32-bit RISC Microcontroller

TXZ+ Family

Reference Manual
Serial Memory Interface (SMIF-B)

Revision 1.3

2023-06

TOSHIBA ELECTRONIC DEVICES & STORAGE CORPORATION

Contents

List of Figures

List of Tables

Preface

Related document

Document name
Clock Control and Operation Mode
Exception
Input/Output Ports
Product Information

Conventions

- Numeric formats follow the rules as shown below:
  Hexadecimal: 0xABC
  Decimal: 123 or 0d123 – Only when it needs to be explicitly shown that they are decimal numbers.
  Binary: 0b111 – It is possible to omit the "0b" when the number of bit can be distinctly understood from a sentence.
- "_N" is added to the end of signal names to indicate low active signals.
- It is called "assert" that a signal moves to its active level, "deassert" to its inactive level.
- When two or more signal names are referred, they are described like as [m: n].
  Example: S[3:0] shows four signal names S3, S2, S1 and S0 together.
- The characters surrounded by ***[ ]*** defines the register.
  Example: ***[ABCD]***
- "n" substitutes suffix number of two or more same kind of registers, fields, and bit names.
  Example: ***[XYZ1], [XYZ2], [XYZ3] → [XYZn]***
- "x" substitutes suffix number or character of units and channels in the Register List.

  In case of unit, "x" means A, B, and C . . .
  Example: ***[ADACR0], [ADBCR0], [ADCCR0] → [ADxCR0]***
  In case of channel, "x" means 0, 1, and 2 . . .
  Example: [***T32A0RUNA], [T32A1RUNA], [T32A2RUNA] → [T32AxRUNA]***
- The bit range of a register is written like as [m: n].
  Example: Bit[3:0] expresses the range of bit 3 to 0.
- The configuration value of a register is expressed by either the hexadecimal number or the binary number.
  Example: ***[ABCD]***<EFG> = 0x01 (hexadecimal), ***[XYZn]***<VW> = 1 (binary)
- Word and Byte represent the following bit length.
  Byte: 8 bits
  Half word: 16 bits
  Word: 32 bits
  Double word: 64 bits
- Properties of each bit in a register are expressed as follows:
  R: Read only
  W: Write only
  R/W: Read and Write are possible
- Unless otherwise specified, register access supports only word access.
- The register defined as reserved must not be rewritten. Moreover, do not use the read value.
- The value read from the bit having default value of "-" is unknown.
- When a register containing both of writable bits and read-only bits is written, read-only bits should be written with their default value, In the cases that default is "-", follow the definition of each register.
- Reserved bits of the Write-only register should be written with their default value. In the cases that default is "-", follow the definition of each register.
- Do not use read-modified-write processing to the register of a definition which is different by writing and read out.

All other company names, product names, and service names mentioned herein may be trademarks of their respective companies.

Terms and Abbreviation

Some of abbreviations used in this document are as follows:

SI	Serial Input
SO	Serial Output
SPI	Serial Peripheral Interface
SMIF	Serial Memory Interface

1. Outlines

The serial memory interface (SMIF) connects to the device which has serial I/O's or multi I/O communication interface device.(SPI Flash memory and others). This manual mainly describes serial memory.

The list of the functions of SMIF is shown in the following table.

Function category	Function	Description
Connection to a serial memory	Connection	- 2 serial memories at maximum can be connected.
	Memory capacity	- 64 KB to 128 MB
	Transfer clock	- 25 MHz at maximum
	Communication Mode	- Communication Mode STR-SPI(Standard SPI compatible) STR-Quad STR-QPI STR-Octal STR-OPI - MSB first - SPI Mode 0 support - Number of address bytes in command with address 2, 3, 4 bytes (Note) Communication using data strobe or data mask signal is not supported
	Memory mapping	- Mapping addresses: 0xA0000000 to 0xA7FFFFFF.
	Access mode	- Direct access - Indirect access
	Command transfer count	- Up to 288 Bytes can be transferred through a register.
	Chip select	- Selection from Serial memory 0 and Serial memory 1. - Deassertion times of SMIxCS0_N and SMIxCS1_N can be set.
	Other functions	- A function to automatically polling the "WIP" field of the status register of the SPI Flash memory before direct access and wait until the "Idle" state. - The number of dummy bytes can be adjusted from 0 to 31 byte.

2. Configuration

The block diagram and the list of the signals of SMIF are shown as follows:

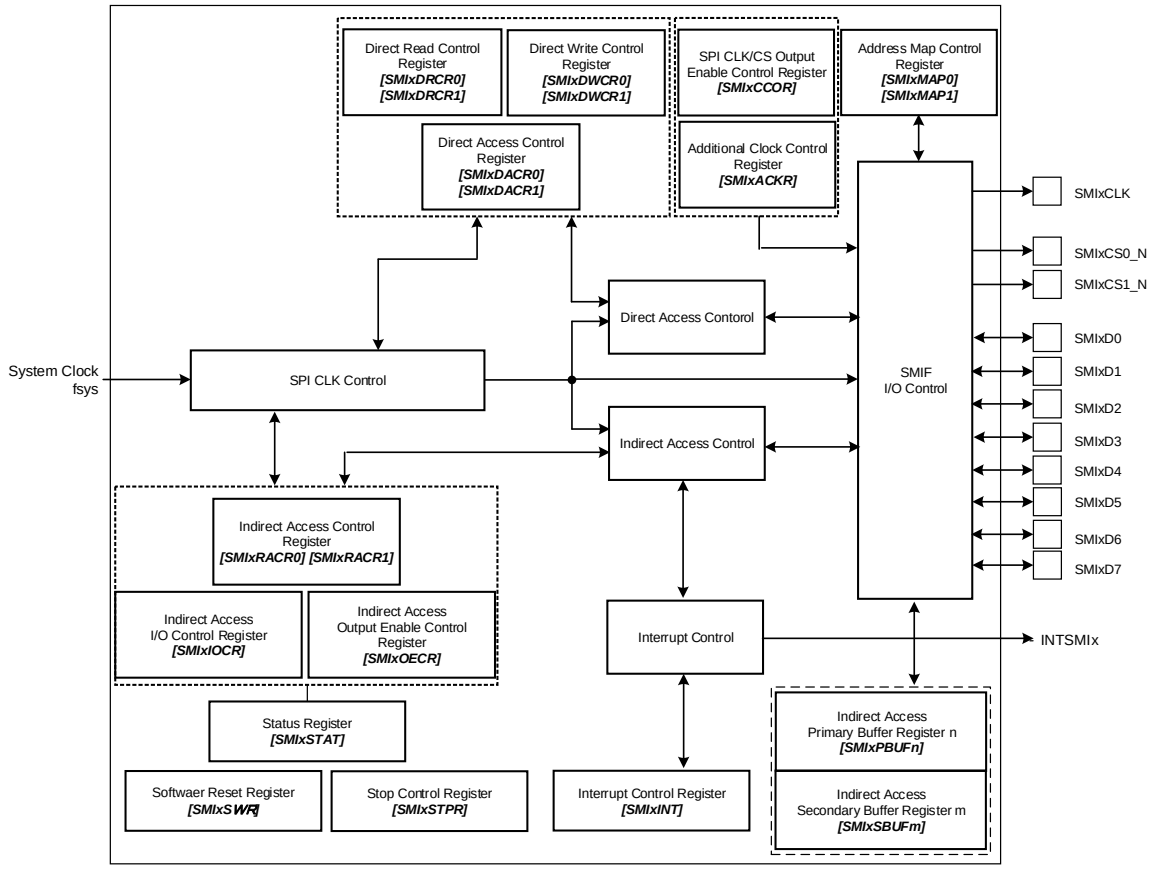

Figure 2.1 Block diagram of SMIF

Table 2.1 List of Signals

No	symbol	Signal name	I/O	Reference manual
1	f_{sys}	System clock	Input	Clock Control and Operation Mode
2	SMIxCLK	Access clock	Output	Product Information, Input/Output ports
3	SMIxCS0_N	Chip select 0	Output	Product Information, Input/Output ports
4	SMIxCS1_N	Chip select 1	Output	Product Information, Input/Output ports
5	SMIxD0	Data input and output	I/O	Product Information, Input/Output ports
6	SMIxD1	Data input and output	I/O	Product Information, Input/Output ports
7	SMIxD2	Data input and output	I/O	Product Information, Input/Output ports
8	SMIxD3	Data input and output	I/O	Product Information, Input/Output ports
9	SMIxD4	Data input and output	I/O	Product Information, Input/Output ports
10	SMIxD5	Data input and output	I/O	Product Information, Input/Output ports
11	SMIxD6	Data input and output	I/O	Product Information, Input/Output ports
12	SMIxD7	Data input and output	I/O	Product Information, Input/Output ports
13	INTSMIx	Interrupt	Output	Exception

3. Function and Operation

SMIF can connect to two serial memories at maximum. Each memory capacity of 64 KB to 128MB is available. The memory can be accessed by "Direct access" to read/write data by address specification and “Indirect access” which issues a command using a program register.

3.1. Clock Supply

When SMIF is used, the corresponding clock enable bits should be set to “1” (Clock supply) in fsys supply stop register A (***[CGFSYSENA], [CGFSYSMENA]***), fsys supply stop register B (***[CGFSYSENB], [CGFSYSMENB]***), and fsys supply stop register C (***[CGFSYSMENC]),***and fc supply stop register (***[CGFCEN]***). The corresponding registers and the bit location depend on a product. Some products do not have all registers. For the details, refer to reference manual “Clock control and operation mode”.
When the clock supply is stopped or the status is transited to STOP1/STOP2 mode, it should be checked that SMIF stops.

3.2. Communication Mode

The communication format between SMIF and a serial memory is STR-SPI(Standard SPI compatible),STR-Quad, STR-QPI, STR-Octal, STR-OPI Read/Write are supported.

The serial memory should satisfy the following conditions.

- Capacity: 64 KB to 128 MB.
- SPI Mode 0 is supported.
- The unused upper addresses are “don’t care”.
- The bit 0 of the status register is the bit of “Write In Progress (WIP)”.
- MSB first

3.3. Memory Mapping

The addresses to access a serial memory can be mapped to any region in “0xA0000000” to “0xA7FFFFFF” (128 MB). If the area which is not mapped is read, an unknown value returns. After the reset assertion, the serial memory 0 is memory-mapped, but the serial memory 1 is not mapped. When using serial memory 1, it is necessary to set the memory mapping. If the serial memory whose capacity is smaller than the mapped region is implemented, a mirror region is read in the serial memory when the unimplemented region is accessed.

Figure 3.1 shows an example that the mapping region of the serial memory 0 is “0xA0000000” to “0xA0BFFFFF” (12 MB) and the mapping region of the serial memory 1 is “0xA0C00000” to “0xA0FFFFFF” (4 MB), and a 4-MB serial memory is connected to each region, respectively.

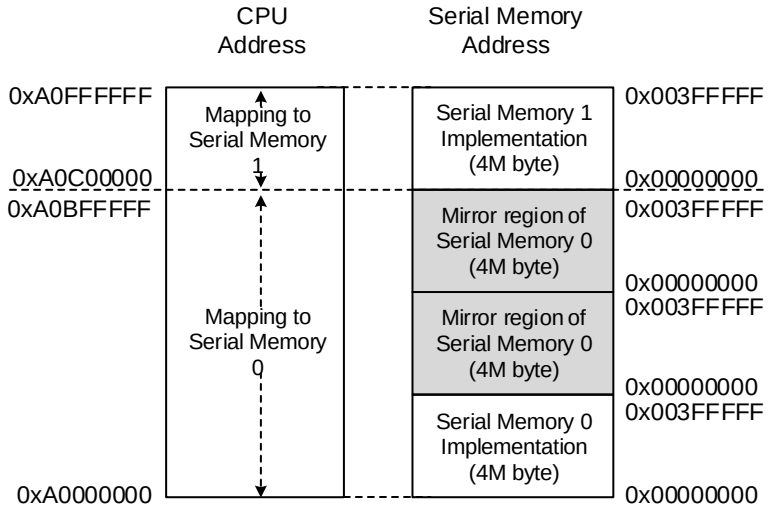

Figure 3.1 Example of memory mapping

3.4. Access Mode

There are two access modes to access a serial memory, “Direct access mode” which reads/writes the data using specified addresses and “Indirect access mode” which issues a command using the control of a program register.

3.4.1. Direct Access Mode

The serial memory can be directly read/write by accessing addresses “0xA0000000” to “0xA7FFFFFF”. This is “Direct access”. When a read to an address in the range of “0xA0000000” to “0xA7FFFFFF” is detected, a read/write command is issued to SPI Flash memory.

3.4.1.1. SPI Flash Command

The initial command which is issued in the direct access mode is Fast Read (Op-code = 0x0B). The command can be replaced to one of the following multi I/O commands. But the supported commands depend on the connected SPI Flash memory. The following shows typical access modes and their timing charts.

- STR-SPI
- STR-Quad
- STR-QPI
- STR-Octal
- STR-OPI

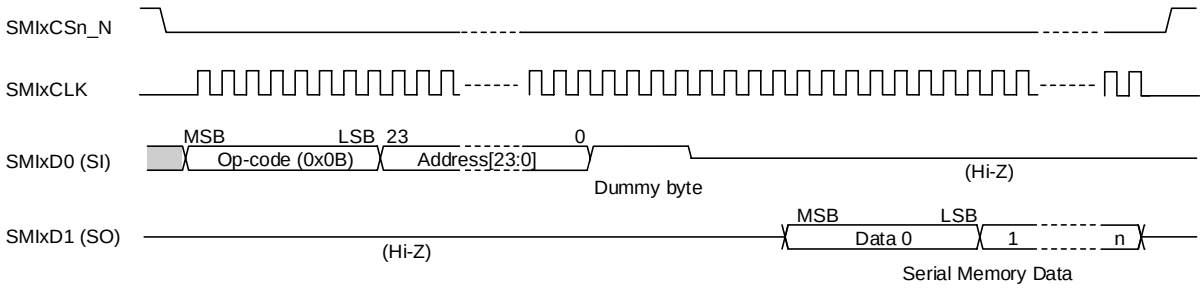

Figure 3.2 STR-SPI: Fast Read

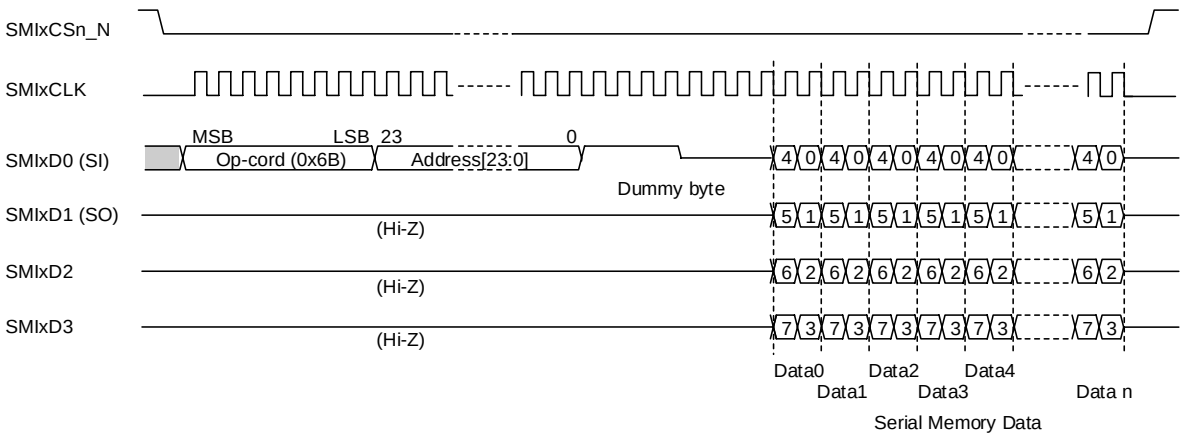

Figure 3.3 STR-Quad: Fast Read Quad output

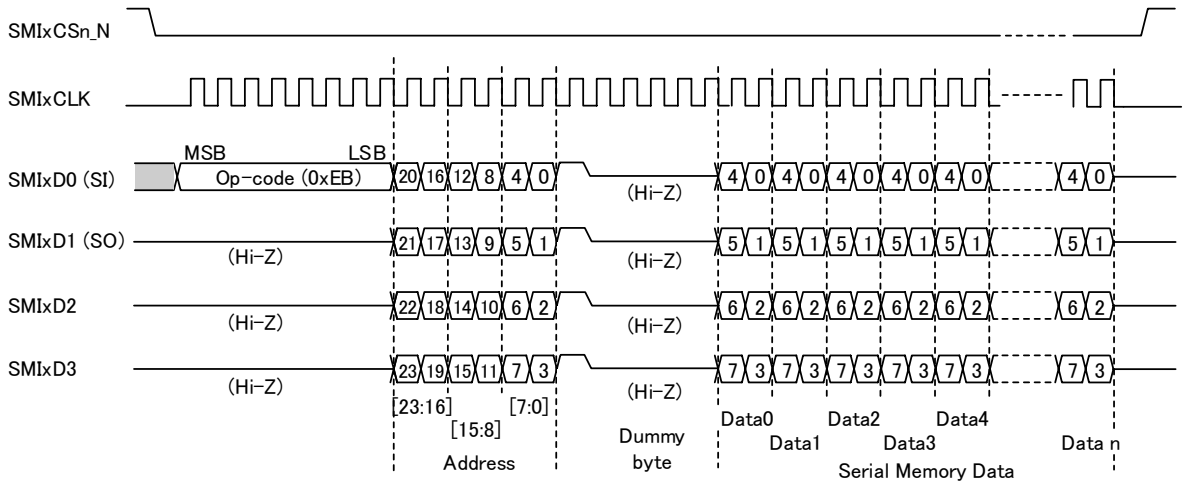

Figure 3.4 STR-Quad: Fast Read Quad I/O

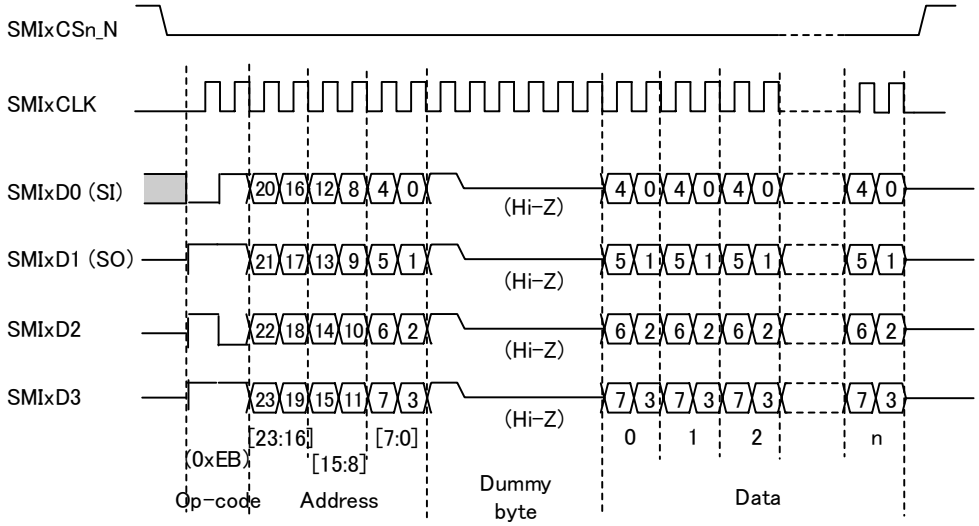

Figure 3.5 STR-QPI: Fast Read

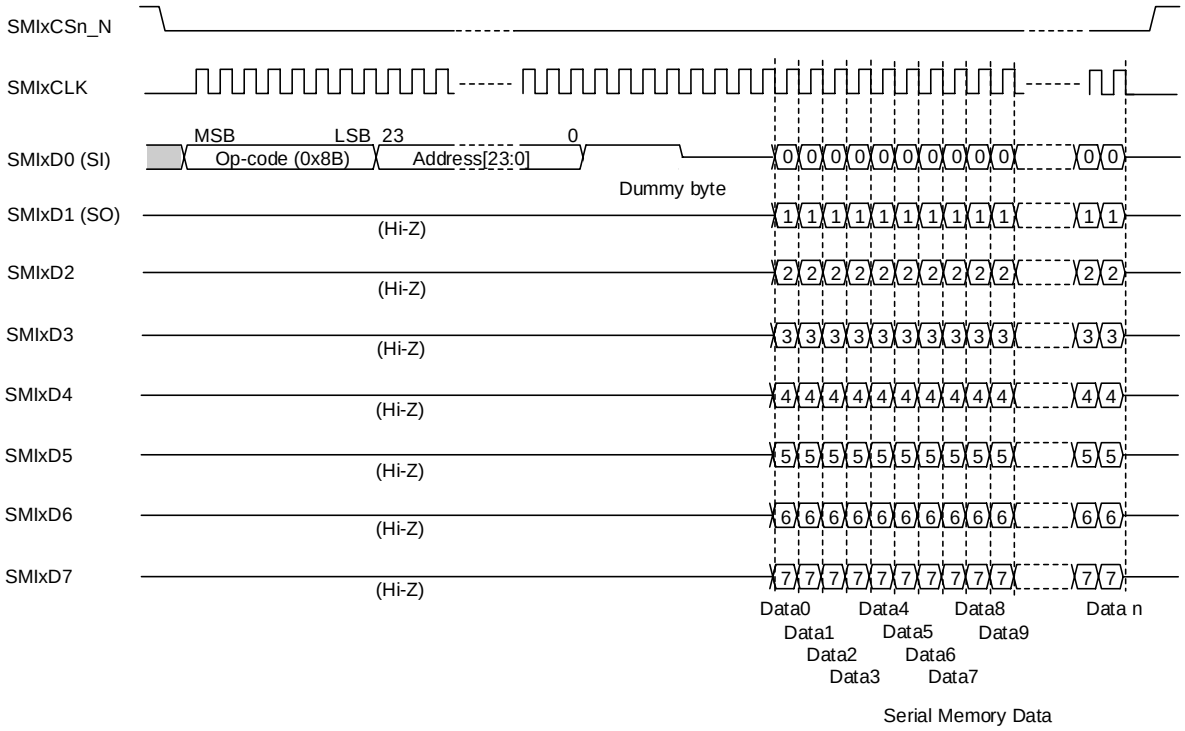

Figure 3.6 STR-Octal: Fast Read Octal Output

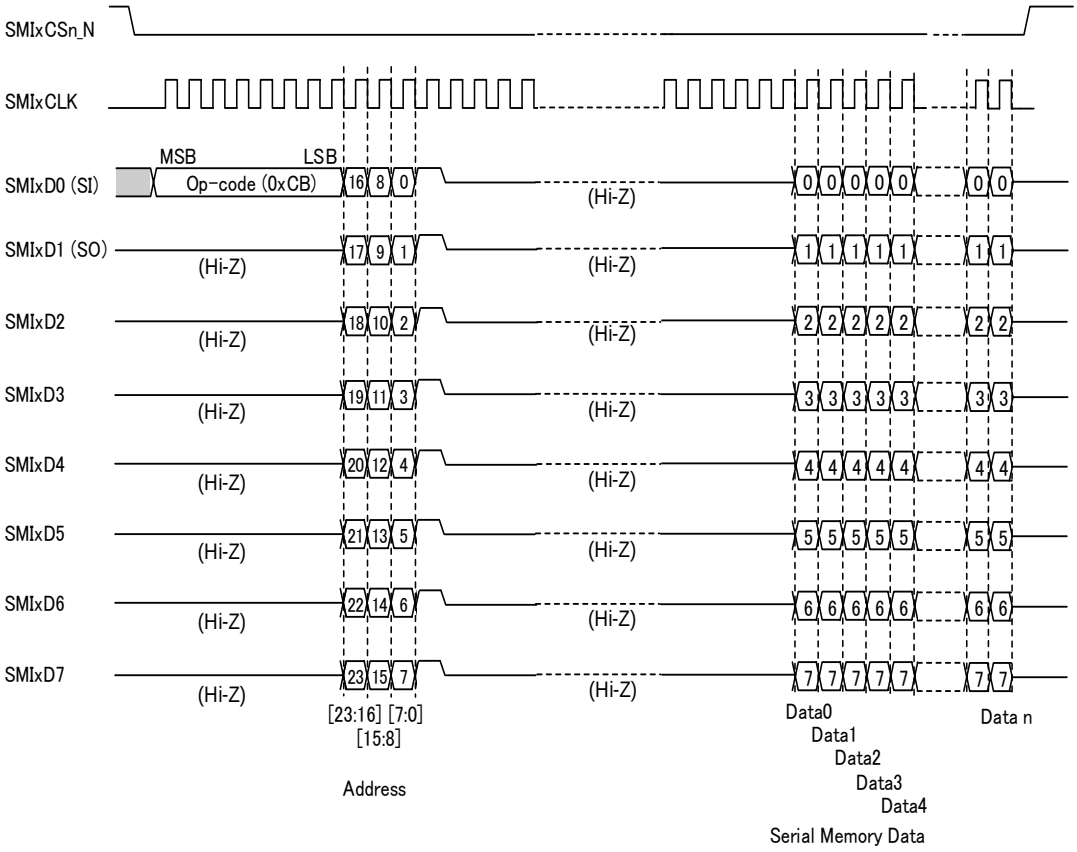

Figure 3.7 STR-Octal: Fast Read Octal I/O

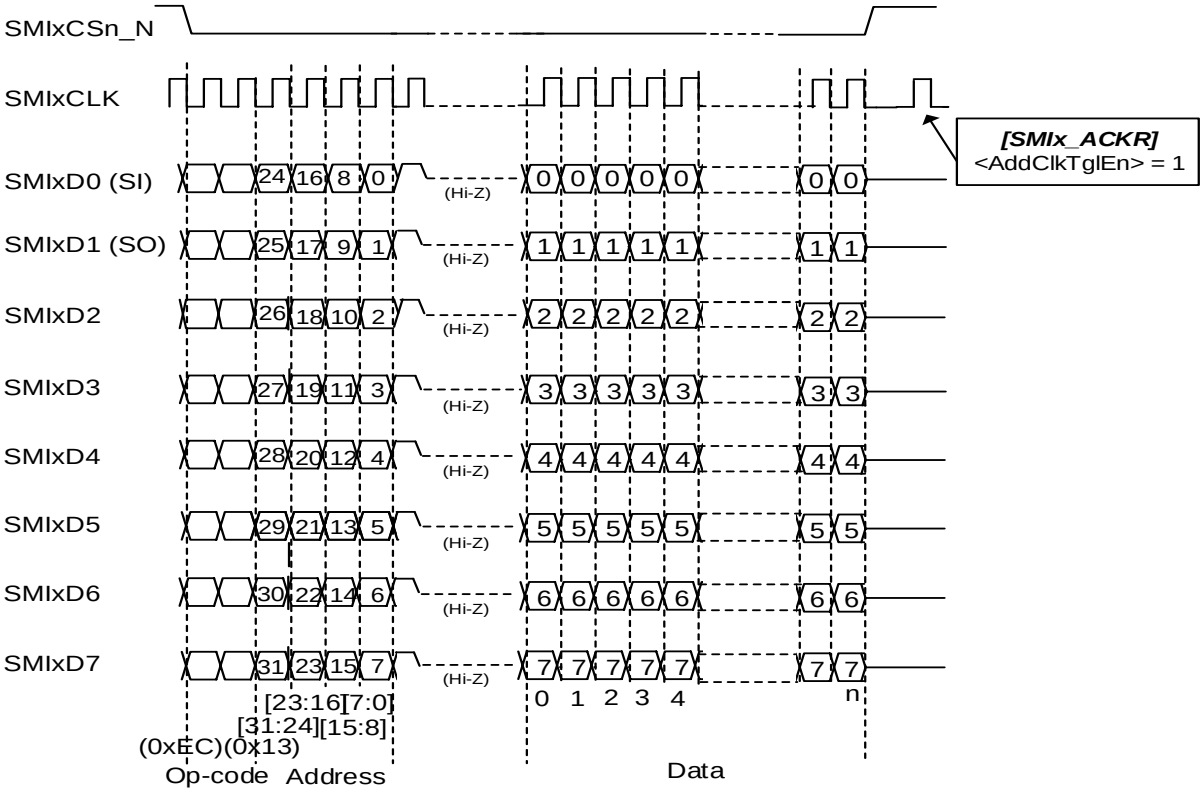

Figure 3.8 STR-OPI: Fast Read

3.4.1.2. Polling of WIP bit

This function is mainly used for debugging.

Generally, SPI Flash memory cannot receive Read command during Write or Erase operation. So, when SMIF issues Read command to SPI Flash memory, the memory does not respond during Write or Erase operation. As a result, invalid data returns. Software should control this access not to be done.

This function enables to receive a correct data even when SPI Flash memory is read during Write or Erase operation. If a read is detected while this function is enabled, SMIF polls WIP bit in Status register in SPI Flash memory until it becomes "0". Then, the Read command is issued. This makes the correct data received. ***[SMIxDACRn]***<PollWIP> should be set to "1" to enable this function.

When the operation starts after the reset deassertion, WIP bit in Status register in SPI Flash memory should be polled to confirm that the state is neither Write nor Erase one. The operation is controlled by the value in ***[SMIxDACRn]***<PollWIP>.

When this function is enabled, the overhead of a read becomes bigger because of the polling per read. And, if the polling result does not return or a timeout error occurs, the operation of SMIF is not guaranteed.

Do not enable this function if no SPI memory is connected or if an SPI memory such as SPI RAM that does not have a WIP bit is connected. In these cases, the hang-up may occur if the read result of the WIP bit is always "1" (e.g.: Pull-up connection to SMIxD1 without SPI memory).

3.4.1.3. Setting Procedure of Direct Access Mode

1. The base address and the capacity are set in ***[SMIxMAP0], [SMIxMAP1]*** according to the implemented serial memory.
2. The transfer clock, CS deassertion time, and WIP polling operation are set in ***[SMIxDACR0], [SMIxDACR1]***.
3. The command op-code, the dummy byte count, and the input/output control are set in ***[SMIxDRCR0], [SMIxDRCR1], [SMIxDWCR0], [SMIxDWCR1]***.
4. Address in the mapping region is read/write in the serial memory.

Note: Immediately after the reset deassertion, the serial memory 0 is mapped to addresses "0xA0000000" to "0xA0FFFFFF" (16 MB space).

3.4.2. Indirect Access

Indirect access issues commands such as Page Program, Erase, Full Chip Erase, Status Read, and Read to SPI Flash memory through registers. Up to 288 bytes commands can be issued to SPI Flash memory using 32 bytes primary buffer and 256 bytes secondary buffer. The setting of indirect access mode is set in ***[SMIxRACR0]*** and ***[SMIxRACR1]*** for the serial memory 0 and the serial memory 1, respectively.

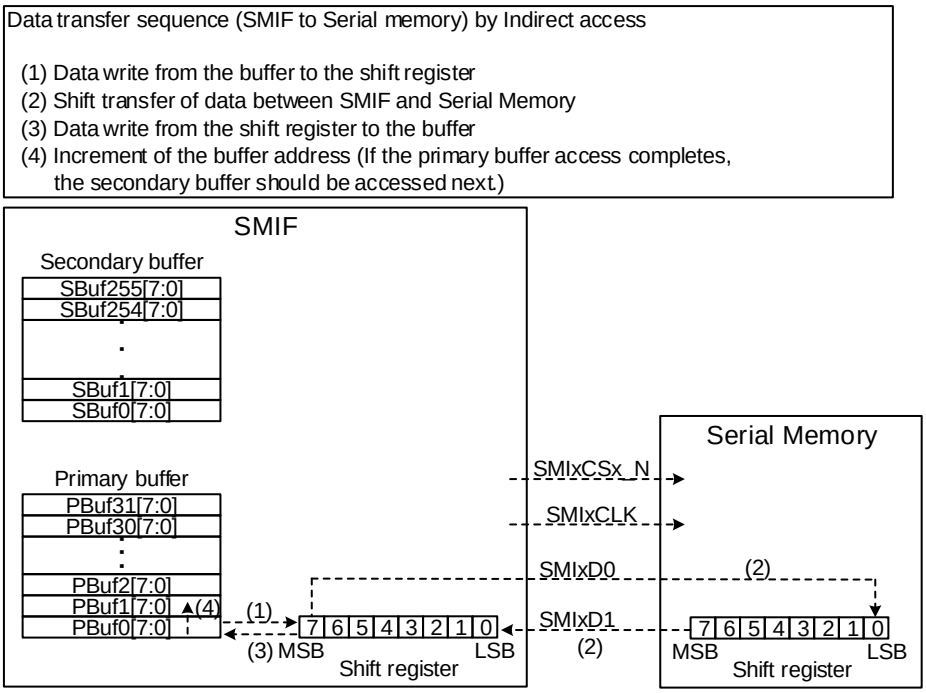

Figure 3.9 Indirect access

3.4.3. Setting Procedure of indirect Access Mode

1. No execution of the indirect access should be checked by reading ***[SMIxSTAT]***<CycProg>.

2. The serial memory which should be accessed, the transfer byte count, and others should be set in ***[SMIxRACR0]***, ***[SMIxRACR1]***, and ***[SMIxINT]***.

3. The command data should be set in the buffers ***[SMIxPBUFn]*** and ***[SMIxSBUFm]***.
   (n = 0 to 7, m = 0 to 63)

4. When ***[SMIxRACR1]***<CycGo> is set in "1", the commands are issued successively.

Note1: When using a command with reception in multi I/O communication, more clocks than the number of transfer data are output, but the operation does not effect.
Note2: When using send only commands in multi-I/O communication, use both primary buffer and secondary buffer.

3.5. Transfer Clock

The frequency of the transfer clock (SMIxCLK) is determined by the value in ***[SMIxDACR0]***<SPR[4:0]> , ***[SMIxDACR1]***<SPR[4:0]>. ***[SMIxRACR0]***<SPR[4:0]> <SPR[4:0]>.

SMIxCLK = fsys frequency / (<SPR[4:0]> + 1)

The frequency of the transfer clock is determined by a dividing value as shown in the following table.

Table 3.1 Transfer clock

<SPR[4:0]>	1	2	3	4	5	6	7	8	9	10	11	12	・・・	28	29	30	31
Dividing / fsys [MHz]	2	3	4	5	6	7	8	9	10	11	12	13	・・・	29	30	31	32
200	100	66.6	50	40	33.3	28.5	25	22.2	20	18.1	16.6	15.3		6.89	6.66	6.45	6.25
160	80	53.3	40.0	32.0	26.7	22.9	20.0	17.8	16.0	14.5	13.3	12.3		5.52	5.33	5.16	5.00
140	70	46.7	35.0	28.0	23.3	20.0	17.5	15.6	14.0	12.7	11.7	10.8		4.83	4.67	4.52	4.38
120	60	40.0	30.0	24.0	20.0	17.1	15.0	13.3	12.0	10.9	10.0	9.2		4.14	4.00	3.87	3.75
100	50	33.3	25.0	20.0	16.7	14.3	12.5	11.1	10.0	9.1	8.3	7.7	・・・	3.45	3.33	3.23	3.13
80	40	26.7	20.0	16.0	13.3	11.4	10.0	8.9	8.0	7.3	6.7	6.2		2.76	2.67	2.58	2.50
60	30	20.0	15.0	12.0	10.0	8.6	7.5	6.7	6.0	5.5	5.0	4.6		2.07	2.00	1.94	1.88
40	20	13.3	10.0	8.0	6.7	5.7	5.0	4.4	4.0	3.6	3.3	3.1		1.38	1.33	1.29	1.25
20	10	6.7	5.0	4.0	3.3	2.9	2.5	2.2	2.0	1.8	1.7	1.5		0.69	0.67	0.65	0.63

Note1: The transfer clock should be 25 MHz or less. The combinations of the fsys frequency and the division value in gray-colored of Table 3.1 should not be set.
Note2: When the dividing value is an odd number, the duty of the transfer clock is not 50 %.
Low width = (Dividing value)/2 + 0.5, and High width = (Dividing value)/2 - 0.5.
Note3: The dividing value is “2” when <SPR[4:0]>=00000.

3.6. Data Input and Output Timing

The data input or output is synchronous with the falling edge of SMIxCLK.

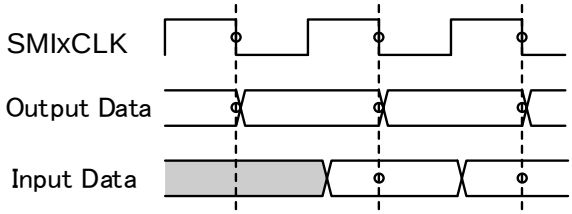

Figure 3.10 Data output and input timing

3.7. Interrupt

When the SPI cycle controlled by the indirect access completes, ***[SMIxSTAT]***<CycDone> is set to “1”. At the same time, the interrupt (INTSMIx) can be generated. The setting of ***[SMIxINT]***<SCDIntEn>=1 enables the interrupt generation.

In order to clear the generated interrupt, ***[SMIxSTAT]***<CycDone> should be set to “0”.

When the SPI cycle controlled by the Stop processing completes, ***[SMIxSTAT]***<StpProgDone> is set to “1”. At the same time, the interrupt (INTSMIx) can be generated. The setting of ***[SMIxINT]***<SDIntEn>=1 enables the interrupt generation.

In order to clear the generated interrupt, ***[SMIxSTAT]***<StpProgDone> should be set to “0”.

4. Registers

4.1. List of Registers

The control registers and their addresses are shown in the following tables.

Peripheral function		Channel/Unit	Base address (Base)	
			TYPE1	TYPE2
Serial memory interface	SMIF	ch 0	0x4000C000	-

Note: The channel/unit and base address type are different by products. Please refer to reference manual "Product Information" for details.

Register name		Address (Base+)
Address Map Control Register 0	***[SMIxMAP0]***	0x0000
Address Map Control Register 1	***[SMIxMAP1]***	0x0004
Direct Access Control Register 0	***[SMIxDACR0]***	0x0008
Direct Access Control Register 1	***[SMIxDACR1]***	0x000C
Direct Read Control Register 0	***[SMIxDRCR0]***	0x0010
Direct Read Control Register 1	***[SMIxDRCR1]***	0x0014
Direct Write Control Register 0	***[SMIxDRCR0]***	0x0018
Direct Write Control Register 1	***[SMIxDRCR1]***	0x001C
Indirect Access Control Register 0	***[SMIxRACR0]***	0x0400
Indirect Access Control Register 1	***[SMIxRACR1]***	0x0404
Indirect Access I/O Control Register 0	***[SMIxIOCR]***	0x0408
Indirect Access Output Enable Control Register 0	***[SMIxOECR]***	0x040C
Interrupt Control Register	***[SMIxINT]***	0x0440
Status Register	***[SMIxSTAT]***	0x0444
Software Reset Register	***[SMIxSWR]***	0x0480
Additional Clock Control Register	***[SMIxACKR]***	0x0484
SMIxCLK/CS Output Enable Control Register	***[SMIxCCOR]***	0x0488
Stop Control Register	***[SMIxSTPR]***	0x048C
Indirect Access Primary Buffer Register n	***[SMIxPBUFn] (n = 0 to 7)***	0x0500 to 0x051F
Indirect Access Secondary Buffer Register m	***[SMIxSBUFm] (m = 0 to 63)***	0x0600 to 0x06FF

4.2. Details of Registers

4.2.1. *[SMIxMAP0]* (Address Map Control Register 0)

Bit	Bit Symbol	After Reset	Type	Description
31:28	-	0	R	Read as “0”
27:16	FBA[11:0]	0x000	R/W	Mapping base address of Serial memory 0. (Note1) (Note2) 0x000 to 0x7FF: Base address <FBA[11:0]> upper base address of Serial memory 0.The lower address is set to “0x0000”. (“0xA0000000” to “0xA7FF0000”)
15:6	-	0	R	Read as “0"
5:2	FDEN[3:0]	1000	R/W	Capacity of Serial memory 0 (Note2) 0000: 64KB 0101: 2MB 1010: 64MB 0001: 128KB 0110: 4MB 1011:128MB 0010: 256KB 0111: 8MB 1100 to 1111: Reserved 0011: 512KB 1000: 16MB 0100: 1MB 1001: 32MB
1	WE	0	R/W	Write Enable 0: Disabled 1: Enabled
0	RE	1	R/W	Read Enable 0: Disabled 1: Enabled

Note1: The value of <FBA[11:0]> should be aligned with the value set by <FDEN[3:0]>.

Note2: The address regions of Serial memory 0 and Serial memory 1 cannot overlap.

4.2.2. *[SMIxMAP1]* (Address Map Control Register 1)

Bit	Bit Symbol	After Reset	Type	Description
31:28	-	0	R	Read as “0”
27:16	FBA[11:0]	0x000	R/W	Mapping base address of Serial memory 1. (Note1) (Note2) 0x000 to 0x7FF: Base address <FBA[11:0]> upper base address of Serial memory 1.The lower address is set to “0x0000”. (“0xA0000000” to “0xA7FF0000”)
15:6	-	0	R	Read as “0"
5:2	FDEN[3:0]	1000	R/W	Capacity of Serial memory 1 (Note2) 0000: 64KB 0101: 2MB 1010: 64MB 0001: 128KB 0110: 4MB 1011:128MB 0010: 256KB 0111: 8MB 1100 to 1111: Reserved 0011: 512KB 1000: 16MB 0100: 1MB 1001: 32MB
1	WE-	0	R/W	Write Enable 0: Disabled 1: Enabled
0	RE	1	R/W	Read Enable 0: Disabled 1: Enabled

Note1: The value of <FBA[15:0]> should be aligned with the value set by <FDEN[3:0]>.

Note2: The address regions of Serial memory 0 and Serial memory 1 cannot overlap.

4.2.3. *[SMIxDACRn]* (Direct Access Control Register n) (n=0,1)

Bit	Bit Symbol	After Reset	Type	Description
31:21	-	0	R	Read as “0”
20:16	SPR[4:0]	11111	R/W	Transfer clock of Serial memory n 00000: 2 divide 00001: 2 divide 00010: 3 divide 00011: 4 divide : 11110: 31 divide 11111: 32 divide
15:8	SCSD[7:0]	0x00	R/W	SMIxCSn_N deassertion time of Serial memory n 0x00 to 0xFF: Deassertion time Deassertion time = fsys cycle time × <SCSD[7:0]> (Note1)
7	-	0	R	Read as “0”
6	PollWIP	0	R/W	WIP polling before Read command is issued to Serial memory n 0: Disabled 1: Enabled For the details, refer to Section “3.4.1.2 Polling of WIP bit”.
5:4	SDCE[1:0]	00	R/W	Write "01" (Note 2)
3	-	0	R	Read as “0”
2		0	R/W	Write “0”
1:0	-	0	R	Read as “0”

Note1: The deassertion time should be set from 0 ns to (fsys cycle time × 255)ns.
Note2: The <SDCE[1:0]> field becomes “00” after the reset. The field should be written to “01” in the initialization setting.

4.2.4. *[SMIxDRCRn]* (Direct Read Control Register n) (n = 0,1)

Bit	Bit Symbol	After Reset	Type	Description
31:24	CmdOp 1[7:0]	0x00	R/W	Op-code1 of SPI command of Serial memory n Op-code of SPI command should be set.
23:16	CmdOp 0[7:0]	0x0B	R/W	Op-code 0 of SPI command of Serial memory n Op-code of SPI command should be set.
15:11	DmyBc[4:0]	00001	R/W	SPI dummy byte count of Serial memory n SPI dummy byte count (“0” to “31”) should be set.
10:9	AddrBc[1:0]	00	R/W	SPI Address byte Count 00: 3 bytes 01: 2 bytes 10: 4 bytes 11: Reserved
8	CmdBc	0	R/WR	SPI Command byte Count 0: 1 byte 1: 2 bytes
7:6	DatIO[1:0]	00	R/W	SPI data input and output control of Serial memory n 00: Single 10: Quad 01: Reserved 11: Octal
5:4	DmyIO[1:0]	00	R/W	SPI dummy input and output control of Serial memory n 00: Single 10: Quad 01: Reserved 11: Octal
3:2	AdrIO[1:0]	00	R/W	SPI address input and output control of Serial memory n 00: Single 10: Quad 01: Reserved 11: Octal
1:0	CmdIO[1:0]	00	R/W	SPI command input and output control of Serial memory n 00: Single 10: Quad 01: Reserved 11: Octal

4.2.5. *[SMIxDWCRn]* (Direct Write Control Register n) (n=0,1)

Bit	Bit Symbol	After Reset	Type	Description
31:24	CmdOp 1[7:0]	0x00	R/W	Op-code1 of SPI command of Serial memory n Op-code of SPI command should be set.
23:16	CmdOp 0[7:0]	0x02	R/W	Op-code 0 of SPI command of Serial memory n Op-code of SPI command should be set.
15:11	DmyBc[4:0]	00001	R/W	SPI dummy byte count of Serial memory n Set <DmyBc[4:0]> to 00000.
10:9	AddrBc[1:0]	00	R/W	SPI Address byte Count 00: 3 bytes 01: 2 bytes 10: 4 bytes 11: Reserved
8	CmdBc	0	R/WR	SPI Command byte Count 0: 1-byte 1: 2-bytes
7:6	DatIO[1:0]	00	R/W	SPI data input and output control of Serial memory n 00: Single 10: Quad 01: Reserved 11: Octal
5:4	DmyIO[1:0]	00	R/W	SPI dummy input and output control of Serial memory n 00: Single 10: Quad 01: Reserved 11: Octal
3:2	AdrIO[1:0]	00	R/W	SPI address input and output control of Serial memory n 00: Single 10: Quad 01: Reserved 11: Octal
1:0	CmdIO[1:0]	00	R/W	SPI command input and output control of Serial memory n 00: Single 10: Quad 01: Reserved 11: Octal

4.2.6. *[SMIxRACR0]* (Indirect Access Control Register 0)

Bit	Bit Symbol	After Reset	Type	Description
31:21	-	0	R	Read as “0”
20:16	SPR[4:0]	11111	R/W	Transfer clock of Serial memory n 00000: 2 divide 00001: 2 divide 00010: 3 divide 00011: 4 divide : 11110: 31 divide 11111: 32 divide
15:8	SCSD[7:0]	0xFF	R/W	SMIxCSn_N deassertion time 0x00 to 0xFF: Deassertion time Deassertion time = fsys frequency × <SCSD[7:0]> (Note1)
7:6	-	0	R	Read as “0”
5:4	SDCE[1:0]	00	R/W	Write "01" (Note2)
3	-	0	R	Read as “0”
2	-	0	R/W	Write “0”
1:0	-	0	R	Read as “0”

Note1: The deassertion time should be set from 0 ns to (fsys cycle time × 255)ns.
Note2: The <SDCE[1:0]> field becomes “00” after the reset. The field should be written to “01” in the initialization setting.

4.2.7. *[SMIxRACR1]* (Indirect Access Control Register 1)

Bit	Bit Symbol	After Reset	Type	Description
31:24	SBufBc[7:0]	0x00	R/W	Transfer Byte count in the secondary buffer The transfer byte count is (the set value +1) bytes from the secondary buffer data. In this field, “0” to “255” can be specified as the 8-bit counter.
23:22	-	0	R	Read as “0”
21	-	0	R/W	Write “0”
20:16	PBufBc[4:0]	00000	R/W	Transfer Byte count in the primary buffer (Note2) “0” to “31”: Transfer byte count The transfer byte count is (the set value +1) bytes.
15:6	-	0	R	Read as “0”
5	SBufEn	0	R/W	Secondary buffer enable (Note3) (Note4) 0: Secondary buffer is not used. 1: Secondary buffer is used.
4	PBufEn	0	R/W	Primary buffer enable 0: Primary buffer is not used. 1: Primary buffer is used.
3:2	-	0	R	Read as “0”
1	CSNum	0	R/W	Asserted CS 0: SMIxCS0_N 1: SMIxCS1_N
0	CycGo	0	R/W	Indirect access control 0: don't care 1: Access start by the indirect access (Note1) Read as “0”

Note1: <CycGo> should not be set to “1” while ***[SMIxSTAT]***<CycProg> is “1” (during SPI cycle).
Note2: When transferring with Quad or Octal using a secondary buffer, use the following settings.
For Quad: PBufBc[4:0] ≥ 1
For Octal: PBufBc[4:0] ≥ 2
Note3: It is not possible to use only secondary buffer. Use in conjunction with primary buffer.
Note4: When using send only commands in multi-I/O communication, use both primary buffer and secondary buffer.

4.2.8. *[SMIxIOCR]* (Indirect Access I/O Control Register)

Bit	Bit Symbol	After Reset	Type	Description
31:21	-	0	R	Read as “0”
20:16	PBufIOCtrl[4:0]	00000	R/W	Transfers the primary buffer data with the data width of <PBufIOCtrlEn [1:0]> from the specified byte transfer position. "0" to “6” can be specified. For example, specify “0” for QPI and OPI where all of the configured commands are multi I/O communication
15:4	-	0	R	Read as “0”
3:2	SBufIOCtrlEn[1:0]	00	R/W	Secondary Buffer I/O Control Register (Note) 00: Single 01: Reserved 10: Quad 11: Octal
1:0	PBufIOCtrlEn[1:0]	00	R/W	Primary Buffer I/O control Register 00: Single 01: Reserved 10: Quad 11: Octal

Note: When not using the secondary buffer, set <SBufIOCtrlEn[1:0]> to the same setting as <PBufIOCtrlEn[1:0]>.

4.2.9. *[SMIxOECR]* (Indirect Access Output Enable Control Register)

Bit	Bit Symbol	After Reset	Type	Description
31:21	-	0	R	Read as “0”
20:16	PBufOEnC[4:0]	00000	R/W	Primary Buffer Output Enable Control Register (Note1) When <PBufOCEn> = 1, the primary buffer data transmission / reception direction is switched from the specified byte transfer position. “0” to “6” can be specified. For example, specify “4” for FAST READ, which has “1” command op-code byte and “3” address bytes. By doing so, the I/O direction is switched to the input direction during the dummy communication period, and a turnaround is realized.
15:1	-	0	R	Read as “0”
0	PBufOCEn	0	R/W	Primary Buffer Output Control Enable Register (Note2) 0: Disabled. 1: Enabled.

Note 1: Set <PBufOEnC[4:0]> to the value equal or greater than ***[SMIxIOCR]***<PBufIOCtrl[4:0]>.

Note 2: Set <PBufOCEn> to "0" when using single for SPI command input/output.

4.2.10. *[SMIxINT]* (Interrupt Control Register)

Bit	Bit Symbol	After Reset	Type	Description
31:5	-	0	R	Read as “0”
4	SDIntEn	0	R/W	Stop Done interrupt Enable Controls the completion interrupt output for forced stop processing. 0: Disabled. 1: Enabled.
3:1	-	0	R	Read as “0”
0	SCDIntEn	0	R/W	Indirect access Done interrupt Enable Controls the completion interrupt output of indirect access. 0: Disabled. 1: Enabled.

4.2.11. *[SMIxSTAT]* (Status Register)

Bit	Bit Symbol	After Reset	Type	Description
31:7	-	0	R	Read as “0”
6	DiAccInProg	0	R	Direct Access in Progress Indicates whether direct access has occurred. 0: No direct access has occurred 1: Direct access has occurred.
5		0	R	Read as “0”
4	StpProgDone	0	R	Forced stop Process Done Indicates the completion status of the forced stop processing. The hardware sets this bit to “1”. It remains “1” until cleared by software. When this bit is set and ***[SMIxINT]*** <SDIntEn> is “1”, an interrupt is asserted. 0: Forced stop processing is not completed or has not occurred 1: Forced stop processing has been completed
			W	Clear forced stop interrupt 0: clear 1: don’t care
3:2	-	0	R	Read as “0”
1	CycProg	0	R	SPI Cycle In Progress This bit is set to “1” after <CycGo> is set. It is reset to “0” when the SPI cycle is completed. 0: Indirect access has not occurred 1: Indirect access has occurred
0	CycDone	0	R	SPI Cycle Done After completing the SPI cycle, the hardware sets this bit to “1”. It remains “1” until cleared by software. If this bit is set and ***[SMIxINT]*** <SCDIntEn> is “1”, an interrupt will be asserted. 0: Indirect access is not completed or has not occurred 1: Indirect access has been completed
			W	Clear the SPI cycle completion interrupt. 0: clear 1: don’t care

4.2.12. *[SMIxSWR]* (Software Reset Register)

Bit	Bit Symbol	After Reset	Type	Description
31:1	-	0	R	Read as “0”
0	RstEn	0	RW	Reset Enable Controls software reset to SMIF. When “1” is written to this field, SMIF immediately executes reset processing. After this, when the reset is completed, this field is cleared to “0” (Note). 0: Software reset completed or not issued 1: Software reset in progress

Note: After software reset, polling until <RstEn> “0” can be read.

4.2.13. *[SMIxACKR]* (Additional Clock Control Register)

Bit	Bit Symbol	After Reset	Type	Description
31:9	-	0	R	Read as “0”
8	AddClkTglEn	0	RW	Additional Clock Toggle Enable When SMIxCSn_N becomes “High” after SPI transfer is completed, it controls whether to output one additional transfer clock pulse. The setting of <FrcClkOutEn> has priority. 0: Additional clock not output 1: Additional clock output
7:1	-	0	R	Read as “0”
0	FrcClkOutEn	0	RW	Force Clock Output Enable If ***[SMIxCCOR]*** <ClkOE> = 1, output SMIxCLK. 0: SMIxCLK is output only when SPI communication occurs 1: SMIxCLK is output without SPI transfer

4.2.14. *[SMIxCCOR]* (SMIxCLK/CS Output Enable Control Register)

Bit	Bit Symbol	After Reset	Type	Description
31:6	-	0	R	Read as “0”
5	Cs1OE	0	RW	Chip select 1 Output Enable Controls the Output Enable of SMIxCS1_N. 0: Disabled(Hi-Z) 1: Enable
4	Cs0OE	0	RW	Chip select 0 Output Enable Controls the Output Enable of SMIxCS0_N. 0: Disabled(Hi-z) 1: Enable
3:1	-	0	R	Read as “0”
0	ClkOE	0	RW	Clock Output Enable Controls the Output Enable of SMIxCLK 0: Disabled(Hi-z) 1: Enable

Note: Set the port. Refer to reference manual "Input/Output PORT" for details.

4.2.15. *[SMIxSTPR]* (Forced Stop Control Register)

Bit	Bit Symbol	After Reset	Type	Description
31:1	-	0	R	Read as “0”
0	StopEn	0	RW	Stop Enable Issues a request to force SMIF to stop the operation. If “1” is written to this field (“0” write is ignored), SMIF immediately executes stop processing. To return, you need to perform a software reset. (Note) 0: Request not issued 1: Request issued

Note: When communication with the SPI memory is occurring, SMIF sets the chip select signal to “High” and interrupts the communication. If a forced stop is performed, the validity of the data is not guaranteed.

4.2.16. *[SMIxPBUFn]* (Indirect Access Primary Buffer Register n) (n = 0 to 7)

[SMIxPBUFn]

Bit	Bit Symbol	After Reset	Type	Description
31:24	PBuf(4n+3)[7:0]	Undefined	R/W	Primary buffer
23:16	PBuf(4n+2)[7:0]	Undefined	R/W	Primary buffer
15:8	PBuf(4n+1)[7:0]	Undefined	R/W	Primary buffer
7:0	PBuf(4n+0)[7:0]	Undefined	R/W	Primary buffer

Note: The capacity of the primary buffer differs depending on the product. For details, refer to the Reference Manual "Product Information".

4.2.17. *[SMIxSBUFm]* (Indirect Access Secondary Buffer Register m) (m = 0 to 63)

[SMIxSBUFm]

Bit	Bit Symbol	After Reset	Type	Description
31:24	SBuf(4m+3)[7:0]	Undefined	R/W	Secondary buffer
23:16	SBuf(4m+2)[7:0]	Undefined	R/W	Secondary buffer
15:8	SBuf(4m+1)[7:0]	Undefined	R/W	Secondary buffer
7:0	SBuf(4m+0)[7:0]	Undefined	R/W	Secondary buffer

Note: The capacity of the secondary buffer differs depending on the product. For details, refer to the Reference Manual "Product Information".

5. Example of Usage

5.1. Initial setting

Software must initialize the module after the module is released from reset. This section describes the following:

In order to use SMIF, a clock must be supplied to SMIF. Refer to the CGRST section related data sheet for the procedure. And it is necessary to assign I/O pin functions so that SMIF can use the I/O pins of the product. For the procedure, refer to the data sheet related to the PORT section.

After the above settings are completed, perform initial settings using the registers of SMIF. The outline of the initial setting is as follows.

- Common settings for direct access and indirect access
- Direct access serial memory 0 setting
- Direct access serial memory 1 setting
- Indirect access settings

5.1.1. Common settings for direct access and indirect access

The following registers are related to both direct access and indirect access.

- ***[SMIxINT]***
  - Controls interrupt output of both forced stop and indirect access complete.
  - The interrupts of SMIF are a level output of polarity “High”. When using interrupt, its clearing must be done in software.

- ***[SMIxACKR]***
  - <AddClkTglEn>: This bit is set to “1” if the SPI memory requires SMIxCLK output after deassertion of chip select to determine the completion of each SPI communication. Normally this bit should be set to “0”, but some special devices need to be set to “1”. Set this bit according to the specifications of the SPI memory.

  - <FrcClkOutEn>: This bit is used when SMIxCLK output is required to confirm the reset release of SPI memory. Set this bit to “1” according to the reset release sequence of the SPI memory, and then set it to “0” again after the wait period. The wait period depends on the SPI memory specifications. Do not perform SPI communication with this bit set to “1”.

  - ***[SMIxCCOR]***
    Controls the I/O direction of SMIxCLK, SMIxCS0_N, SMIxCS1_N. Set the pins to be used to output enable. It is different from the control of the port part. Note that both controls are required.

5.1.2. Direct Access serial memory 0 setting

Shows the initial state for direct access serial memory 0.

- Memory map: Refer to the product address map. This module has a 128MB area.
- Read command issuing function: Enabled.
- Write command issuing function: Disabled.
- Read command: Number of op-code bytes: 1, Op-code: 0x0B, Number of address bytes: 3, Number of dummy byte: 1.
- Write command: Number of op-code bytes: 1, Op-code: 0x02, Number of address bytes: 3, Number of dummy byte: 0.
- Chip select deassertion time: 0 ns.
- Transfer clock: Divide the bus clock by 32.

Set SMIF according to the SPI memory connected to the chip select pin (SMIxCS0_N). Perform the following procedure.

1. Set base address and size, according to the SPI memory specification. Set it by ***[SMIxMAP0]*** <FBA [11:0]>, <FDEN [3:0]>.
2. Set the transfer clock, chip select deassertion time according to the SPI memory AC timing. Set it by ***[SMIxDACR0]*** <SPR [4:0]>, <SCSD [7:0]>.
3. If necessary, change the command configuration issued to the SPI memory (for example, when using multi I/O as 4×I/O or 8×I/O). Set it by ***[SMIxDRCR0].***
4. If you want to issue a write command to the SPI memory, set the command configuration to issue with ***[SMIxDWCR0]***. Then, set ***[SMIxMAP0]*** <WE> to "1".

5.1.3. Direct Access serial memory 1 Setting

The function of direct access ch1 is disabled after reset. Software must set ***[SMIxMAP1]*** <FBA [11:0]>, <FDEN [3:0]>, ***[SMIxDACR1]*** <SPR [4:0]>, <SCSD [7:0]>, <SDCE [1:0]>, ***[SMIxDRCR1]***, and ***[SMIxDWCR1]*** according to the SPI memory specification. After these settings are completed, set ***[SMIxMAP1]*** <RE> and <WE> to "1" to enable SMIF to issue read and write commands.

5.1.4. Indirect Access Setting

The indirect access setting includes the selection of the chip select that issues the command. In indirect access, different settings can be made each time a transfer is performed.

The transfer clock, chip select deassert time must be set according to the SPI memory.
Set them by ***[SMIxRACR0]*** <SPR [4:0]>, <SCSD [7:0]>, <SDCE [1:0]> respectively. Use ***[SMIxINT]*** <SCDIntEn> to set whether to generate the interrupt when the indirect access is completed.

5.1.5. Transfer clock

The frequency of the transfer clock (SMIxCLK) is determined by the dividing ratio of ***[SMIxDACR0]*** <SPR [4:0]>, ***[SMIxDACR1]*** <SPR [4:0]>, and ***[SMIxRACR0]*** <SPR [4:0]>.

Transfer clock = fsys frequency / dividing ratio

Note 1: Set the SMIxCLK frequency within the range that meets the AC specifications of the product.
Note 2: When odd division (<SPR [4:0]> value is even), the duty ratio of SMIxCLK is not 50%.
During odd frequency division, the “High” width of SMIxCLK is shorter than the “Low” width by one system clock cycle.
Note 3: <SPR [4:0]> = 00000: SMIxCLK is divided by “2”.

5.2. Programming Method for Direct Access

The software does not need to change the direct access settings as long as it continues to communicate with the SPI memory.
When SMIF detects a read or write access to the direct access area, it issues a command to the SPI memory based on the following register settings. For details on the settings, refer to 5.1.2 and 5.1.3.

- ***[SMIxMAP0]***
- ***[SMIxMAP1]***
- ***[SMIxDACR0]***
- ***[SMIxDACR1]***
- ***[SMIxDRCR0]***
- ***[SMIxDRCR1]***
- ***[SMIxDWCR0]***
- ***[SMIxDWCR1]***

To change ***[SMIxDACR0]*** <SPR [4:0]> and ***[SMIxDACR1]*** <SPR [4:0]> , perform a dummy read (read without using the return value of read) for the corresponding direct access area after changing the setting. This dummy read is not required during initial setting.

Note: When the software changes the direct access setting, instructions cannot be executed in the target direct access area.

5.3. Programming Method for Indirect Access

5.3.1. Basic procedure

Follow the procedure below to program indirect access.

1. To wait for the completion of indirect access by interrupt, write “1” to ***[SMIxINT]*** <SCDIntEn>.
2. Check ***[SMIxSTAT]*** <CycPrgrs> for no indirect access in progress.
3. If necessary, set the transfer clock, chip select deassertion time by ***[SMIxRACR0]*** <SPR [4:0]>, <SCSD [7:0]> respectively.
4. Change the command configuration issued to the SPI memory.
5. Set ***[SMIxRACR1]*** <CycGo> to “1” to start indirect access (Note 1).
6. Wait for the indirect access to complete. (Wait for the indirect access completion interrupt or wait until ***[SMIxSTAT]*** <CycDone> is set to “1” (Note 2).)
7. Write “0” to ***[SMIxSTAT]*** <CycDone>.

Set the command configuration for the supported device as follows.

- The data width and transfer direction of I/O used for communication with SPI memory are set as follows using ***[SMIxIOCR]*** and ***[SMIxOECR]***.

(1) ***[SMIxIOCR]***

(a) <PBufIOCtrlEn [1:0]>: Set this field to “00” for SPI communication, “10” (4×I/O) or “11” (8×I/O) for multi I/O communication .

(b) <PBufIOCtrl [4:0]>: When <PBufIOCtrlEn> is set for SPI communication, this field setting is invalid. When multi I/O communication is set, specify the byte number at which the I/O width switches. For example, for QPI and OPI communication where the entire command is multi I/O communication, set this field to “00000” .

(2) ***[SMIxOECR]***

(a)<PBufOCEn>: Set this field to “0” for transmission, “0” for SPI communication for reception, “1” for multi I/Ocommunication for reception.

(b) <PBufOEnC [4:0]>: When transmission or <PBufOCEn> is set to SPI communication, this field is invalid. Multi I/O communication for reception, specify the byte number at which the I/O direction switches from OUT to IN. For example, if the command op-code byte count is “1” and the address byte count is “3” for FAST READ, set this field to “00100”.

- Set the command op-code and address to be used for transfer in ***[SMIxPBUFn]***.
- The transfer direction is transmission, set the transmission data in ***[SMIxSBUFm]***.
- ***[SMIxRACR1]*** <SBufBc [7:0]>, <PBufBc [4:0]>, <SBufEn>, <PBufEn> to set the number of bytes in the primary and secondary buffers used for transfer (Note 1)).
- Set the chip select number to which the communication destination device is connected to ***[SMIxRACR1]*** <CSNum> (Note 1).

Note 1: Writing “1” to ***[SMIxRACR1]*** <CycGo> and setting values of other fields may be changed at the same time.
Note 2: Writing to the following registers is prohibited until ***[SMIxSTAT]*** <CycDone> is set to “1” after writing “1” to ***[SMIxRACR1]*** <CycGo>.

- ***[SMIxRACR0]***
- ***[SMIxRACR1]***
- ***[SMIxIOCR]***
- ***[SMIxOECR]***
- ***[SMIxPBUFn]***
- ***[SMIxSBUFm]***

5.3.2. Programming example: PAGE PROGRAM, WRITE

The procedure for transmitting 256-byte data (Data [2047:0]) to the 3-byte physical address (Addr [23:0]) of the SPI memory is shown below.

Note: The SPI Flash enters the BUSY state after executing this command. A device in the BUSY state can normally communicate only with some commands such as status register read. If store of instruction in the SPI Flash, do not fetch the instruction from the SPI Flash until the BUSY state is released. If a memory read access to the SPI memory during BUSY state, good data cannot be read from the SPI Flash.

The command configuration is as follows: command op-code byte count: 1, address byte count: 3, dummy byte count: 0, payload data byte count: 256. The I/O width of the command is all SPI. The command op-code is "0x02".

1. Set the following values in the primary buffer.
   -<PBuf0> = 0x02
   -<PBuf1> = Addr [23:16]
   -<PBuf2> = Addr [15:8]
   -<PBuf3> = Addr [7:0]

2. Set the following values in the secondary buffer.
   -<SBuf0> = Data [7:0] (Data written to Addr [23:0] + 0x00)
   -<SBuf1> = Data [15:8] (Data written to Addr [23:0] + 0x01)
   -
   -<SBuf254> = Data [2039: 2032] (Data written to Addr [23:0] + 0xFE)
   -<SBuf255> = Data [2047: 2040] (Data written to Addr [23:0] + 0xFF)

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 00011
   -<SBufEn> = 1, <SBufBc [7:0]> = 0xFF
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 00 (SPI communication)
   -<SBufIOCtrlEn [1:0]> = 00 (SPI communication)
   -<PBufIOCtrl [4:0]> = don't care

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 0 (transmit)
   -<PBufOEnC [4:0]> = don't care

6. Set ***[SMIxRACR1]*** <CycGo> to "1" to start indirect access.

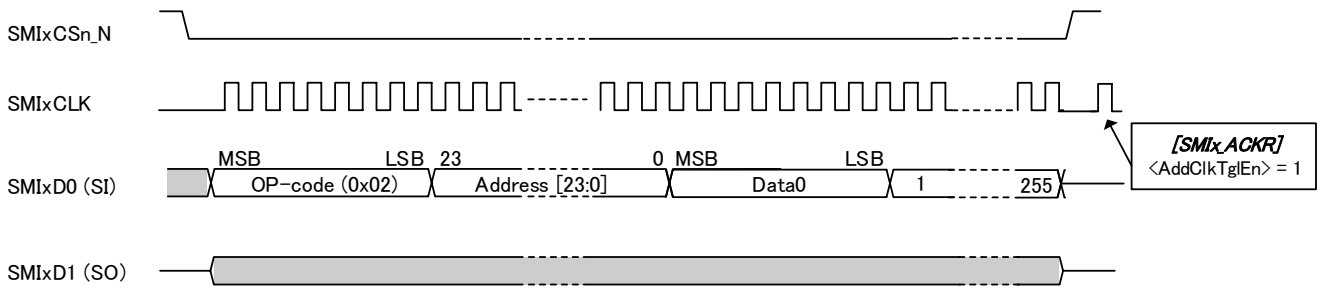

Figure 5.1 PAGE PROGRAM, WRITE

5.3.3. Programming example: Quad I/O PAGE PROGRAM, Quad I/O WRITE

The procedure for transmitting 256-byte of data (Data [2047:0]) to the 3-byte physical address (Addr [23:0]) of the SPI memory using 4×I/O is shown below.

Note: The SPI Flash enters the BUSY state after executing this command. A device in the BUSY state can normally communicate only with some commands such as status register read. If store of instruction in the SPI Flash, do not fetch the instruction from the SPI Flash until the BUSY state is released. If a memory read access is made to the SPI memory during BUSY, good data cannot be read from the SPI Flash

The command configuration is as follows: command op-code byte count: 1, address byte count: 3, dummy byte count: 0, payload data byte count: 256. For the command I/O width, the command op-code is SPI, and 4×I/O after the address. The command op-code is “0x38”.

1. Set the following values in the primary buffer.
   -<PBuf0 [7:0]> = 0x38
   -<PBuf1 [7:0]> = Addr [23:16]
   -<PBuf2 [7:0]> = Addr [15:8]
   -<PBuf3 [7:0]> = Addr [7:0]

2. Set the following values in the secondary buffer.
   -<SBuf0 [7:0]> = Data [7:0] (Data written to Addr [23:0] + 0x00)
   -<SBuf1 [7:0]> = Data [15:8] (Data written to Addr [23:0] + 0x01)
   -
   -<SBuf254 [7:0]> = Data [2039: 2032] (Data written to Addr [23:0] + 0xFE)
   -<SBuf255 [7:0]> = Data [2047: 2040] (Data written to Addr [23:0] + 0xFF)

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 00011
   -<SBufEn> = 1, <SBufBc [7:0]> = 0xFF
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 10 (4×I/O communication)
   -<SBufIOCtrlEn [1:0]> = 10 (4×I/O communication)
   -<PBufIOCtrl [4:0]> = 00001 (command op-code is single, address to multi I/O communication)

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 0 (transmit)
   -<PBufOEnC [4:0]> = don't care

6. Set ***[SMIxRACR1]*** <CycGo> to “1” to start indirect access.

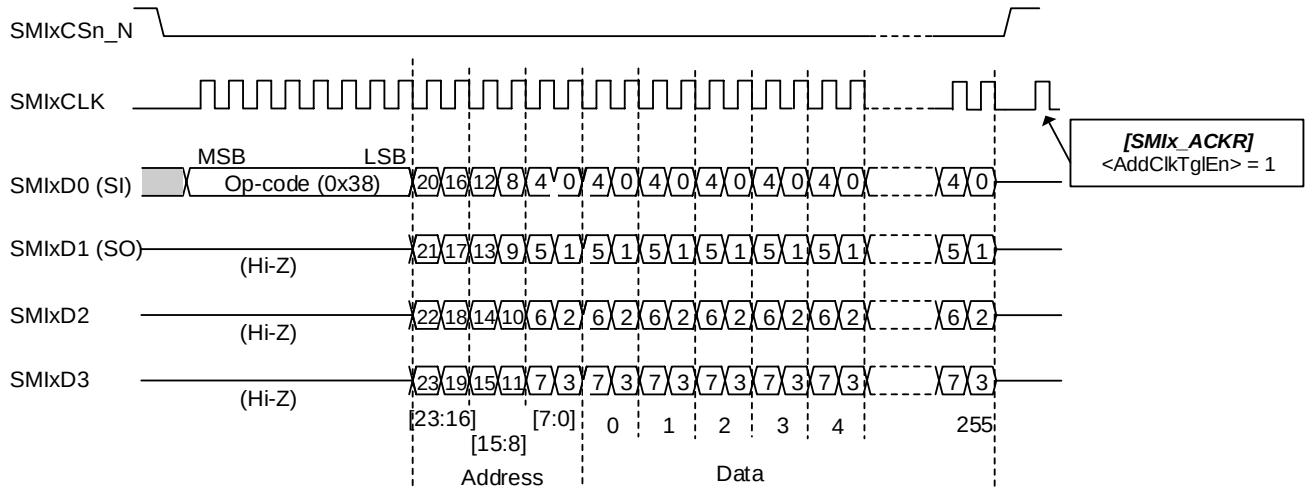

Figure 5.2 Quad I/O PAGE PROGRAM, Quad I/O WRITE

5.3.4. Programming example: QPI PAGE PROGRAM, QPI WRITE

The procedure for transmitting 256-byte of data (Data [2047:0]) to the 3-byte physical address (Addr [23:0]) of the SPI memory using 4×I/O is shown below.

Note: The SPI Flash enters the BUSY state after executing this command. A device in the BUSY state can normally communicate only with some commands such as status register read. If store of instruction in the SPI Flash, do not fetch the instruction from the SPI Flash until the BUSY state is released. If a memory read access is made to the SPI memory during BUSY, good data cannot be read from the SPI Flash

The command configuration is as follows: command op-code byte count: 1, address byte count: 3, dummy byte count:0, payload data byte count: 256. For the command I/O width, the command op-code is SPI, and 4×I/O after the address. The command op-code is “0x02”.

1. Set the following values in the primary buffer.
   -<PBuf0 [7:0]> = 0x02
   -<PBuf1 [7:0]> = Addr [23:16]
   -<PBuf2 [7:0]> = Addr [15:8]
   -<PBuf3 [7:0]> = Addr [7:0]

2. Set the following values in the secondary buffer.
   -<SBuf0 [7:0]> = Data [7:0] (Data written to Addr [23:0] + 0x00)
   -<SBuf1 [7:0]> = Data [15:8] (Data written to Addr [23:0] + 0x01)
   -
   -<SBuf254 [7:0]> = Data [2039: 2032] (Data written to Addr [23:0] + 0xFE)
   -<SBuf255 [7:0]> = Data [2047: 2040] (Data written to Addr [23:0] + 0xFF)

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 00011
   -<SBufEn> = 1, <SBufBc [7:0]> = 0xFF
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 10 (4×I/O communication)
   -<SBufIOCtrlEn [1:0]> = 10 (4×I/O communication)
   -<PBufIOCtrl [4:0]> = 00001 (command op-code is single, address to multi I/O communication)

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 0 (transmit)
   -<PBufOEnC [4:0]> = don't care

6. Set ***[SMIxRACR1]*** <CycGo> to “1” to start indirect access.

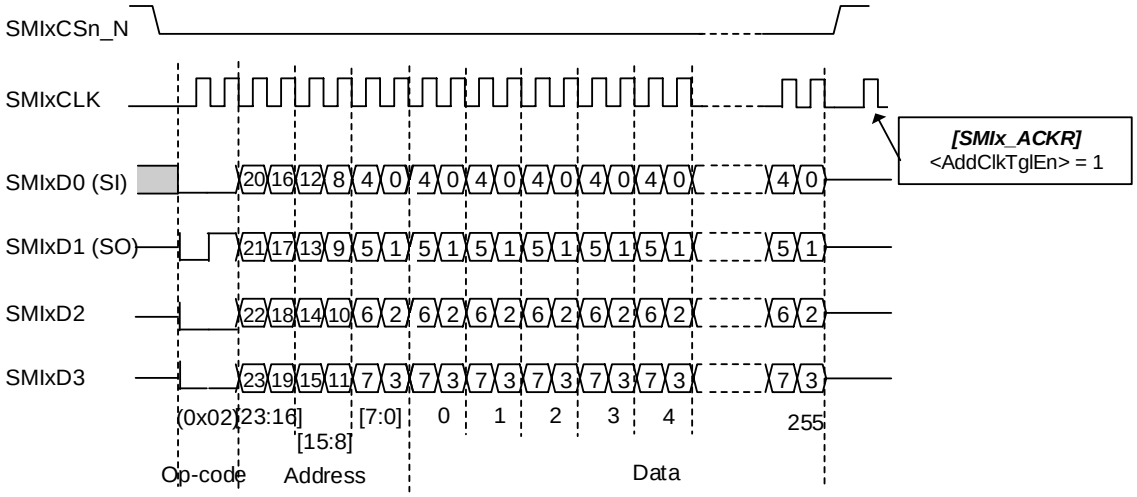

Figure 5.3 QPI PAGE PROGRAM, QPI WRITE

5.3.5. Programming example: Octal I/O PAGE PROGRAM, Octal I/O WRITE

The procedure for transmitting 256-byte of data (Data [2047:0]) to the 3-byte physical address (Addr [23:0]) of the SPI memory using 8×I/O is shown below.

Note: The SPI Flash enters the BUSY state after executing this command. A device in the BUSY state can normally communicate only with some commands such as status register read. If store of instruction in the SPI Flash, do not fetch the instruction from the SPI Flash until the BUSY state is released. If a memory read access is made to the SPI memory during BUSY, good data cannot be read from the SPI Flash

The command configuration is as follows: command op-code byte count: 1, address byte count: 3, dummy byte count: 0, payload data byte count: 256. For the command I/O width, the command op-code is SPI, and 8×I/O after the address. The command op-code is “0xC2”.

1. Set the following values in the primary buffer.
   -<PBuf0 [7:0]> = 0xC2
   -<PBuf1 [7:0]> = Addr [23:16]
   -<PBuf2 [7:0]> = Addr [15:8]
   -<PBuf3 [7:0]> = Addr [7:0]

2. Set the following values in the secondary buffer.
   -<SBuf0 [7:0]> = Data [7:0] (Data written to Addr [23:0] + 0x00)
   -<SBuf1 [7:0]> = Data [15:8] (Data written to Addr [23:0] + 0x01)
   -
   -<SBuf254 [7:0]> = Data [2039: 2032] (Data written to Addr [23:0] + 0xFE)
   -<SBuf255 [7:0]> = Data [2047: 2040] (Data written to Addr [23:0] + 0xFF)

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 00011
   -<SBufEn> = 1, <SBufBc [7:0]> = 0xFF
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 11 (8×I/O communication)
   -<SBufIOCtrlEn [1:0]> = 11 (8×I/O communication)
   -<PBufIOCtrl [4:0]> = 00001 (command op-code is single, address to multi I/O communication)

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 0 (transmit)
   -<PBufOEnC [4:0]> = don't care

6. Set ***[SMIxRACR1]*** <CycGo> to “1” to start indirect access.

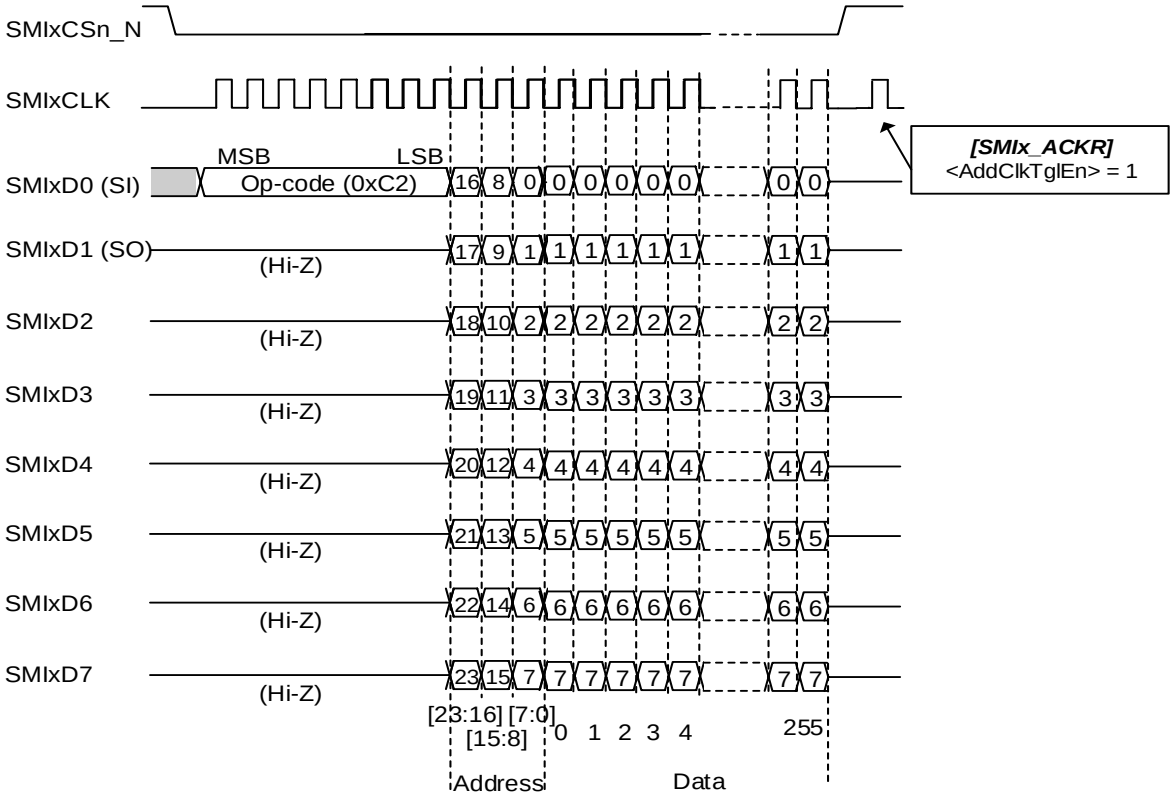

Figure 5.4 Octal I/O PAGE PROGRAM, Octal I/O WRITE

5.3.6. Programming example: OPI PAGE PROGRAM, OPI WRITE

The procedure for transmitting 256-byte of data (Data [2047:0]) to the 4-byte physical address (Addr [31:0]) of the SPI memory using OPI is shown below.

Note: The SPI Flash enters the BUSY state after executing this command. A device in the BUSY state can normally communicate only with some commands such as status register read. If store of instruction in the SPI Flash, do not fetch the instruction from the SPI Flash until the BUSY state is released. If a memory read access is made to the SPI memory during BUSY, good data cannot be read from the SPI Flash

The command configuration is as follows: command op-code byte count: 2, address byte count: 4, dummy byte count: 0, payload data byte count: 256. The I/O width of the command is all 8×I/O. The command op-code is "0x12ED".

1. Set the following values in the primary buffer.
   -<PBuf0 [7:0]> = 0x12
   -<PBuf1 [7:0]> = 0xED
   -<PBuf2 [7:0]> = Addr [31:24]
   -<PBuf3 [7:0]> = Addr [23:16]
   -<PBuf4 [7:0]> = Addr [15:8]
   -<PBuf5 [7:0]> = Addr [7:0]

2. Set the following values in the secondary buffer.
   -<SBuf0 [7:0]> = Data [7:0] (Data written to Addr [31:0] + 0x00)
   -<SBuf1 [7:0]> = Data [15:8] (Data written to Addr [31:0] + 0x01)
   -
   -<SBuf254 [7:0]> = Data [2039: 2032] (Data written to Addr [31:0] + 0xFE)
   -<SBuf255 [7:0]> = Data [2047: 2040] (Data written to Addr [31:0] + 0xFF)

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 00101
   -<SBufEn> = 1, <SBufBc [7:0]> = 0xFF
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 11 (8×I/O communication)
   -<SBufIOCtrlEn [1:0]> = 11 (8×I/O communication)
   -<PBufIOCtrl [4:0]> = 00000 (All multi I/O communication)

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 0 (transmit)
   -<PBufOEnC [4:0]> = don't care

6. Set ***[SMIxRACR1]*** <CycGo> to "1" to start indirect access.

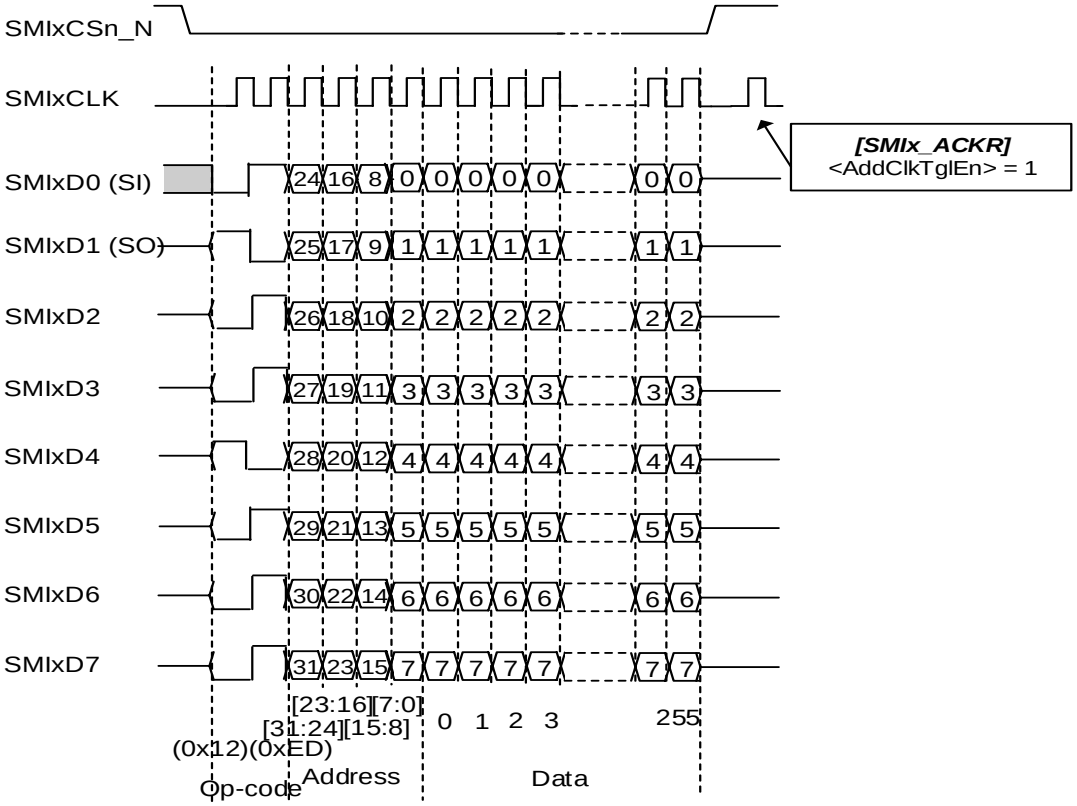

Figure 5.5 OPI PAGE PROGRAM, OPI WRITE

5.3.7. Programming example: SECTOR ERASE

The procedure for issuing SECTOR ERASE (op-code: 0x20) to the 3-byte physical address Addr [23:0] of the SPI memory is shown below.

Note: The SPI Flash enters the BUSY state after executing this command. A device in the BUSY state can normally communicate only with some commands such as status register read. If store of instruction in the SPI Flash, do not fetch the instruction from the SPI Flash until the BUSY state is released. If a memory read access is made to the SPI memory during BUSY, good data cannot be read from the SPI Flash

1. Set the following values in the primary buffer.
   -<PBuf0 [7:0]> = 0x20
   -<PBuf1 [7:0]> = Addr[23:16]
   -<PBuf2 [7:0]> = Addr [15:8]
   -<PBuf3 [7:0]> = Addr [7:0]

2. Not use the secondary buffer.

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 00011
   -<SBufEn> = 0, <SBufBc [7:0]> = don't care
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 00 (SPI communication)
   -<SBufIOCtrlEn [1:0]> = 00 (SPI communication)
   -<PBufIOCtrl [4:0]> = don't care

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 0 (transmit)
   -<PBufOEnC [4:0]> = don't care

6. Set ***[SMIxRACR1]*** <CycGo> to “1” to start indirect access.

5.3.8. Programming example: CHIP ERASE

The procedure to issue CHIP ERASE (op-code: 0xC7) is shown below.

Note: The SPI Flash enters the BUSY state after executing this command. A device in the BUSY state can normally communicate only with some commands such as status register read. If store of instruction in the SPI Flash, do not fetch the instruction from the SPI Flash until the BUSY state is released. If a memory read access is made to the SPI memory during BUSY, good data cannot be read from the SPI Flash

1. Set the following values in the primary buffer.
   -<PBuf0 [7:0]> = 0xC7

2. Not use the secondary buffer.

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 00000
   -<SBufEn> = 0, <SBufBc [7:0]> = don't care
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 00 (SPI communication)
   -<SBufIOCtrlEn [1:0]> = 00 (SPI communication)
   -<PBufIOCtrl [4:0]> = don't care

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 0 (transmit)
   -<PBufOEnC [4:0]> = don't care

6. Set ***[SMIxRACR1]*** <CycGo> to “1” to start indirect access.

5.3.9. Programming example: READ STATUS

The procedure to read one byte data by READ STATUS (op-code: 0x05) is shown below.

1. Set the following values in the primary buffer.
   -<PBuf0 [7:0]> = 0x05

2. The secondary buffer stores status data returned by the device after indirect access is completed
   -<SBuf0 [7:0]> = Data [7:0] (The status register bit [7:0] of the device is stored.)

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 00000
   -<SBufEn> = 1, <SBufBc [7:0]> = 0x00
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 00 (SPI communication)
   -<SBufIOCtrlEn [1:0]> = 00 (SPI communication)
   -<PBufIOCtrl [4:0]> = don't care

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 0 (receive)
   -<PBufOEnC [4:0]> = don't care

6. Set ***[SMIxRACR1]*** <CycGo> to “1” to start indirect access.

When the indirect access is completed, the value of the status register of the SPI memory is stored in <SBuf0 [7:0]>.

5.3.10. Programming example: FAST READ

The following shows the procedure for receiving 256-byte data (Data [2047:0]) from the 3-byte physical address (Addr [23:0]) of the SPI memory.

The command configuration is as follows: command Op-code byte count: 1, address byte count: 3, dummy byte count: 1, payload data byte count: 256. The I/O width of the command is all SPI. The command Op-code is "0x0B".

1. Set the following values in the primary buffer.
   -<PBuf0 [7:0]> = 0x0B
   -<PBuf1 [7:0]> = Addr [23:16]
   -<PBuf2 [7:0]> = Addr [15:8]
   -<PBuf3 [7:0]> = Addr [7:0]

2. The read data from SPI memory is stored in the secondary buffer after indirect access is completed.
   -<SBuf0 [7:0]> = Data [7:0] (Data read from Addr [23:0] + 0x00)
   -<SBuf1 [7:0]> = Data [15:8] (Data read from Addr [23:0] + 0x01)
   -
   -<SBuf254 [7:0]> = Data [2039: 2032] (Data read from Addr [23:0] + 0xFE)
   -<SBuf255 [7:0]> = Data [2047: 2040] (Data read from Addr [23:0] + 0xFF)

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 00100
   -<SBufEn> = 1, <SBufBc [7:0]> = 0xFF
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 00 (SPI communication)
   -<SBufIOCtrlEn [1:0]> = 00 (SPI communication)
   -<PBufIOCtrl [4:0]> = don't care

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 0 (receive)
   -<PBufOEnC [4:0]> = don't care

6. Set ***[SMIxRACR1]*** <CycGo> to "1" to start indirect access.

When the indirect access is completed, the data read from the SPI memory is stored in <SBuf0 [7:0]> to <SBuf255 [7:0]>.

Note: The actual number of dummy bytes to be set depends on the model number of the SPI memory and the setting of its configuration register.

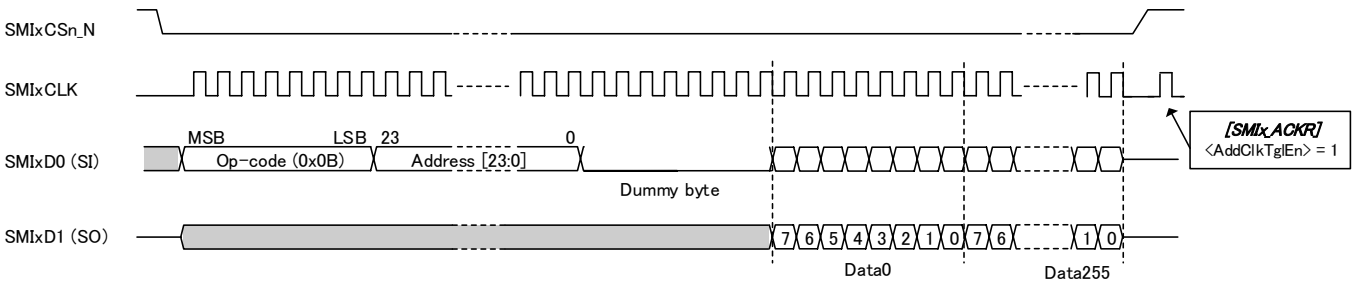

Figure 5.6 FAST READ

5.3.11. Programming example: Quad I/O FAST READ

The following shows the procedure for receiving 256-byte data (Data [2047:0]) from the 3-byte physical address (Addr [23:0]) of the SPI memory using 4×I/O.

The command configuration is as follows: command Op-code byte count: 1, address byte count: 3, dummy byte count: 3, payload data byte count: 256. For the command I/O width, the command Op-code is SPI, and 4×I/O after the address. The command Op-code is “0xEB”.

1. Set the following values in the primary buffer.
   -<PBuf0 [7:0]> = 0xEB
   -<PBuf1 [7:0]> = Addr [23:16]
   -<PBuf2 [7:0]> = Addr [15:8]
   -<PBuf3 [7:0]> = Addr [7:0]

2. The read data from SPI memory is stored in the secondary buffer after indirect access is completed.
   -<SBuf0 [7:0]> = Data [7:0] (Data read from Addr [23:0] + 0x00)
   -<SBuf1 [7:0]> = Data [15:8] (Data read from Addr [23:0] + 0x01)
   -
   -<SBuf254 [7:0]> = Data [2039: 2032] (Data read from Addr [23:0] + 0xFE)
   -<SBuf255 [7:0]> = Data [2047: 2040] (Data read from Addr [23:0] + 0xFF)

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 00110
   -<SBufEn> = 1, <SBufBc [7:0]> = 0xFF
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 10 (4×I/O communication)
   -<SBufIOCtrlEn [1:0]> = 10 (4×I/O communication)
   -<PBufIOCtrl [4:0]> = 00001

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 1
   -<PBufOEnC [4:0]> = 00100 (Switch cycle to input from dummy byte)

6. Set ***[SMIxRACR1]*** <CycGo> to “1” to start indirect access.

When the indirect access is completed, the data read from the SPI memory is stored in <SBuf0 [7:0]> to <SBuf255 [7:0]>.

Note: The actual number of dummy bytes to be set depends on the model number of the SPI memory and the setting of its configuration register.

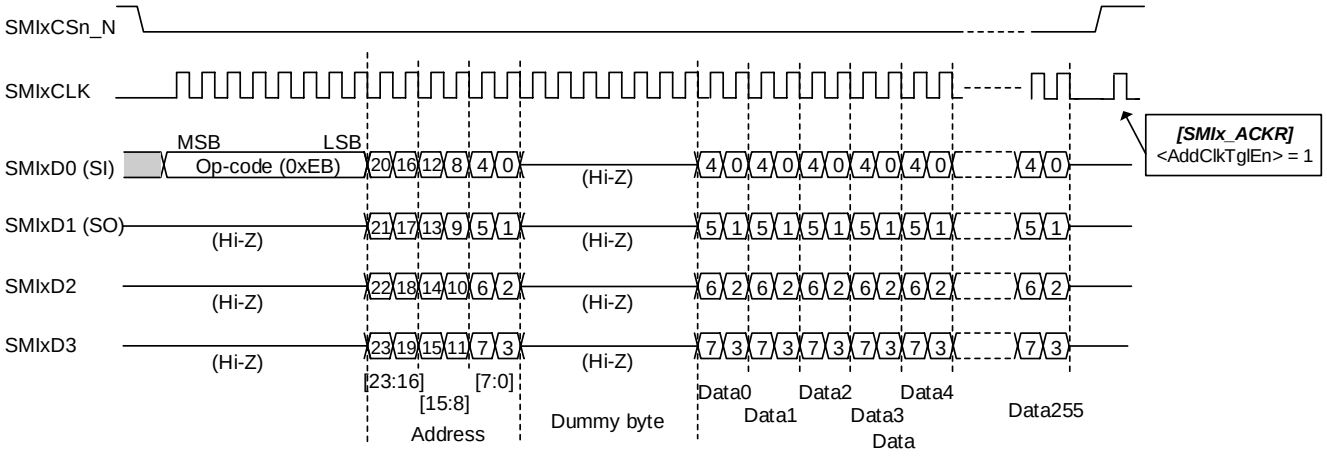

Figure 5.7 Quad I/O FAST READ

5.3.12. Programming example: QPI FAST READ

The following shows the procedure for receiving 256-bytes data (Data [2047:0]) from the 3-byte physical address (Addr [23:0]) of the SPI memory using 4×I/O.

The command configuration is as follows: command Op-code byte count: 1, address byte count: 3, dummy byte count: 3, payload data byte count: 256. The I/O width of all commands is 4×I/O. The command Op-code is "0xEB".

1. Set the following values in the primary buffer.
   -<PBuf0 [7:0]> = 0xEB
   -<PBuf1 [7:0]> = Addr [23:16]
   -<PBuf2 [7:0]> = Addr [15:8]
   -<PBuf3 [7:0]> = Addr [7:0]

2. The read data from SPI memory is stored in the secondary buffer after indirect access is completed.
   -<SBuf0 [7:0]> = Data [7:0] (Data read from Addr [23:0] + 0x00)
   -<SBuf1 [7:0]> = Data [15:8] (Data read from Addr [23:0] + 0x01)
   -
   -<SBuf254 [7:0]> = Data [2039: 2032] (Data read from Addr [23:0] + 0xFE)
   -<SBuf255 [7:0]> = Data [2047: 2040] (Data read from Addr [23:0] + 0xFF)

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 00110
   -<SBufEn> = 1, <SBufBc [7:0]> = 0xFF
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 10 (4×I/O communication)
   -<SBufIOCtrlEn [1:0]> = 10 (4×I/O communication)
   -<PBufIOCtrl [4:0]> = 00001 (Command op-code is single, address to multi I/O communication)

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 1
   -<PBufOEnC [4:0]> = 00100 (Switch cycle to input from dummy byte)

6. Set ***[SMIxRACR1]*** <CycGo> to "1" to start indirect access.

When the indirect access is completed, the data read from the SPI memory is stored in <SBuf0 [7:0]> to <SBuf255 [7:0]>.

Note: The actual number of dummy bytes to be set depends on the model number of the SPI memory and the setting of its configuration register.

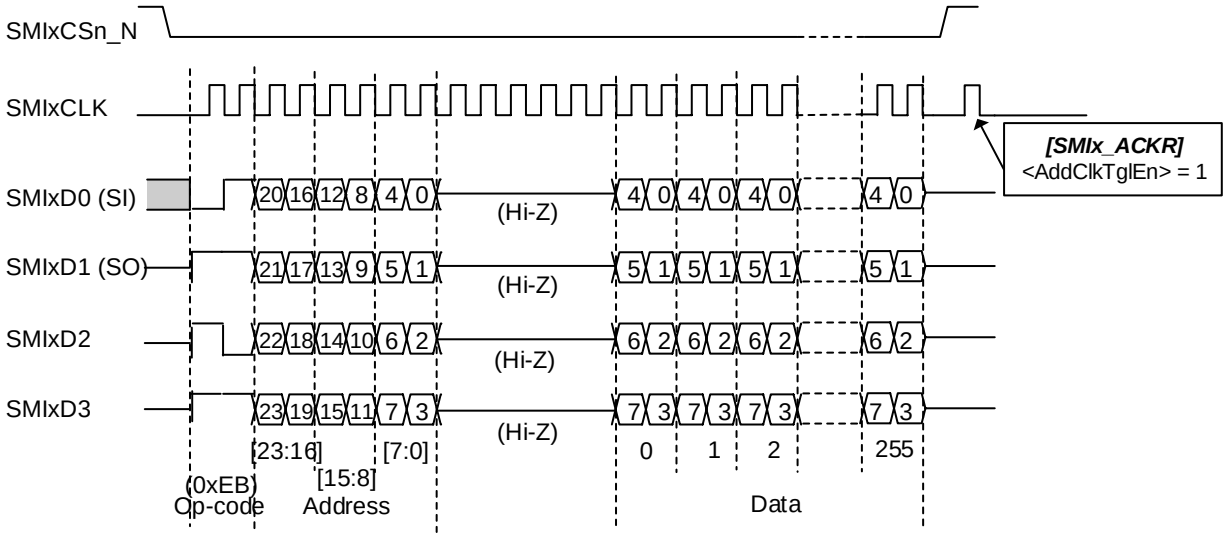

Figure 5.8 QPI FAST READ

5.3.13. Programming example: Octal FAST READ

The following shows the procedure for receiving 256-byte data (Data [2047:0]) from the 3byte physical address (Addr [23:0]) of the SPI memory using 8×I/O.

The command configuration is as follows: command Op-code byte count: 1, address byte count: 3, dummy byte count: 16, payload data byte count: 256. For the command I/O width, the command Op-code is SPI, and 8×I/O after the address. The command Op-code is “0xCB”.

1. Set the following values in the primary buffer.
   -<PBuf0 [7:0]> = 0xCB
   -<PBuf1 [7:0]> = Addr [23:16]
   -<PBuf2 [7:0]> = Addr [15:8]
   -<PBuf3 [7:0]> = Addr [7:0]

2. The read data from SPI memory is stored in the secondary buffer after indirect access is completed.
   -<SBuf0 [7:0]> = Data [7:0] (Data read from Addr [23:0] + 0x00)
   -<SBuf1 [7:0]> = Data [15:8] (Data read from Addr [23:0] + 0x01)
   -
   -<SBuf254 [7:0]> = Data [2039: 2032] (Data read from Addr [23:0] + 0xFE)
   -<SBuf255 [7:0]> = Data [2047: 2040] (Data read from Addr [23:0] + 0xFF)

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 10011
   -<SBufEn> = 1, <SBufBc [7:0]> = 0xFF
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 11 (8×I/O communication)
   -<SBufIOCtrlEn [1:0]> = 11 (8×I/O communication)
   -<PBufIOCtrl [4:0]> = 00001 (Command op-code is single, address to multi I/O communication)

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 1
   -<PBufOEnC [4:0]> = 00100 (Switch cycle to input from dummy byte)

6. Set ***[SMIxRACR1]*** <CycGo> to “1” to start indirect access.

When the indirect access is completed, the data read from the SPI memory is stored in <SBuf0 [7:0]> to <SBuf255 [7:0]>.

Note: The actual number of dummy bytes to be set depends on the model number of the SPI memory and the setting of its configuration register.

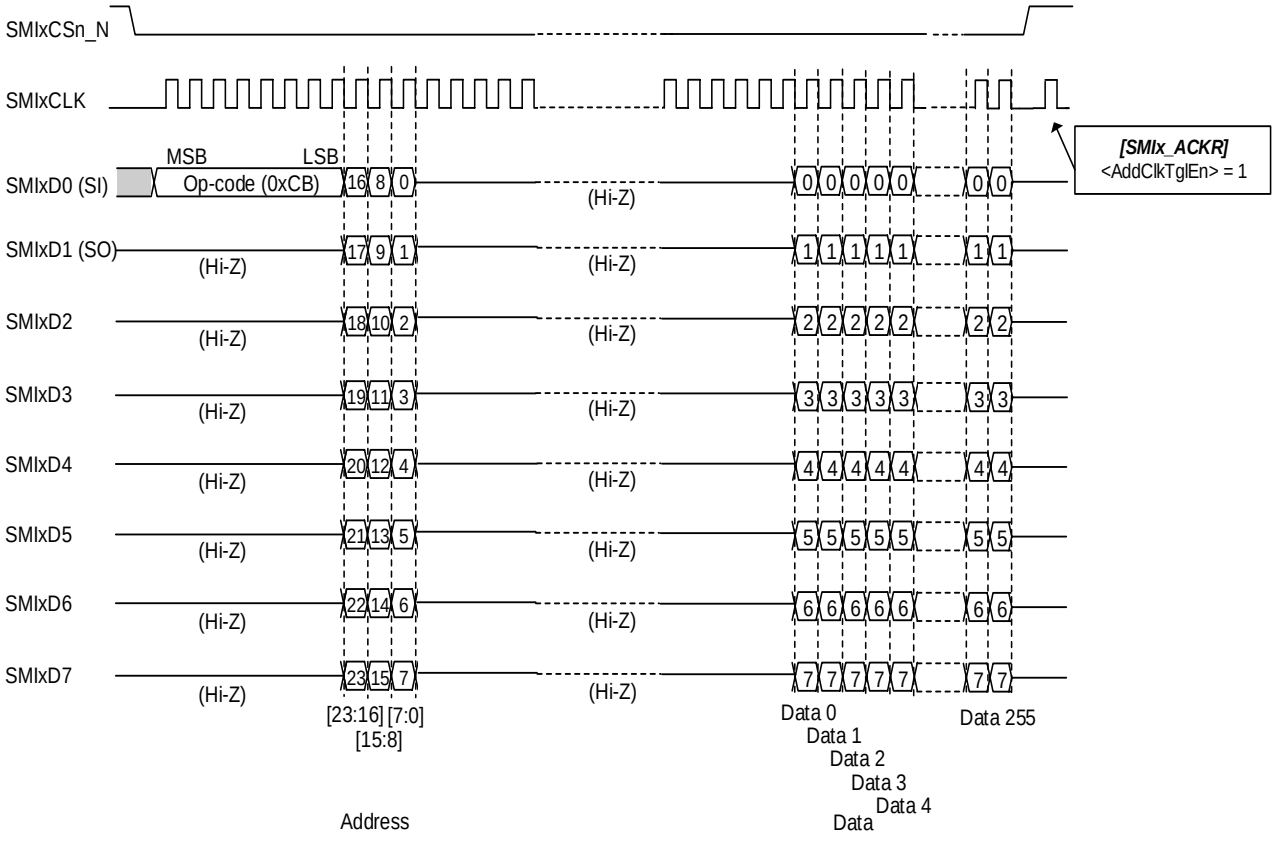

Figure 5.9 Octal FAST READ

5.3.14. Programming example: OPI FAST READ

The following shows the procedure for receiving 256-bytes data (Data [2047:0]) from the 4-byte physical address (Addr [31:0]) of the SPI memory using 8×I/O.

The command configuration is as follows: command Op-code byte count: 2, address byte count: 4, dummy byte count: 20, payload data byte count: 256. The I/O width of all commands is 8×I/O. The command Op-code is "0xEC13".

1. Set the following values in the primary buffer.
   -<PBuf0 [7:0]> = 0x13
   -<PBuf1 [7:0]> = 0xEC
   -<PBuf2 [7:0]> = Addr [31:24]
   -<PBuf3 [7:0]> = Addr [23:16]
   -<PBuf4 [7:0]> = Addr [15:8]
   -<PBuf5 [7:0]> = Addr [7:0]

2. The read data from SPI memory is stored in the secondary buffer after indirect access is completed.
   -<SBuf0 [7:0]> = Data [7:0] (Data read from Addr [31:0] + 0x00)
   -<SBuf1 [7:0]> = Data [15:8] (Data read from Addr [31:0] + 0x01)
   -
   -<SBuf254 [7:0]> = Data [2039: 2032] (Data read from Addr [31:0] + 0xFE)
   -<SBuf255 [7:0]> = Data [2047: 2040] (Data read from Addr [31:0] + 0xFF)

3. ***[SMIxRACR1]*** settings
   -<PBufEn> = 1, <PBufBc [4:0]> = 11001
   -<SBufEn> = 1, <SBufBc [7:0]> = 0xFF
   -<CSNum> = chip select number connected to SPI memory

4. ***[SMIxIOCR]*** settings
   -<PBufIOCtrlEn [1:0]> = 11 (8×I/O communication)
   -<SBufIOCtrlEn [1:0]> = 11 (8×I/O communication)
   -<PBufIOCtrl [4:0]> = 00000 (All multi I/O communication)

5. ***[SMIxOECR]*** settings
   -<PBufOCEn> = 1
   -<PBufOEnC [4:0]> = 00110 (Switch cycle to input from dummy byte)

6. Set ***[SMIxRACR1]*** <CycGo> to "1" to start indirect access.

When the indirect access is completed, the data read from the SPI memory is stored in <SBuf0 [7:0]> to <SBuf255 [7:0]>.

Note: The actual number of dummy bytes to be set depends on the model number of the SPI memory and the setting of its configuration register.

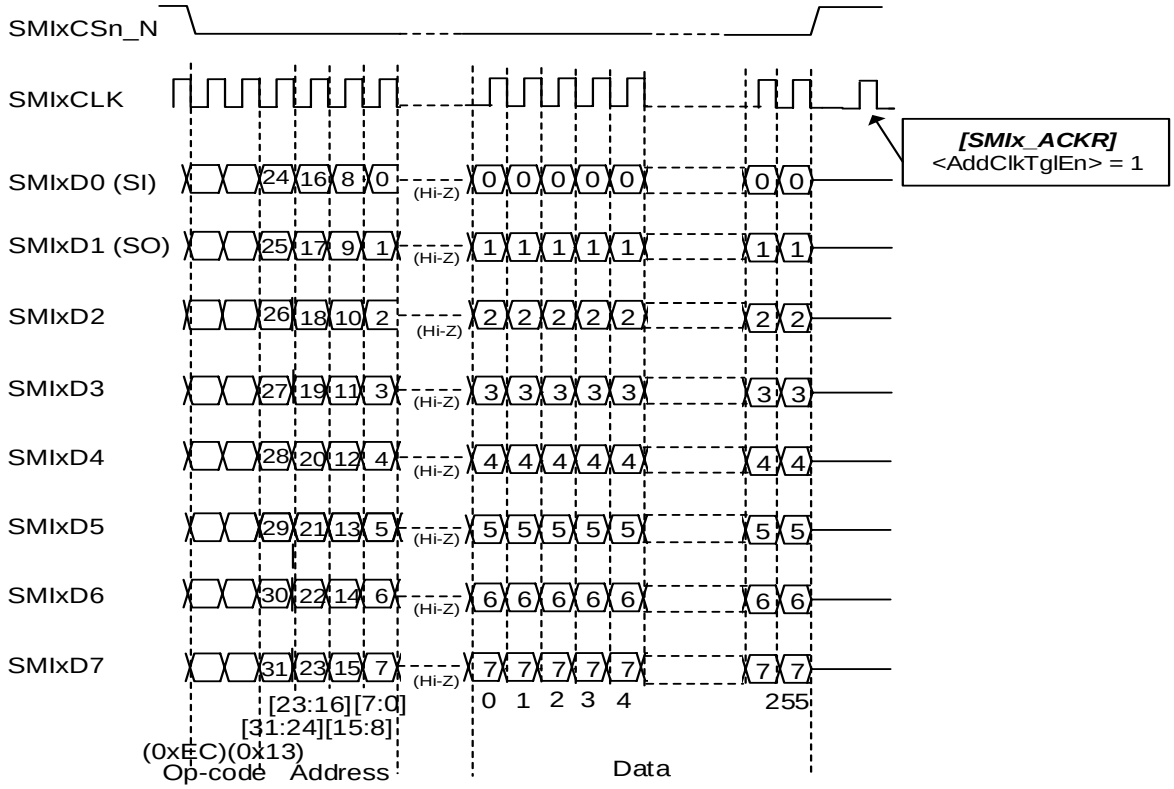

Figure 5.10 OPI FAST READ

5.4. Others

5.4.1. Forced Stop

A forced stop is a process for urgently stopping the operation of SMIF. When the communication with the SPI memory is in progress, the communication is aborted and the chip select signal is set to “High”. At this time, communication data is not guaranteed.

Even after a forced stop, the internal registers of SMIF can be accessed. However, SMIF can’t communicate to SPI memory. To re-communicate, it is necessary to perform software reset and configure communication settings again.

The procedure for performing a forced stop is shown below.

1. If using interrupt, write “1” to ***[SMIxINT]*** <SDIntEn>.
2. Write “1” to ***[SMIxSTPR]*** <StopEn>.
3. Wait for an interrupt or until ***[SMIxSTAT]*** <StpProgDone> is set to “1”.
4. Write “0” to ***[SMIxSTAT]*** <StpProgDone>
5. Perform a software reset according to the procedure in section 5.4.2**.**

5.4.2. Software Reset

The procedure for performing a software reset is shown below.

1. Check that both ***[SMIxSTAT]*** <DirAccInPrgrs> and <CycPrgrs> are “0”.
2. Write “1” to ***[SMIxSWR]*** <RstEn>.
3. Wait until “0” is read from ***[SMIxSWR]*** <RstEn>.

Note1: When you perform a forced stop, step 1 can be omitted.
Note2: During this procedure, a new direct access or indirect access request must not be issued to SMIF.

5.5. Connection Example with Serial Memory

The following shows an example of connection with serial memory.
Each pin is assumed to be processed externally as shown below. Determine the processing of each pin according to the actual connected memory and external circuit.

SMIxCS0_N, SMIxCS1_N: Pull-up
SMIxCLK: Pull-down
SMIxD0,1,2,3: Pull-up (Note)

Note: When SMIxD2 and SMIxD3 are not used for connection to serial memory, they can be used as input/output ports.

5.5.1. Connection example with Standard SPI

1) Serial memory 0

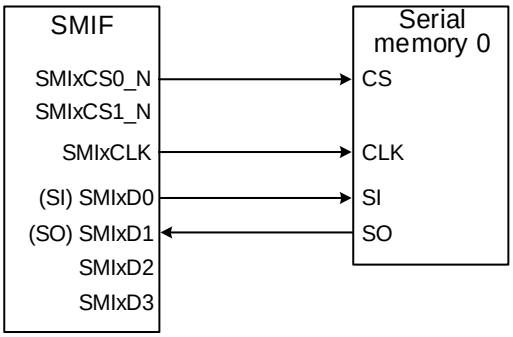

Figure 5.11 Connection example with Standard SPI (Serial memory 0)

2) Serial memory 0, Serial memory 1

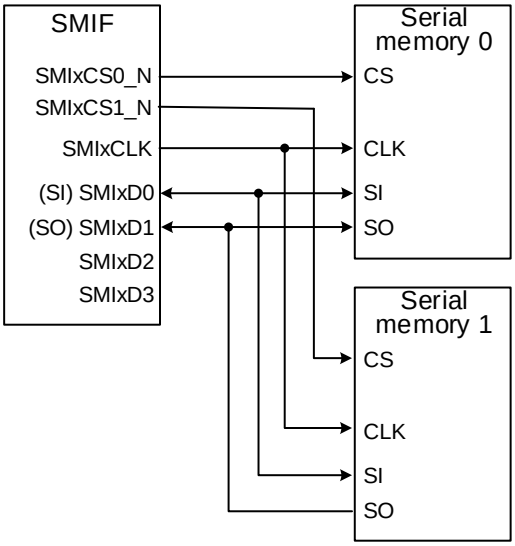

Figure 5.12 Connection example with Standard SPI (Serial memory 0, 1)

5.5.2. Connection example with Quad/QPI

1) Serial memory 0

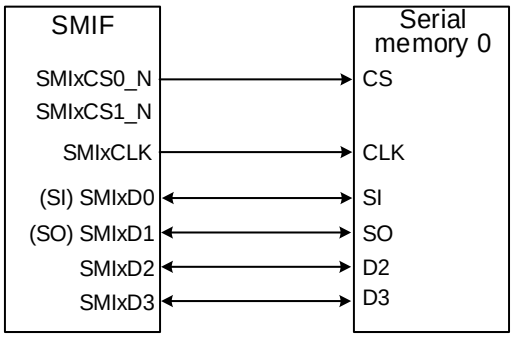

Figure 5.13 Connection example with Quad/QPI (Serial memory 0)

2) Serial memory 0, Serial memory 1

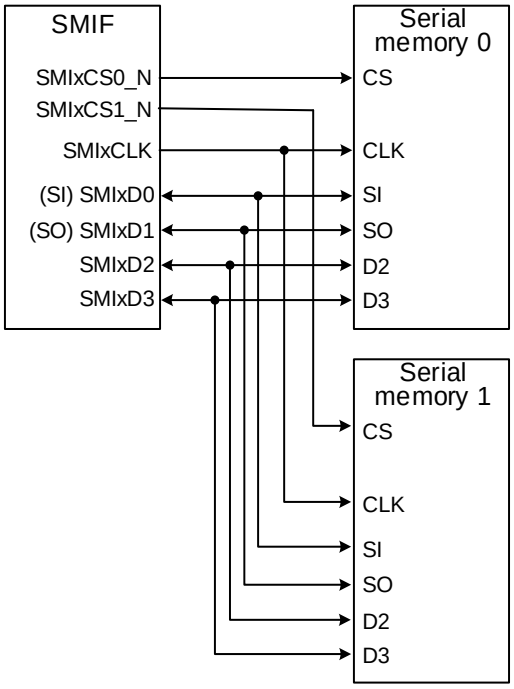

Figure 5.14 Connection example Quad/QPI (Serial memory 0, 1)

5.5.3. Connection example with Octal/OPI

1) Serial memory 0

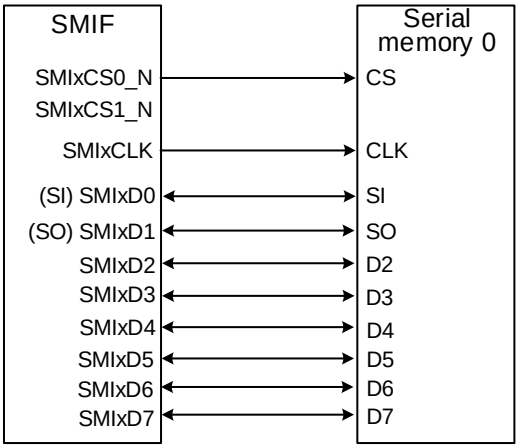

Figure 5.15 Connection example with Octal/OPI (Serial memory 0)

2) Serial memory 0, Serial memory 1

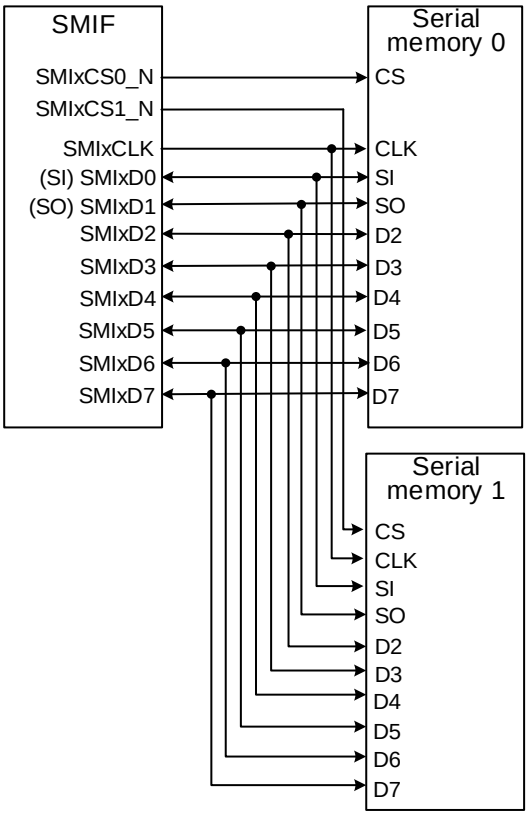

Figure 5.16 Connection example with Octal/OPI (Serial memory 0, 1)

6. Precaution for Usage

- Do not access addresses to which no registers have been assigned.
- Some products do not have SMIxCSn_N pin and SMIxDn pin.

7. Revision History

Table 7.1 Revision History

Revision	Date	Description
1.0	2020-11-16	First release
1.1	2021-02-01	-Table 2.1 SMIxCLK,SMIxCS0_N,SMIxCS1_N : 'I/O' to 'Output' -4.2.1 FBA[11:0] 0x000～0x0FF → 0x000～0x7FF -4.2.2 FBA[11:0] 0x000～0x0FF → 0x000～0x7FF -4.2.4 bit 7:0 SPI → Single Quad/QPI → Quad Octal/OPI → Octal -4.2.5 bit 7:0 SPI → Single Quad/QPI → Quad Octal/OPI → Octal -4.2.8 bit 3:0 SPI → Single Quad/QPI → Quad Octal/OPI → Octal -4.2.14. (SMIxCLK,CS pin direction Control register) to (SMIxCLK/CS Output Enable Control Register) -4.2.14. Controls the I/O direction of SMIxCS1_N. 0: input 1: output change to Controls the Output Enable of SMIxCS1_N. 0: Disabled(Hi-z) 1: Enable Controls the I/O direction of SMIxCS0_N. 0: input 1: output change to Controls the Output Enable of SMIxCS0_N. 0: Disabled(Hi-z) 1: Enable Controls the I/O direction of SMIxCLK 0: input 1: output change to Controls the Output Enable of SMIxCLK 0: Disabled(Hi-z) 1: Enable

Revision	Date	Description
1.2	2023-03-31	- 3. Function and Operation Changed description. - 3.3. Memory Mapping Changed description. - 3.4. Access Mode Changed description. - 3.4.1.1. SPI Flash Command Changed Figure 3.2 to Figure 3.8 - 3.4.3. Setting Procedure of indirect Access Mode Deleted Note. Added Note1,2. - 4.1. List of Registers Changed table (0x051C → 0x051F, 0x06FC → 0x06FF) 4.2.5. ***[SMIxDWCRn]*** (Direct Write Control Register n) (n=0,1) Changed table (changed description column for bit 15:11) - 4.2.7. ***[SMIxRACR1]*** (Indirect Access Control Register 1) Changed table. Added Note2 to Note4. - 4.2.8. ***[SMIxIOCR]*** (Indirect Access I/O Control Register) Changed table. Added Note. - 4.2.9. ***[SMIxOECR***] (Indirect Access Output Enable Control Register) Changed table. Deleted Note. Added Note1,2. - 5.1.2. Direct Access serial memory 0 setting Changed description. - 5.1.4. Indirect Access Setting Changed description. - 5.3.1. Programming Method for Indirect Access Changed description. - 5.3.2. Programming example: PAGE PROGRAM, WRITE Changed figure 5.1. - 5.3.3. Programming example: Quad I/O PAGE PROGRAM, Quad I/O WRITE Changed figure 5.2. (unified to (Hi-Z) notation) - 5.3.5. Programming example: Octal I/O PAGE PROGRAM, Octal I/O WRITE Changed figure 5.4. (unified to (Hi-Z) notation) - 5.3.10. 5.3.10. Programming example: FAST READ Changed figure 5.6. - 5.3.11. Programming example: Quad I/O FAST READ Changed figure 5.7. - 5.3.12. Programming example: QPI FAST READ Changed figure 5.8. - 5.3.13. Programming example: Octal FAST READ Changed figure 5.9. (unified to (Hi-Z) notation) - 5.3.14. Programming example: OPI FAST READ Changed figure 5.10. - 5.5.1. Connection example with Standard SPI Changed figure 5.11.
1.3	2023-06-19	- 4.2.6. ***[SMIxRACR0]*** (Indirect Access Control Register 0) Changed initial value of <SPR[4:0]> to “11111”.

RESTRICTIONS ON PRODUCT USE

Toshiba Corporation and its subsidiaries and affiliates are collectively referred to as “TOSHIBA”.
Hardware, software and systems described in this document are collectively referred to as “Product”.

- TOSHIBA reserves the right to make changes to the information in this document and related Product without notice.
- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA's written permission, reproduction is permissible only if reproduction is without alteration/omission.
- Though TOSHIBA works continually to improve Product's quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before customers use the Product, create designs including the Product, or incorporate the Product into their own applications, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the "TOSHIBA Semiconductor Reliability Handbook" and (b) the instructions for the application with which the Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating all operating parameters for such designs and applications. **TOSHIBA ASSUMES NO LIABILITY FOR CUSTOMERS' PRODUCT DESIGN OR APPLICATIONS.**
- **PRODUCT IS NEITHER INTENDED NOR WARRANTED FOR USE IN EQUIPMENTS OR SYSTEMS THAT REQUIRE EXTRAORDINARILY HIGH LEVELS OF QUALITY AND/OR RELIABILITY, AND/OR A MALFUNCTION OR FAILURE OF WHICH MAY CAUSE LOSS OF HUMAN LIFE, BODILY INJURY, SERIOUS PROPERTY DAMAGE AND/OR SERIOUS PUBLIC IMPACT ("UNINTENDED USE").** Except for specific applications as expressly stated in this document, Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, lifesaving and/or life supporting medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, and devices related to power plant. **IF YOU USE PRODUCT FOR UNINTENDED USE, TOSHIBA ASSUMES NO LIABILITY FOR PRODUCT.** For details, please contact your TOSHIBA sales representative or contact us via our website.
- Do not disassemble, analyze, reverse-engineer, alter, modify, translate or copy Product, whether in whole or in part.
- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.
- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.
- **ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.**
- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the applicable export laws and regulations including, without limitation, the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.
- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. **TOSHIBA ASSUMES NO LIABILITY FOR DAMAGES OR LOSSES OCCURRING AS A RESULT OF NONCOMPLIANCE WITH APPLICABLE LAWS AND REGULATIONS.**