HB0089
Handbook
CoreSPI v5.2

Microsemi
Power Matters.

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, California, and has approximately 4,800 employees globally. Learn more at www.microsemi.com.

51700089. 6.0 4/18

Contents

1 Revision History . . . 1
1.1 Revision 6.0 . . . 1
1.2 Revision 5.0 . . . 1
1.3 Revision 4.0 . . . 1
1.4 Revision 3.0 . . . 1
1.5 Revision 2.0 . . . 1
1.6 Revision 1.0 . . . 1

2 Introduction . . . 2
2.1 Features . . . 2
2.2 Core Version . . . 2
2.3 Supported Families . . . 2
2.4 Supported Interfaces . . . 3
2.5 Utilization and Performance . . . 3

3 Design Description . . . 4
3.1 Verilog/VHDL Parameters . . . 4
3.2 I/O Signals . . . 5
3.3 Register Map and Descriptions . . . 7
3.3.1 Register Summary . . . 7
3.3.2 Control Register 1 . . . 9
3.3.3 Interrupt Clear Register . . . 10
3.3.4 RX Data Register . . . 10
3.3.5 Interrupt Masked Register . . . 11
3.3.6 Interrupt Raw Register . . . 11
3.3.7 Control Register 2 . . . 12
3.3.8 Command Register . . . 12
3.3.9 Status Register . . . 13
3.3.10 Slave Select Register . . . 13
3.3.11 Aliased TX Data Register . . . 14
3.3.12 Clock Rate Register . . . 14

4 Design Details . . . 15
4.1 Transfer Protocols . . . 15
4.1.1 SPI Transfer Protocols . . . 15
4.1.2 Output Enable Control . . . 20
4.1.3 Transmission of Multiple Frames . . . 20
4.1.4 APB Interface Timing . . . 20

5 Tool Flows . . . 22
5.1 Licensing . . . 22
5.1.1 RTL . . . 22
5.2 SmartDesign . . . 22
5.3 Design Migration . . . 23
5.4 Simulation Flows . . . 26
5.4.1 User Testbench . . . 26
5.5 Synthesis . . . 27
5.6 Place-and-Route . . . 27

6 System Integration 28

Figures

Figure 1 CoreSPI I/O Signal Diagram . . . 2
Figure 2 Single-Frame Transfer in Motorola Mode 0 . . . 15
Figure 3 Multi-Frame Transfer in Motorola Mode 0 . . . 15
Figure 4 Single Frame Transfer in Motorola Mode 1 . . . 16
Figure 5 Single Frame Transfer in Motorola Mode 2 . . . 16
Figure 6 Single Frame Transfer in Motorola Mode 3 . . . 16
Figure 7 Single-Frame Transfer in NSC SPI Mode . . . 17
Figure 8 Multi Frame Transfer in NSC SPI Mode (Master Mode Operation) . . . 17
Figure 9 Continuous Transfer in NSC Mode for Large Frames . . . 18
Figure 10 Continuous Transfer in NSC Mode with Idle Cycle . . . 18
Figure 11 Single Frame Transfer in TI Synchronous SPI Mode . . . 19
Figure 12 Continues Transfer in TI Synchronous SPI Mode . . . 19
Figure 13 Continuous Transfer in TI mode for Jumbo Frames . . . 19
Figure 14 APB Data Write Cycle . . . 21
Figure 15 APB Data Read Cycle . . . 21
Figure 16 CoreSPI Full I/O View . . . 22
Figure 17 CoreSPI SmartDesign Configuration to Associate Parameters . . . 23
Figure 18 CoreSPI User Testbench . . . 27
Figure 19 CoreSPI Integration . . . 28

Tables

Table 1 CoreSPI Device Utilization and Performance . . . 3
Table 2 CoreSPI General Parameter and Generic Descriptions . . . 4
Table 3 CoreSPI Serial Transfer Parameter and Generic Descriptions . . . 4
Table 4 CoreSPI I/O Signal Descriptions . . . 5
Table 5 CoreSPI Internal Register Address Map . . . 7
Table 6 Control Register 1 . . . 9
Table 7 Control Register 1 Bit Definition . . . 9
Table 8 Interrupt Clear Register . . . 10
Table 9 Interrupt Clear Register Bit Definition . . . 10
Table 10 RX Data Register . . . 10
Table 11 TX Data Register . . . 10
Table 12 Interrupt Masked Register . . . 11
Table 13 Interrupt Raw Register . . . 11
Table 14 Interrupt Raw Register Bit Definition . . . 11
Table 15 Control Register 2 . . . 12
Table 16 Control Register 2 Bit Definition . . . 12
Table 17 Command Register . . . 12
Table 18 Command Register Bit Definition . . . 12
Table 19 Status Register . . . 13
Table 20 Status Register Bit Definition . . . 13
Table 21 Slave Select Register . . . 13
Table 22 Aliased TX Data Register . . . 14
Table 23 Clock Rate Register . . . 14
Table 24 Motorola Mode - Default Slave Select Behavior . . . 16
Table 25 NSC SPI mode bit mapping . . . 17
Table 26 Output Enable Behavior . . . 20
Table 27 Parameter Migration Table . . . 24
Table 28 Configuration GUI parameter mapping . . . 26

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 6.0

The following is a summary of the changes in revision 5.0 of this document.

- Updated Core version to v5.2
- Updated Register Map and Description section with new register CLK_DIV.

1.2 Revision 5.0

The following is a summary of the changes in revision 5.0 of this document.

- Updated Core version to v5.1
- Updated reset value of registers in Table 5, page 7.

1.3 Revision 4.0

Updated Table 1, page 3 with the RTG4 device information.

1.4 Revision 3.0

The following is a summary of the changes in revision 3.0 of this document.

- Updated Core version to v5.0. Support for RTG4 devices added to the supported families section. Updated resource Utilization data.
- Updated Table 2, page 4 and added Table 3, page 4, detailing the descriptions of the generics/parameters used to configure the SPI transfer.
- Renamed RXAVAIL interrupt source to DATA_RX in Table 9, page 10, Table 14, page 11, and Table 16, page 12.
- Updated Table 14, page 11 to reflect new CoreSPI configurator GUI in v5.0.

1.5 Revision 2.0

The following is a summary of the changes in revision 2.0 of this document.

- The Core Version was updated to v4.2. Supports for the SmartFusion2 and IGLOO2 devices were added to the supported families section.
- Fields modified within Interrupt Raw and Interrupt Clear Registers.
- Updated Tool Flows, page 22 section.

1.6 Revision 1.0

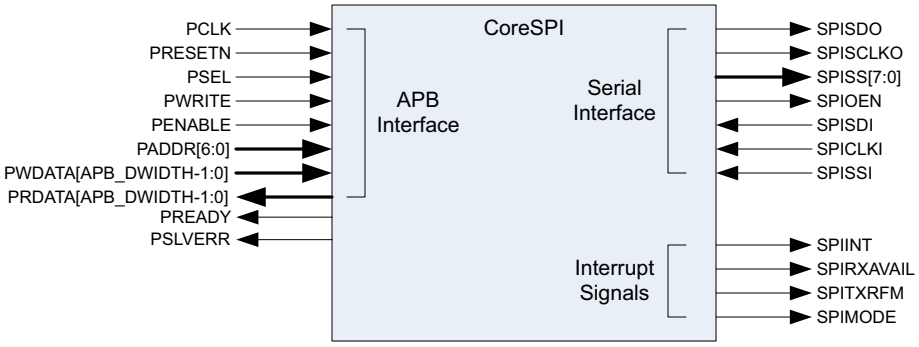
Revision 1.0 was the first publication of this document.

2 Introduction

CoreSPI is a controller core designed for synchronous serial communication using a Motorola, TI, or NSC serial peripheral interface (SPI). The core is parameterized to allow user specification of the operating mode, FIFO depth, and frame width. Operation is fully synchronous and operates on the system clock as well as the external SPI clock for slave mode.

CoreSPI consists of an advanced peripheral bus (APB) interface designed to connect to an APB bus. Its registers, including transmit and receive FIFOs can be accessed by an APB master.

Figure 1 • **CoreSPI I/O Signal Diagram**



2.1 Features

Following are the key features of CoreSPI:

- SPI clock rate configurable
  - From PCLK/512 to PCLK/2 in steps of 2
  - Maximum data rate of PCLK/2 in Master mode and PCLK/8 in Slave mode.
- SPI protocol configurable
  - Master and slave operation
  - Supports up to eight slaves
  - Motorola SPI support
  - TI SPI support
  - NSC SPI support
  - Slave select behavior configurable during Idle cycles
  - Supports broadcast operation
  - Configurable frame size (4 to 32 bits)
- FIFO
  - Width set to frame size for optimal core size
  - Depth configurable through the parameter
- Interrupt generation
  - Receive/transmit data interrupts
  - FIFO overflow and under run
  - Command transmitted interrupt
- APB3 compliant

2.2 Core Version

This handbook provides information on CoreSPI version 5.2.

2.3 Supported Families

CoreSPI version 5.2 is a generic core and supports all the device families.

2.4 Supported Interfaces

CoreSPI is available with the following interfaces:

- APB slave interface
- Interrupt request interface
- Serial (SPI) interface

These interfaces are further described in I/O Signals, page 5.

2.5 Utilization and Performance

Utilization and performance data is listed in Table 1, for several Microsemi device families. The data listed in this table is indicative only. The overall device utilization and performance of the core is system dependent.

Table 1 • **CoreSPI Device Utilization and Performance**

		Utilization				
Family	**Device**	**Sequential**	**Combinatorial**	**Total**	**%**	**Performance PCLK (MHz)**
Fusion	AFS600	226	542	768	5.56	81
SmartFusion	A2F500M3G	226	542	768	6.67	81
ProASIC3 / E	A3P600 / A3PE600	226	542	768	5.56	81
ProASIC3L	A3P600L	226	545	771	5.58	68
IGLOO / e	AGL600V2 / AGLE600V2	227	549	776	5.61	34
IGLOO+	AGLP125V2	227	549	776	24.87	34
Axcelerator	AX500	234	337	571	7.08	100
RTAX-S	RTAX1000S	234	343	577	3.18	127
SmartFusion2 / IGLOO2	M2S050 / M2GL050	258	404	662	0.59	160
RTG4	RT4G150	258	421	679	0.22	73
PolarFire	MPF300T_ES	210	347	557	0.09	182

Note: Data in this table was achieved using the Verilog RTL with typical synthesis and layout settings. Frequency (in MHz) was set to 100 and speed grade was standard. Top-level parameters/generics were set as follows: APB_DWIDTH = 8, CFG_FRAME_SIZE = 4, CFG_FIFO_DEPTH = 4, CFG_CLK = 7, CFG_MODE = 0, CFG_MOT_MODE = 0, CFG_MOT_SSEL = 0, CFG_TI_NSC_CUSTOM = 0, CFG_TI_NSC_FRC = 0, CFG_TI_JMB_FRAMES = 0, CFG_NSC_OPERATION = 0.

3 Design Description

3.1 Verilog/VHDL Parameters

CoreSPI has parameters (Verilog) or generics (VHDL) for configuring the RTL code, described in Table 2, page 4 and Table 3, page 4. All parameters and generics are integer types. The parameters listed in the following table are used to configure the general features of CoreSPI.

Table 2 • **CoreSPI General Parameter and Generic Descriptions**

Parameter Name	Valid Range	Default	Description
APB_DWIDTH	8, 16, 32	8	APB data width can be 8, 16, or 32 bits. Operation in NSC mode is only possible with an APB data width of 32.
CFG_FRAME_SIZE	4 to 32	4	SPI frame size, in bits. For Motorola and TI modes, this is the actual required frame size but for NSC mode this is set to 9 + the required data frame size.
CFG_FIFO_DEPTH	1 to 32	4	Number of frames that can be stored in the FIFO at any given time (both TX and RX FIFOs)
CFG_CLK	0 to 255	7	Clock rate parameter, which determines the generated SPI master clock by: SPICLK = PCLK/(2*(CFG_CLK+1))

The parameters listed in the following table are used to configure the SPI serial transfer variant that CoreSPI communicates in.

Table 3 • **CoreSPI Serial Transfer Parameter and Generic Descriptions**[1]

Parameter Name	Valid Range	Default	Dependencies	Description
CFG_MODE	0 – 2	0	N/A	Determines operating mode: 0: Motorola mode 1: TI mode 2: NSC mode
CFG_MOT_MODE	0 – 3	0	CFG_MODE = 0	Motorola mode selection: 0: Mode 0 1: Mode 1 2: Mode 2 3: Mode 3
CFG_MOT_SSEL	0 – 1	0	CFG_MODE = 0	Slave select active between back to back transfers in Motorola mode. 0: Slave select behavior varies depending on the Motorola mode selected. Refer to Table 27, page 24 for more details. 1: Active – Remains active between back to back transfers
CFG_TI_NSC_CUSTOM	0 – 1	0	CFG_MODE = 1 or 2	Enable custom transfer configuration in TI/NSC mode. 0: Normal transfer 1: Custom transfer

Table 3 • **CoreSPI Serial Transfer Parameter and Generic Descriptions[1] *(continued)***

Parameter Name	Valid Range	Default	Dependencies	Description
CFG_TI_NSC_FRC	0 – 1	0	CFG_MODE = 1 or 2 CFG_TI_NSC_CUSTOM = 1	Free running clock in TI/NSC mode. 0: Clock in-active between transfers 1: Clock remains active
CFG_TI_JMB_FRAMES	0 – 1	0	CFG_MODE = 1 CFG_TI_NSC_CUSTOM = 1	Concatenate frames in a TI mode back-to-back transfer 0: Standard TI transfers 1: Jumbo frame transfers
CFG_NSC_OPERATION	0 – 2	0	CFG_MODE = 2 CFG_TI_NSC_CUSTOM = 1	NSC specific transfer settings 0: Standard NSC transfers 1: Idle cycles inserted between frames in back-to-back transfers 2: Large response frames. Response frames stored in the TX_FIFO are concatenated to form a single large response frame.

1. Refer to Table 28, page 26 to determine how these parameters are configured in the CoreSPI configuration GUI.

3.2 I/O Signals

The port signals for the CoreSPI macro are shown in Figure 1, page 2 and listed in the following table.

Table 4 • **CoreSPI I/O Signal Descriptions[1]**

Port Name	Type	Description
APB Interface		
PCLK	Input	APB system clock; Reference clock for all internal logics
PRESETN	Input	APB active low asynchronous reset.
PADDR[6:0]	Input	APB address bus; address internal registers.
PSEL	Input	APB slave select; select signal to registers for APB reads and writes.
PENABLE	Input	APB strobe. This signal indicates the second cycle of an APB transfer.
PWRITE	Input	APB write or read. If high, a write occurs when an APB transfer takes place. If low, a read takes place.
PWDATA[APB_DWIDTH-1:0]	Input	APB write data
PRDATA[APB_DWIDTH-1:0]	Output	APB read data
PREADY	Output	APB ready. Used to insert wait states. Not used in CoreSPI but part of APB3 interface. Tied high (always ready).
PSLVERR	Output	APB Error. Not used in CoreSPI but part of APB3 interface. Tied Low.
Interrupts		
SPIINT	Output	Interrupt pending: This active high output signal is the interrupt output signal from CoreSPI. It can be programmed to become active on certain events, informing the CPU that such an event has occurred. The CPU can then take appropriate action.

Table 4 • **CoreSPI I/O Signal Descriptions[1] *(continued)***

Port Name	Type	Description
SPIRXAVAIL	Output	Indicates receive data is available to be read. This signal is normally used for DMA transfer control but there is a corresponding maskable interrupt available through the SPIINT signal for software detection of the condition.
SPITXRFM	Output	Indicates that transmit FIFO is not full – ready for more data. This signal is normally used for DMA transfer control but there is a corresponding maskable interrupt available through the SPIINT signal for software detection of the condition.
Serial (SPI) Interface		
SPISSI	Input	Slave select in: Used to address CoreSPI in Slave mode. Active low in Motorola and NSC modes and active high in TI mode.
SPISDI	Input	Serial data in: Shift data input, Master or Slave mode.
SPICLKI	Input	Serial clock in: Shift clock input, for Slave mode operation.
SPISS[7:0]	Output	Slave select: Generated by CoreSPI in master mode. Active low in Motorola and NSC modes and active high in TI mode.
SPISCLK0	Output	Serial clock out: Generated by SPI in Master mode.
SPIOEN	Output	Data output enable: When de-asserted output pad for SPISDO tri-stated). This is active when the SPI is writing output data and deactivated when there is not data to write. This signal is active high.
SPISDO	Output	Serial data out: Data output in Master or Slave mode.
SPIMODE	Output	CoreSPI mode indicator: When 1 – master and 0 – slave.

1. All signals are active high (logic 1) unless otherwise noted.

3.3 Register Map and Descriptions

This section describes the registers that are used in CoreSPI.

3.3.1 Register Summary

Values listed in the following tables are in hexadecimal format; type designations: R = read only; W = write only; R/W = read/write.

Table 5 • **CoreSPI Internal Register Address Map**

Address	Register Name	Type	Width	Reset Value	Description
0x00	CONTROL	R/W	8	0x00	Control Register 1
0x04	INTCLEAR	W	8	0x00	Interrupt Clear Register
0x08	RXDATA	R	32	0x00	Receive Data Register Reading from this register reads one frame from the RX FIFO.
0x0C	TXDATA	W	32	0x00	Transmit Data Register Writing to this register writes one frame to the TX FIFO.
0x10	INTMASK	R	8	0x00	Masked interrupt status These bits indicate the masked interrupt status by ANDING the interrupt enables in the CONTROL and CONTROL2 registers with the raw interrupt register. When any of these bits are set, the INTERRUPT output will be active. The bits are cleared by writing to the Interrupt clear register.
0x14	INTRAW	R	8	0x80	Raw interrupt status
0x18	CONTROL2	R/W	8	0x00	Control Register 2
0x1C	COMMAND	W	8	0x00	Command Register
0x20	STAT	R	8	0x44	Status Register
0x24	SSEL	R/W	8	0x00	Slave Select Register Specifies the slaves selected Default 0 (nothing selected). Write 1 to each bit to select one or more slaves. Slave select output pin is active Low. In TI mode the slave select outputs are inverted to become active High.
0x28	TXDATA_LAST	W	32	0x00	Transmit Data Register Writing to this register writes one frame to the TX FIFO. Also indicates to CoreSPI that this is the last frame in this packet before SSEL is supposed to go inactive, effectively allowing for the specification of the number of transmitted frames.

Table 5 • **CoreSPI Internal Register Address Map**

Address	Register Name	Type	Width	Reset Value	Description
0x2C	CLK_DIV	R/W	8	CFG_CLK	Clock rate register. Writing to this register will update clock division factor of CoreSPI generated clock (SPICLKO) in master mode.

3.3.2 Control Register 1

Table 6 • **Control Register 1**

PADDR[5:0]	Register Name	Type	Width	Reset Value	Description
0x00	CONTROL	R/W	8	0x00	Control Register 1

Table 7 • **Control Register 1 Bit Definition**

Bits	Name	Type	Description
7	OENOFF	R/W	0: SPI output enable active as required 1: The core will not assert the SPI output enable. This allows multiple slaves to be connected to a single master sharing a single slave select and software protocol implemented that can enable the slaves transmit data when a certain broadcast address SPI command is received.
6	FRAMEURUN	R/W	0: Under runs are generated whenever a read is attempted from an empty transmit FIFO 1: Under run condition will be ignored for the complete frame if the first data frame read resulted in a potential overflow, that is, the slave was not ready to transmit any data. If the first data frame is read from the FIFO and transmitted then an under run will be generated if the FIFO becomes empty for any of the remaining packet frames, that is, while SSEL is active. Master operation will never create a transmit FIFO under run condition.
5	INTTXURUN	R/W	Interrupt on transmit under run 0: Interrupt disabled 1: Interrupt enabled.
4	INTRXOVFLOW	R/W	Interrupt on receive overflow 0: Interrupt disabled 1: Interrupt enabled.
3	INTTXDONE	R/W	Interrupt on transmit data of data which has been placed in TX FIFO through the TXDATA_LAST register. 0: Interrupt disabled 1: Interrupt enabled.
2	–	–	Reserved
1	MASTER	R/W	0: Run CoreSPI in Slave mode 1: Run CoreSPI in Master mode
0	ENABLE	R/W	0: Core does not respond to external signals until this bit is enabled. SPISCLKO driven to zero and SPIOEN, SPISS (slave select) driven inactive. 1: Core is active

3.3.3 Interrupt Clear Register

Table 8 • Interrupt Clear Register

PADDR[5:0]	Register Name	Type	Width	Reset Value	Description
0x04	INTCLEAR	W	8	0x00	Interrupt Clear Register

Table 9 • Interrupt Clear Register Bit Definition

Bits	Name	Type	Description
7	TXRFM	W	Writing 1 clears the TXRFM interrupt
6	DATA_RX	W	Writing 1 clears the DATA_RX interrupt
5	SSEND	W	Writing 1 clears the SSEND interrupt.
4	CMDINT	W	Writing 1 clears the CMDINT interrupt.
3	TXUNDERRUN	W	Writing 1 clears the TXUNDERRUN interrupt.
2	RXOVERFLOW	W	Writing 1 clears the RXOVERFLOW interrupt.
1	–	–	Reserved
0	TXDONE	W	Writing 1 clears the TXDONE interrupt.

3.3.4 RX Data Register

Table 10 • RX Data Register

PADDR[5:0]	Register Name	Type	Width	Reset Value	Description
0x08	RXDATA	R	32	0x00	Receive Data Register Reading from this register reads one frame from the RX FIFO

Table 11 • TX Data Register

PADDR[5:0]	Register Name	Type	Width	Reset Value	Description
0x0C	TXDATA	W	32	0x00	Transmit Data Register Writing to this register writes one frame to the TX FIFO.

3.3.5 Interrupt Masked Register

Table 12 • **Interrupt Masked Register**

PADDR[5:0]	Register Name	Type	Width	Reset Value	Description
0x10	INTMASK	R	8	0x00	Masked interrupt status These bits indicate the masked interrupt status by ANDING the interrupt enables in the CONTROL registers with the raw interrupt register. When any of these bits are set, the INTERRUPT output will be Active. The bits are cleared by writing to the Interrupt clear register.

3.3.6 Interrupt Raw Register

Table 13 • **Interrupt Raw Register**

PADDR[5:0]	Register Name	Type	Width	Reset Value	Description
0x14	INTRAW	R	8	0x80	Raw interrupt status

Table 14 • **Interrupt Raw Register Bit Definition[1]**

Bits	Name	Type	Description
7	TXRFM	R	Indicates that there is at least one frame free in the transmit FIFO for writing.
6	DATA_RX	R	Indicates that at least one byte is received. Check the RXEMPTY bit in the Status register to determine if there is more RX data available in the RX FIFO. Writing a 1 to the corresponding bit in the Interrupt clear register clears this bit, provided that the RX FIFO is empty.
5	SSEND	R	Indicates that SSEL is Inactive.
4	CMDINT	R	Indicates that the number of frames set by the CMDSIZE register have been received as a single packet of frames (SSEL held active).
3	TXUNDERRUN	R	Indicates that in Slave mode that the data was not available when required in the transmit FIFO.
2	RXOVERFLOW	R	Indicates that in Master and Slave mode the receive FIFO is overflowed.
1	–	–	Reserved
0	TXDONE	R	Indicates that all frames, including last frame (see Aliased TX Data Register), have been transmitted

1. Writing a 1 to the corresponding bit in the Interrupt Clear register clears the associated Interrupt Raw register bit, provided that the hardware condition that triggered the interrupt in the first instance is resolved.

3.3.7 Control Register 2

Table 15 • **Control Register 2**

PADDR[5:0]	Register Name	Type	Width	Reset Value	Description
0x18	CONTROL2	R/W	8	0x00	Control Register 2

Table 16 • **Control Register 2 Bit Definition**

Bits	Name	Type	Description
7	INTEN_TXRFM	R/W	0: No effect 1: Enables the interrupt when there is room in the TX FIFO.
6	INTEN_DATA_RX	R/W	0: No effect 1: Enables the interrupt when at least one byte is received.
5	INTEN_SSEND	R/W	0: No effect 1: Enables the interrupt as SSEL goes High. SPI master and slave modes.
4	INTEN_CMD	R/W	0: No effect 1: Enables Interrupt after the number of frames set by CMDSIZE (above) has been received as a single packet of frames (SSEL held active).
3	–	–	Reserved
2 : 0	CMDSIZE	R/W	Number of frames sent before interrupt is generated when INTEN_CMD is set (see above).

3.3.8 Command Register

Table 17 • **Command Register**

PADDR[5:0]	Register Name	Type	Width	Reset Value	Description
0x1C	COMMAND	W	8	0x00	Command Register (write-only)

Table 18 • **Command Register Bit Definition**

Bits	Name	Type	Description
7 : 2	–	–	Reserved
1	TXFIFORST	W	Writing 1 will reset the TX FIFO. This bit always reads as zero
0	RXFIFORST	W	Writing 1 will reset the RX FIFO This bit always reads as zero

3.3.9 Status Register

Table 19 • **Status Register**

PADDR[5:0]	Register Name	Type	Width	Reset Value	Description
0x20	STAT	R	8	0x44	Status Register (read-only)

Table 20 • **Status Register Bit Definition**

Bits	Name	Type	Description
7	ACTIVE	R	Core is still transmitting data
6	SSEL	R	Current state of SSEL
5	TXUNDERRUN	R	Transmit FIFO under flowed
4	RXOVFLOW	R	Receive FIFO overflowed
3	TXFULL	R	Transmit FIFO is full, that is, no space for more data
2	RXEMPTY	R	Receive FIFO is empty, that is, no data available to read
1	DONE	R	No of requested frames have been transmitted and received.
0	FIRSTFRAME	R	Next frame in Receive FIFO was first received after SSEL went active (Command Frame)

3.3.10 Slave Select Register

Table 21 • **Slave Select Register**

PADDR[5:0]	Register Name	Type	Width	Reset Value	Description
0x24	SSEL	R/W	8	0x00	Slave Select Register Specifies the slaves selected. Default 0 (nothing selected). Write 1 to each bit to select one or more slaves. Slave select outputs are active low in Motorola and NSC modes. In TI mode the slave select outputs are inverted to become active High. For example, to select Slave 0 and Slave 5, write the value 0x00100001 to this register.

3.3.11 Aliased TX Data Register

Table 22 • **Aliased TX Data Register**

PADDR[5:0]	Register Name	Type	Width	Reset Value	Description
0x28	TXDATA_LAST	W	32	0x00	Transmit Data Register Writing to this register writes one frame to the TX FIFO. Also indicates to CoreSPI that this is the last frame in this packet before SSEL is supposed to go inactive, effectively allowing for the specification of the number of transmitted frames.

3.3.12 Clock Rate Register

Table 23 • **Clock Rate Register**

PADDR[5:0]	Register Name	Type	Width	Reset Value	Description
0x2C	CLK_DIV	R/W	8	CFG_CLK	Clock rate register. Writing to this register will update clock division factor of CoreSPI generated clock (SPICLKO) in master mode. Clock rate of generated SPI master clock is determined by the formula: **SPICLK = PCLK / (2 * (CLK_DIV + 1))** The register value overrides the parameter value (CFG_CLK) set in the core configuration window. At the power on, the CLK_DIV register will have the value of configurable parameter CFG_CLK and this value will be used to determine the clock rate of generated SPI master clock. Whenever the CLK_DIV register is updated, the new value is used to determine the clock rate of the generated SPI master clock. **Note:** It is recommended that the user updates this register when the core is not performing any SPI transactions.

4 Design Details

4.1 Transfer Protocols

This section describes the CoreSPI transfer protocols.

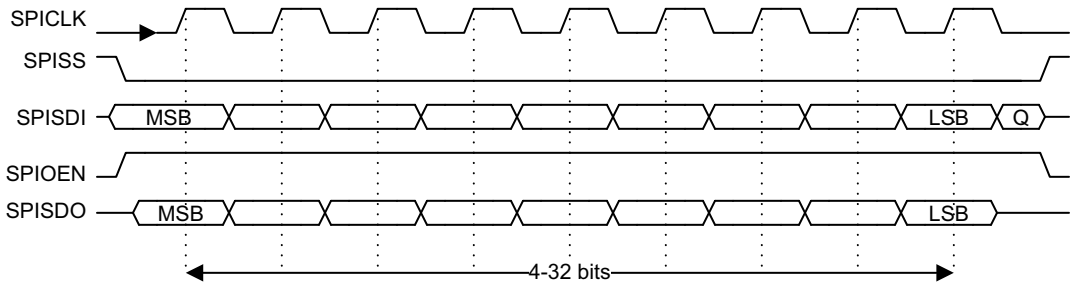
4.1.1 SPI Transfer Protocols

CoreSPI supports Motorola, NSC, and TI serial protocols.

4.1.1.1 Motorola SPI Mode

The following figure shows an example of the timing diagram single-frame transfer for both master (SPISDO) and slave (SPISDI) operation in Motorola SPI mode.

Figure 2 • **Single-Frame Transfer in Motorola Mode 0**

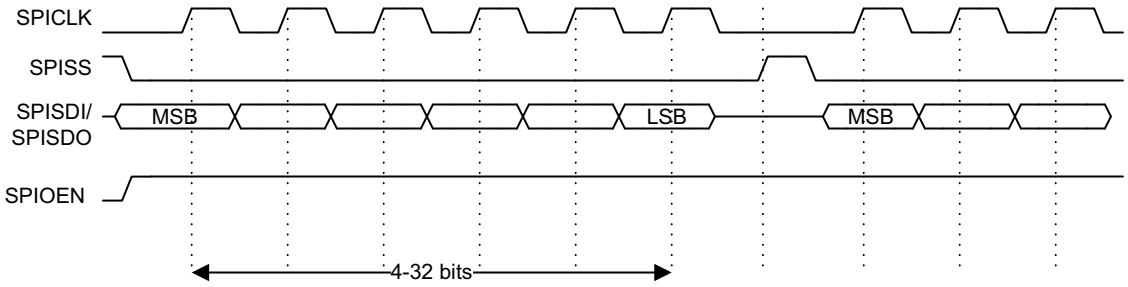


Note: The number of bits transferred is actually set by the CFG_FRAME_SIZE parameter. Refer Verilog/VHDL Parameters, page 4 for more information.

Data is captured on the rising edge of SPICLK and loaded on the falling.

The following figure shows an example of the timing diagram multiple-frame transfer for both master (SPISDO) and slave (SPISDI) operation in Motorola SPI mode.

Figure 3 • **Multi-Frame Transfer in Motorola Mode 0**



Note: SPI CLK is low between frames

Note: Data transferred MSB first

Note: SPIOEN signal is asserted during the transmission, de-asserted at the end of transfer (after last frame sent)

Note: Slave select (either input or output) is asserted for a pulse between frames (CFG_MOT_SSEL = 0)

Table 24 • **Motorola Mode - Default Slave Select Behavior**

Motorola Mode	Slave Select Behavior (CFG_MOT_SSEL = 0)
Mode 0	Pulses between all frames.
Mode 2	Pulses between all frames.
Mode 1	Pulses when the transmit FIFO becomes empty. Otherwise remains asserted during back to back transfers.
Mode 3	Pulses when the transmit FIFO becomes empty. Otherwise remains asserted during back to back transfers.

The waveforms for Motorola modes 1, 2, and 3 are shown in the following figure.

Figure 4 • **Single Frame Transfer in Motorola Mode 1**

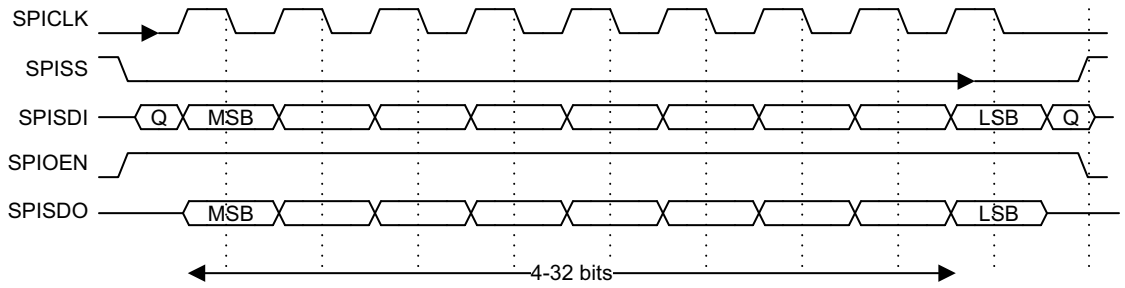


Figure 5 • **Single Frame Transfer in Motorola Mode 2**

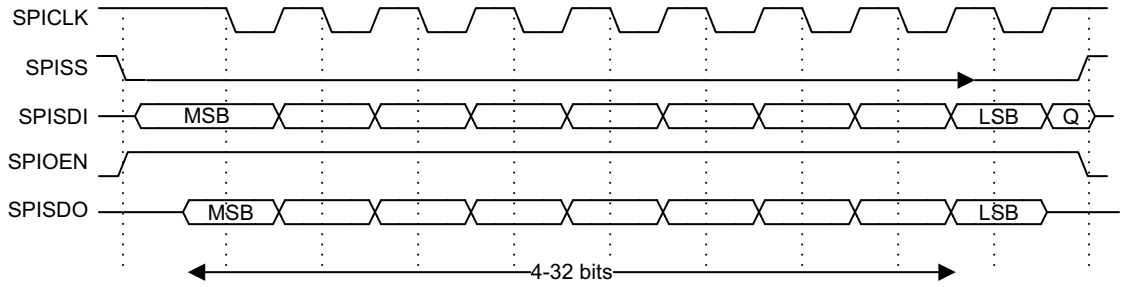
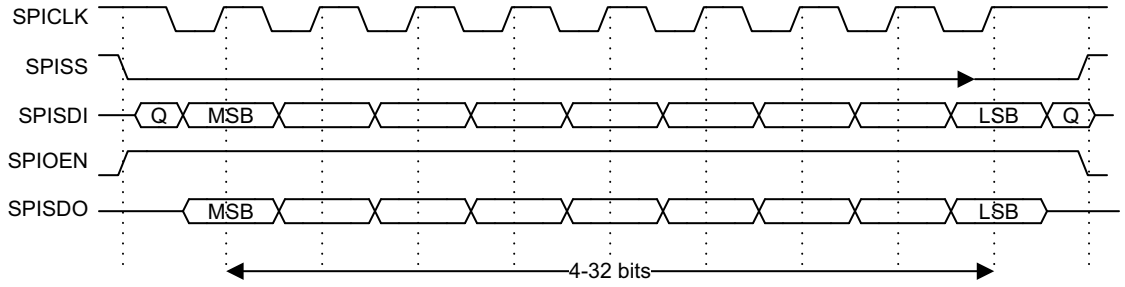


Figure 6 • **Single Frame Transfer in Motorola Mode 3**

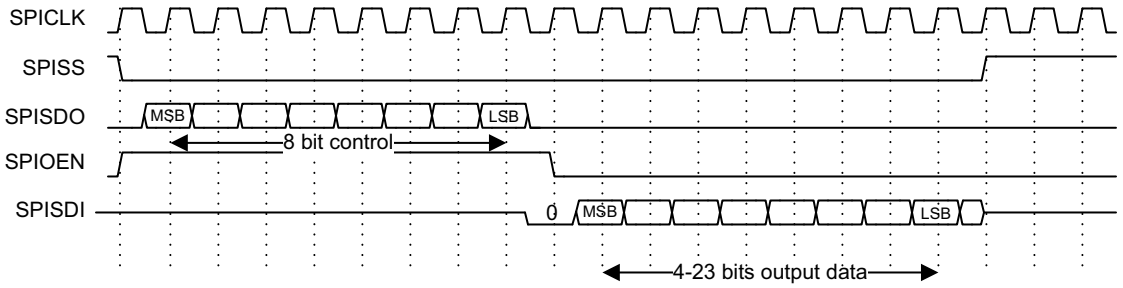


For a complete Motorola SPI protocol description, refer to the *ARM PrimeCell Synchronous Serial Port (PL022) Technical Reference Manual*.

4.1.1.2 NSC Microwire SPI Mode

The following figure shows a single-frame transfer in NSC mode. In this case, the master generates SPISDO and the slave responds with SPISDI.

Figure 7 • **Single-Frame Transfer in NSC SPI Mode**



NSC MicroWire format is half duplex using a master-slave message passing technique. Each serial transmission begins with an 8-bit control word during which, no incoming data is received.

After the control word is sent, the slave decodes it and after waiting one serial clock from the end of control word, responds with required data, which may be 4 to 23 bits in length.

The frame size is set as follows in NSC mode; the frame consists of 8 bits of control, 1-bit of bus idle and N bits of output data.

The following settings are required for NSC mode:

- Frame size = 8+1+N = 9+N
- Master transmit data (C) is written to bits [N+8:N+1] of TXDATA
- Master receive data (D) is the least significant bits of RXDATA [N-1:0]
- Slave transmit data (D) is written to TXDATA [N-1:0]
- Slave receive data (C) is bits [N+8:N+1] of RXDATA

The bit mapping shows the 8-bit command in the most significant 8 bits, 1 idle bit and N data bits.

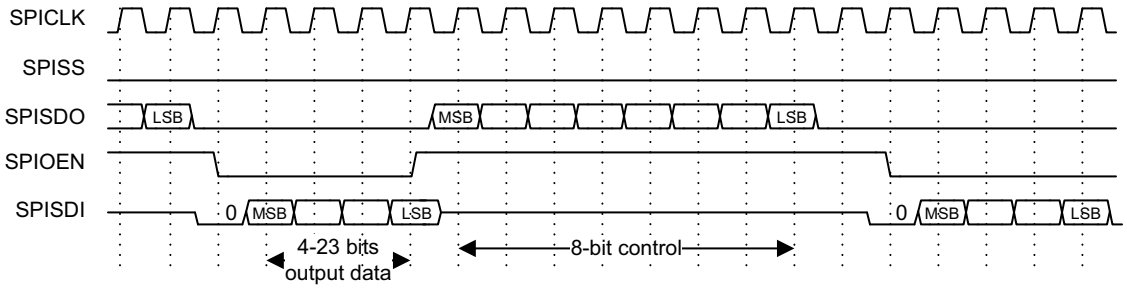
Table 25 • **NSC SPI mode bit mapping**

N+8	N+7	N+6	N+5	N+4	N+3	N+2	N+1	N	N-1	N-2	1	0
C[7]	C[6]	C[5]	C[4]	C[3]	C[2]	C[1]	C[0]	X	D[N-1]	D[N-1]	D[1]	D[0]

Bit N is undefined, it can either be written as a 0 or 1. During reads it will reflect the actual SPI data line value, this may be 0 or 1, and in simulation a 'Z' is possible as the SPI data will normally be floating at this time.

The following figure shows a multiple-frame transfer in NSC SPI mode.

Figure 8 • **Multi Frame Transfer in NSC SPI Mode (Master Mode Operation)**

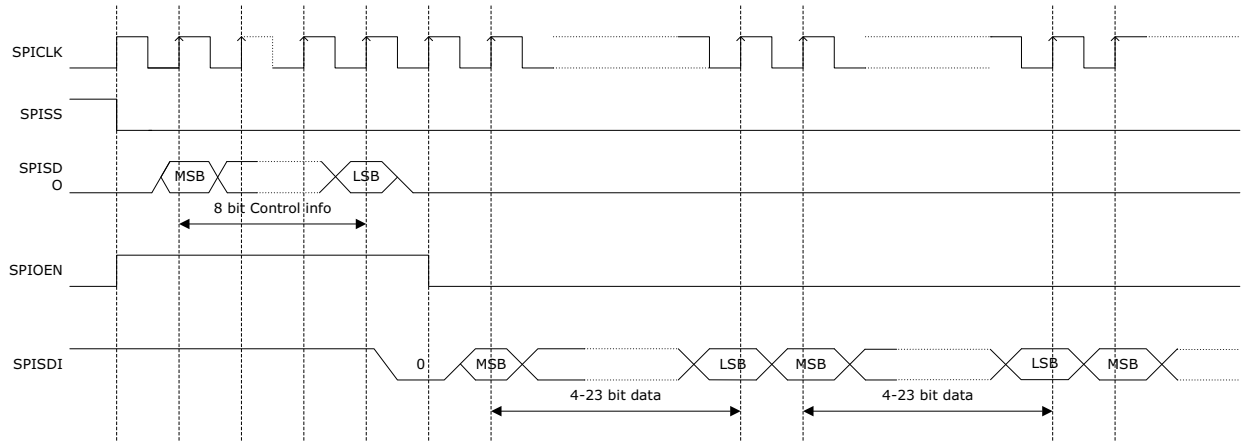


Note: Slave select is continuously asserted (held low)

Note: SPIOEN is asserted for duration of each control byte.

Note: Transfers proceed in back to back manner.

Figure 9 • **Continuous Transfer in NSC Mode for Large Frames**



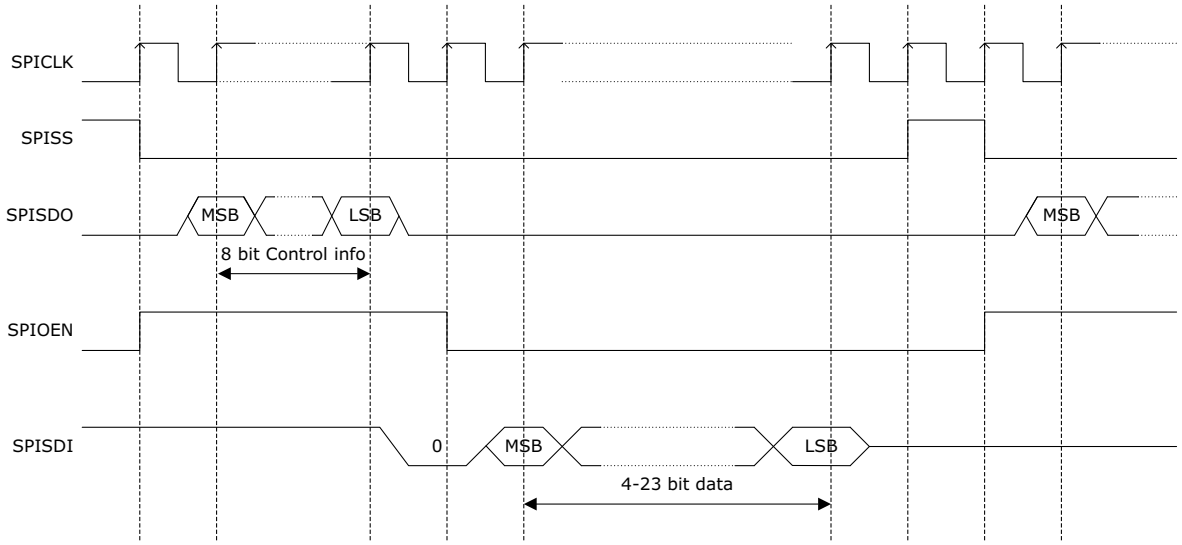
Note: Slave select is continuously asserted (held low).

Note: Control byte is transferred only once.

Note: Output enable asserted for duration of control byte.

Note: Frames are transferred back to back without transmitting control byte for each frame, making the data look like a single large frame.

Figure 10 • **Continuous Transfer in NSC Mode with Idle Cycle**



Note: Idle cycles are inserted in between the frames.

Note: Slave select is pulsed in between the frames during idle period.

Note: Output enable asserted for duration of each control byte.

4.1.1.3 TI Synchronous Serial SPI Mode

The following figure shows a single and continuous serial transfers in TI Synchronous SPI mode.

Figure 11 • **Single Frame Transfer in TI Synchronous SPI Mode**

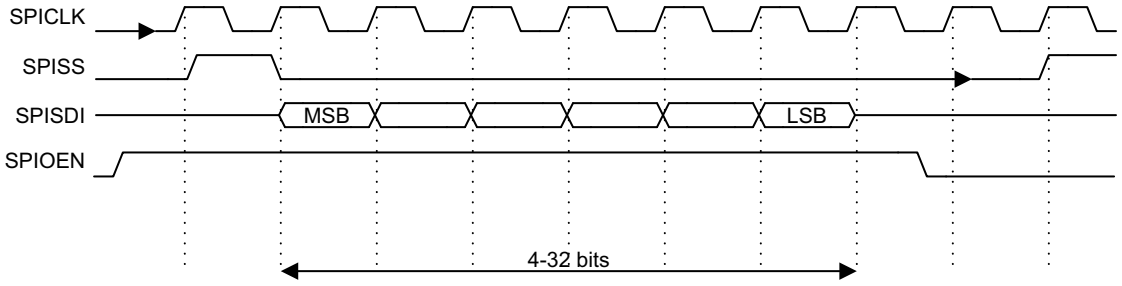
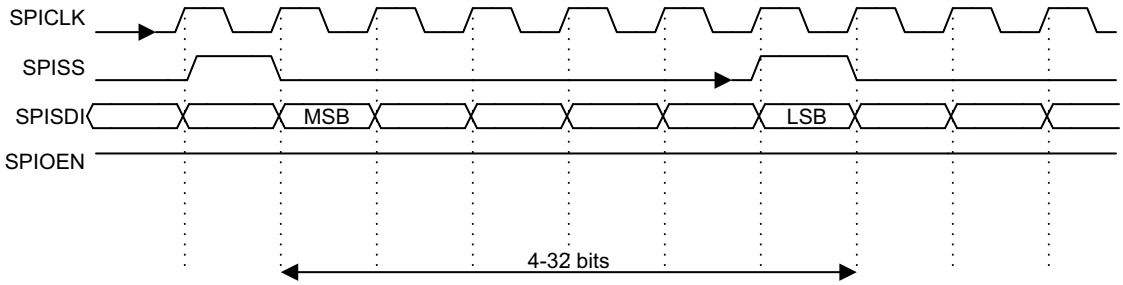


Figure 12 • **Continues Transfer in TI Synchronous SPI Mode**



Note: The slave select is pulsed between transfers to guarantee a high to low transition between each frame.

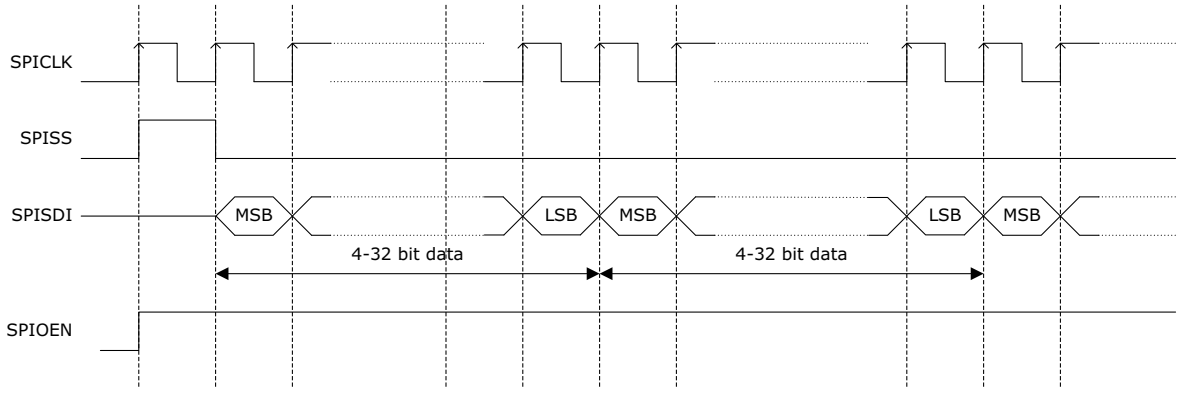
Note: Slave select low in idle.

Note: The data is available on the clock cycle immediately following the slave select assertion.

Note: Both the SPI master and the SPI slave clock each data bit into their serial shift registers on the falling edge of the SPICLK. The received data is latched on the rising edge of the SPICLK.

Note: The output enable is held active high throughout the transfers.

Figure 13 • **Continuous Transfer in TI mode for Jumbo Frames**



Note: Slave select is pulsed (high to low transition) only for the first frame.

Note: Output enable held active throughout the transfer.

Note: Frames are transferred back to back without slave select transition, making the data look like a single large frame.

4.1.2 Output Enable Control

The following table lists output enable operation in each CoreSPI operating mode.

Table 26 • **Output Enable Behavior**

Mode	Master	Slave
Motorola	SPIOEN is asserted with identical timing to SSEL. This provides an additional ½ SPICLK cycle of data turn on and off relative to the data bit valid requirements	The incoming SSEL signal is used to directly generate SPIOEN. Like for the master case this provides an additional ½ clock cycle of data turn on and off.
TI	SPIOEN is asserted on the negative clock edge prior to the MSB (whilst SSEL is asserted) and if non-continuous data de-asserted on the falling SPICLK edge following the LSB. This provides ½ a clock cycle of data turn on off time.	SPIOEN is asserted on the positive SPI clock edge as the MSB is output. SPIOEN is de-asserted on the positive SPI clock edge at the end of the LSB data bit assuming no consecutive data. This assumes a rising edge occurs at the end of the LSB data window as per ARM documents.
NSC	SPIOEN is asserted with SSEL, and then removed at the start of the 9th data bit (turn around cycle).	SPIOEN is asserted at the start of the 10th bit as data becomes valid. SPIOEN is de-asserted at the end of the LSB if a falling clock edge occurs or when SSEL is de-asserted otherwise.

4.1.3 Transmission of Multiple Frames

When transmitting, ensure that you write the last frame in a transmission to the aliased TX Data Register, rather than the TX Data Register. This indicates to CoreSPI that this is end of the current packet. In Motorola mode 1 or Motorola mode 3, the SSEL signal is held active until the last frame is transmitted.

4.1.4 APB Interface Timing

The following figures depict typical write cycle and read cycle timing relationships relative to the system clock, PCLK.

Figure 14 • **APB Data Write Cycle**

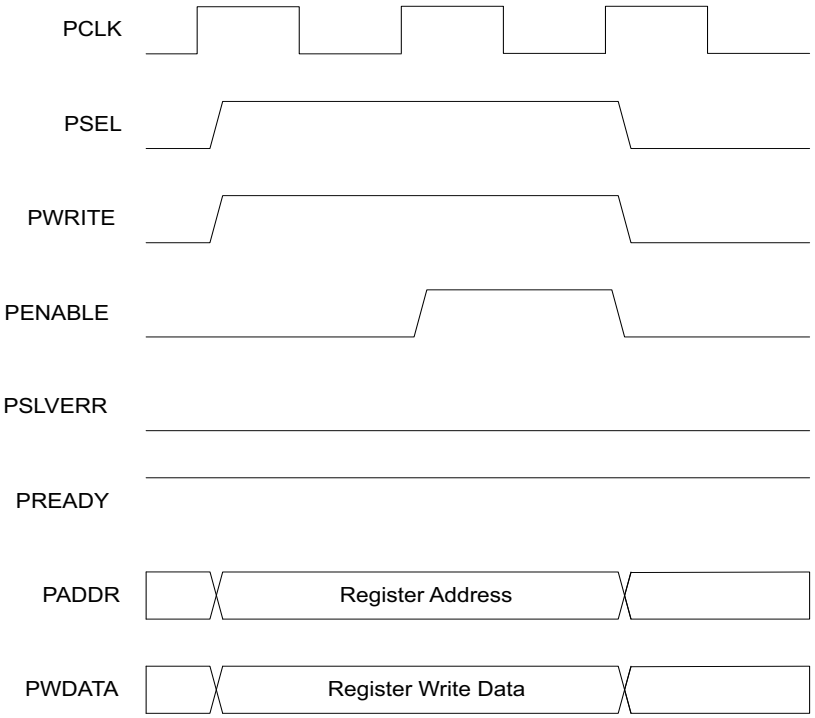
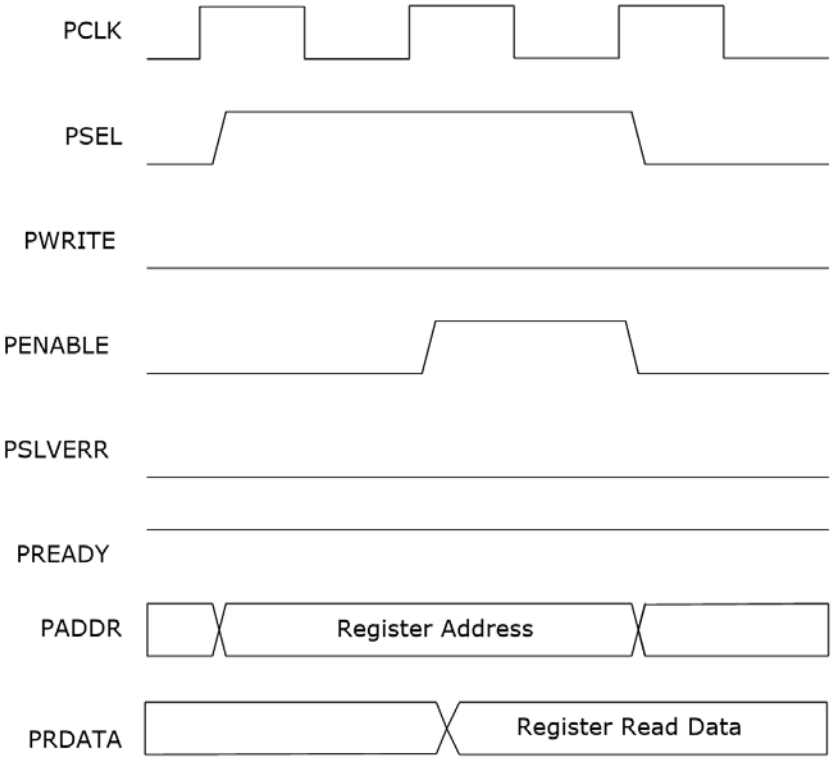


Figure 15 • **APB Data Read Cycle**



5 Tool Flows

5.1 Licensing

No licence is required for the use of this core.

5.1.1 RTL

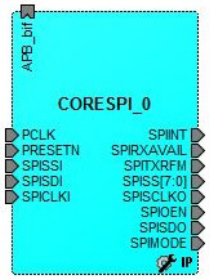
Complete RTL source code is provided for the core and testbenches.

5.2 SmartDesign

CoreSPI is preinstalled in the SmartDesign IP Deployment design environment.

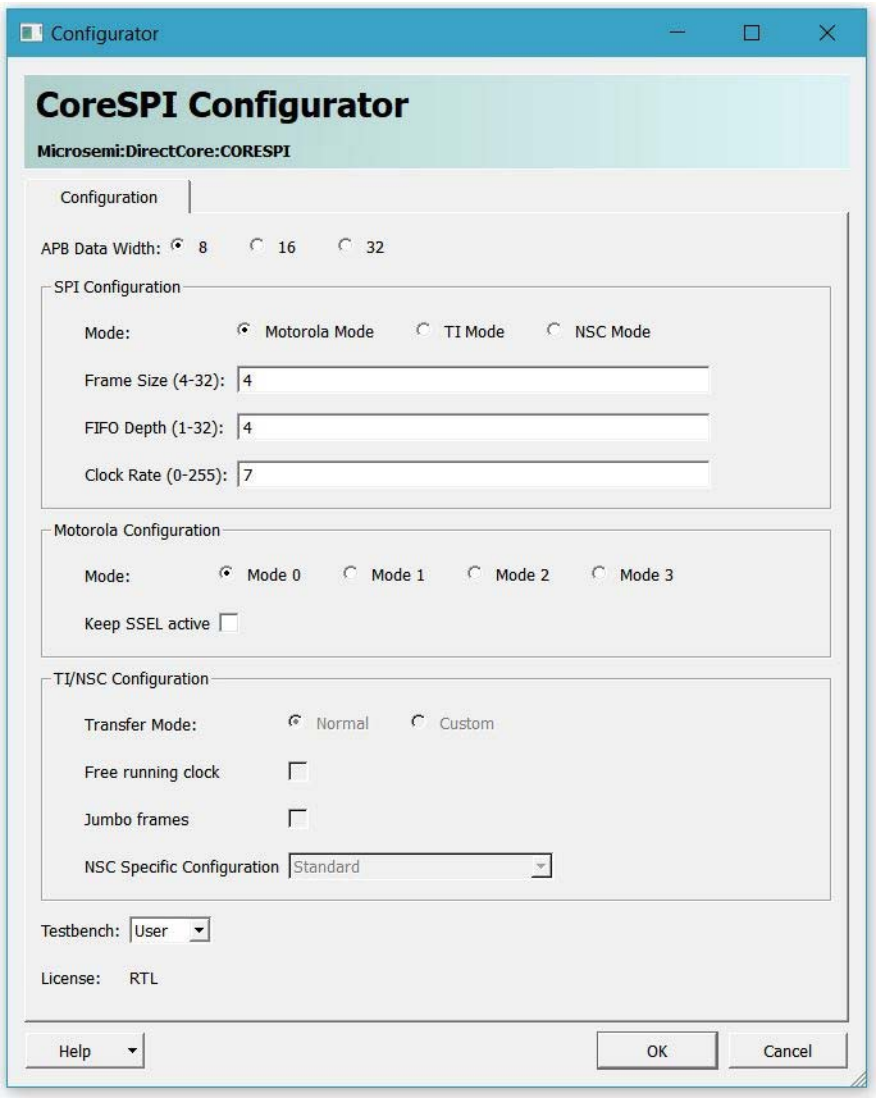
For information on using SmartDesign to instantiate and generate cores, refer to the *Using DirectCore in Libero User Guide*.

Figure 16 • **CoreSPI Full I/O View**



The core is configured using the configuration GUI within SmartDesign, as shown in the following figure.

Figure 17 • **CoreSPI SmartDesign Configuration to Associate Parameters**



5.3 Design Migration

Migrating a design using a previously released version of CoreSPI (Core v4.2 or lower versions) to CoreSPI v5.0 or later requires the following steps to be completed.

1. Note/record the CoreSPI configuration settings used in the design. Right-click SmartDesign component in the Libero software and select **Configure**.
2. Right-click SmartDesign component and select **Replace Instance Version**. Ensure that the latest version is listed in the **Change to Version** window and then click **OK**.
3. Right-click updated instance of CoreSPI and select **Configure** to configure it with the same parameters values and settings as the previous version. Refer to the following table to determine the mapping between the old parameters and the new parameters contained in CoreSPI v5.0 or later versions.

The following table lists parameter migration of different modes.

Table 27 • **Parameter Migration Table**[1]

Mode	Configuration	Old Parameters				New Configuration Parameters							Functional Operation
		CFG_MODE	CFG_SPO	CFG_SPH	CFG_SPS	CFG_MODE	CFG_MOT_MODE	CFG_MOT_SSEL	CFG_TI_NSC_CUSTOM	CFG_TI_NSC_FRC	CFG_TI_JMB_FRAMES	CFG_NSC_OPERATION	
Motorola	Mode 0	0	0	0	0	0	0	0	0	0	0	0	SSEL toggles between transfers
		0	0	0	1	0	0	1	0	0	0	0	SSEL remains active between back to back transfers
	Mode 1	0	0	1	0	0	1	0	0	0	0	0	SSEL toggles when TX FIFO empties, otherwise active
		0	0	1	1	0	1	1	0	0	0	0	SSEL remains active between back to back transfers
	Mode 2	0	1	0	0	0	2	0	0	0	0	0	SSEL toggles between transfers
		0	1	0	1	0	2	1	0	0	0	0	SSEL remains active between back to back transfers
	Mode 3	0	1	1	0	0	3	0	0	0	0	0	SSEL toggles when TX FIFO empties, otherwise active
		0	1	1	1	0	3	1	0	0	0	0	SSEL remains active between back to back transfers
TI	Normal	1	0	0	0	1	0	0	0	0	0	0	Normal
	Custom	1	1	0	0	1	0	0	1	1	0	0	Free running clock
		1	0	1	0	1	0	0	1	0	1	0	Jumbo frames. Removes SSEL on consecutive frames (back-to-back) making them look like big frames
		1	1	1	0	1	0	0	1	1	1	0	Free running clock & jumbo frames

Table 27 • **Parameter Migration Table[1] *(continued)***

Mode	Configuration	CFG_MODE	CFG_SPO	CFG_SPH	CFG_SPS	CFG_MODE	CFG_MOT_MODE	CFG_MOT_SSEL	CFG_TI_NSC_CUSTOM	CFG_TI_NSC_FRC	CFG_TI_JMB_FRAMES	CFG_NSC_OPERATION	Functional Operation
		Old Parameters				New Configuration Parameters							
NSC	Normal	2	0	0	0	2	0	0	0	0	0	0	Normal
	Custom	2	1	0	0	2	0	0	1	1	0	0	Free running clock
		2	0	1	0	2	0	0	1	0	0	1	Idle cycles. Forces idle cycles (SSEL deactivated) between back-to-back frames
		2	1	1	0	2	0	0	1	1	0	1	Free running clock & idle cycles
		2	0	0	1	2	0	0	1	0	0	2	Large response frames. After sending the command part of the frame, subsequent frames are concatenated to create a single large data frame (master operation only)
		2	1	0	1	2	0	0	1	1	0	2	Free running clock & large response frames

1. The FIFO_DEPTH parameter (in core v4.2 or lower versions) is renamed to CFG_FIFO_DEPTH (in core v5.0 or later). Manually update this parameter when migrating a design using a version of CoreSPI released prior to CoreSPI v5.0, if the FIFO_DEPTH parameter was set at anything other than the default setting in the previous design.

The following table lists how the generics/parameters used to configure CoreSPI v5.0 or later version are configured from the CoreSPI configurator GUI when the core is instantiated in SmartDesign.

Table 28 • **Configuration GUI parameter mapping**

Parameter	CoreSPI SmartDesign Configuration GUI Field
CFG_MODE	SPI Configuration -> Mode
CFG_MOT_MODE	Motorola Configuration -> Mode
CFG_MOT_SSEL	Motorola Configuration -> Keep SSEL active
CFG_TI_NSC_CUSTOM	TI/NSC Configuration -> Transfer Mode
CFG_TI_NSC_FRC	TI/NSC Configuration -> Free running clock
CFG_TI_JMB_FRAMES	TI/NSC Configuration -> Jumbo frames
CFG_NSC_OPERATION	TI/NSC Configuration -> NSC specific Configuration

5.4 Simulation Flows

The user testbench for CoreSPI is included in all releases.

To run simulations:

1. Select the user testbench flow within SmartDesign
2. Click **Save and Generate** in the Generate pane. The user testbench is selected through the Core Testbench Configuration GUI.

When SmartDesign generates the Libero project, it installs the user testbench files.

To run the user testbench:

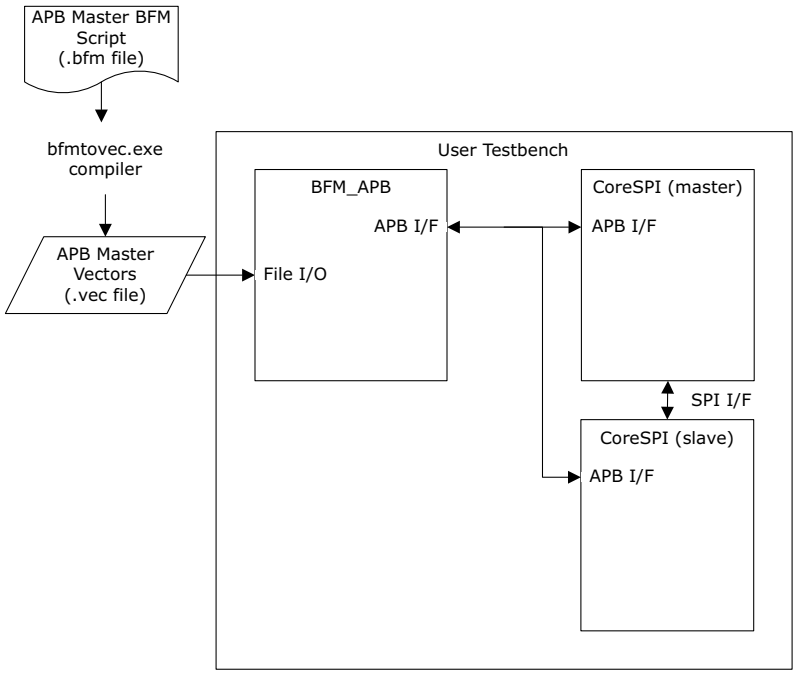
1. Set the design root to the CoreSPI instantiation in the Libero design hierarchy pane.
2. Click **Simulation** in the Libero Design Flow window. This invokes ModelSim and automatically run the simulation.

5.4.1 User Testbench

The following figure shows the simulation testbench. It includes and instantiates the CoreSPI macro and a BFM-based APB-driven block. All BFM compilers are included for both the Linux and Windows operating system as well as the BFM vector source files. These are editable.

Note: The user testbench supports fixed configurations only.

Figure 18 • **CoreSPI User Testbench**



5.5 Synthesis

To run synthesis on the CoreSPI, set the design root to the IP component instance and run the synthesis tool from the Libero Design Flow pane. This will invoke Synplify Pro and automatically runs the synthesis.

5.6 Place-and-Route

After design is synthesized, click **Place and Route** in the Libero Design Flow pane to run place and route on the CoreSPI. No special place and route settings are required.

6 System Integration

This chapter provides hints to ease the integration of CoreSPI.

- The design created using CoreSPI which is interfaced with SPI Winbond (w25q64fv) flash on the SmartFusion2 Security Evaluation kit. The SPI flash works as a slave and supports Motorola mode 0 and 3.
- FABRIC RESET (SYSRESET_0) resets CoreResetP_0
- MSS_HPMS_READY resets CoreSPI_0.
- CORESPI_0 has PCLK, SPISCLKO, and SPICLKI clocks. In Master mode, PCLK and SPISCLKO clocks are used.
- 50 MHz PCLK required for the APB system in CORESPI_0 is driven from the FCCC_0/GL0.

Figure 19 • **CoreSPI Integration**