: Region is not full. (Bit 15 ~ Bit 8: Region 7 ~ Region 0)
[7:0]	region_empty	R	Region empty status 1: Region is empty. 0: Region is not empty. (Bit 7 ~ Bit 7: Region 7 ~ Region 0)

4.2.31 Region *n* User Mode Pointer Adjustment Register (Offset = 0x0404 + *n* * 0x4, *n* = 0 ~ 7)

When the ignore bit is set in a command, the operation will enter the user mode. The BMC region will be used as a memory but not as a FIFO. By setting this register, the read and write positions can be adjusted. In the user mode, one BMC region contains 32 grids. The size of each grid is 512 bytes.

The pointers should be adjusted to the correct position every time before issuing a user-mode command. When processing a user-mode command with 1 kB per sector, the pointer must be set with an even value (e.g. 0, 2, 4, …, and so on).

Because each region only contains 16K bytes space, the length of one user mode read/write operation should be keep less than 16K bytes (Pointer offset + read/write sector count * n ≤ 32, n = 1 for 512 byte per sector; n = 2 for 1 kB per sector).

Table 4-47. Region *n* User Mode Pointer Adjustment Register (Offset = 0x0404 + *n* * 0x4, *n* = 0 ~ 7)

Bit	Name	Type	Description
[31:13]	-	-	Reserved
[12:8]	region_um_wrptr	W	Region *n* user mode write pointer (For the NAND Flash read command)
[7:5]	-	-	Reserved
[4:0]	region_um_rdptr	W	Region *n* user mode read pointer (For the NAND Flash write command)

4.2.32 DMA Mode Write Data Fill/Read Data Pop Register (Offset = 0x0424)

By writing to this register once (Write '1' and cleared by itself), one 512 bytes data (Dummy data) will be filled into or popped from the specified region. This is useful in Flash with 1 kB sector and only the 512 bytes data is required to be written to or read from the Flash. The pop or fill operation must be executed only once before the movement of the read/write data of the DMA mode or after all read/write data are transferred to/from AHB.

region_buf_empty should be '1' and region_full should be '0' before setting the fill_data register. The fill_data register is not allowed to be set during executing the NAND Flash read command.

region_empty should be '0' before setting the pop_data register. The pop_data register is not allowed to be set during executing the NAND Flash write command.

Table 4-48. DMA Mode Write Data Fill/Read Data Pop Register (Offset = 0x0424)

Bit	Name	Type	Description
[31:16]	-	-	Reserved
[15:8]	pop_data	W	This signal pops 512 bytes of data from specific regions (Used only in the DMA mode). Write 1: Pops 512 bytes of data Write 0: No effect (Bit 15 ~ Bit 8: Region 7 ~ Region 0)
[7:0]	fill_data	W	This signal fills 512 bytes of data into specific regions (Used only in the DMA mode). Write 1: Fills 512 bytes of data Write 0: No effect (Bit 7 ~ Bit 0: Region 7 ~ Region 0)

4.2.33 Region Software Reset Register (Offset = 0x0428)

The region can be reset by writing to this register.

Table 4-49. Region Software Reset Register (Offset = 0x0428)

Bit	Name	Type	Description
[31:8]	-	-	Reserved
[7:0]	region_sw_rst	W	Region software reset (This reset is cleared by itself.) Write 1: Reset Write 0: No effect (Bit 7 ~ Bit 0: Region 7 ~ Region 0)

4.2.34 Force Region Fill Read Data Register (Offset = 0x042C)

By setting this register to '1', the region will enter a non-empty state and return meaningless data to a requested AHB master. This register can only be set only when a region is not in the DMA handshake mode. (A region in the DMA handshake mode means that the command processed by a NANDC is with the DMA handshake mode enabled).

This register is useful in solving the condition that a DMA master reads data from a region in which an ECC uncorrectable event happens. Because the uncorrectable data cannot be read from this region, this bit must be set to prevent the DMA master from hanging.

Table 4-50. Force Region Fill Read Data Register (Offset = 0x042C)

Bit	Name	Type	Description
[31:8]	-	-	Reserved
[7:0]	force_fill_rd	R/W	Force region fill read data 1: Enable 0: Disable (Bit 7 ~ Bit 0: Region 7 ~ Region 0)

4.2.35 Region *x* Remaining Sector Count of Read Data Register (Offset = 0x0430 + *x* * 0x4, *x* = 0 ~ 7)

This register represents the remaining data in one region. The counting unit is in Sector (512 bytes or 1K bytes). Once an ECC uncorrectable error happens, the corresponding region will be halted in the uncorrected sector. The remaining sector counts can be read from this register.

Table 4-51. Region *x* Remaining Sector Count of Read Data Register (Offset = 0x0430 + *x* * 0x4, *x* = 0 ~ 7)

Bit	Name	Type	Description
[31:17]	-	-	Reserved
[16:0]	regionx_rd_cnt	R	Remaining sector count of the read data in region *x*

4.2.36 Revision Number Register (Offset = 0x0500)

Table 4-52. Revision Number Register (Offset = 0x0500)

Bit	Name	Type	Description
[31:0]	rev_num	R	Revision number of FTNANDC024

4.2.37 Feature 1 Register (Offset = 0x0504)

Table 4-53. Feature 1 Register (Offset = 0x0504)

Bit	Name	Type	Description
31	DDR_IF	R	DDR interface selection 0: Disable DDR_IF 1: Enable DDR_IF

Bit	Name	Type	Description
30	ahb_s3_clk_mode	R	AHB Slave3 interface clock mode selection 1: Asynchronous mode 0: Same clock mode
29	ahb_s2_clk_mode	R	AHB Slave2 interface clock mode selection 1: Asynchronous mode 0: Same clock mode
28	ahb_s1_clk_mode	R	AHB Slave1 interface clock mode selection 1: Asynchronous mode 0: Same clock mode
27	ahb_s0_clk_mode	R	AHB Slave0 interface clock mode selection 1: Asynchronous mode 0: Same clock mode
26	rgf_clk_mode	R	Register interface clock mode selection 1: Asynchronous mode 0: Same clock mode
[25:24]	mem_clk_mode	R	NAND interface clock mode selection 2'b00: Synchronous mode 2'b01: The mode is selected by the port. 2'b10: Same clock mode
[23:21]	ecc_cs_unfold	R	Unfold value of the ECC Chien search correction engine 3'b000: 4 3'b001: 8 3'b010: 16 3'b011: 32 3'b100: 64 Others: Reserved
[20:18]	ecc_cs_num	R	Number of the ECC Chien search correction engines 3'b000: 1 3'b001: 2 3'b010: 3 3'b011: 4 3'b100: 5 3'b101: 6 3'b110: 7 3'b111: 8

Bit	Name	Type	Description
[17:16]	bma_ffm_number	R	Number of BMA multipliers 2'b00 8 FFMs 2'b01: 16 FFMs 2'b10: 32 FFMs
[15:14]	ch_data_width	R	Channel data width 2'b00: 8 bits 2'b01: 16 bits 2'b10: 32 bits
[13:12]	pfc_depth	R	Programmable flow control SRAM depth 2'b00: 128 bytes 2'b01: 256 bytes
11	EBI_on	R	EBI function is turned on. 1'b1: Turned on 1'b0: Turned off
[10:8]	cmdq_depth	R	Command queue depth Possible values: 1, 2, and 4
7	-	-	Reserved
[6:4]	ahb_slave_num	R	Number of the AHB data slave ports Possible values: 1, 2, 3, and 4
[3:0]	nandc_ch_num	R	Number of the NANDC channels Possible values: 1, 2, 4, and 8

4.2.38 AHB Slave Memory Space Range Register (Offset = 0x0508)

Each AHB slave data port contains eight memory spaces corresponded to each region. The range of the memory space can be set from 512 bytes to 64K bytes in this register.

Table 4-54. AHB Slave Memory Space Range Register (Offset = 0x0508)

Bit	Name	Type	Description
[31:25]	-	-	Reserved
[24:16]	ahb_rd_len	R/W	AHB slave read data pre-fetch length To improve the performance of a read operation, the read data can be pre-fetched with a length set in this register. 9'h1 ~ 9'h80: 1 ~ 128 words

Bit	Name	Type	Description
			Other value is not allowed. Notes: • The value must be set to the power of 2. For example, set the value to 64 words or 128 words. The value will only be effective for the asynchronous AHB slave port. • The AHB read start address offset adds the pre-fetch value must not exceed the AHB memory range. • ahb_rd_len must be less than the memory range. • The prefetch value must not be set '1' (One word) when the AHB slave data width =64 bits. • The prefetch length must be equal to or less than the sector size. For example, when ECC base = '0', sector size = 512 byte, the prefetch length must be equal to or less than 512 byte.
[15:12]	ahb_force_len	R/W	Enable the read pre-fetch by the length set in ahb_rd_len If this bit is set to '1', the AHB slave port will pre-fetch the length specified in "ahb_rd_len". If this bit is set to '0', the pre-fetched length will be decided by HBUSRT at each AHB transaction. 1: Pre-fetch length depends on ahb_rd_len. 0: Pre-fetch length depends on HBURST. (If HBURST = INCR, the pre-fetch length will be '1', and other pre-fetch lengths will be defined by the HBURST value. That is; if HBURST is INCR4, the pre-fetch length will be four words) (Bits[15:12] correspond to the AHB Slave 3 to AHB Slave 0, respectively.) Notes: • This value is only effective for the asynchronous AHB slave port. • Because the BMC region is FIFO base, data ordering will be sequential even if HBURST = WRAP.
[11:8]	ahb_retry_en	R/W	AHB bus retry protocol enable 1: Enable (Default) 0: Disable Note: If the AHB slave port is connected to the single AHB bus, this bit must be set to '1'.
[7:0]	ahb_mem_range	R/W	AHB slave memory space range 8'b0000_0001: 512 bytes 8'b0000_0010: 1K bytes 8'b0000_0100: 2K bytes 8'b0000_1000: 4K bytes 8'b0001_0000: 8K bytes 8'b0010_0000: 16K bytes 8'b0100_0000: 32K bytes 8'b1000_0000: 64K bytes

4.2.39 Global Software Reset Register (Offset = 0x050C)

Writing '1' to this register resets all the modules in FTNANDC024, except for the asynchronous AHB data salve and the following registers:

- ECC control related register
- NANDC control-related register ("NANDC Auto Compare Pattern Register" is cleared.)
- Spare register
- Programmable flow control register
- Programmable OPCODE register

Global software reset will reset the following registers:

- ECC interrupt status of data and spare region
- ECC threshold interrupt status of data and spare region
- ECC error count of data and spare region
- NAND controller reset
- ECC engine reset
- BMC reset
- Programming/erase status
- Auto compare patterns and auto compare fail status
- Fill RD register

After the above registers are set, the NANDC software reset register must be '0' before further operations.

Table 4-55. Global Software Reset Register (Offset = 0x050C)

Bit	Name	Type	Description
[31:1]	-	-	Reserved
0	glb_sw_rst	W	Global software reset Write 1: Reset Write 0: No effect

4.2.40 AHB Data Slave Reset Register (Offset = 0x0510)

Table 4-56. AHB Data Slave Reset Register (Offset = 0x0510)

Bit	Name	Type	Description
[31:4]	-	-	Reserved
[3:0]	ahb_rst	R/W	AHB data slave software reset This reset is only effective for the asynchronous AHB data slave port. After setting this bit to '1', F/W must wait until this bit is cleared to '0' by H/W. It should be noted that when the AHB slave port is in the synchronous mode, this register should not be written. Write 1: Reset Write 0: No effect

4.2.41 ECC Correction Capability Register 1 (Offset = 0x0514)

Table 4-57. ECC Correction Capability Register 1 (Offset = 0x0514)

Bit	Name	Type	Description
31	ECC_CORR_32BIT_EN	R	ECC correction 32 bit
30	ECC_CORR_31BIT_EN	R	ECC correction 31 bit
29	ECC_CORR_30BIT_EN	R	ECC correction 30 bit
28	ECC_CORR_29BIT_EN	R	ECC correction 29 bit
27	ECC_CORR_28BIT_EN	R	ECC correction 28 bit
26	ECC_CORR_27BIT_EN	R	ECC correction 27 bit
25	ECC_CORR_26BIT_EN	R	ECC correction 26 bit
24	ECC_CORR_25BIT_EN	R	ECC correction 25 bit
23	ECC_CORR_24BIT_EN	R	ECC correction 24 bit
22	ECC_CORR_23BIT_EN	R	ECC correction 23 bit
21	ECC_CORR_22BIT_EN	R	ECC correction 22 bit
20	ECC_CORR_21BIT_EN	R	ECC correction 21 bit
19	ECC_CORR_20BIT_EN	R	ECC correction 20 bit
18	ECC_CORR_19BIT_EN	R	ECC correction 19 bit
17	ECC_CORR_18BIT_EN	R	ECC correction 18 bit
16	ECC_CORR_17BIT_EN	R	ECC correction 17 bit
15	ECC_CORR_16BIT_EN	R	ECC correction 16 bit
14	ECC_CORR_15BIT_EN	R	ECC correction 15 bit
13	ECC_CORR_14BIT_EN	R	ECC correction 14 bit

Bit	Name	Type	Description
12	ECC_CORR_13BIT_EN	R	ECC correction 13 bit
11	ECC_CORR_12BIT_EN	R	ECC correction 12 bit
10	ECC_CORR_11BIT_EN	R	ECC correction 11 bit
9	ECC_CORR_10BIT_EN	R	ECC correction 10 bit
8	ECC_CORR_9BIT_EN	R	ECC correction 9 bit
7	ECC_CORR_8BIT_EN	R	ECC correction 8 bit
6	ECC_CORR_7BIT_EN	R	ECC correction 7 bit
5	ECC_CORR_6BIT_EN	R	ECC correction 6 bit
4	ECC_CORR_5BIT_EN	R	ECC correction 5 bit
3	ECC_CORR_4BIT_EN	R	ECC correction 4 bit
2	ECC_CORR_3BIT_EN	R	ECC correction 3 bit
1	ECC_CORR_2BIT_EN	R	ECC correction 2 bit
0	ECC_CORR_1BIT_EN	R	ECC correction 1 bit

4.2.42 ECC Correction Capability Register 2 (Offset = 0x0518)

Table 4-58. ECC Correction Capability Register 2 (Offset = 0x0518)

Bit	Name	Type	Description
31	ECC_CORR_64BIT_EN	R	ECC correction 64 bit
30	ECC_CORR_63BIT_EN	R	ECC correction 63 bit
29	ECC_CORR_62BIT_EN	R	ECC correction 62 bit
28	ECC_CORR_61BIT_EN	R	ECC correction 61 bit
27	ECC_CORR_60BIT_EN	R	ECC correction 60 bit
26	ECC_CORR_59BIT_EN	R	ECC correction 59 bit
25	ECC_CORR_58BIT_EN	R	ECC correction 58 bit
24	ECC_CORR_57BIT_EN	R	ECC correction 57 bit
23	ECC_CORR_56BIT_EN	R	ECC correction 56 bit
22	ECC_CORR_55BIT_EN	R	ECC correction 55 bit
21	ECC_CORR_54BIT_EN	R	ECC correction 54 bit
20	ECC_CORR_53BIT_EN	R	ECC correction 53 bit
19	ECC_CORR_52BIT_EN	R	ECC correction 52 bit
18	ECC_CORR_51BIT_EN	R	ECC correction 51 bit
17	ECC_CORR_50BIT_EN	R	ECC correction 50 bit

Bit	Name	Type	Description
16	ECC_CORR_49BIT_EN	R	ECC correction 49 bit
15	ECC_CORR_48BIT_EN	R	ECC correction 48 bit
14	ECC_CORR_47BIT_EN	R	ECC correction 47 bit
13	ECC_CORR_46BIT_EN	R	ECC correction 46 bit
12	ECC_CORR_45BIT_EN	R	ECC correction 45 bit
11	ECC_CORR_44BIT_EN	R	ECC correction 44 bit
10	ECC_CORR_43BIT_EN	R	ECC correction 43 bit
9	ECC_CORR_42BIT_EN	R	ECC correction 42 bit
8	ECC_CORR_41BIT_EN	R	ECC correction 41 bit
7	ECC_CORR_40BIT_EN	R	ECC correction 40 bit
6	ECC_CORR_39BIT_EN	R	ECC correction 39 bit
5	ECC_CORR_38BIT_EN	R	ECC correction 38 bit
4	ECC_CORR_37BIT_EN	R	ECC correction 37 bit
3	ECC_CORR_36BIT_EN	R	ECC correction 36 bit
2	ECC_CORR_35BIT_EN	R	ECC correction 35 bit
1	ECC_CORR_34BIT_EN	R	ECC correction 34 bit
0	ECC_CORR_33BIT_EN	R	ECC correction 33 bit

4.2.43 ECC Correction Capability Register 3 (Offset = 0x051C)

Table 4-59. ECC Correction Capability Register 3 (Offset = 0x051C)

Bit	Name	Type	Description
[31:10]	-	-	Reserved
9	ECC_CORR_74BIT_EN	R	ECC correction 74 bit
8	ECC_CORR_73BIT_EN	R	ECC correction 73 bit
7	ECC_CORR_72BIT_EN	R	ECC correction 72 bit
6	ECC_CORR_71BIT_EN	R	ECC correction 71 bit
5	ECC_CORR_70BIT_EN	R	ECC correction 70 bit
4	ECC_CORR_69BIT_EN	R	ECC correction 69 bit
3	ECC_CORR_68BIT_EN	R	ECC correction 68 bit
2	ECC_CORR_67BIT_EN	R	ECC correction 67 bit
1	ECC_CORR_66BIT_EN	R	ECC correction 66 bit
0	ECC_CORR_65BIT_EN	R	ECC correction 65 bit

4.2.44 Programmable OPCODE Register (Offset = 0x0700 ~ 0x0704)

The OPCODE can be set in this register.

Table 4-60. Programmable OPCODE Register (Offset = 0x0700)

Bit	Name	Type	Description
[31:24]	opcode_3	R/W	Programmable OP-code byte 3
[23:16]	opcode_2	R/W	Programmable OP-code byte 2
[15:8]	opcode_1	R/W	Programmable OP-code byte 1
[7:0]	opcode_*0*	R/W	Programmable OP-code byte *0*

Table 4-61. Programmable OPCODE Register (Offset = 0x0704)

Bit	Name	Type	Description
[31:24]	opcode_*7*	R/W	Programmable OP-code byte 7
[23:16]	opcode_*6*	R/W	Programmable OP-code byte 6
[15:8]	opcode_*5*	R/W	Programmable OP-code byte 5
[7:0]	opcode_*4*	R/W	Programmable OP-code byte *4*

4.2.45 dqs_in_delay Register (Offset = 0x0520)

Table 4-62. dqs_in_delay Register (Offset = 0x0520)

Bit	Name	Type	Description
[31:5]	-	-	Reserved
[4:0]	R_dqs_delay	R/W	Read DQS delay counter Note: This register can exist when FPGA is set to ON. 0: Not delayed 1: Delayed for 1 T 2: Delayed for 2T Note: Legal value is between 1 and 31.

4.2.46 Spare Access Register (Offset = 0x1000 ~ 0x10FF)

The spare data can be read or written by accessing this register. The Flash ID is also read from this register. Every channel contains 32 bytes spare data.

The channel offset is as below:
Channel 0: Spare register offset is from 1000 ~ 101F.
Channel 1: Spare register offset is from 1020 ~ 103F.
Channel 2: Spare register offset is from 1040 ~ 105F.
Channel 3: Spare register offset is from 1060 ~ 107F.
Channel 4: Spare register offset is from 1080 ~ 109F.
Channel 5: Spare register offset is from 10A0 ~ 10BF.
Channel 6: Spare register offset is from 10C0 ~ 10DF.
Channel 7: Spare register offset is from 10E0 ~ 10FF.

4.2.47 Micro-Code SRAM Access Register (Offset = 0x2000 ~ 0x20FF)

Users can program the Micro-Code SRAM for flash flows that are not supported by the fixed flow. If the command index (Offset 30C bits[18:8]) bit[18] is set to '1', the programming flow by Micro-Code SRAM will be executed, and bits[17:8] will be the address of the Micro-Code SRAM address. Micro-Code SRAM will be 128 bytes or 256 bytes by hardware configuration.

4.2.48 Data SRAM Access Register (Offset = 0x2_0000 ~ 0x3_FFFF)

The data in BMC FIFO can be read or written by accessing this register. The mapping of the address offset to the exact region position is depicted in Figure 4-18.

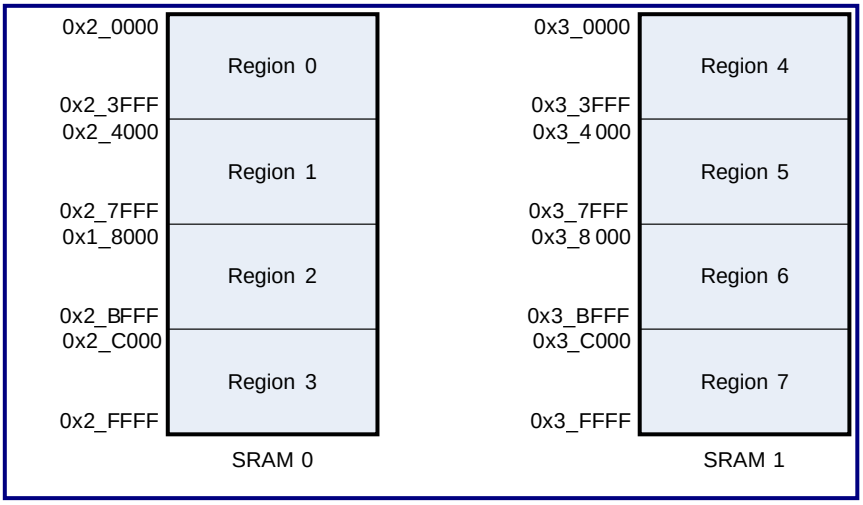


Figure 4-18. Data SRAM Address Offset Mapped to BMC Region

Chapter 5

Function Description

This chapter contains the following sections:

- 5.1 AHB Data Slave Port
- 5.2 Buffer Management Controller
- 5.3 AHB Register Slave
- 5.4 ECC Correction Error Handling
- 5.5 Auto Compare Error Handling
- 5.6 Auto-compare Check Mechanism
- 5.7 Scramble and Data Inverter
- 5.8 Performance
- 5.9 Abort Sequence
- 5.10 EBI (External Bus Interface) Operation
- 5.11 ODT/Warm-up Cycles Function
- 5.12 SRAM Behavior
- 5.13 Host Controller Data Format
- 5.14 Description of NANDC MicroCode

5.1 AHB Data Slave Port

FTNANDC024 comprises of up to four data slave ports. Each data slave port has a memory space ranging from 4 kB (512 bytes * 8 channels) to 512 kB (64k bytes * 8 channels). The memory space is equally divided into eight regions. Each region is directly mapped to one BMC region. One BMC region cannot be simultaneously accessed by two external AHB masters. It is suggested terminating all transfers in one region from a specific AHB master before starting another AHB master transfer.

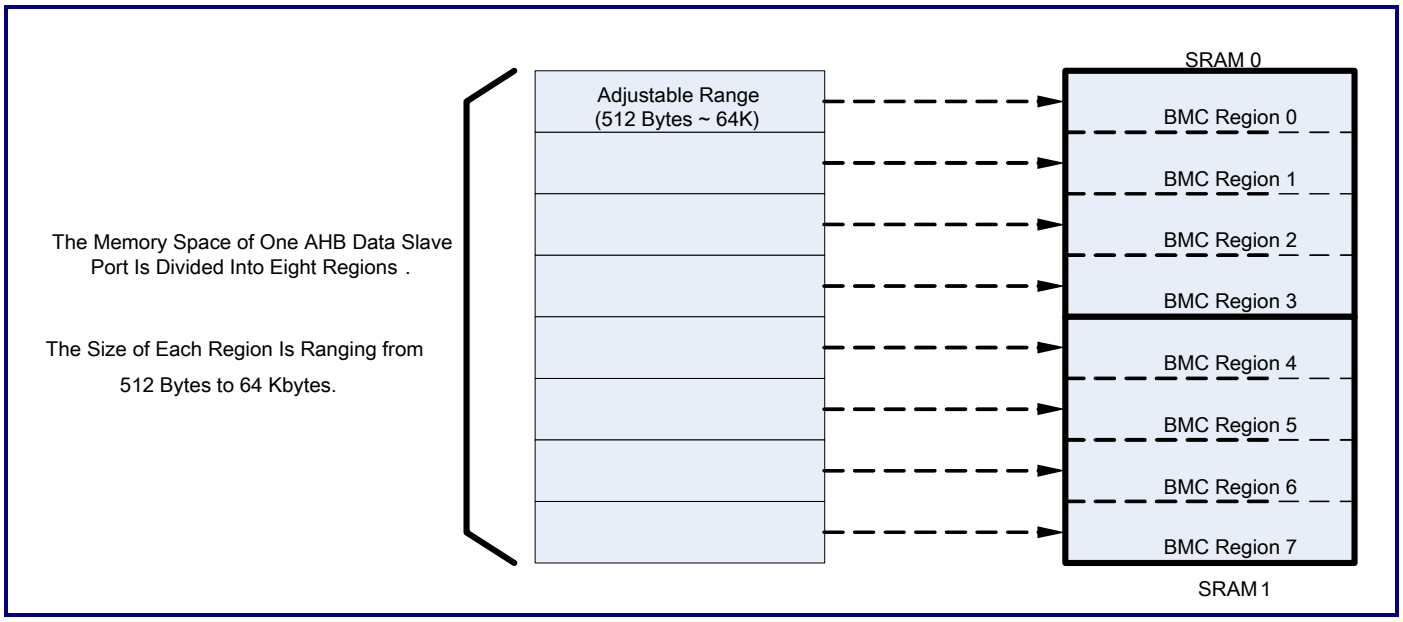


Figure 5-1. Mapping between AHB Slave and BMC Region

Example 1: Writing data to the BMC region 0 when the memory space is 512 bytes.
Data can be written to the BMC region 0 by issuing a write transaction with address offsets between 0x0000 and 0x01FF.

Example 2: Reading data from the BMC region 1 when the memory space is 1K bytes.
Data can be read from the BMC region 1 by issuing a read transaction with address offsets between 0x0400 and 0x07FF.

Note: The BMC region acts as FIFO in the DMA mode and there is no correlation between the address on the AHB bus and the data location in the BMC region.

The AHB data slave port supports the AHB retry protocol. When "ahb_retry_en" is set, the AHB data slave port will retry if the selected BMC region is not available for accessing.

The AHB data slave port can be in the same mode or asynchronous mode by using hardware configuration. If a data slave port is configured as the same mode, the AHB clock provided to the slave must be the same as the core clock. If the data slave port is configured as the asynchronous mode, the AHB clock has no relation with the core clock.

To improve the performance of the data read operation of the asynchronous AHB slave port, the data pre-fetch mechanism is adopted. The pre-fetch length can be decided by using the register setting or the AHB HBURST value. For example, if the pre-fetch length is set in the register as 64 words, the AHB slave will pre-fetch 64 words (512 bytes) of data from BMC when HBURST is INCR. The external DMA must read these 512 bytes of data before accessing other regions through the AHB slave. Another example is when INCR16 is issued, the AHB slave will pre-fetch 16 words (64 bytes) and this transaction must not be terminated before it is finished (It must not use two or more AHB masters to access one slave port at the same time, that is, before one AHB master finishing of transferring the pre-fetched data, the slave port cannot accept the access from another AHB master).

The address issued by the external DMA must be the fixed address or cannot cross the region boundary before all data with the pre-fetched length are accessed. The address must be incremental from the first word to the last word in the pre-fetch length when the fixed address is not used.

5.2 Buffer Management Controller

The Buffer Management Controller (BMC) is in charge of controlling data accessed from NANDC or the AHB data slave port. The data storage space is divided into eight regions (When the number of configured NANDC is eight, four regions will be used for four NANDC configurations and one region will be used for one NANDC configuration).

The "bmc_region_sel" field in the 4th word of a NANDC command decides the region to be accessed during processing a NANDC command. Please note that NANDC channel 0 to channel 3 can only access BMC region 0 to region 3; and NANDC channel 4 to channel 7 can only access BMC region 4 to region 7.

Setting the "bmc_ignore" bit in the 4th word of a NANDC command to '1' indicates that this command acts in the user mode. Read/Write from NANDC will not affect the region status used in the normal mode when operating in the user mode. The user-mode pointer can be adjusted by writing "User Mode Pointer Adjustment Register".

It shows a user-mode read operation from the NANDC channel 0 to region 0. Please note that the data will be ready in the region 0 only after the NAND command is completed when ECC is enabled.

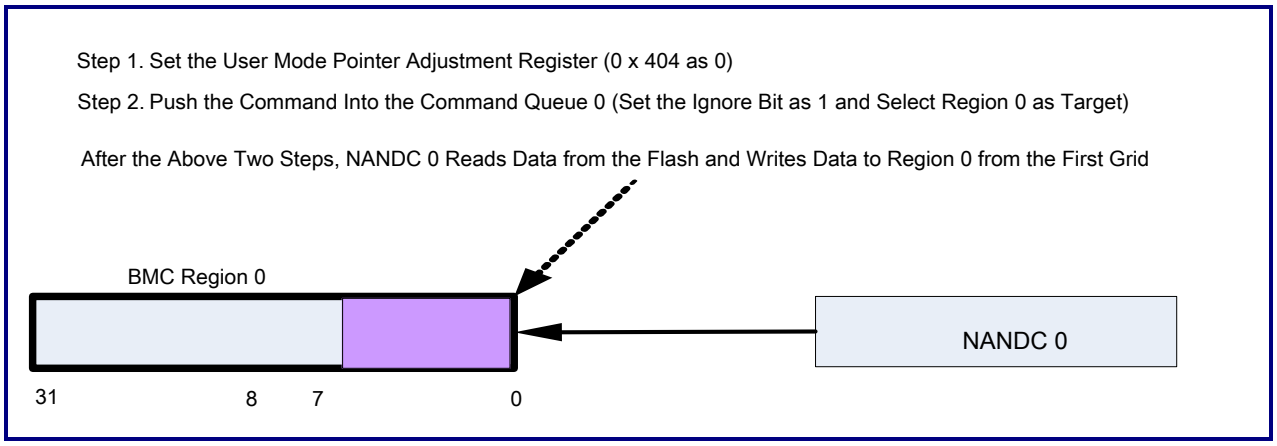


Figure 5-2. User Mode Operation

When operating in the normal mode (That is, the NANDC command with the "bmc_ignore" bit is set to '0'), the region will act as FIFO. For a Flash write operation, the data will be pushed from the AHB data slave port and NANDC will pop the data from the designated region. For a Flash read operation, the data will be pushed from NANDC and will be popped by the AHB data slave port after the ECC checking and correction.

When a host issues two commands, one is in the normal mode another is in the ignore mode, the host must make sure that the normal mode command complete includes the data transfer. Afterwards, the host can issue the ignore command. Otherwise, the data in the normal mode will be destroyed when the host will execute the ignore command.

The DMA handshake protocol is supported in FTNANDC024. When a NANDC command is issued with "cmd_hsk_mode" (In the 6th word of a NANDC command) as '1', the dma_req signal will be asserted under the following conditions:

- For a write operation, the dma_req signal will be asserted when at least a space of 512/1024 bytes is available in a region.
- For a read operation, the dma_req signal will be asserted when at least one sector can be read by the AHB data slave port. Each BMC region has one dedicated dma_req/dma_ack I/O, that is, bits[7:0] of dma_req/dma_ack correspond to region 0 to region 7 of BMC, respectively. The transfer length of one pair of dma_req/dma_ack (DMA burst size) is 512/1024 bytes of data.

Please note that the designated BMC region must not conflict with different NAND channels when issuing a NAND command. For example, if the NAND channel 0 designates region 0 as a target, channel 1 must not designate region 0 as a target before channel 0 finishes.

Please note that if the AHB slave port is configured as the ASync. mode, users can read the data with the following two methods:

- Users can read data after the AHB master poll the region empty (Offset 0x400) as '0' in the PIO mode.
- Users can directly read data by using the DMA handshake mode.

5.3 AHB Register Slave Port

FTNANDC024 comprises one AHB register slave port. This port can be used to access the following items:

- FTNANDC024 register
- Command queue
- Spare register
- Programmable OPCODE register
- MSC SRAM
- Data SRAM

Please note that the command queue can only be accessed with WORD/DWORD as HSIZE. The AHB register slave port can be configured as the asynchronous mode or synchronous mode. If configured as the synchronous mode, the AHB clock will be the same as the core clock. If configured as the asynchronous mode, the AHB clock will have no relation with the core clock.

5.4 ECC Correction Error Handling

When an ECC correction fail event is encountered, the ECC correction fail interrupt status can be checked by reading the register offset 0x0024. Consequently, the ECC correction failed channel will be halted, that is, the corresponded NAND channel controller will stop further operations. However, the designated BMC region may still contain the correct sectors that can be read out. Once the correct sectors in the BMC region are all read out, the "BMC region halt" and "region_buf_empty" statuses will be set to '1' and can be checked by reading the offset 0x0400.

Please keep the following procedure in mind to deal with the ECC correction fail event:

1. Check the "ECC Interrupt Status Register" (Offset = 0x0024) to decide the channel in which the ECC correction fails.
2. Poll "BMC Region Status Register" (Offset = 0x0400) to make sure that the correct data in the designated region are all read out.
3. Check "Region x Remaining Sector Count of Read Data" (Offset = 0x0430). With the values obtained in this register and row address in the command queue, the precise page location of the ECC un-correctable error can be calculated.
4. Reset the designated BMC region (The external DMA must be aborted before reset the BMC region reset)
5. Reset the AHB Slave (If the AHB Slave port is asynchronous.) and wait for the AHB reset to be cleared
6. Flush the corresponded command queue (This step is optional. If the command queue is not flushed, NANDC will repeat the current command after reset.)
7. Poll "0x100" to check if the target channel is at the ready state.
8. Reset the NANDC (This step can be skipped if Step 6 is executed.)
9. Poll the NANDC software reset register until it returns to '0'.

5.5 Auto Compare Error Handling

When the auto compare fail event occurs at manipulating blanking check, command complete will not be asserted, and F/W read register (0x0154) will find out auto compare fail. The host controller will hang on and F/W must perform the following sequence.

1. Check if the auto_cmp_fail (0x0154) status occurred
2. Flush the command queue (Including the reset controller and ECC engine)
3. Poll the NANDC software reset register until it returns to '0'

5.6 Auto-compare Check Mechanism

If the ECC correction bit is 6, the ECC parity bit will be 6 * 14/8 = 10.5 bytes and the auto-compare check mechanism will check total 11 bytes, not 10.5 bytes only. This mechanism is suitable for both the spare and data region ECC protection.

5.7 Scramble and Data Inverter

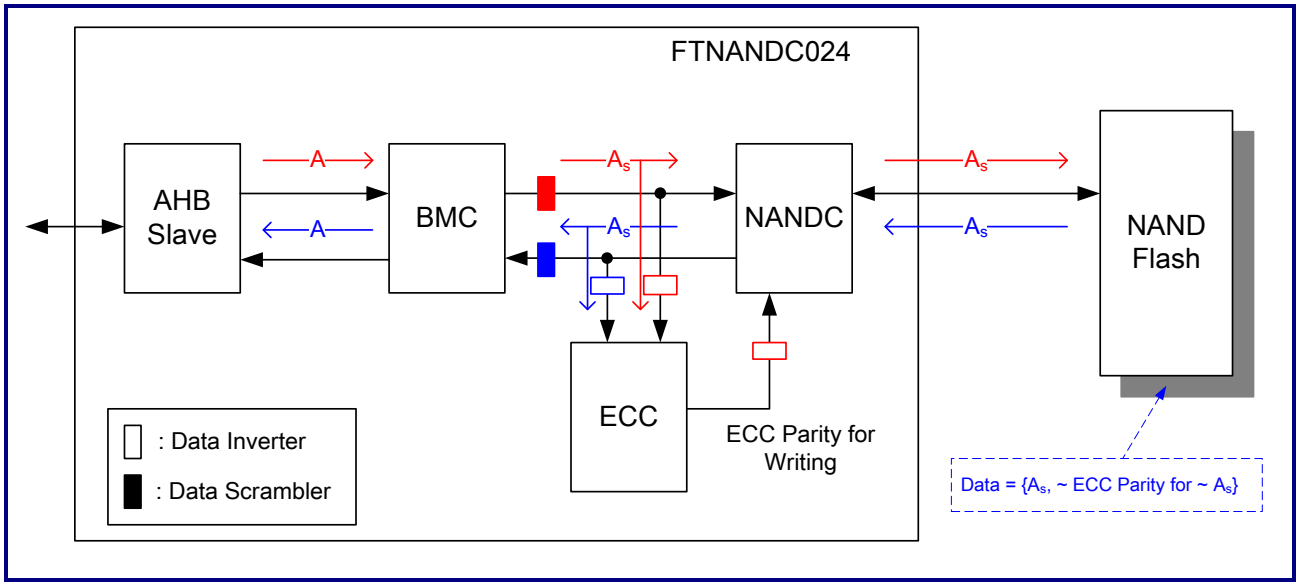


Figure 5-3. Scramble and Data Inverter

When reading data from the erase block region at scramble and data inverter enable, ECC will not fail and the BMC read data will not be FF. The returned data will be FF through the descrambler. Both data region and spare region can be applied to the scrambler and data inverter function shown in Figure 5-3.

5.8 Performance

BMC can only offer a throughput of 800 MB/s for NANDC four channels at 200 MHz of core_clk. If data access is simultaneously performed by two channels, the maximum throughput of each channel will be 800/2 = 400 MB/s; if data access is simultaneously performed by four channels, the maximum throughput of each channel will be 800/4 = 200 MB/s; if eight channels are configured, channel 0 ~ channel 3 will share 800MB/s and channel 4 ~ channel 7 will share 800MB/s since channel 0 ~ channel 3 are for DATA SRAM0 and channel 4 ~ channel 7 are for DATA SRAM1.

5.9 Abort Sequence

When the host controller wants to abort this command, please follow the sequence below to guarantee that the flow is correct.

1. Flush the command queue
2. Poll the NANDC software to '0'
3. Reset the BMC region
4. Reset the AHB Slave (If the AHB Slave port is asynchronous.) and wait for the AHB reset to be cleared
5. Issue the "RESET FLASH" command before issuing a new command

5.10 EBI (External Bus Interface) Operation

To share the pins of the chip with other memory controllers, FTNANDC024 provides an external bus interface to request for the data bus when the EBI capability is configured. An external arbiter is required to arbitrate the requests from different memory controllers.

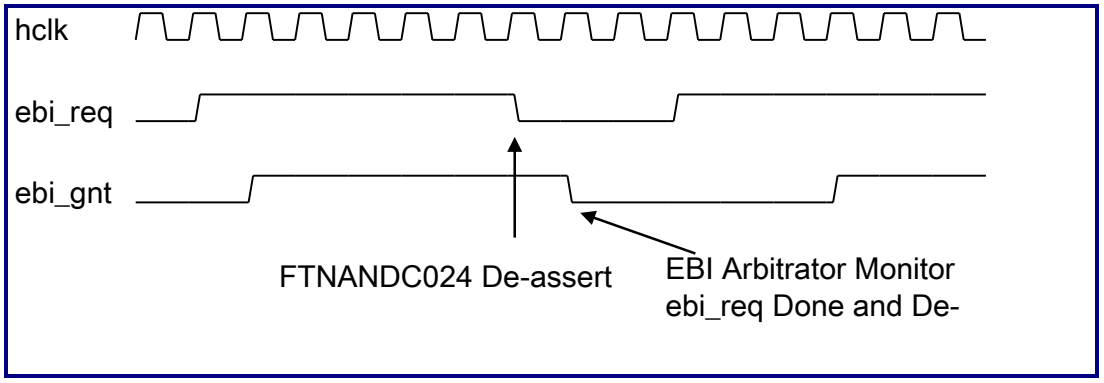


Figure 5-4. External Bus Interface Timing

Normally, "ebi_gnt" should not be de-asserted during the assertion of "ebi_req". However, FTNANDC024 supports the non-preemptive grant stop handshaking from the EBI arbiter, as shown in Figure 5-5. If a high priority EBI request is made, the EBI arbiter may de-assert "ebi_gnt" even if "ebi_req" is set to high. FTNANDC024 will release "ebi_req" as soon as possible.

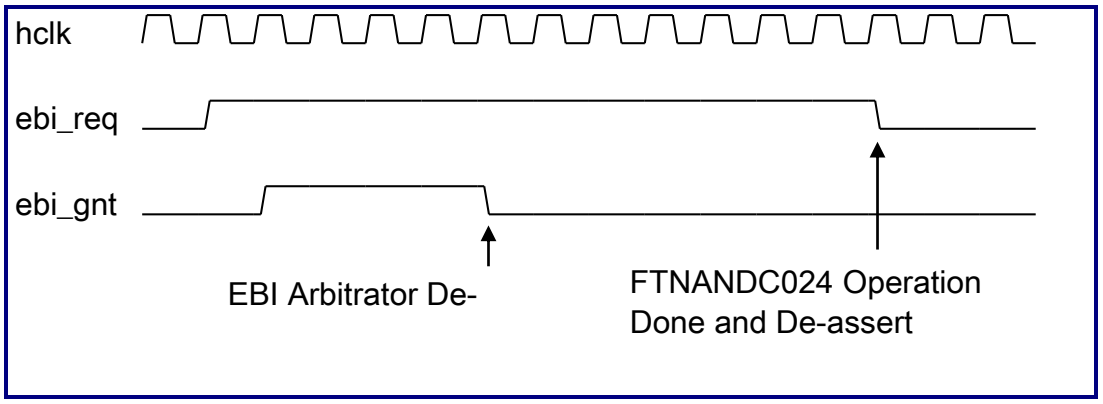


Figure 5-5. Grant Stop Handshaking with EBI Arbiter

If the EBI function is configured but not used, users should directly feedback "ebi_req" to "ebi_gnt", as shown in Figure 5-6.

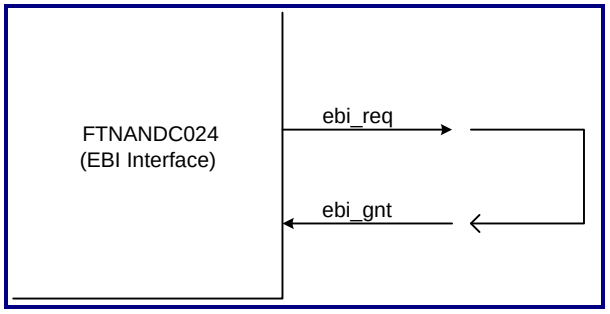


Figure 5-6. Example of EBI Interface Connection

Meanwhile, FTNANDC024 provides one pair of EBI per channel. EBI is only used at SDR I/Fs and DDR I/Fs will ignore it.

5.11 ODT/Warm-up Cycles Function

The DDR I/Fs of Toggle 2.0 provide ODT and the warm-up cycle function. These features are supported by FTNANDC024. Please refer to the specification of Toggle 2.0 for details.

5.12 SRAM Behavior

SRAM must be included in FTNANDC024 and the behavior of SRAM must be the same as the waveform shown in Figure 5-7 to meet the design timing requirement, especially when users generate SRAM at the FPGA stage.

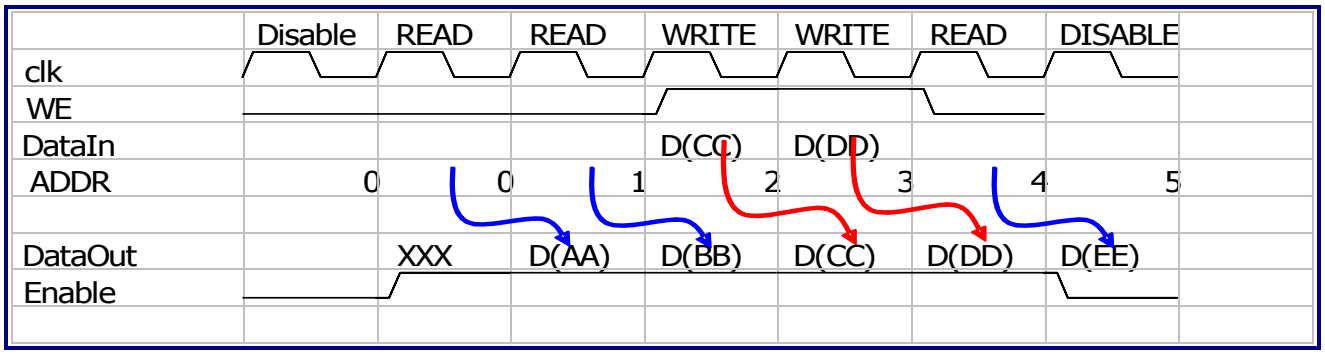


Figure 5-7. Waveform for SRAM Behavior

5.13 Host Controller Data Format

Figure 5-8 depicts the data format of the host controller.

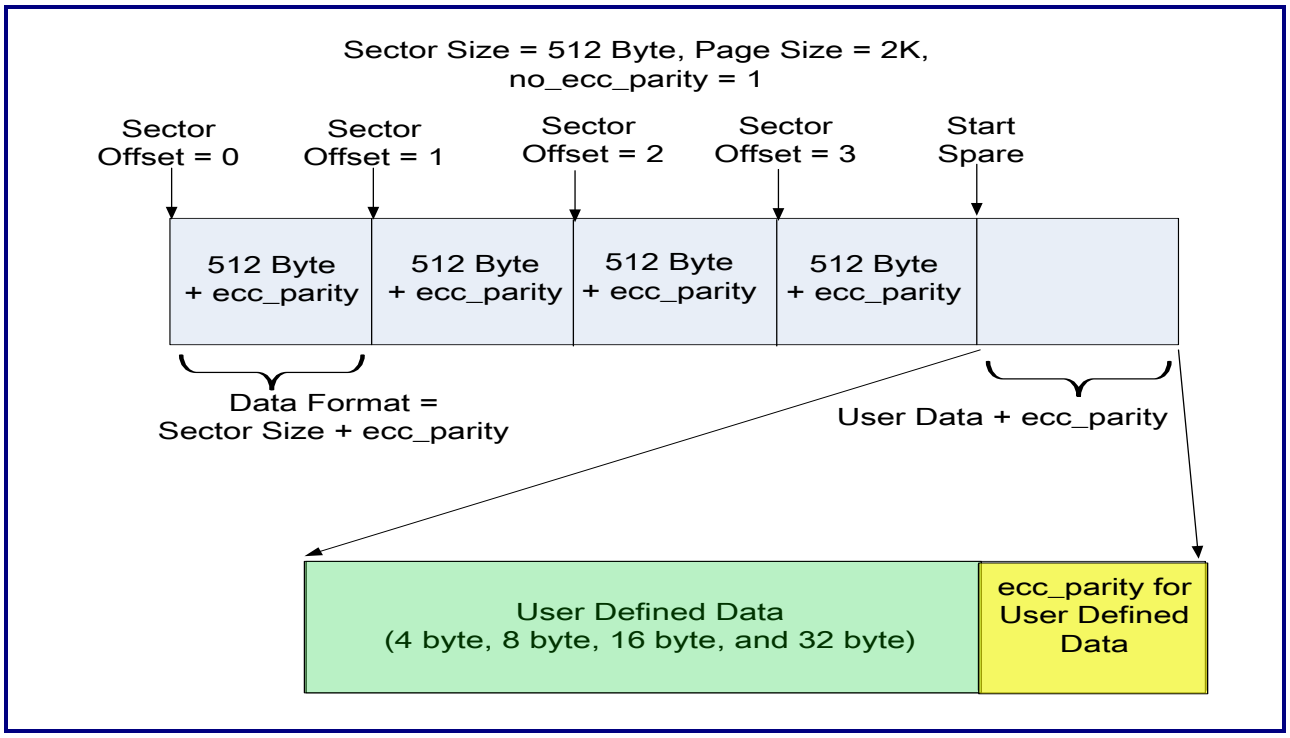


Figure 5-8. Data Format at no_ecc_parity = '0'

Legacy Flash Operation

The ECC parity is combined with the Flash data and the spare data and ecc_parity are placed after the data format to reduce the random data access time in order to improve the performance.

High-speed Flash Operation

If the number of ecc_parity is programmed by F/W to be an odd number, the data strobe must be even, as specified in the high-speed Flash specification. Therefore, ecc_parity will be automatically added by '1' by using the host controller that meets the even number of the high-speed Flash. Thus, the spare data used will be compressed. For example, if the page size is 4K, sector size is 512 bytes, spare data size is 128 bytes, and ECC correct is 6 bits (ecc_parity is 11 bytes); 4K/512 byte = 8 sectors, the remaining spare data size will be 128 - (8 * (11 + 1)) = 32 bytes at the high-speed Flash operation. However, for the legacy Flash operation, the remaining spare data size will be 128 - (11 * 8) = 40 bytes.

5.13.1 Spare Register Organization

Figure 5-9 depicts the spare register organization. Every channel has 32 bytes spare data used.

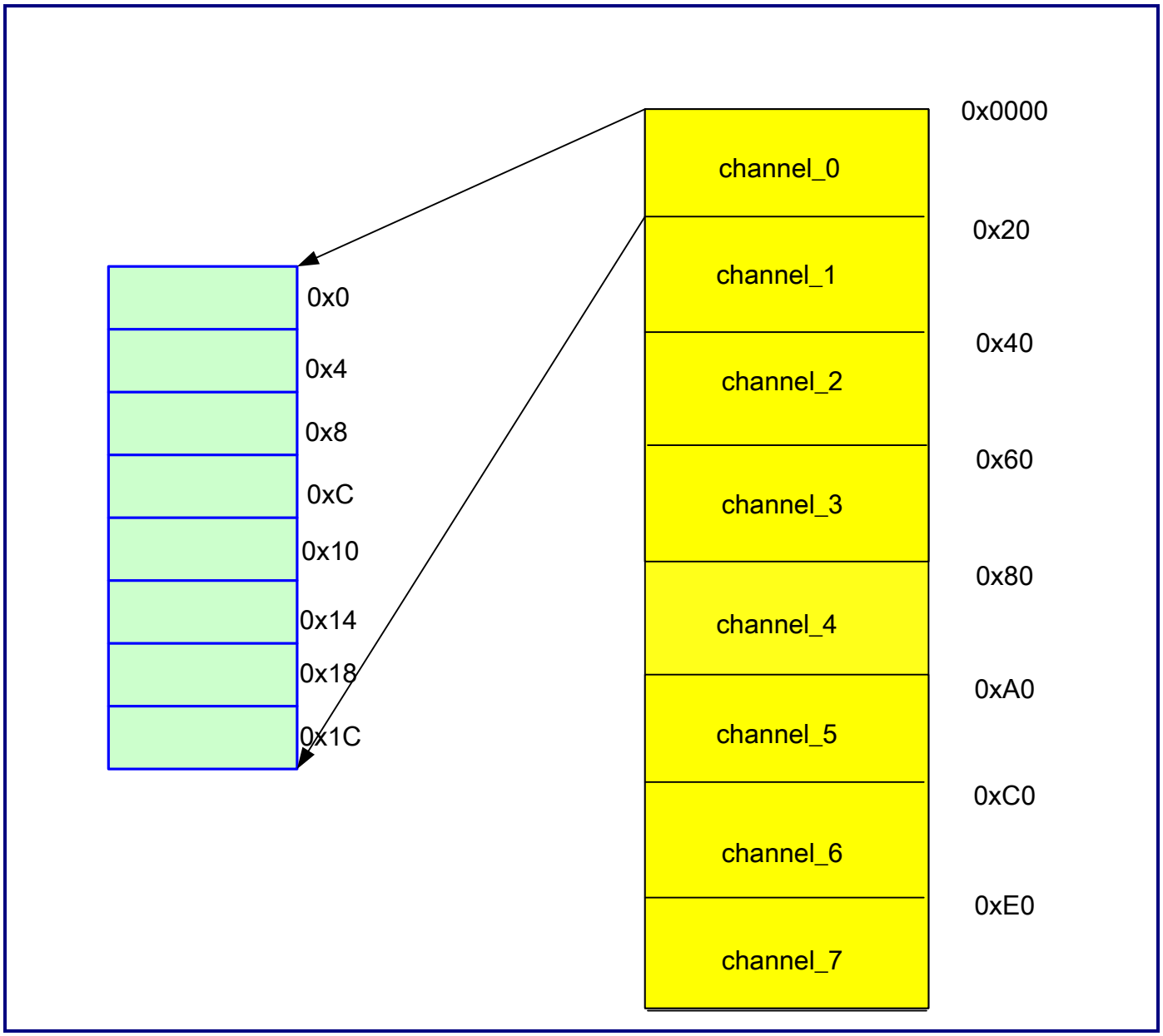


Figure 5-9. Spare Register Organization

5.13.2 User Data and ECC Usages

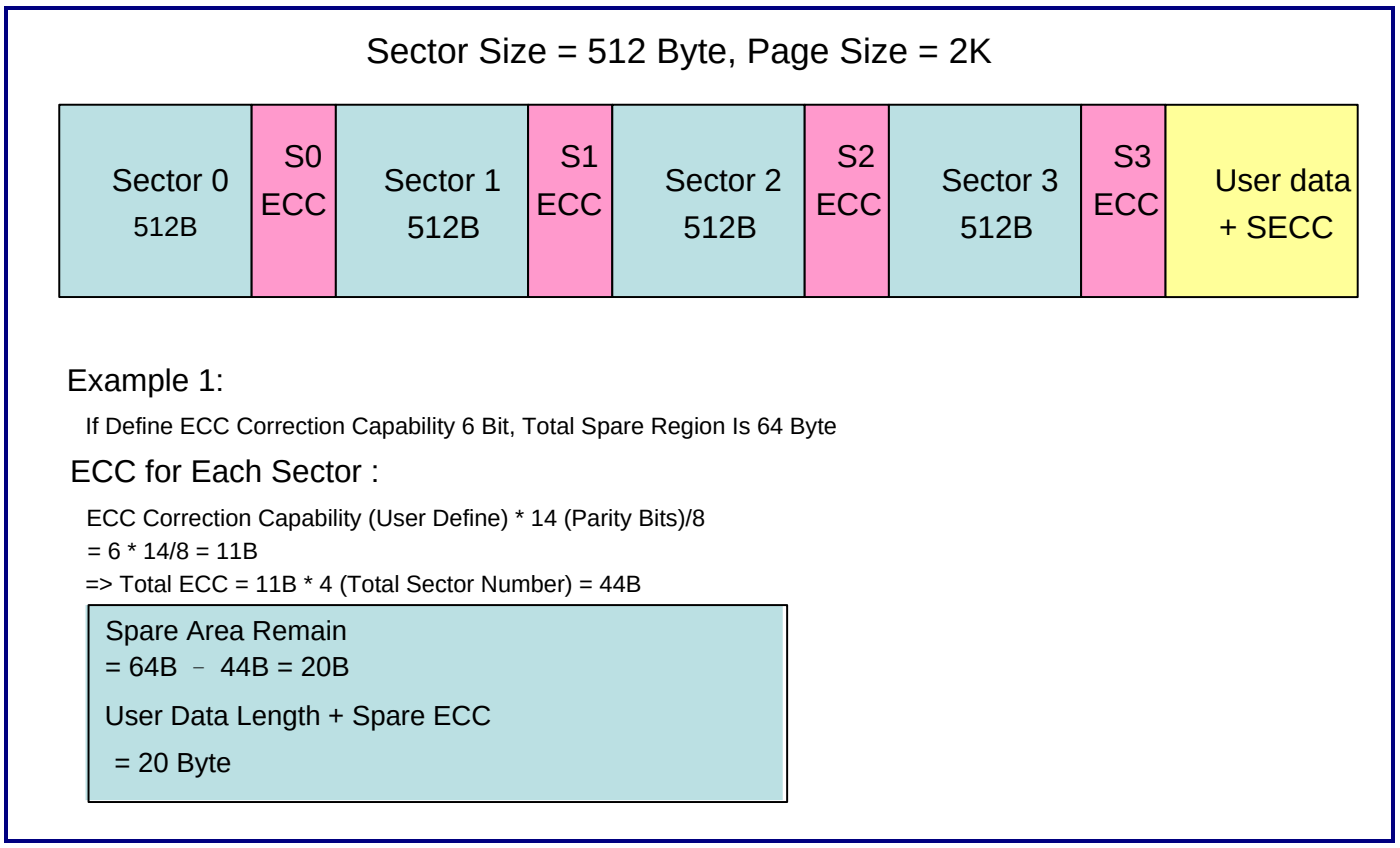


Figure 5-10. User Data and ECC Usages

5.14 Description of NANDC MicroCode

Table 5-1. Summary of MicroCode

MircoCode	bit[7:0]	operation comment
opcode	8'b00_??????	bit[5:0]means different opcode command, at most 64 opcode
zero_addr	8'b010_00000	issue address with all zero value, cycle number is setting by col_cyc
1st_row_addr	8'b010_00001	1st_row address cycles, cycle number is setting by row_cyc
1st_col_addr	8'b010_00010	1st_col address cycles is setting by col_cyc, and colmun address is generated by hardware base on 1st_sec_offset
2nd_row_addr	8'b010_00011	2nd_row address cycles is setting by row_cyc
2nd_col_addr	8'b010_00100	2nd_col address cycles is setting by col_cyc, and colmun address is generated by hardware base on 2nd_sec_offset
tog_1st_addr	8'b010_00101	toggle 1st_row_addr, and toggle address cycles is setting by row_cyc. toggle bit deponds on user setting
tog_2nd_addr	8'b010_00110	toggle 2nd_row_addr, and toggle address cycles is setting by row_cyc. toggle bit deponds on user setting
sp_col_addr	8'b010_00111	spare colmun address for flash random opcode . depends on 1st_sec_offset. col_cyc means source sp_col_addr cycles.
fix_1st_addr	8'b010_01000	fix start 1st_row_addr, row_cyc is fix addr cycles.
fix_2nd_addr	8'b010_01001	fix start 2nd_row_addr row_cyc is fix addr cycles.
var_addr1	8'b010_01010	variable byte address which indicates variable address register[7:0]
var_addr2	8'b010_01011	variable byte address which indicates variable address register[15:8]
var_addr3	8'b010_01100	variable byte address which indicates variable address register[23:16]
var_addr4	8'b010_01101	variable byte address which indicates variable address register[31:24]
3rd_row_addr	8'b010_01110	3rd row block address for eD3 flash
targ_1st_addr	8'b010_01111	target 1st block address for eD3 flash
targ_2nd_addr	8'b010_10000	target 2nd block address for eD3 flash
targ_3rd_addr	8'b010_10001	target 3rd block address for eD3 flash
order_row_addr	8'b010_10010	order number of row address for eD3
tog_3rd_addr	8'b010_10011	toggle 3rd row address, toggle block bit
tog_targ_1st_add	8'b010_10100	toggle target 1st row address, toggle block bit
var_opcode	8'b010_10101	variable opcode, 09h for 1st program, 0Dh for 2nd program for Toshiba eD3

MircoCode	bit[7:0]	operation comment
buffer1	8'b011_00000	buffer1: max{tADL,tCCS}~60ns
buffer2	8'b011_00001	buffer2: max{tAR,tRR,tCLR}~20ns
buffer3	8'b011_00010	buffer3: max{tRHW,tRHZ}~100ns
buffer4	8'b011_00011	buffer4: max{tWHR}~60ns
write data	8'b011_00100	write data state including data+ecc_parity
read data	8'b011_00101	read data state including data+ecc_parity
busy	8'b011_00110	busy state means wait busy ready by hardware
end	8'b011_00111	control flow end state
write spare	8'b011_01000	write spare state
read spare	8'b011_01001	read spare state
read status	8'b011_01010	read status once
multi read status	8'b011_01011	read status until busy/ready OK
blank write data	8'b011_01100	write whole page data for blanking check using auto compare pattern.
blank read data	8'b011_01101	read whole page data for blanking check using auto compare pattern
inc_ce	8'b011_01110	inc ce state means increase ce
dec_ce	8'b011_01111	dec ce state means decrease ce
RETURN	8'b011_10001	Return to Main Rountine from Sub-rountine
chk_rdy	8'b011_10010	only check flash ready bit
LINK	8'b011_10011	Link 2 or more commands become one entire flash command
goto_fix	8'b10_0?????	Goto fix flow sub-rountine. Following a microcode [7:0] means goto sub_rountine index
goto_pro	8'b10_1?????	Goto programming flow sub-rountine. Following a microcode [7:0] means goto sub_rountine in
goback	8'b11_??????	goback state means goback which index of control flow table. bit[5:0]means goback index number.

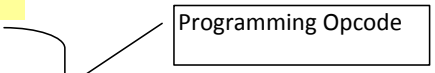
5.14.1 MicroCode Description

- OPCODE

Users can use the OPCODE index to choose the proper opcode based on Table 5-2.

Table 5-2. OPCODE

index('d)	OPCODE('h)	index('d)	OPCODE('h)	index('d)	OPCODE('h)	index('d)	OPCODE('h)
0	00	16	3A	32	8A	48	EE
1	01	17	3C	33	8B	49	EF
2	02	18	3F	34	8C	50	F1
3	03	19	50	35	90	51	F2
4	05	20	55	36	A2	52	FA
5	06	21	5C	37	C0	53	FC
6	09	22	5D	38	C5	54	FF
7	0D	23	60	39	D0	55	Byte0
8	10	24	69	40	D1	56	Byte1
9	11	25	70	41	DA	57	Byte2
10	15	26	71	42	DF	58	Byte3
11	1A	27	78	43	E0	59	Byte4
12	30	28	7B	44	E1	60	Byte5
13	31	29	80	45	E2	61	Byte6
14	32	30	81	46	EC	62	Byte7
15	35	31	85	47	ED	63	



For example, if users want to issue the opcode of 10, which the index is 8, to fill 8'b000_01000.

- zero_addr:

  When users fill zero address in a flow, it indicates that the zero address will be issued to Flash.
- 1st_row_addr:

  When users fill the first row address in a flow, it indicates that the host will issue the first-row address from the command table. After the goback state, 1st_row_addr will be increased using the page or block offset by programming the setting. At eD3 copy back, 1st row addr is 1st source address of SLC
- 1st_col_addr:

  When it is not in the byte mode, the host will issue the first column address according to the first sec. offset in the command table. For example, if the first sec. offset is '1', the sector size will be 512 bytes and the ECC parity will be 3 bytes. Then, the first column address can be calculated as (512 + 3).

  When it is in the byte mode, the first column address will be concatenated with the second offset and the first offset will become a 2-byte address, of which the address can be set to any value by users. If the host operates longer than the second time of 1st_col_addr in a flow, 1st_col_addr will become '0'.

- 2nd_row_addr:
  When users fill the second row address in a flow, it indicates that the host will issue the second row address from the command table. After the goback state, 2nd_row_addr will be increased using a page or block offset by programming the settings. At eD3 copy back, 2nd row address is 2nd source address of SLC
- toggle_1st_addr:
  The host will issue an address to toggle the specified addresses. When 1st_row_addr = 24'h0000_0000_0000, and toggle bit is 7, toggle_1st_addr will be 24'h0000_0000_0080.
- toggle_2nd_addr:
  The host will issue an address to toggle the specified addresses. When 2nd_row_addr = 24'h0000_0000_0000, and toggle bit is 6, toggle_2nd_addr will be 24'h0000_0000_0040.
- sp_col_addr:
  sp_col_addr = (sector_size + ecc_parity) * (page_size/sector_size).
- fix_1st_addr:
  The host will issue the fixed specific 1st_row_addr.
- fix_2nd_addr:
  The host will issue the fixed specific 2nd_row_addr.
- var_addr1, var_addr2, var_addr3, var_addr4:
  Variable byte addresses. It indicates variable address register (0x1D0) var_B1, var_B2, var_B3 and var_B4.
- 3rd_row_addr: Only for the eD3 copy back flow, and 3rd row address is 3rd source address of SLC.
- targ_1st_addr: Only for the eD3 copy back flow, and targ 1st row address is 1st target address of TLC.
- targ_2nd_addr: Only for the eD3 copy back flow, and targ 2nd row address is 2nd target address of TLC.
- targ_3rd_addr: Only for the eD3 copy back flow, and targ 3rd row address is 3rd target address of TLC.
- order_row_addr: Only for the eD3 copy back flow, it is row address of TLC programming order.
- tog_3rd_addr: Only for the eD3 copy back flow, it will toggle the 3rd row address.
- tog_targ_1st_addr: Only for the eD3 copy back flow, it will toggle target address of TLC
- var_opcode: Variable opcode microcode is only for Toshiba eD3 copy back flow, 1st program is 09h and 2nd program is 0Dh, others are buffer1 microcode.
- Buffer1:
  The buffer1 state issued by the host indicates that Flash will delay the AC timing by using the AC timing programming register.
- Buffer2:

The buffer2 state issued by the host indicates that Flash will delay the AC timing by using the AC timing programming register.

- Buffer3:
  The buffer3 state issued by the host indicates that Flash will delay the AC timing by using the AC timing programming register.
- Buffer4:
  The buffer4 state issued by the host indicates that Flash will delay the C timing by using the AC timing programming register.
- write data:
  The host will perform the write data strobe and ECC parity strobe until the page boundary or sector counter is reached.
- read data:
  The host will perform the read data strobe and ECC parity strobe until the page boundary or sector counter is reached.
- busy:
  The host will wait for the Flash busy/ready until busy/ready is set to high.
- end:
  The host issues the end state to perform the end command of the Flash and the controller will finish this command.
- write spare:
  The host will issue proper write spare length according to the sector counter until the page boundary or counter is reached.
- read spare:
  The host will issue proper read spare length according to the sector counter until the page boundary or counter is reached.
- multi-read status:
  The host will issue re_n to poll the Flash status until this status is ready and the check status passes/fails and report its status.
- read status :
  The host will issue one re_n and check the status regardless Flash is ready or busy and check status fail/pass and report status.
- blank write data:
  The host will issue the blank write data strobe in a blanking flow and its length is the total data format

including the data ECC parity and spare ECC parity.

- blank read data:

  The host will issue the blank read data strobe in a blanking flow and its length is the total data format including the data ECC parity and spare ECC parity.

  inc_ce:

  This state simply increases one CE.
- dec_ce:

  This state simply decreases one CE.
- Return:

  From subroutine return to the main flow
- Chk_rdy:

  Only continuous checks Flash ready bit till flash is ready and not check command fail/pass and not report status.
- LINK:

  Link 2 or more command flows become one entire Flash command. Link will be similar to the end state only when ce_n is active low at the link state and ce_n is active high at the end state. One flash entire command must be finished by end not by link. As a result, the valid command flow will be LINK-END, LINK-LINK-END, or LINK-LINK-.....-END.
- goto_fix:

  Go to fixed flow sub-routine from main flow. The following microcode[7:0] is sub-routine index.
- goto_pro:

  Go to programming flow sub-routine from main flow. The following microcode[7:0] is sub-routine index.
- goback:

  The goback state will continually perform a flow if the total data performed exceed one page. Bit 5 to Bit 0 are the goback index numbers. The maximum goback index is 63.

Note:

The goback state is 8'b11. Bit 5 to Bit 0 represent the goback index number. If one flow includes a 10-microcode state, and the first state is opcode, the second state will be the 1st_col address and the 9th state will be goback. The state will go to the goback state with index = 8 and the current goback index = 9. Consequently, goback to 9 - 8 = 1, the next state is the opcode state.

5.14.2 Flow Rules

The fixed flow and programming flow are comprised by MicoCode, as indicated in Table 5-1. The following rules must be followed.

- In the byte mode, only the read/write spare state can be used. The goback state cannot be used in a flow and the sector counter can be ignored. spare_num is programmed from 0 to 31 (1 byte ~ 32 bytes).
- The row/column address is decided by the flow state. When the state goes to a row/column state, the row/column address will be issued.
- In the byte mode, the first column address is decided by concatenating 1st_sec_offset and 2nd_sec_offset. 2nd_sec_offset is the high byte. When it is not in the byte mode, 1st_sec_offset will be the sector offsets. They can be programmed from 0 to 31. The first column address will be decided.
- The sector format of the read/write data state is (Data + ecc_parity); however, ecc_parity will always appear regardless ecc_en is enabled and no_ecc_parity is disabled. If no_ecc_parity is '1' and ecc_en is '0', ecc_parity will disappear in the data format.
- "intr_en" means that users must clear the command complete interrupt when a command finishes. The controller will then issue the next command if the command queue is not empty.
- The address will not be automatically generated across the chip boundary. This operation should be performed by users.
- The command queue will pop when a command finishes normally. However, when auto-compare fails or ECC fails, the command queue will not pop.
- The 2-plane commands only operate in one block. For the next block, users must re-issue a new command.
    - Example: As shown in Figure 5-11, block size = 64 pages. If users issue a 2-plane page write command to write four pages from page 63, this command will not be finished correctly by the host controller. Users must issue two commands; the 2-plane page writes to write two pages from page 63. Another command is from page 128.

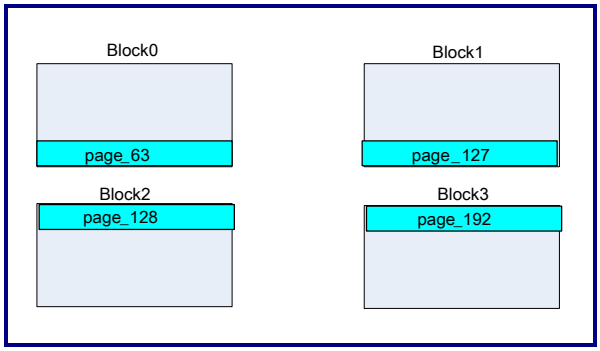


Figure 5-11. 2-plane Write 4 Pages Cross Block

- The row address can be increased by page or block after the goback state.
- The cache page read/write must not cross the blocks.
- One flow does not include two goback states.
- Each flow must have an end/link state. If a flow is finished with link, which is not the entire flash command, it must be finished with the end state.
- The programming flow cannot continuously include two busy states.
- The return state is returned to the main flow and the return state must be the last state of the subroutine.
- The goto state goes to the subroutine flow. The index of the goto state has at most 63. The subroutine must not be programmed more than 63 indexes. sub_routine is only allocated with fixed flow.
- The first MicroCode of flow must not be read data or write data.
- The MicroCode of the flow next to the read data or write data must not be read data or write data.
- The microcode of the flow next to the read data or write data must not be read spare or write spare.

5.14.3 Command Register Setting for Fixed Flow Command 1

Before executing a command, it is recommended referring to the command register setting (As shown in Table 5-3) to set the counter.

Table 5-3. Command Register Setting

Coommand Register setting	cmd_index ('h)	bmc_ignore	byte_mode	cmd_hsk_mode	1st_offset	2nd_offset	counter unit	update spare
PAGE READ	1C	V	0	V	V	don't care	sector	X
PAGE WRITE WITH SPARE	26	V	0	V	V	don't care	sector	V
SPARE WRITE	33	don't care	0	0	V	don't care	page	V
SPARE READ	3E	don't care	0	0	V	don't care	page	X
PAGE READ with SPARE	48	V	0	V	V	don't care	sector	X
PAGE WRITE	54	V	0	V	V	don't care	sector	X
READ ID (byte mode)	5F	don't care	1	0	V	V	don't care	X
RESET	65	don't care	0	0	don't care	don't care	don't care	X
BLOCK ERASE	68	don't care	0	0	don't care	don't care	block	X
INTERNAL COPY BACK	70	don't care	0	0	V	don't care	page	V
byte write(byte mode)	80	don't care	1	0	V	V	don't care	X
byte read(byte mode)	8A	don't care	1	0	V	V	don't care	X
multi read status	93	don't care	0	0	don't care	don't care	don't care	X
read status	96	don't care	0	0	don't care	don't care	times	X
2P Write/Micron	B0	V	0	V	V	don't care	sector	V
2P Erase/Sam	C6	don't care	0	0	don't care	don't care	2 block	X
2P PAGE READ	D0	V	0	V	V	don't care	sector	X
2P Write/Tog1	E6	V	0	V	V	don't care	sector	V
2P Erase/Tog1	FC	don't care	0	0	don't care	don't care	2 block	X
2P COPY BACK/Tog1	106	don't care	0	0	don't care	don't care	2 page	V
I2 Page Write	11A	V	0	V	V	don't care	sector	V
I2 BLOCK ERASE	135	don't care	0	0	don't care	don't care	2 block	X
2 LUN Page Write/Micron	146	V	0	V	V	don't care	sector	V
2 LUN Page Write/Samsung	167	V	0	V	V	don't care	sector	V
CACHE READ_1/Toshiba	185	V	0	V	0	don't care	sector	X
CACHE READ_2/Toshiba	194	V	0	V	0	don't care	sector	X
CAHCE WRITE_1/Toshiba	19C	V	0	V	0	don't care	sector	V
CAHCE WRITE_2/Toshiba	1A9	V	0	V	0	don't care	sector	V
COPY BACK WITH CAHCE_1/Toshib	1B5	don't care	0	0	0	don't care	don't care	X
COPY BACK WITH CAHCE_2/Toshib	1C2	don't care	0	0	0	don't care	2page + (page/counter)	X
COPY BACK WITH CAHCE_3/Toshib	1D0	don't care	0	0	0	don't care	don't care	X
HW COPY BACK	1DF	1	0	0	0	don't care	sector	V
Blanking Write	1F4	don't care	0	0	don't care	don't care	page	X
Blanking Read	1FF	don't care	0	0	don't care	don't care	page	X
BLANKING CHECK	209	don't care	0	0	don't care	don't care	page	X
SYN RESET	21A	don't care	0	0	don't care	don't care	don't care	X
READ PARAMETER PAGE	21D	V	0	V	don't care	don't care	sector	X
READ UNIQUE ID	224	V	0	V	don't care	don't care	sector	X
GET FEATURE	22B	don't care	1	0	don't care	don't care	don't care	X
SET FEATURE	232	don't care	1	0	don't care	don't care	don't care	X
SELECT LUN WITH STATUS	238	don't care	0	0	don't care	don't care	don't care	X

Coommand Register setting	cmd_index	bmc_ignore	byte_mode	cmd_hsk_mode	1st_offset	2nd_offset	counter unit	update spare
Small Page Command								
Small Page Read	23E	V	0	V	0	don't care	sector	X
Small Spare Read	249	don't care	0	0	don't care	don't care	sector	X
Small Byte Read_50	253	don't care	1	0	V	don't care	don't care	X
Small Byte Read_00	25C	don't care	1	0	V	don't care	don't care	X
Small Byte Read_01	264	don't care	1	0	V	don't care	don't care	X
Small Page Write	26C	V	0	V	0	don't care	sector	V
Small Spare Write	278	don't care	0	0	don't care	don't care	sector	V
Small Byte Write_50	283	don't care	1	0	V	don't care	sector	X
Small Byte Write_00	28D	don't care	1	0	V	don't care	sector	X
Small Byte Write_01	297	don't care	1	0	V	don't care	sector	X
Small Copy Back	2A1	don't care	0	0	don't care	don't care	sector	X
Small HW Copy Back	2AE	1	0	0	0	don't care	sector	V
Small Block Erase	2C1	don't care	0	0	don't care	don't care	block	X
Small Blanking Check	2C8	don't care	0	0	don't care	don't care	page	X

Notes:

1. "don't care" indicates that users can fill with anything.
2. "V" indicates the value-affect behavior.
3. "2 block/loop" indicates that "loop" is the counter number if the counter is set to '2', which will be totally 2 * 2 = 4 blocks.
4. Update spare is the spare data written to Flash from the spare register.
5. The number behind the command name is the command index. Users can fill in specified value in the command table to perform a flow.
6. Cache read operation: The cache_read_1 counter must be filled with multiplier of page by sector and cache_read_2 counter can be filled with sector unit. Example if one page has four sectors and users want to perform cache read with nine sectors, then the cache_read_1 counter must be eight sectors and the cache_read_2 counter will be 9 – 8 = 1 sector. If users want to read three pages, the cache_read_1 counter will be eight sectors and the cache_read_2 counter will be four sectors. Consequently, the cache_read_2 counter will always be less than or equal to one page and the cache_read_1 counter will always be the multiplier of page
7. Cache write operation: The cache_write_1 counter must fill multiple pages by sector and the cache_write_2 counter can be filled with the sector unit. For example, if one page has eight sectors and users want to perform the cache write with 19 sectors, then the cache_write_1 counter must be 16 sectors and the cache_write_2 counter will be 19 – 16 = 3 sectors. If users want to write five pages, the cache_write_1 counter will be four pages (4 * 8 = 32 sectors) and the cache_write_2 counter will be one page (Eight sectors). Consequently, the cache_write_2 counter will always be less than or equal to one page the and cache_write_1 counter will always be the multiplier of page.
8. Cache write address : If users want perform cache write M page, then row1_addr of cache_write_1 will be N, and row1_addr of cache_write_2 will be (N + M - 1)
9. CACHE Copy Back_2, if counter = '1', total copy back 3 page, and if counter = 3, total copy 3 + 2 = 5 pages. If users want to copy eight pages, the counter must be filled as 8 – 2 = 6.
10. Copy Back with Cache address: Please refer to the following figure to decide the row address.

if Copy M Page	ROW_1ST_ADDR	ROW_2ND_ADDR
COPY BACK WITH CAHCE_1/Toshib	X	Y
COPY BACK WITH CAHCE_2/Toshib	X+1	Y+1
COPY BACK WITH CAHCE_3/Toshib	X+M-1	Y+M-1

5.14.4 Fixed Flow Command 1 (CTD) and Usage

Before users issue a command, please refer to the following fixed flow, which contains some MicroCode, and is filled with the corresponding register to execute this command. For example, the first row address and second row address can act as a counter. For other settings, please refer to Table 5-3.

Subroutine Command Index

cmd_index('d)	0	5	10	16	23
offset	FIX_RDST	FIX_MRDST	FIX_RDI	FIX_RDO	FIX_CRDY
0	OPCODE_70	OPCODE_70	OPCODE_85	OPCODE_05	OPCODE_70
1	BUFFER4	BUFFER4	BUFFER4	SP_COL_ADDR	BUFFER4
2	RD_ST	MRD_ST	SP_COL_ADDR	OPCODE_E0	CHK_RDY
3	BUFFER3	BUFFER3	BUFFER1	BUFFER4	BUFFER3
4	FLW_RETURN	FLW_RETURN	WR_SP	RD_SP	FLW_RETURN
5			FLW_RETURN	BUFFER3	
6				FLW_RETURN	

Subroutine Usage: When the main flow executes FIX_MRDST sub-routine, it will issue multiple read status sub-routines and finally return to the main flow.

Basic_1

offset	Page Read	Page Write with Spare	Spare Write	Spare Read	Page Read with Spare	Page Write
0	OPCODE_00	OPCODE_80	OPCODE_80	OPCODE_00	OPCODE_00	OPCODE_80
1	COL_1ST_ADDR	COL_1ST_ADDR	SP_COL_ADDR	SP_COL_ADDR	COL_1ST_ADDR	COL_1ST_ADDR
2	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
3	OPCODE_30	BUFFER1	BUFFER1	OPCODE_30	OPCODE_30	BUFFER1
4	BUSY	WDATA	WR_SP	BUSY	BUSY	WDATA
5	BUFFER2	GOTO_FIX	OPCODE_10	BUFFER2	BUFFER2	OPCODE_10
6	RDATA	FIX_RDI	BUSY	RD_SP	RDATA	BUSY
7	BUFFER3	OPCODE_10	GOTO_FIX	BUFFER3	BUFFER3	GOTO_FIX
8	GOBACK_8	BUSY	FIX_RDST	GOBACK_8	GOTO_FIX	FIX_RDST
9	FLW_END	GOTO_FIX	GOBACK_9	FLW_END	FIX_RDO	GOBACK_9
10		FIX_RDST	FLW_END		GOBACK_10	FLW_END
11		GOBACK_11			FLW_END	
12		FLW_END				

Note: The offset number is the offset of a flow. For example, if users want to perform a PAGE READ flow and issue 1st_row_addr instead of starting from '0', only (Index + offset) should be issued, which is equivalent to 0 + 2 = 2. Consequently, users can fill two command indexes in the command table.

Basic_2

offset	Read ID(Byte Mode)	RESET Flash	Block Erase	Copy Back
0	OPCODE_90	OPCODE_FF	OPCODE_60	OPCODE_00
1	COL_1ST_ADDR	BUSY	ROW_1ST_ADI	ZERO_ADDR
2	BUFFER4	FLW_END	OPCODE_D0	ROW_1ST_ADDR
3	RD_SP		BUSY	OPCODE_35
4	BUFFER3		GOTO_FIX	BUSY
5	FLW_END		FIX_RDST	OPCODE_85
6			GOBACK_6	SP_COL_ADDR
7			FLW_END	ROW_2ND_ADDR
8				BUFFER1
9				WR_SP
10				OPCODE_10
11				BUSY
12				GOTO_FIX
13				FIX_RDST
14				GOBACK_14
15				FLW_END

Basic_3

offset	Byte Write	Byte Read	Multi Read Status	Read Status
0	OPCODE_80	OPCODE_00	GOTO_FIX	GOTO_FIX
1	COL_1ST_ADDR	COL_1ST_ADDR	FIX_MRDST	FIX_RDST
2	ROW_1ST_ADDR	ROW_1ST_ADDR	FLW_END	GOBACK_2
3	BUFFER1	OPCODE_30		FLW_END
4	WR_SP	BUSY		
5	OPCODE_10	BUFFER2		
6	BUSY	RD_SP		
7	GOTO_FIX	BUFFER3		
8	FIX_RDST	FLW_END		
9	FLW_END			

2Plane_1

offset	2P Write/Micron	2P Erase/Sam	2P Page Read
0	OPCODE_80	OPCODE_60	OPCODE_00
1	COL_1ST_ADDR	FIX_2ND_ADDR	COL_1ST_ADDR
2	ROW_1ST_ADDR	OPCODE_60	ROW_1ST_ADDR
3	BUFFER1	TOG_1ST_ADDR	OPCODE_30
4	WDATA	OPCODE_D0	BUSY
5	GOTO_FIX	BUSY	BUFFER2
6	FIX_RDI	GOTO_FIX	RDATA
7	OPCODE_11	FIX_RDST	BUFFER3
8	BUSY	GOBACK_8	GOTO_FIX
9	OPCODE_80	FLW_END	FIX_RDO
10	ZERO_ADDR		OPCODE_00
11	ROW_2ND_ADDR		ZERO_ADDR
12	BUFFER1		TOG_1ST_ADDR
13	WDATA		OPCODE_30
14	GOTO_FIX		BUSY
15	FIX_RDI		BUFFER2
16	OPCODE_10		RDATA
17	BUSY		BUFFER3
18	GOTO_FIX		GOTO_FIX
19	FIX_RDST		FIX_RDO
20	GOBACK_20		GOBACK_20
21	FLW_END		FLW_END

Note: “2P PAGE WRITE” has two different fix flows, one is for Samsung and the other is for Micron. “2P PAGE WRITE/Samsung” is for Samsung and “2P PAGE WRITE/Micron” is for Micron. “2P PAGE WRITE” depends on the Samsung Flash specifications to generate an easy and simple fixed flow. If users want to use this flow, please check if the flow matches the specifications. Users should treat “2P PAGE WRITE/Micron” as the one for Samsung.

2Plane_2

offse	2P Write/Tog1	2P Erase/Tog1	2P COPY BACK/Tog1
0	OPCODE_80	OPCODE_60	OPCODE_60
1	COL_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
2	ROW_1ST_ADDR	OPCODE_60	OPCODE_60
3	BUFFER1	ROW_2ND_ADDI	TOG_1ST_ADDR
4	WDATA	OPCODE_D0	OPCODE_35
5	GOTO_FIX	BUSY	BUSY
6	FIX_RDI	GOTO_FIX	OPCODE_85
7	OPCODE_11	FIX_RDST	ZERO_ADDR
8	BUSY	GOBACK_8	ROW_2ND_ADDR
9	OPCODE_81	FLW_END	OPCODE_11
10	ZERO_ADDR		BUSY
11	ROW_2ND_ADDR		OPCODE_81
12	BUFFER1		ZERO_ADDR
13	WDATA		TOG_2ND_ADDR
14	GOTO_FIX		OPCODE_10
15	FIX_RDI		BUSY
16	OPCODE_10		GOTO_FIX
17	BUSY		FIX_RDST
18	GOTO_FIX		GOBACK_18
19	FIX_RDST		FLW_END
20	GOBACK_20		
21	FLW_END		

Interleaving_1

offset	I2 Page Write	I2 Block Erase	2 LUN Page Write/Micron	2 LUN Page Write/Samsur
0	OPCODE_80	OPCODE_60	OPCODE_80	OPCODE_80
1	COL_1ST_ADDR	ROW_1ST_ADDR	COL_1ST_ADDR	COL_1ST_ADDR
2	ROW_1ST_ADDR	OPCODE_D0	ROW_1ST_ADDR	ROW_1ST_ADDR
3	BUFFER1	INC_CE	BUFFER1	BUFFER1
4	WDATA	GOTO_FIX	WDATA	WDATA
5	GOTO_FIX	FIX_CRDY	GOTO_FIX	GOTO_FIX
6	FIX_RDI	OPCODE_60	FIX_RDI	FIX_RDI
7	OPCODE_10	ROW_2ND_ADDR	OPCODE_10	OPCODE_10
8	INC_CE	OPCODE_D0	OPCODE_78	OPCODE_F2
9	GOTO_FIX	DEC_CE	FIX_2ND_ADDR	BUFFER4
10	FIX_CRDY	GOTO_FIX	BUFFER4	CHK_RDY
11	OPCODE_80	FIX_MRDST	CHK_RDY	BUFFER3
12	ZERO_ADDR	GOBACK_12	BUFFER3	OPCODE_80
13	ROW_2ND_ADDR	INC_CE	OPCODE_80	ZERO_ADDR
14	BUFFER1	GOTO_FIX	ZERO_ADDR	ROW_2ND_ADDR
15	WDATA	FIX_MRDST	ROW_2ND_ADDR	BUFFER1
16	GOTO_FIX	FLW_END	BUFFER1	WDATA
17	FIX_RDI		WDATA	GOTO_FIX
18	OPCODE_10		GOTO_FIX	FIX_RDI
19	DEC_CE		FIX_RDI	OPCODE_10
20	GOTO_FIX		OPCODE_10	OPCODE_F1
21	FIX_MRDST		OPCODE_78	BUFFER4
22	GOBACK_22		FIX_1ST_ADDR	MRD_ST
23	INC_CE		BUFFER4	BUFFER3
24	GOTO_FIX		MRD_ST	GOBACK_24
25	FIX_MRDST		BUFFER3	OPCODE_F2
26	FLW_END		GOBACK_26	BUFFER4
27			OPCODE_78	MRD_ST
28			FIX_2ND_ADDR	BUFFER3
29			BUFFER4	FLW_END
30			MRD_ST	
31			BUFFER3	
32			FLW_END	

Cache

offset	Cache Read/Toshiba/Micron	Cache Write/Toshiba/Micron	Cache Copy Back
0	OPCODE_00(CACHE Read_1)	OPCODE_80(CACHE_Write_1	OPCODE_00(CACHE_COPY_1)
1	COL_1ST_ADDR	COL_1ST_ADDR	COL_1ST_ADDR
2	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
3	OPCODE_30	BUFFER1	OPCODE_30
4	BUSY	WDATA	BUSY
5	BUFFER2	GOTO_FIX	BUFFER2
6	OPCODE_31	FIX_RDI	OPCODE_8C
7	BUSY	OPCODE_15	ZERO_ADDR
8	BUFFER2	BUSY	ROW_2ND_ADDR
9	RDATA	GOTO_FIX	BUFFER1
10	BUFFER3	FIX_RDST	OPCODE_15
11	GOTO_FIX	GOBACK_11	BUSY
12	FIX_RDO	LINK	LINK
13	GOBACK_7	OPCODE_80(CACHE_WRITE_	OPCODE_00(CAHCE_COPY_2)
14	LINK	ZERO_ADDR	ZERO_ADDR
15	OPCODE_3F(CACHE Read_2)	ROW_1ST_ADDR	ROW_1ST_ADDR
16	BUSY	BUFFER1	OPCODE_3A
17	BUFFER2	WDATA	BUSY
18	RDATA	GOTO_FIX	BUFFER3
19	BUFFER3	FIX_RDI	OPCODE_8C
20	GOTO_FIX	OPCODE_10	ZERO_ADDR
21	FIX_RDO	BUSY	ROW_2ND_ADDR
22	FLW_END	GOTO_FIX	BUFFER1
23		FIX_RDST	OPCODE_15
24		FLW_END	BUSY
25			GOBACK_12
26			LINK
27			OPCODE_00(CACHE_COPY_3)
28			ZERO_ADDR
29			ROW_1ST_ADDR
30			OPCODE_3A
31			BUSY
32			BUFFER2
33			OPCODE_8C
34			ZERO_ADDR
35			ROW_2ND_ADDR
36			BUFFER1
37			OPCODE_10
38			BUSY
39			GOTO_FIX
40			FIX_RDST
41			FLW_END

MISC

offset	HW Copy Back	Blanking Write	Blanking Read	Blanking Check
0	OPCODE_00	OPCODE_80	OPCODE_00	OPCODE_80
1	ZERO_ADDR	ZERO_ADDR	ZERO_ADDR	ZERO_ADDR
2	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
3	OPCODE_30	BUFFER1	OPCODE_30	BUFFER1
4	BUSY	BLK_WR	BUSY	BLK_WR
5	BUFFER2	OPCODE_10	BUFFER2	OPCODE_10
6	RDATA	BUSY	BLK_RD	BUSY
7	BUFFER3	GOTO_FIX	BUFFER3	OPCODE_00
8	OPCODE_80	FIX_RDST	GOBACK_8	ZERO_ADDR
9	ZERO_ADDR	GOBACK_9	FLW_END	ROW_1ST_ADDR
10	ROW_2ND_ADDR	FLW_END		OPCODE_30
11	BUFFER1			BUSY
12	WDATA			BUFFER2
13	GOTO_FIX			BLK_RD
14	FIX_RDI			BUFFER3
15	OPCODE_10			GOBACK_15
16	BUSY			FLW_END
17	GOTO_FIX			
18	FIX_RDST			
19	GOBACK_19			
20	FLW_END			

DDR_1

offset	SYN Reset	Read Parameter Page	Read Uniqe ID	Get Feature	Set Feature	Select LUN with Status
0	OPCODE_FC	OPCODE_EC	OPCODE_ED	OPCODE_EE	OPCODE_EF	OPCODE_78
1	BUSY	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADD	ROW_1ST_ADDR	ROW_1ST_ADDR
2	FLW_END	BUSY	BUSY	BUSY	BUFFER1	BUFFER4
3		BUFFER2	BUFFER2	BUFFER2	WR_SP	RD_ST
4		RDATA	RDATA	RD_SP	BUSY	BUFFER3
5		BUFFER3	BUFFER3	BUFFER3	FLW_END	FLW_END
6		FLW_END	FLW_END	FLW_END		

Note: Users must set no_ecc_parity and the sector counter according to the specification when executing READ_PARAMSTER_PAGE/READ UNIQUE ID.

Small Page Flow

Page Read

offset	Small Page Read	Small Spare Read	Small Byte Read_50	Small Byte Read_00	Small Byte Read_01
0	OPCODE_00	OPCODE_50	OPCODE_50	OPCODE_00	OPCODE_01
1	COL_1ST_ADDR	SP_COL_ADDR	COL_1ST_ADDR	COL_1ST_ADDR	COL_1ST_ADDR
2	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
3	BUSY	BUSY	BUSY	BUSY	BUSY
4	BUFFER2	BUFFER2	BUFFER2	BUFFER2	BUFFER2
5	RDATA	RD_SP	RD_SP	RD_SP	RD_SP
6	RD_SP	BUFFER3	BUFFER3	BUFFER3	BUFFER3
7	BUFFER3	BUSY	BUSY	FLW_END	FLW_END
8	BUSY	GOBACK_8	FLW_END		
9	GOBACK_9	FLW_END			
10	FLW_END				

Page Write

offset	Small Page Write	Small Spare Write	Small Byte Write_50	Small Byte Write_00	Small Byte Write_01
0	OPCODE_00	OPCODE_50	OPCODE_50	OPCODE_00	OPCODE_01
1	OPCODE_80	OPCODE_80	OPCODE_80	OPCODE_80	OPCODE_80
2	COL_1ST_ADDR	SP_COL_ADDR	COL_1ST_ADDR	COL_1ST_ADDR	COL_1ST_ADDR
3	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
4	BUFFER1	BUFFER1	BUFFER1	BUFFER1	BUFFER1
5	WDATA	WR_SP	WR_SP	WR_SP	WR_SP
6	WR_SP	OPCODE_10	OPCODE_10	OPCODE_10	OPCODE_10
7	OPCODE_10	GOTO_FIX	GOTO_FIX	GOTO_FIX	GOTO_FIX
8	GOTO_FIX	FIX_MRDST	FIX_MRDST	FIX_MRDST	FIX_MRDST
9	FIX_MRDST	GOBACK_8	FLW_END	FLW_END	FLW_END
10	GOBACK_9	FLW_END			
11	FLW_END				

MISC_1

offset	Small Copy Back	Small HW Copy Back	Small Block Erase	Small Blanking Check
0	OPCODE_00	OPCODE_00	OPCODE_60	OPCODE_00
1	COL_1ST_ADDR	COL_1ST_ADDR	ROW_1ST_ADDR	OPCODE_80
2	ROW_1ST_ADDR	ROW_1ST_ADDR	OPCODE_D0	ZERO_ADDR
3	BUSY	BUSY	GOTO_FIX	ROW_1ST_ADDR
4	BUFFER2	BUFFER2	FIX_MRDST	BUFFER1
5	OPCODE_8A	RDATA	GOBACK_5	BLK_WR
6	COL_2ND_ADDR	BUFFER3	FLW_END	OPCODE_10
7	ROW_2ND_ADDR	BUSY		GOTO_FIX
8	BUSY	OPCODE_80		FIX_MRDST
9	GOTO_FIX	COL_2ND_ADDR		OPCODE_00
10	FIX_MRDST	ROW_2ND_ADDR		ZERO_ADDR
11	GOBACK_11	BUFFER1		ROW_1ST_ADDR
12	FLW_END	WDATA		BUSY
13		WR_SP		BUFFER2
14		OPCODE_10		BLK_RD
15		GOTO_FIX		BUFFER3
16		FIX_MRDST		BUSY
17		GOBACK_17		GOBACK_17
18		FLW_END		FLW_END

5.14.5 Command Register Setting for Fixed Flow Command 2

Coommand Register setting Large Page	cmd_index ('h)	bmc_ignore	byte_mode	cmd_hsk_mode	1st_offset	2nd_offset	counter unit	update spare
PAGE READ	79	V	0	V	V	don't care	sector	X
PAGE WRITE WITH SPARE	85	V	0	V	V	don't care	sector	V
SPARE WRITE	91	don't care	0	0	V	don't care	page	V
SPARE READ	9B	don't care	0	0	V	don't care	page	X
PAGE READ with SPARE	A7	V	0	V	V	don't care	sector	X
READ ID (byte mode)	B5	don't care	1	0	V	V	don't care	X
RESET	BB	don't care	0	0	don't care	don't care	don't care	X
BLOCK ERASE	BF	don't care	0	0	don't care	don't care	block	X
BLANKING CHECK	C6	don't care	0	0	don't care	don't care	page	X
byte write(byte mode)	DA	don't care	1	0	V	V	don't care	X
byte read(byte mode)	E3	don't care	1	0	V	V	don't care	X
multi read status	EE	don't care	0	0	don't care	don't care	don't care	X
read status	F1	don't care	0	0	don't care	don't care	times	X
Coommand Register setting Small Page	cmd_index	bmc_ignore	byte_mode	cmd_hsk_mode	1st_offset	2nd_offset	counter unit	update spare
Small Page Command								
Small Page Read	F5	V	0	V	0	don't care	sector	X
Small Spare Read	100	don't care	0	0	don't care	don't care	sector	X
Small Byte Read_50	10A	don't care	1	0	V	don't care	don't care	X
Small Byte Read_00	113	don't care	1	0	V	don't care	don't care	X
Small Byte Read_01	11B	don't care	1	0	V	don't care	don't care	X
Small Page Write	123	V	0	V	0	don't care	sector	V
Small Spare Write	12F	don't care	0	0	don't care	don't care	sector	V
Small Byte Write_50	13A	don't care	1	0	V	don't care	sector	X
Small Byte Write_00	144	don't care	1	0	V	don't care	sector	X
Small Byte Write_01	14E	don't care	1	0	V	don't care	sector	X
Small HW Copy Back	158	1	0	0	0	don't care	sector	V
Small Block Erase	16B	don't care	0	0	don't care	don't care	block	X
Small Blanking Check	172	don't care	0	0	don't care	don't care	page	X
GET FEATURE	185	don't care	1	0	don't care	don't care	don't care	X
SET FEATURE	18E	don't care	1	0	don't care	don't care	don't care	X

Coommand Register setting eD3 command: Prefix T: Toshiba Prefix S: Samsung	cmd_index	bmc_ignore	byte_mode	cmd_hsk_mode	1st_offset	2nd_offset	counter unit	update spare
T_SLC Page Read with Spare	195	V	0	V	V	don't care	sector	X
T_SLC Spare Read	1BA	don't care	0	0	V	don't care	don't care	X
T_SLC Byte Read	1C9	don't care	1	0	V	V	don't care	X
T_SLC Page Write with spare	1D8	V	0	V	V	don't care	sector	V
T_SLC Byte Write	1F7	don't care	1	0	V	V	don't care	X
T_SLC Erase	201	don't care	0	0	don't care	don't care	1 block	X
T_TLC Lower Page Read with Spare	209	V	0	V	V	don't care	sector	X
T_TLC Lower Spare Read	22D	don't care	0	0	V	don't care	don't care	X
T_TLC Middle Page Read with Spar	23C	V	0	V	V	don't care	sector	X
T_TLC Middle Spare Read	260	don't care	0	0	V	don't care	don't care	X
T_TLC Upper Page Read with Spare	26F	V	0	V	V	don't care	sector	X
T_TLC Upper Spare Read	293	don't care	0	0	V	don't care	don't care	X
T_TLC 3SLC copy TLC	2A2	don't care	0	0	0	don't care	zigzag time	X
S_SLC Page Read with Spare	2AA	V	0	V	V	don't care	sector	X
S_SLC Spare Read	2AD	don't care	0	0	V	don't care	page	X
S_SLC Byte Read	2B0	don't care	1	0	V	V	don't care	X
S_SLC Page Write with Spare	2B3	V	0	V	V	don't care	sector	V
S_SLC Byte Write	2B6	don't care	1	0	V	V	don't care	X
S_SLC Erase	2B9	don't care	0	0	don't care	don't care	block	X
S_TLC Erase	2BC	don't care	0	0	don't care	don't care	block	X
S_TLC Page Read with spare	2BF	V	0	V	V	don't care	sector	X
S_TLC Spare Read	2C2	don't care	0	0	V	don't care	page	X
S_TLC 3SLC copy TLC	2C5	don't care	0	0	0	don't care	zigzag time	X

5.14.6 Fixed Flow Command 2 (MTD) and Usage

Sub-routine 1

cmd_index('d)	0	5	10	16	23
offset	FIX_RDST	FIX_MRDST	FIX_RDI	FIX_RDO	FIX_CRDY
0	OPCODE_70	OPCODE_70	OPCODE_85	OPCODE_05	OPCODE_70
1	BUFFER4	BUFFER4	BUFFER4	SP_COL_ADDI	BUFFER4
2	RD_ST	MRD_ST	SP_COL_ADDR	OPCODE_E0	CHK_RDY
3	BUFFER3	BUFFER3	BUFFER1	BUFFER4	BUFFER3
4	FLW_RETURN	FLW_RETURN	WR_SP	RD_SP	FLW_RETURN
5			FLW_RETURN	BUFFER3	
6				FLW_RETURN	

Sub-routine 2

cmd_cindex	28	59	90
offset	//T_TLC Copy Lower	//T_TLC Copy Middle	//T_TLC Copy Upper
0	OPCODE_A2	OPCODE_A2	OPCODE_A2
1	OPCODE_60	OPCODE_60	OPCODE_60
2	ROW_1ST_ADDR	ROW_2ND_ADDR	ROW_3RD_ADDR
3	OPCODE_60	OPCODE_60	OPCODE_60
4	TOG_1ST_ADDR	TOG_2ND_ADDR	TOG_3RD_ADDR
5	OPCODE_30	OPCODE_30	OPCODE_30
6	OPCODE_70	OPCODE_70	OPCODE_70
7	BUFFER4	BUFFER4	BUFFER4
8	MRD_ST	MRD_ST	MRD_ST
9	BUFFER3	BUFFER3	BUFFER3
10	VAR_OPCODE	VAR_OPCODE	VAR_OPCODE
11	OPCODE_01	OPCODE_02	OPCODE_03
12	OPCODE_85	OPCODE_85	OPCODE_85
13	ZERO_ADDR	ZERO_ADDR	ZERO_ADDR
14	TARG_1ST_ADDR	TARG_1ST_ADDR	TARG_1ST_ADDR
15	OPCODE_11	OPCODE_11	OPCODE_11
16	OPCODE_70	OPCODE_70	OPCODE_70
17	BUFFER4	BUFFER4	BUFFER4
18	MRD_ST	MRD_ST	MRD_ST
19	BUFFER3	BUFFER3	BUFFER3
20	VAR_OPCODE	VAR_OPCODE	VAR_OPCODE
21	OPCODE_01	OPCODE_02	OPCODE_03
22	OPCODE_85	OPCODE_85	OPCODE_85
23	ZERO_ADDR	ZERO_ADDR	ZERO_ADDR
24	TOG_TARG1_ADDR	TOG_TARG1_ADDR	TOG_TARG1_ADDR
25	OPCODE_1A	OPCODE_1A	OPCODE_10
26	OPCODE_70	OPCODE_70	OPCODE_70
27	BUFFER4	BUFFER4	BUFFER4
28	MRD_ST	MRD_ST	MRD_ST
29	BUFFER3	BUFFER3	BUFFER3
30	FLW_RETURN	FLW_RETURN	FLW_RETURN

Sub-routine summary

sub-routine	cmd_index('d)
FIX_RDST	0
FIX_MRDST	5
FIX_RDI	10
FIX_RDO	16
FIX_CRDY	23
T_TLC Copy Lower	28
T_TLC Copy Middle	59
T_TLC Copy Upper	90
FIX_PWS	133
FIX_SR	155
FIX_PRS	167
FIX_ER	191
FIX_BW	218
FIX_BR	227

FIX_PWS is page write with spare fixed flow.
FIX_SR is spare write fixed flow.
FIX_PRS is page read with spare fixed flow.
FIX_ER is block erase fixed flow.
FIX_BW is byte write fixed flow.
FIX_BR is byte read fixed flow.

Large page 1

offset	Page Read	Page Write with Spare	Spare Write	Spare Read	Page Read with Spare
0	OPCODE_00	OPCODE_80	OPCODE_80	OPCODE_00	OPCODE_00
1	COL_1ST_ADDR	COL_1ST_ADDR	SP_COL_ADDR	SP_COL_ADDR	COL_1ST_ADDR
2	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
3	OPCODE_30	BUFFER1	BUFFER1	OPCODE_30	OPCODE_30
4	GOTO_FIX	WDATA	WR_SP	GOTO_FIX	GOTO_FIX
5	FIX_MRDST	GOTO_FIX	OPCODE_10	FIX_MRDST	FIX_MRDST
6	OPCODE_00	FIX_RDI	GOTO_FIX	OPCODE_00	OPCODE_00
7	BUFFER4	OPCODE_10	FIX_MRDST	BUFFER4	BUFFER4
8	RDATA	GOTO_FIX	GOBACK_8	RD_SP	RDATA
9	BUFFER3	FIX_MRDST	FLW_END	BUFFER3	BUFFER3
10	GOBACK_10	GOBACK_10		GOBACK_10	GOTO_FIX
11	FLW_END	FLW_END		FLW_END	FIX_RDO
12					GOBACK_12
13					FLW_END

Large page 2

offset	Read ID(Byte Mode)	RESET Flash	Block Erase	Blanking Check
0	OPCODE_90	OPCODE_FF	OPCODE_60	OPCODE_80
1	COL_1ST_ADDR	GOTO_FIX	ROW_1ST_ADDR	ZERO_ADDR
2	BUFFER4	FIX_MRDST	OPCODE_D0	ROW_1ST_ADDR
3	RD_SP	FLW_END	GOTO_FIX	BUFFER1
4	BUFFER3		FIX_MRDST	BLK_WR
5	FLW_END		GOBACK_5	OPCODE_10
6			FLW_END	GOTO_FIX
7				FIX_MRDST
8				OPCODE_00
9				ZERO_ADDR
10				ROW_1ST_ADDR
11				OPCODE_30
12				GOTO_FIX
13				FIX_MRDST
14				OPCODE_00
15				BUFFER4
16				BLK_RD
17				BUFFER3
18				GOBACK_18
19				FLW_END

Large page 3

offset	Byte Write	Byte Read	Multi Read Status	Read Status
0	OPCODE_80	OPCODE_00	GOTO_FIX	GOTO_FIX
1	COL_1ST_ADDR	COL_1ST_ADDR	FIX_MRDST	FIX_RDST
2	ROW_1ST_ADDR	ROW_1ST_ADDR	FLW_END	GOBACK_2
3	BUFFER1	OPCODE_30		FLW_END
4	WR_SP	GOTO_FIX		
5	OPCODE_10	FIX_MRDST		
6	GOTO_FIX	OPCODE_00		
7	FIX_MRDST	BUFFER4		
8	FLW_END	RD_SP		
9		BUFFER3		
10		FLW_END		

Small page 1

offset	Small Page Read	Small Spare Read	Small Byte Read_50	Small Byte Read_00	Small Byte Read_01
0	OPCODE_00	OPCODE_50	OPCODE_50	OPCODE_00	OPCODE_01
1	COL_1ST_ADDR	SP_COL_ADDR	COL_1ST_ADDR	COL_1ST_ADDR	COL_1ST_ADDR
2	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
3	BUSY	BUSY	BUSY	BUSY	BUSY
4	BUFFER2	BUFFER2	BUFFER2	BUFFER2	BUFFER2
5	RDATA	RD_SP	RD_SP	RD_SP	RD_SP
6	RD_SP	BUFFER3	BUFFER3	BUFFER3	BUFFER3
7	BUFFER3	BUSY	BUSY	FLW_END	FLW_END
8	BUSY	GOBACK_8	FLW_END		
9	GOBACK_9	FLW_END			
10	FLW_END				

Small page 2

offset	Small Page Write	Small Spare Write	Small Byte Write_50	Small Byte Write_00	Small Byte Write_01
0	OPCODE_00	OPCODE_50	OPCODE_50	OPCODE_00	OPCODE_01
1	OPCODE_80	OPCODE_80	OPCODE_80	OPCODE_80	OPCODE_80
2	COL_1ST_ADDR	SP_COL_ADDR	COL_1ST_ADDR	COL_1ST_ADDR	COL_1ST_ADDR
3	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
4	BUFFER1	BUFFER1	BUFFER1	BUFFER1	BUFFER1
5	WDATA	WR_SP	WR_SP	WR_SP	WR_SP
6	WR_SP	OPCODE_10	OPCODE_10	OPCODE_10	OPCODE_10
7	OPCODE_10	GOTO_FIX	GOTO_FIX	GOTO_FIX	GOTO_FIX
8	GOTO_FIX	FIX_MRDST	FIX_MRDST	FIX_MRDST	FIX_MRDST
9	FIX_MRDST	GOBACK_8	FLW_END	FLW_END	FLW_END
10	GOBACK_9	FLW_END			
11	FLW_END				

Small page 3

offset	Small Copy Back	Small Block Erase	Small Blanking Check
0	OPCODE_00	OPCODE_60	OPCODE_00
1	COL_1ST_ADDR	ROW_1ST_ADDR	OPCODE_80
2	ROW_1ST_ADDR	OPCODE_D0	ZERO_ADDR
3	BUSY	GOTO_FIX	ROW_1ST_ADDR
4	BUFFER2	FIX_MRDST	BUFFER1
5	RDATA	GOBACK_5	BLK_WR
6	BUFFER3	FLW_END	OPCODE_10
7	BUSY		GOTO_FIX
8	OPCODE_80		FIX_MRDST
9	COL_2ND_ADDR		OPCODE_00
10	ROW_2ND_ADDR		ZERO_ADDR
11	BUFFER1		ROW_1ST_ADDR
12	WDATA		BUSY
13	WR_SP		BUFFER2
14	OPCODE_10		BLK_RD
15	GOTO_FIX		BUFFER3
16	FIX_MRDST		BUSY
17	GOBACK_17		GOBACK_17
18	FLW_END		FLW_END

MISC

offset	Get Feature	Set Feature
0	OPCODE_EE	OPCODE_EF
1	ROW_1ST_ADDR	ROW_1ST_ADDR
2	GOTO_FIX	BUFFER1
3	FIX_MRDST	WR_SP
4	OPCODE_00	GOTO_FIX
5	BUFFER4	FIX_MRDST
6	RD_SP	FLW_END
7	BUFFER3	
8	FLW_END	

Toshiba eD3 1

offset	T_SLC Page Read with Spare	T_SLC Spare Read	T_SLC Byte Read
0	OPCODE_A2	OPCODE_A2	OPCODE_A2
1	OPCODE_00	OPCODE_00	OPCODE_00
2	COL_1ST_ADDR	COL_1ST_ADDR	ZERO_ADDR
3	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
4	OPCODE_30	OPCODE_30	OPCODE_30
5	GOTO_FIX	GOTO_FIX	GOTO_FIX
6	FIX_MRDST	FIX_MRDST	FIX_MRDST
7	OPCODE_05	OPCODE_05	OPCODE_05
8	COL_1ST_ADDR	SP_COL_ADDR	COL_1ST_ADDR
9	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
10	OPCODE_E0	OPCODE_E0	OPCODE_E0
11	BUFFER4	BUFFER4	BUFFER4
12	RDATA	RD_SP	RD_SP
13	BUFFER3	BUFFER3	BUFFER3
14	OPCODE_05	FLW_END	FLW_END
15	SP_COL_ADDR		
16	ROW_1ST_ADDR		
17	OPCODE_E0		
18	BUFFER4		
19	RD_SP		
20	BUFFER3		
21	OPCODE_05		
22	ZERO_ADDR		
23	TOG_1ST_ADDR		
24	OPCODE_E0		
25	BUFFER4		
26	RDATA		
27	BUFFER3		
28	OPCODE_05		
29	SP_COL_ADDR		
30	TOG_1ST_ADDR		
31	OPCODE_E0		
32	BUFFER4		
33	RD_SP		
34	BUFFER3		
35	GOBACK_35		
36	FLW_END		

Toshiba eD3 2

offset	T_SLC Page Write	T_SLC Byte Write	T_SLC Erase
0	OPCODE_A2	OPCODE_A2	OPCODE_A2
1	OPCODE_80	OPCODE_80	OPCODE_60
2	COL_1ST_ADDR	COL_1ST_ADDR	ROW_1ST_ADDR
3	ROW_1ST_ADDR	ROW_1ST_ADDR	OPCODE_D0
4	BUFFER1	BUFFER1	GOTO_FIX
5	WDATA	WR_SP	FIX_MRDST
6	OPCODE_85	OPCODE_10	GOBACK_6
7	BUFFER4	GOTO_FIX	FLW_END
8	SP_COL_ADDR	FIX_MRDST	
9	ROW_1ST_ADDR	FLW_END	
10	BUFFER1		
11	WR_SP		
12	OPCODE_11		
13	GOTO_FIX		
14	FIX_MRDST		
15	OPCODE_80		
16	ZERO_ADDR		
17	TOG_1ST_ADDR		
18	BUFFER1		
19	WDATA		
20	OPCODE_85		
21	BUFFER4		
22	SP_COL_ADDR		
23	TOG_1ST_ADDR		
24	BUFFER1		
25	WR_SP		
26	OPCODE_10		
27	GOTO_FIX		
28	FIX_MRDST		
29	GOBACK_29		
30	FLW_END		

Toshiba eD3 3

offset	T_TLC Lower Page Read with Spare	T_TLC Lower Spare Read	T_TLC Middle Page Read with Spar	T_TLC Middle Spare Read
0	OPCODE_01	OPCODE_01	OPCODE_02	OPCODE_02
1	OPCODE_00	OPCODE_00	OPCODE_00	OPCODE_00
2	ZERO_ADDR	ZERO_ADDR	ZERO_ADDR	ZERO_ADDR
3	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
4	OPCODE_30	OPCODE_30	OPCODE_30	OPCODE_30
5	GOTO_FIX	GOTO_FIX	GOTO_FIX	GOTO_FIX
6	FIX_MRDST	FIX_MRDST	FIX_MRDST	FIX_MRDST
7	OPCODE_05	OPCODE_05	OPCODE_05	OPCODE_05
8	COL_1ST_ADDR	SP_COL_ADDR	COL_1ST_ADDR	SP_COL_ADDR
9	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR	ROW_1ST_ADDR
10	OPCODE_E0	OPCODE_E0	OPCODE_E0	OPCODE_E0
11	BUFFER4	BUFFER4	BUFFER4	BUFFER4
12	RDATA	RD_SP	RDATA	RD_SP
13	BUFFER3	BUFFER3	BUFFER3	BUFFER3
14	OPCODE_05	FLW_END	OPCODE_05	FLW_END
15	SP_COL_ADDR		SP_COL_ADDR	
16	ROW_1ST_ADDR		ROW_1ST_ADDR	
17	OPCODE_E0		OPCODE_E0	
18	BUFFER4		BUFFER4	
19	RD_SP		RD_SP	
20	BUFFER3		BUFFER3	
21	OPCODE_05		OPCODE_05	
22	ZERO_ADDR		ZERO_ADDR	
23	TOG_1ST_ADDR		TOG_1ST_ADDR	
24	OPCODE_E0		OPCODE_E0	
25	BUFFER4		BUFFER4	
26	RDATA		RDATA	
27	BUFFER3		BUFFER3	
28	OPCODE_05		OPCODE_05	
29	SP_COL_ADDR		SP_COL_ADDR	
30	TOG_1ST_ADDR		TOG_1ST_ADDR	
31	OPCODE_E0		OPCODE_E0	
32	BUFFER4		BUFFER4	
33	RD_SP		RD_SP	
34	BUFFER3		BUFFER3	
35	FLW_END		FLW_END	

Toshiba eD3 4

offset	T_TLC Upper Page Read with Spare	T_TLC Upper Spare Read	T_TLC 3SLC copy TLC
0	OPCODE_03	OPCODE_03	GOTO_FIX
1	OPCODE_00	OPCODE_00	FIX_TLC_CP_LO
2	ZERO_ADDR	ZERO_ADDR	GOTO_FIX
3	ROW_1ST_ADDR	ROW_1ST_ADDR	FIX_TLC_CP_MI
4	OPCODE_30	OPCODE_30	GOTO_FIX
5	GOTO_FIX	GOTO_FIX	FIX_TLC_CP_UP
6	FIX_MRDST	FIX_MRDST	GOBACK_6
7	OPCODE_05	OPCODE_05	FLW_END
8	COL_1ST_ADDR	SP_COL_ADDR	
9	ROW_1ST_ADDR	ROW_1ST_ADDR	
10	OPCODE_E0	OPCODE_E0	
11	BUFFER4	BUFFER4	
12	RDATA	RD_SP	
13	BUFFER3	BUFFER3	
14	OPCODE_05	FLW_END	
15	SP_COL_ADDR		
16	ROW_1ST_ADDR		
17	OPCODE_E0		
18	BUFFER4		
19	RD_SP		
20	BUFFER3		
21	OPCODE_05		
22	ZERO_ADDR		
23	TOG_1ST_ADDR		
24	OPCODE_E0		
25	BUFFER4		
26	RDATA		
27	BUFFER3		
28	OPCODE_05		
29	SP_COL_ADDR		
30	TOG_1ST_ADDR		
31	OPCODE_E0		
32	BUFFER4		
33	RD_SP		
34	BUFFER3		
35	FLW_END		

Samsung eD3 1

offset	S_SLC Page Read with Spare	S_SLC Spare Read	S_SLC Byte Read	S_SLC Page Write with Spar	S_SLC Byte Write
0	OPCODE_DA	OPCODE_DA	OPCODE_DA	OPCODE_DA	OPCODE_DA
1	GOTO_FIX	GOTO_FIX	GOTO_FIX	GOTO_FIX	GOTO_FIX
2	FIX_PRS	FIX_SR	FIX_BR	FIX_PWS	FIX_BW

Samsung eD3 2

offset	S_SLC Erase	S_TLC Erase	S_TLC Read with spare	S_TLC Spare Read	S_TLC 3SLC copy TLC
0	OPCODE_DA	OPCODE_DF	OPCODE_DF	OPCODE_DF	OPCODE_DA
1	GOTO_FIX	GOTO_FIX	GOTO_FIX	GOTO_FIX	OPCODE_00
2	FIX_ER	FIX_ER	FIX_PRS	FIX_SR	ZERO_ADDR
3					ROW_1ST_ADDR
4					OPCODE_35
5					GOTO_FIX
6					FIX_MRDST
7					OPCODE_DF
8					OPCODE_85
9					ZERO_ADDR
10					TARG_1ST_ADDR
11					OPCODE_C0
12					VAR_ADDR1
13					GOTO_FIX
14					FIX_MRDST
15					OPCODE_DA
16					OPCODE_00
17					ZERO_ADDR
18					ROW_2ND_ADDR
19					OPCODE_35
20					GOTO_FIX
21					FIX_MRDST
22					OPCODE_DF
23					OPCODE_85
24					ZERO_ADDR
25					TARG_2ND_ADDR
26					OPCODE_C0
27					VAR_ADDR2
28					GOTO_FIX
29					FIX_MRDST
30					OPCODE_DA
31					OPCODE_00
32					ZERO_ADDR
33					ROW_3RD_ADDR
34					OPCODE_35
35					GOTO_FIX
36					FIX_MRDST
37					OPCODE_DF
38					OPCODE_85
39					ZERO_ADDR
40					TARG_3RD_ADDR
41					OPCODE_C0
42					VAR_ADDR3
43					GOTO_FIX
44					FIX_MRDST
45					OPCODE_DF
46					OPCODE_8B
47					ZERO_ADDR
48					ORDER_ADDR
49					OPCODE_10
50					GOTO_FIX
51					FIX_MRDST
52					GOBACK_52
53					FLW_END

Chapter 6

Initialization/Application

This chapter contains the following sections:

- 6.1 Initial Steps
- 6.2 Command Queue Access Method
- 6.3 Selection Restriction of BMC Region

6.1 Initial Steps

Although FTNANDC024 does not require specific initialization flow, a number of points are required to be checked before accessing Flash:

- Enable ECC
- Enable the data inverting and scramble functions
- The AC timing must be correctly set based on the current NAND clock speed and requirement of Flash.
- The memory attribute must be correctly set based on the specification of Flash.

6.2 Command Queue Access Method

FTNANDC024 provides a command queue of each channel. Each Flash can be accessed by issuing a command into the command queue. The 1st word to the 3rd word and 5th, 6th word can be arbitrarily changed before writing the 4th word. After writing the 4th word, a command will be pushed into the command queue.

Please note the following considerations:

- Always check the command status (Should not be full) before preparing a new command (Preparing a new command that includes writing from the 1st word to the 3rd word and 5th, 6th word)
- By using the general command queue, the command content will be the same to be written to all the command queues. With the possibility of leading to different NAND channels mapped to the same BMC region, the general command queue should not be used by a data read/write command that needed to be pushed into the command queue.

6.3 Selection Restriction of BMC Region

Following are two restrictions for selecting the BMC region:

- From the NAND channel side, it is not allowed to assign two NAND channels to set one BMC region as the target in the same period.
- From the AHB side, it is not allowed to let other AHB master access one BMC region before the current AHB master finishes all data transfers.