AM17x/AM18x ARM Microprocessor Serial Peripheral Interface (SPI)

User's Guide

Literature Number: SPRUFW1B
March 2011

List of Figures

List of Tables

Read This First

About This Manual

This document describes the serial peripheral interface (SPI). The SPI is in effect a programmable-length shift register used for high speed communication between external peripherals or other devices.

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (decimal 64): 40h.
- Registers in this document are shown in figures and described in tables.
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure designate a bit that is used for future device expansion.

Related Documentation From Texas Instruments

Copies of these documents are available on the Internet at www.ti.com. *Tip:* Enter the literature number in the search box provided at www.ti.com.

The current documentation that describes the DSP, related peripherals, and other technical collateral, is available in the C6000 DSP product folder at: www.ti.com/c6000.

SPRUGU3 — ***AM1705 ARM Microprocessor System Reference Guide.*** Describes the ARM subsystem, system memory, memory protection unit (MPU), device clocking, phase-locked loop controller (PLLC), power and sleep controller (PSC), power management, ARM interrupt controller (AINTC), and system configuration module.

SPRUGR6 — ***AM1707 ARM Microprocessor System Reference Guide.*** Describes the ARM subsystem, system memory, memory protection unit (MPU), device clocking, phase-locked loop controller (PLLC), power and sleep controller (PSC), power management, ARM interrupt controller (AINTC), and system configuration module.

SPRUGX5 — ***AM1802 ARM Microprocessor System Reference Guide.*** Describes the ARM subsystem, system memory, memory protection unit (MPU), device clocking, phase-locked loop controller (PLLC), power and sleep controller (PSC), power management, ARM interrupt controller (AINTC), and system configuration module.

SPRUGU4 — ***AM1806 ARM Microprocessor System Reference Guide.*** Describes the ARM subsystem, system memory, memory protection unit (MPU), device clocking, phase-locked loop controller (PLLC), power and sleep controller (PSC), power management, ARM interrupt controller (AINTC), and system configuration module.

SPRUGM9 — ***AM1808/AM1810 ARM Microprocessor System Reference Guide.*** Describes the ARM subsystem, system memory, memory protection unit (MPU), device clocking, phase-locked loop controller (PLLC), power and sleep controller (PSC), power management, ARM interrupt controller (AINTC), and system configuration module.

SPRUFU0 — ***AM17x/AM18x ARM Microprocessor Peripherals Overview Reference Guide.*** Provides an overview and briefly describes the peripherals available on the AM17x/AM18x ARM Microprocessors.

Serial Peripheral Interface (SPI)

1 Introduction

This document describes the serial peripheral interface (SPI) module. See your device-specific data manual to determine how many SPIs are available on your device.

1.1 Purpose of the Peripheral

The SPI is a high-speed synchronous serial input/output port that allows a serial bit stream of programmed length (2 to 16 bits) to be shifted into and out of the device at a programmed bit-transfer rate. The SPI is normally used for communication between the device and external peripherals. Typical applications include interface to external I/O or peripheral expansion via devices such as shift registers, display drivers, SPI EPROMS and analog-to-digital converters.

1.2 Features

The SPI has the following features:

- 16-bit shift register
- 16-bit Receive buffer register (SPIBUF) and 16-bit Receive buffer emulation 'alias' register (SPIEMU)
- 16-bit Transmit data register (SPIDAT0) and 16-bit Transmit data and format selection register (SPIDAT1)
- 8-bit baud clock generator
- Serial clock (SPIx_CLK) I/O pin
- Slave in, master out (SPIx_SIMO) I/O pin
- Slave out, master in (SPIx_SOMI) I/O pin
- SPI enable ($\overline{\text{SPIx\_ENA}}$) I/O pin (4-pin or 5-pin mode only)
- Multiple slave chip select ($\overline{\text{SPIx\_SCS[n]}}$) I/O pins (4-pin or 5-pin mode only)
- Programmable SPI clock frequency range
- Programmable character length (2 to 16 bits)
- Programmable clock phase (delay or no delay)
- Programmable clock polarity (high or low)
- Interrupt capability
- DMA support (read/write synchronization events)

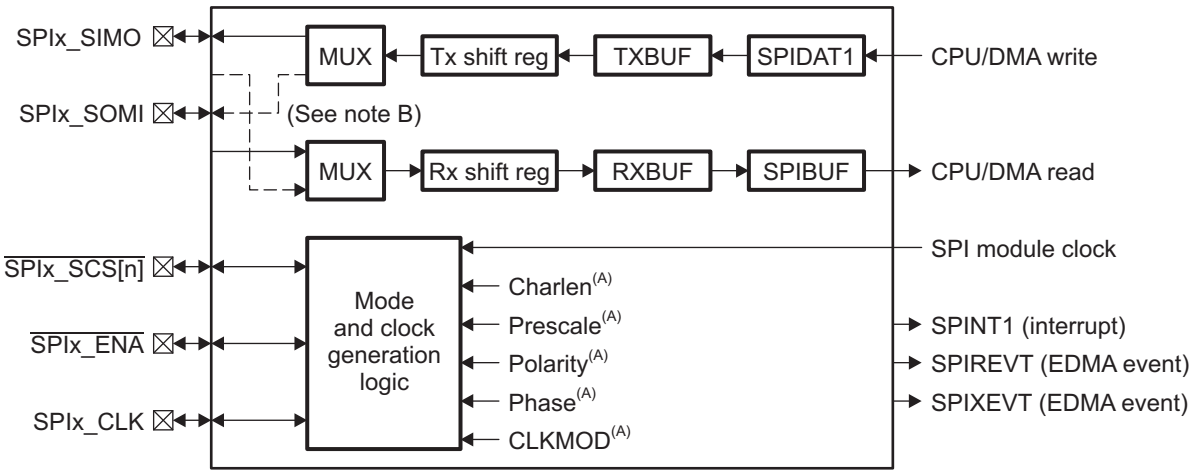
The SPI allows software to program the following options:

- SPI pins as functional or digital I/O pins
- SPI Master or Slave mode
- SPIx_CLK frequency (SPI module clock/3 through SPI module clock/256)
- 3-pin, 4-pin, and 5-pin options
- Character length (2 to 16 bits) and shift direction (MSB/LSB first)
- Clock phase (delay or no delay) and polarity (high or low)
- Delay between transmissions in master mode.
- Chip select setup and hold times in master mode
- Chip select hold in master mode

1.3 Functional Block Diagram

The Figure 1 shows the SPI block diagram.

Figure 1. SPI Block Diagram



NOTE: The value *x* indicates the applicable SPI; that is, SPI0, SPI1, etc. See your device-specific data manual to determine how many SPIs are available on your device. The value *n* indicates the SPI pins available. See your device-specific data manual to determine how many SPI pins are available on your device.

A Indicates the log controlled by SPI register bits.

B Solid line represents data flow for SPI master mode. Dashed line represents data flow for SPI slave mode.

1.4 Industry Standard(s) Compliance Statement

The programmable configuration capability of the SPI allows it to gluelessly interface to a variety of SPI format devices. The SPI does not conform to a specific industry standard.

2 Architecture

This section describes the SPI operation modes. It gives an overview of SPI operation and then provides details on the 3-pin, 4-pin, and 5-pin options, as well as more specific details on the supported data formats.

2.1 Clock

The SPI clock (SPIx_CLK) is derived from the SPI module clock. The maximum clock bit rate supported is SPI module clock/3, as determined by the PRESCALE field in the SPI data format register *n* (SPIFMT*n*). The SPIx_CLK frequency is calculated as:

SPIx_CLK frequency = [SPI module clock] / [SPIFMT*n*.PRESCALE + 1]

2.2 Signal Descriptions

Table 1 shows the SPI pins used to interface to external devices.

Table 1. SPI Pins

Pin [1]	Type	Function
SPIx_SIMO	Input/Output	Serial data input in slave mode, serial data output in master mode
SPIx_SOMI	Input/Output	Serial data output in slave mode, serial data input in master mode
SPIx_CLK	Input/Output	Serial clock input in slave mode, serial clock output in master mode
$\overline{\text{SPIx\_SCS[n]}}$ [2]	Input/Output	Slave chip select output in master mode, input in slave mode
$\overline{\text{SPIx\_ENA}}$	Input/Output	Input in master mode, output in slave mode indicates slave is ready

(1) The value *x* indicates the applicable SPI; that is, SPI0, SPI1, etc. See your device-specific data manual to determine how many SPIs are available on your device.

(2) The value *n* indicates the SPI pins available; that is, $\overline{\text{SPIx\_SCS[0]}}$, $\overline{\text{SPIx\_SCS[1]}}$, etc. See your device-specific data manual to determine how many SPI pins are available on your device.

2.3 Operation Modes

The SPI operates in master or slave mode. The SPI bus master is the device that drives the SPIx_CLK, SPIx_SIMO, and optionally the $\overline{\text{SPIx\_SCS[n]}}$ signals, and therefore initiates SPI bus transfers. The CLKMOD and MASTER bits in the SPI global control register 1 (SPIGCR1) select between master and slave mode. In both master and slave mode, the SPI supports four options:

- 3-pin option
- 4-pin with chip select option
- 4-pin with enable option
- 5-pin with enable and chip select option

The 3-pin option is the basic clock, data in, and data out SPI interface and uses the SPIx_CLK, SPIx_SIMO, and SPIx_SOMI pins. The 4-pin with chip select option adds the $\overline{\text{SPIx\_SCS[n]}}$ pin that is used to support multiple SPI slave devices on a single SPI bus. The 4-pin with enable option adds the $\overline{\text{SPIx\_ENA}}$ pin that is used to increase the overall throughput by adding hardware handshaking. The 5-pin option uses all the SPI pins and is a superset of the different options.

2.4 Programmable Registers

A general representation of the SPI programmable registers is shown in Table 2. For details on registers, see Section 3.

Table 2. SPI Registers

Offset Address [1]	Acronym	Name	Description	Section
0h	SPIGCR0	Global control register 0	Contains the software reset bit for the module	Section 3.1
4h	SPIGCR1	Global control register 1	Controls basic configurations of the module	Section 3.2
8h	SPIINT0	Interrupt register	Enable bits for interrupts, error, DMA and other functionality.	Section 3.3
Ch	SPILVL	Level register	SPI interrupt levels are set in this register.	Section 3.4
10h	SPIFLG	Flag register	Shows the status of several events during the operation.	Section 3.5
14h	SPIPC0	Pin control register 0	Determines if pins operate as general I/O or SPI functional pin	Section 3.6
18h	SPIPC1	Pin control register 1	Controls the direction of data on the I/O pins	Section 3.7
1Ch	SPIPC2	Pin control register 2	Reflects the values on the I/O pins	Section 3.8
20h	SPIPC3	Pin control register 3	Controls the values sent to the I/O pins	Section 3.9
24h	SPIPC4	Pin control register 4	Sets data values in the SPIPC3 register	Section 3.10
28h	SPIPC5	Pin control register 5	Clears values in the SPIPC3 register	Section 3.11
38h	SPIDAT0	Transmit data register 0	Transmit data register	Section 3.12
3Ch	SPIDAT1	Transmit data register 1	Transmit data with format selection register	Section 3.13
40h	SPIBUF	Receive buffer register	Holds received word	Section 3.14
44h	SPIEMU	Receive buffer emulation register	Mirror of SPIBUF. Read does not clear flags	Section 3.15
48h	SPIDELAY	Delay register	Sets $\overline{\text{SPIx\_SCS[n]}}$ mode, $\overline{\text{SPIx\_SCS[n]}}$ pre-/post-transfer delay time and $\overline{\text{SPIx\_ENA}}$ time-out	Section 3.16
4Ch	SPIDEF	Chip select default register	In $\overline{\text{SPIx\_SCS[n]}}$ decoded mode only: sets high low/active $\overline{\text{SPIx\_SCS[n]}}$ signal	Section 3.17
50h	SPIFMT0	Format 0 register	Configuration of data word format 0	Section 3.18
54h	SPIFMT1	Format 1 register	Configuration of data word format 1	Section 3.18
58h	SPIFMT2	Format 2 register	Configuration of data word format 2	Section 3.18
5Ch	SPIFMT3	Format 3 register	Configuration of data word format 3	Section 3.18
64h	INTVEC1	Interrupt vector register 1	Interrupt vector for line INT1	Section 3.19

(1) The actual address of these registers is device specific and CPU specific. See your device-specific data manual to verify the SPI register addresses.

2.5 Master Mode Settings

The four master mode options are defined by the configuration bit settings listed in Table 3. Other configuration bits may take any value in the range listed in Table 4. The values listed in Table 3 and Table 4 should not be changed while the ENABLE bit in the SPI global control register 1 (SPIGCR1) is set to 1. Note that in certain cases the allowed values may still be ignored. For example, Table 4 indicates that SPIDELAY may take a range of values in Master 3-pin mode; however, SPIDELAY has no effect in Master 3-pin mode. For complete details on each mode, see the following sections that explain the SPI operation for each of the master modes.

Table 3. SPI Register Settings Defining Master Modes

Register	Bit(s)	Master 3-pin	Master 4-pin Chip Select	Master 4-pin Enable	Master 5-pin
SPIGCR0	RESET	1	1	1	1
SPIGCR1	ENABLE	1	1	1	1
SPIGCR1	LOOPBACK	0	0	0	0
SPIGCR1	CLKMOD	1	1	1	1
SPIGCR1	MASTER	1	1	1	1
SPIPC0	SOMIFUN	1	1	1	1
SPIPC0	SIMOFUN	1	1	1	1
SPIPC0	CLKFUN	1	1	1	1
SPIPC0	ENAFUN	0	0	1	1
SPIPC0	SCS0FUN	0	1	0	1

Table 4. Allowed SPI Register Settings in Master Modes

Register	Bit(s)	Master 3-pin	Master 4-pin Chip Select	Master 4-pin Enable	Master 5-pin
SPIINT0	ENABLEHIGHZ	0,1	0,1	0,1	0,1
SPIFMT*n*	WDELAY	0 to 3Fh	0 to 3Fh	0 to 3Fh	0 to 3Fh
SPIFMT*n*	PARPOL	0,1	0,1	0,1	0,1
SPIFMT*n*	PARENA	0,1	0,1	0,1	0,1
SPIFMT*n*	WAITENA	0	0	1	1
SPIFMT*n*	SHIFTDIR	0,1	0,1	0,1	0,1
SPIFMT*n*	DISCSTIMERS	0,1	0,1	0,1	0,1
SPIFMT*n*	POLARITY	0,1	0,1	0,1	0,1
SPIFMT*n*	PHASE	0,1	0,1	0,1	0,1
SPIFMT*n*	PRESCALE	2 to FFh	2 to FFh	2 to FFh	2 to FFh
SPIFMT*n*	CHARLEN	2 to 10h	2 to 10h	2 to 10h	2 to 10h
SPIDELAY	C2TDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh
SPIDELAY	T2CDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh
SPIDELAY	T2EDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh
SPIDELAY	C2EDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh

2.5.1 Master Mode Timing Options

The SPI in master mode supports several options to modify the timing of its generation of the chip select signal ($\overline{\text{SPIx\_SCS[n]}}$). This allows the SPI to support the timing requirements of various slave devices without adding additional overhead to the CPU by generating the appropriate delays automatically.

2.5.1.1 Chip Select Setup Time

The master can be configured to provide a (slow) slave device a certain chip select setup time to the first edge on SPIx_CLK. This delay is controlled by the C2TDELAY field in the SPI delay register (SPIDELAY) and can be configured between 3 and 257 SPI module clock cycles. The C2TDELAY is applicable only in 4-pin with chip select and 5-pin SPI master modes. The C2TDELAY begins when the SPI master asserts $\overline{\text{SPIx\_SCS[n]}}$. The C2T delay period is specified by:

Maximum duration of C2TDELAY period = SPIDELAY.C2TDELAY + 2 (SPI module clock cycles)

Note that if SPIDELAY.C2TDELAY = 0, then the C2TDELAY period = 0.

The previous value of the CSHOLD bit in the SPI transmit data register (SPIDAT1) must be cleared to 0 for the C2T delay to be enabled.

NOTE: If the SPIDAT1.CSHOLD bit is set within the control field, the current hold time and the following setup time will not be applied in between transaction.

2.5.1.2 Chip Select Hold Time

The master can be configured to provide a (slow) slave device a certain chip select hold time after the last edge on SPIx_CLK. This delay is controlled by the T2CDELAY bit in the SPI delay register (SPIDELAY) and can be configured between 2 and 256 SPI module clock cycles. The T2CDELAY is applicable only in 4-pin with chip select and 5-pin SPI master modes. The T2CDELAY begins after the data shifting period ends. The T2C delay period is specified by:

Maximum duration of T2CDELAY period = SPIDELAY.T2CDELAY + 1 (SPI module clock cycle)

Note that if SPIDELAY.T2CDELAY = 0, then the T2CDELAY period = 0. If the PHASE bit in the SPI data format register *n* (SPIFMT*n*) is 0, then the T2CDELAY period lasts for an additional 1/2 SPIx_CLK time over that specified by the above equation.

The current value of the CSHOLD bit in the SPI transmit data register (SPIDAT1) must be cleared to 0 for T2C delay to be enabled.

NOTE: If the SPIDAT1.CSHOLD bit is set within the control field, the current hold time and the following setup time will not be applied in between transaction.

2.5.1.3 Automatic Delay Between Transfers

The SPI master can automatically insert a delay of between 2 and 65 SPI module clock cycles between transmissions. This delay is controlled by the WDELAY field in the SPI data format register *n* (SPIFMT*n*) and is enabled by setting the WDEL bit in the SPI transmit data register (SPIDAT1) to 1. The WDELAY period begins when the T2EDELAY period terminates (if T2E delay period is enabled) or when the T2CDELAY period terminates (if T2E delay period was disabled and T2C delay period was enabled) or when the master deasserts $\overline{\text{SPIx\_SCS[n]}}$ (if T2E and T2C delay periods are disabled). If a transfer is initiated by writing a 32-bit value to SPIDAT1, then the new values of SPIDAT1.WDEL and SPIFMT*n*.WDELAY are used; otherwise, the old values of SPIDAT1.WDEL and SPIFMT*n*.WDELAY are used. The WDELAY delay period is specified by:

Maximum duration of WDELAY period = SPIFMTn.WDELAY + 2 (SPI module clock cycles)

2.5.1.4 Chip Select Hold Option

There are slave devices available that require the chip select signal to be held continuously active during several consecutive data word transfers. Other slave devices require the chip select signal to be deactivated between consecutive data word transfers. The SPI can support both types of slave devices. The CSHOLD bit in the SPI transmit data register (SPIDAT1) selects between the two options.

If the chip select hold option is enabled, the chip select will not toggle between two consecutive accesses; therefore, the SPIDELAY.T2CDELAY of the first transfer and the SPIDELAY.C2TDELAY of the second transfer will not be applied. However, the wait delay could still be applied between the two transactions, if the WDEL bit in SPIDAT1 is set to 1.

The current and previous values of the CSHOLD bit are retained. Though the current value of the CSHOLD bit is initialized to 0 when the RESET bit in the SPI global control register 0 (SPIGCR0) is cleared to 0, the previous value of the CSHOLD bit is not initialized. The previous value of the CSHOLD bit must be explicitly initialized by writing twice to the CSHOLD bit.

2.6 Slave Mode Settings

The four slave mode options are defined by the configuration bit settings listed in Table 5. Other configuration bits may take any value in the range listed in Table 6. The values listed in Table 5 and Table 6 should not be changed while the ENABLE bit in the SPI global control register 1 (SPIGCR1) is set to 1. Note that in certain cases the allowed values may still be ignored. For complete details on each mode, see the following sections that explain the SPI operation for each of the slave modes.

Table 5. SPI Register Settings Defining Slave Modes

Register	Bit(s)	Slave 3-pin	Slave 4-pin Chip Select	Slave 4-pin Enable	Slave 5-pin
SPIGCR0	RESET	1	1	1	1
SPIGCR1	ENABLE	1	1	1	1
SPIGCR1	LOOPBACK	0	0	0	0
SPIGCR1	CLKMOD	0	0	0	0
SPIGCR1	MASTER	0	0	0	0
SPIPC0	SOMIFUN	1	1	1	1
SPIPC0	SIMOFUN	1	1	1	1
SPIPC0	CLKFUN	1	1	1	1
SPIPC0	ENAFUN	0	0	1	1
SPIPC0	SCS0FUN	0	1	0	1

Table 6. Allowed SPI Register Settings in Slave Modes

Register	Bit(s)	Slave 3-pin	Slave 4-pin Chip Select	Slave 4-pin Enable	Slave 5-pin
SPIINT0	ENABLEHIGHZ	0,1	0,1	0,1	0,1
SPIFMT*n* (1)	WDELAY	0 to 3Fh	0 to 3Fh	0 to 3Fh	0 to 3Fh
SPIFMT*n* (1)	PARPOL	0,1	0,1	0,1	0,1
SPIFMT*n* (1)	PARENA	0,1	0,1	0,1	0,1
SPIFMT*n* (1)	WAITENA	0,1	0,1	0,1	0,1
SPIFMT*n* (1)	SHIFTDIR	0,1	0,1	0,1	0,1
SPIFMT*n* (1)	DISCSTIMERS	0,1	0,1	0,1	0,1
SPIFMT*n* (1)	POLARITY	0,1	0,1	0,1	0,1
SPIFMT*n* (1)	PHASE	0,1	0,1	0,1	0,1
SPIFMT*n* (1)	PRESCALE	2 to FFh	2 to FFh	2 to FFh	2 to FFh
SPIFMT*n* (1)	CHARLEN	2 to 10h	2 to 10h	2 to 10h	2 to 10h
SPIDELAY	C2TDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh
SPIDELAY	T2CDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh
SPIDELAY	T2EDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh
SPIDELAY	C2EDELAY	0 to FFh	0 to FFh	0 to FFh	0 to FFh

(1) In slave mode, only SPIFMT0 is used. When SPIDAT1 is written, the DFSEL field in SPIDAT1 is cleared to 0 to select SPIFMT0.

2.7 SPI Operation: 3-Pin Mode

NOTE: If only unidirectional communication is required, the SPIx_CLK pin and the two data pins (SPIx_SOMI and SPIx_SIMO) must all be configured as functional pins. A 2-pin unidirectional mode is not supported.

The SPI 3-pin mode uses only the clock (SPIx_CLK) and data (SPIx_SOMI and SPIx_SIMO) pins for bidirectional communication between master and slave devices. Figure 2 shows the basic 3-pin SPI option.

To select the 3-pin SPI option, the SPIx_CLK, SPIx_SOMI, and SPIx_SIMO pins should be configured as functional pins by configuring the SPI pin control register 0 (SPIPC0). The $\overline{\text{SPIx\_SCS[n]}}$ and $\overline{\text{SPIx\_ENA}}$ pins can be used as general-purpose I/O pins by configuring the SPIPC1 through SPIPC5 registers.

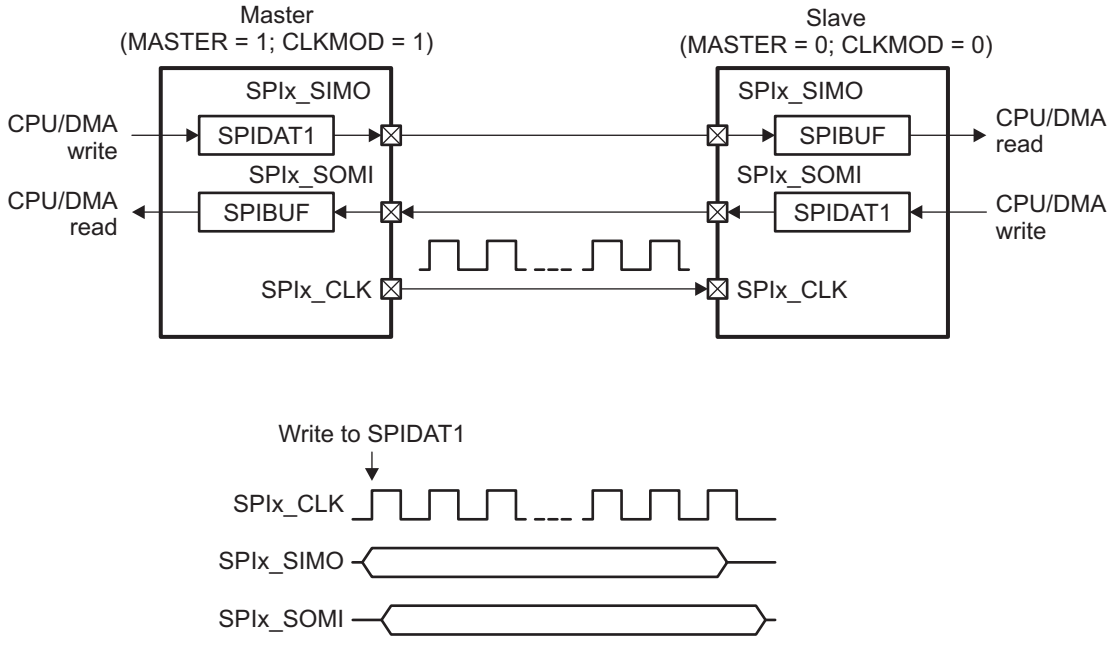
The SPI operates in either master or slave mode. The CLKMOD and MASTER bits in the SPI global control register 1 (SPIGCR1) select between master and slave mode; both must be programmed to 1 to configure the SPI for master mode or to 0 to configure the SPI for slave mode. The SPI bus master is the device that drives the SPIx_CLK signal and initiates SPI bus transfers. In SPI master mode, the SPIx_SOMI pin output buffer is in a high-impedance state and the SPIx_CLK and the SPIx_SIMO pin output buffer is enabled. In SPI slave mode, the SPIx_SIMO and SPIx_CLK pin output buffer is in a high-impedance state and the SPIx_SOMI pin output buffer is enabled.

In master mode with the 3-pin option, the CPU writes transmit data to the SPI transmit data registers (SPIDAT0[15:0] or SPIDAT1[15:0]). This initiates a transfer. A series of clocks pulses will be driven out on the SPIx_CLK pin to complete the transfer. Each clock pulse on the SPIx_CLK pin causes the simultaneous transfer (in both directions) of one bit by both the master and slave SPI devices. CPU writes to the configuration bits in SPIDAT1 (not writing to SPIDAT1[15:0]) do not result in a new transfer. When the selected number of bits has been transmitted, the received data is transferred to the SPI receive buffer register (SPIBUF) for the CPU to read. Data is stored right-justified in SPIBUF.

In slave mode with 3-pin option, CPU writes to SPIDAT0[15:0] or SPIDAT1[15:0] makes the slave ready to transmit. CPU writes to the configuration bits in SPIDAT1 (not writing to SPIDAT1[15:0]) do not make the slave ready to transmit.

NOTE: Either SPIDAT0 or SPIDAT1 can be used on both master and slaves sides.

Figure 2. SPI 3-Pin Option



2.8 SPI Operation: 4-Pin with Chip Select Mode

The 4-pin with chip select option is a superset of the 3-pin option and uses the chip select (SPIx_SCS[n]) pin in addition to the clock (SPIx_CLK) and data (SPIx_SOMI and SPIx_SIMO) pins. Figure 3 shows the SPI 4-pin chip select option.

To select the 4-pin with chip select option, the SPIx_CLK, SPIx_SOMI, SPIx_SIMO, and SPIx_SCS[n] pins should be configured as functional pins by configuring the SPI pin control register 0 (SPIPC0). The SPIx_ENA pin can be used as a general-purpose I/O pin by configuring the SPIPC1 through SPIPC5 registers.

In SPI master mode, the SPIx_SOMI pin output buffer is in a high-impedance state and the SPIx_CLK, SPIx_SIMO, and SPIx_SCS[n] pin output buffer is enabled. In SPI slave mode, the SPIx_CLK, SPIx_SIMO, and SPIx_SCS[n] pin output buffer is in a high-impedance state, and the SPIx_SOMI pin output buffer is enabled when SPIx_SCS[n] is asserted and in a high-impedance state when SPIx_SCS[n] is deasserted.

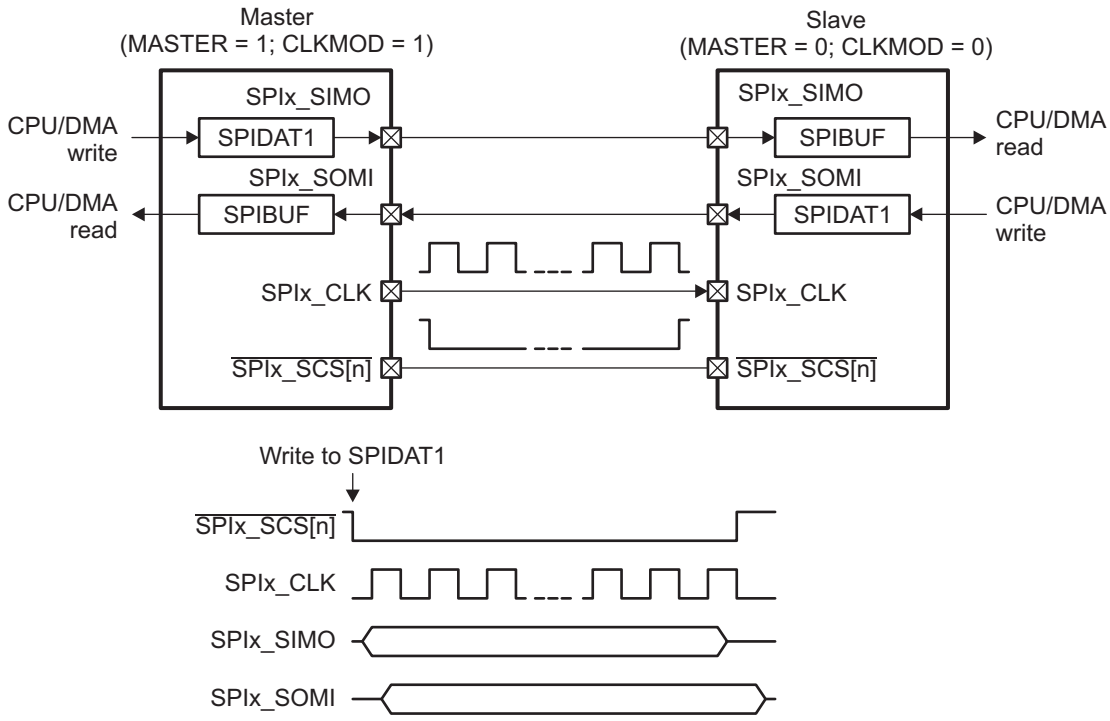
In slave mode with the chip select option enabled, the SPI ignores all transactions on the bus unless SPIx_SCS[n] is asserted by the bus master. It also 3-states its output pin when SPIx_SCS[n] is deasserted by the master to avoid conflicting with the active slave device on the bus.

In master mode, the SPIx_SCS[n] pin functions as an output, and toggles when a specific slave device is selected. However, this is most useful on devices that support multiple SPIx_SCS[n] pins. The SPI only supports a single SPIx_SCS[n] and so the usefulness of this pin in master mode is limited. In practice, general-purpose I/O pins are needed to support multiple slave device chip selects.

However, one reason to use the SPIx_SCS[n] pin as a functional pin for the SPI master is to take advantage of the timing parameters that can be set using the SPI delay register (SPIDELAY). The SPIDELAY allows delays to be added automatically so that the slave timing requirements between clock and chip select may be more easily met. Another reason would be to make use of the error detection built into the SPI.

NOTE: Either SPIDAT0 or SPIDAT1 can be used on both master and slaves sides.

Figure 3. SPI 4-Pin Option with SPIx_SCS[n]



NOTE: During an SPI transfer, if the slave mode SPI detects a deassertion of its chip select even before its internal character length counter overflows, then it 3-states its SPIx_SOMI pin. Once this condition has occurred, if a SPIx_CLK edge is detected while the chip select is deasserted, the SPI stops the transfer and sets an error flag DLENERR (data length) and generates an interrupt if enabled.

2.9 SPI Operation: 4-Pin with Enable Mode

The 4-pin with enable option is a superset of the 3-pin option and uses the enable ($\overline{\text{SPIx\_ENA}}$) pin in addition to the clock (SPIx_CLK) and data (SPIx_SOMI and SPIx_SIMO) pins. Figure 4 shows the SPI 4-pin enable option.

To select the 4-pin with enable option, the SPIx_CLK, SPIx_SOMI, SPIx_SIMO, and $\overline{\text{SPIx\_ENA}}$ pins should be configured as functional pins by configuring the SPI pin control register 0 (SPIPC0). The $\overline{\text{SPIx\_SCS[n]}}$ pins can be used as general-purpose I/O pins by configuring the SPIPC1 through SPIPC5 registers.

In SPI master mode, the SPIx_SOMI and $\overline{\text{SPIx\_ENA}}$ pin output buffer is in a high-impedance state and the SPIx_CLK and SPIx_SIMO pin output buffer is enabled. In SPI slave mode, the SPIx_CLK and SPIx_SIMO pin output buffer is in a high-impedance state, and the SPIx_SOMI pin output buffer is enabled. In SPI slave mode, the $\overline{\text{SPIx\_ENA}}$ pin output buffer enable depends upon the status of the transmit buffer and the configuration of the ENABLEHIGHZ bit in the SPI interrupt register (SPIINT0).

The handshake operation works this way:

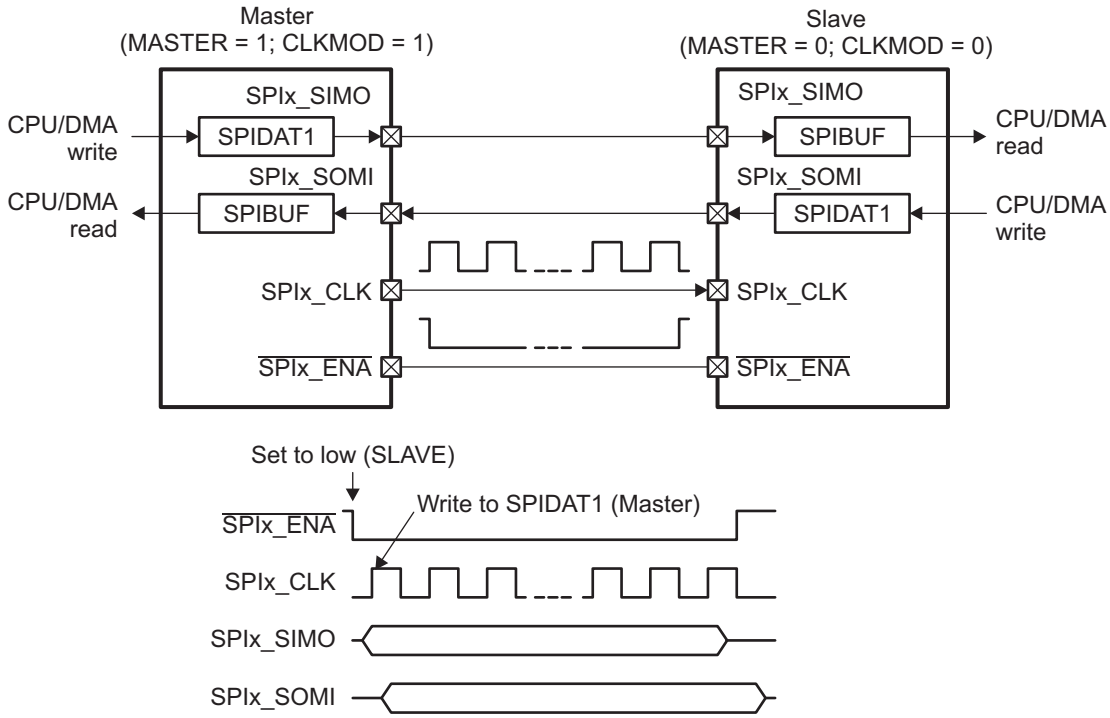
- After a transfer completes, both the master and slave SPI modules need to be serviced.
- The slave SPI deasserts $\overline{\text{SPIx\_ENA}}$ after the transfer, indicating it requires servicing and is not ready.
- The slave should begin servicing its SPI by first reading receive data from the SPI receive buffer register (SPIBUF).
- Next, the slave device should write transmit data to the SPI transmit data registers (SPIDAT0 or SPIDAT1). This causes the slave SPI to assert $\overline{\text{SPIx\_ENA}}$ indicating it is ready for the next transmission.
- In parallel, the master device can service its SPI at any time. It does not need to insert a delay before writing to its SPIDAT0 or SPIDAT1 in order to avoid overrunning the slave device. Instead, the master SPI module will automatically delay the next transfer until the slave has asserted $\overline{\text{SPIx\_ENA}}$ again to indicate it is ready for the transmission.

This handshake allows the two SPIs to communicate at the maximum rate possible. Without the handshake pin, the master must insert a delay between each transfer long enough to support the worst case response time of the slave servicing its SPI or risk an overrun condition. With the handshake, the throughput is determined by the average response time of the two devices servicing their SPI ports.

The $\overline{\text{SPIx\_ENA}}$ pin can be driven in a push-pull or open-drain mode, depending upon the setting of the ENABLEHIGHZ bit.

NOTE: Either SPIDAT0 or SPIDAT1 can be used on both master and slaves sides.

Figure 4. SPI 4-Pin Option with $\overline{\text{SPIx\_ENA}}$



2.10 SPI Operation: 5-Pin Mode

The 5-pin mode is a superset of both 4-pin modes. To use the 5-pin mode, both the $\overline{\text{SPIx\_ENA}}$ and the $\overline{\text{SPIx\_SCS[n]}}$ pins must be configured as functional pins, in addition to the SPIx_CLK, SPIx_SIMO, and SPIx_SOMI pins by configuring the SPI pin control register 0 (SPIPC0). Figure 5 shows the SPI 5-pin option.

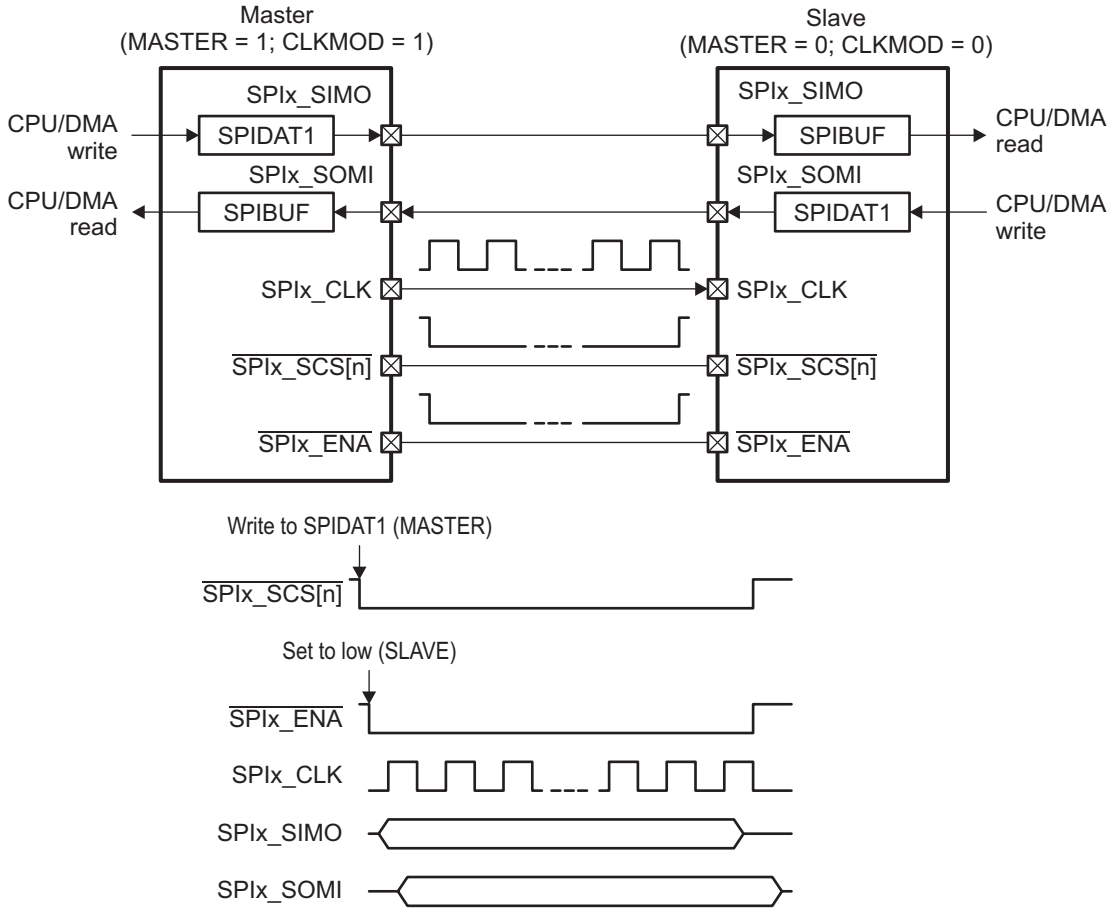
In SPI master mode, the SPIx_SOMI and $\overline{\text{SPIx\_ENA}}$ pin output buffer is in a high-impedance state and the SPIx_CLK, SPIx_SIMO, and $\overline{\text{SPIx\_SCS[n]}}$ pin output buffer is enabled. In SPI slave mode, the SPIx_CLK, SPIx_SIMO, and $\overline{\text{SPIx\_SCS[n]}}$ pin output buffer is in a high-impedance state, and the SPIx_SOMI pin output buffer is enabled and disabled asynchronously by the $\overline{\text{SPIx\_SCS[n]}}$ input and the $\overline{\text{SPIx\_ENA}}$ pin output buffer enable depends upon the status of the transmit buffer and the state of the $\overline{\text{SPIx\_SCS[n]}}$ input. In SPI slave mode, the assertion of the $\overline{\text{SPIx\_ENA}}$ pin by the slave is delayed until the master asserts $\overline{\text{SPIx\_SCS[n]}}$, thereby, allowing multiple SPI slaves on a single SPI bus, each slave with its own enable pin.

If the $\overline{\text{SPIx\_ENA}}$ pin is in high-impedance mode (ENABLEHIGHZ = 1 in the SPI interrupt register (SPIINT0)), the slave SPI will put this signal into the high-impedance state by default. The slave SPI will drive the $\overline{\text{SPIx\_ENA}}$ signal low when new data is written to the slave transmit shift register and the slave has been selected by the master ($\overline{\text{SPIx\_SCS[n]}}$ is low).

If the $\overline{\text{SPIx\_ENA}}$ pin is in push-pull mode (ENABLEHIGHZ = 0), the slave SPI will drive this pin high by default when it is in functional mode. The slave SPI will drive the $\overline{\text{SPIx\_ENA}}$ signal low when new data is written to the slave transmit shift register and the slave is selected by the master ($\overline{\text{SPIx\_SCS[n]}}$ is low). If the slave is deselected by the master ($\overline{\text{SPIx\_SCS[n]}}$ goes high), the slave $\overline{\text{SPIx\_ENA}}$ signal is driven high automatically.

NOTE: Either SPIDAT0 or SPIDAT1 can be used on both master and slaves sides.

Figure 5. SPI 5-Pin Option with $\overline{\text{SPIx\_ENA}}$ and $\overline{\text{SPIx\_SCS[n]}}$



NOTE: Push-Pull mode of the $\overline{\text{SPIx\_ENA}}$ pin can be used only when there is a single slave in the system. When there are multiple SPI slave devices connected to the common $\overline{\text{SPIx\_ENA}}$ pin, all the slaves should configure their $\overline{\text{SPIx\_ENA}}$ pins in high-impedance mode.

During an SPI transfer, if slave mode SPI detects a deassertion of its chip select even before its internal character length counter overflows, then it 3-states its SPIx_SOMI and $\overline{\text{SPIx\_ENA}}$ (if SPIINT0.ENABLEHIGHZ bit is set to 1) pins. Once this condition has occurred, if a SPIx_CLK edge is detected while the chip select is deasserted, then the SPI stops that transfer and sets an error flag DLENERR (data length) and generates an interrupt if enabled.

2.11 Data Formats

The SPI provides the capability to configure four independent data formats. These formats are configured by programming the corresponding SPI data format registers (SPIFMT*n*). In each data format, the following characteristics of the SPI operation are selected:

- Character length from 2 to 16 bits: The character length is configured by the SPIFMT*n*.CHARLEN field.
- Shift direction (MSB first or LSB first): The shift out direction is configured by the SPIFMT*n*.SHIFTDIR bit.
- Clock polarity: The clock polarity is configured by the SPIFMT*n*.POLARITY bit.
- Clock phase: The clock phase is configured by the SPIFMT*n*.PHASE bit.

The data format is chosen on each transaction. Transmit data is written to the SPI transmit data register 1 (SPIDAT1) and in the same write the data word format select (DFSEL) bit in SPIDAT1 indicates which data format is to be used for the next transaction. Alternatively, the data format can be configured once and applies to all transactions that follow until the data format is changed.

2.11.1 Character Length

The character length is configured by the SPIFMT*n*.CHARLEN bit. Legal values are 2 bits (2h) to 16 bits (10h). The character length is independently configured for each of the four data formats; and it must be programmed in both master mode and slave mode.

Transmit data is written to SPIDAT1. The transmit data must be written right-justified irrespective of the character length. The SPI automatically sends out the data correctly based on the chosen data format.

Figure 6 shows how a 12-bit word (EC9h) needs to be written to the transmit buffer in order to be transmitted correctly.

Figure 6. Format for Transmitting 12-Bit Word

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
x	x	x	x	1	1	1	0	1	1	0	0	1	0	0	1

The data received in SPIBUF is right-justified irrespective of the character length and is padded with 0s when character length is less than 16.

Figure 7 shows how a 10-bit word (3A2h) is stored in the buffer once it is received.

Figure 7. Format for 10-Bit Received Word

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	1	1	1	0	1	0	0	0	1	0

2.11.2 Shift Direction

The shift out direction is configured as most-significant bit (MSB) first or least significant bit (LSB) first. The shift out direction is selected by the SPIFMT*n*.SHIFTDIR bit. The shift out direction is independently configured for each of the four data formats.

- When SPIFMT*n*.SHIFTDIR is 0, the transmit data is shifted out MSB first.
- When SPIFMT*n*.SHIFTDIR is 1, the transmit data is shifted out LSB first.

2.11.3 Clock Phase and Polarity

The SPI provides the flexibility to program four different clock mode combinations that SPIx_CLK may operate, enabling a choice of the clock phase (delay or no delay) and the clock polarity (rising edge or falling edge). When operating with PHASE active, the SPI makes the first bit of data available after SPIDAT1 is written and before the first edge of SPIx_CLK. The data input and output edges depend on the values of both the POLARITY and PHASE bits as shown in Table 7.

Table 7. Clocking Modes

POLARITY	PHASE	Action
0	0	Data is output on the rising edge of SPIx_CLK. Input data is latched on the falling edge.
0	1	Data is output one half-cycle before the first rising edge of SPIx_CLK and on subsequent falling edges. Input data is latched on the rising edge of SPIx_CLK.
1	0	Data is output on the falling edge of SPIx_CLK. Input data is latched on the rising edge.
1	1	Data is output one half-cycle before the first falling edge of SPIx_CLK and on subsequent rising edges. Input data is latched on the falling edge of SPIx_CLK.

Figure 8 to Figure 11 illustrate the four possible signals of SPIx_CLK corresponding to each mode. Having four signal options allows the SPI to interface with different types of serial devices. Also shown are the SPIx_CLK control bit polarity and phase values corresponding to each signal.

Figure 8. Clock Mode with POLARITY = 0 and PHASE = 0

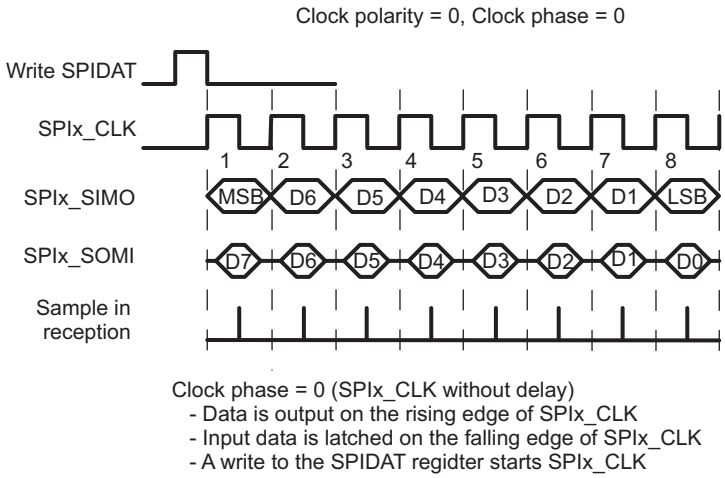
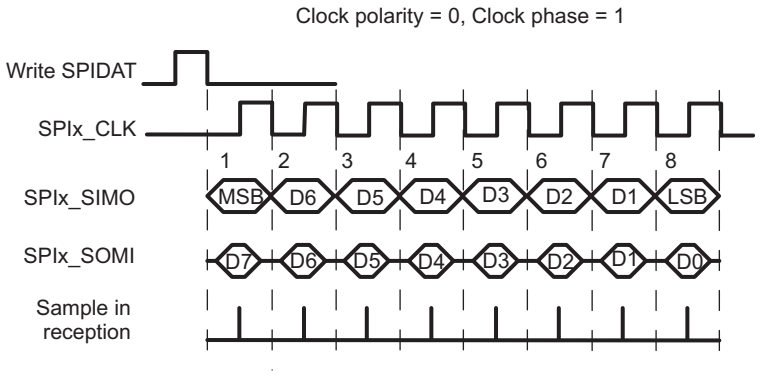
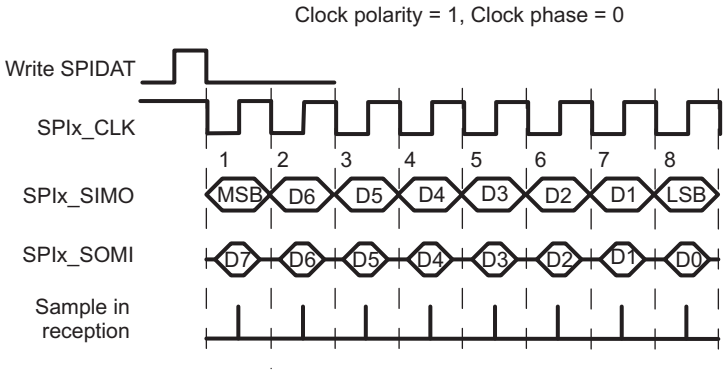


Figure 9. Clock Mode with POLARITY = 0 and PHASE = 1



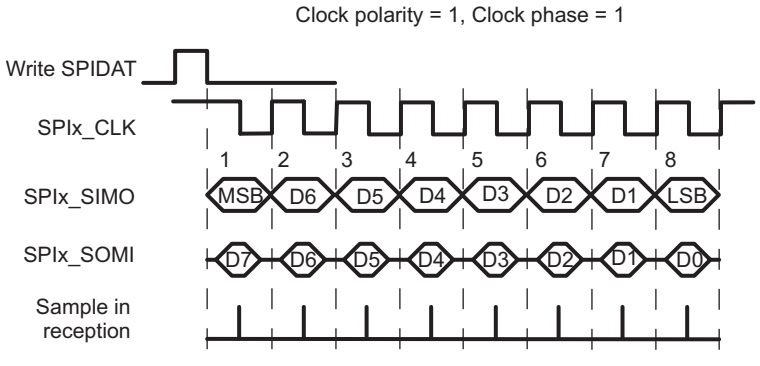
Clock phase = 1 (SPIx_CLK with delay)
- Data is output one-half cycle before the first rising of SPIx_CLK and on subsequent falling edges of SPIx_CLK
- Input data is latched on the rising edge of SPIx_CLK

Figure 10. Clock Mode with POLARITY = 1 and PHASE = 0



Clock phase = 0 (SPIx_CLK without delay)
- Data is output on the falling edge of SPIx_CLK
- Input data is latched on the rising edge of SPIx_CLK
- A write to the SPIDAT register starts SPIx_CLK

Figure 11. Clock Mode with POLARITY = 1 and PHASE = 1

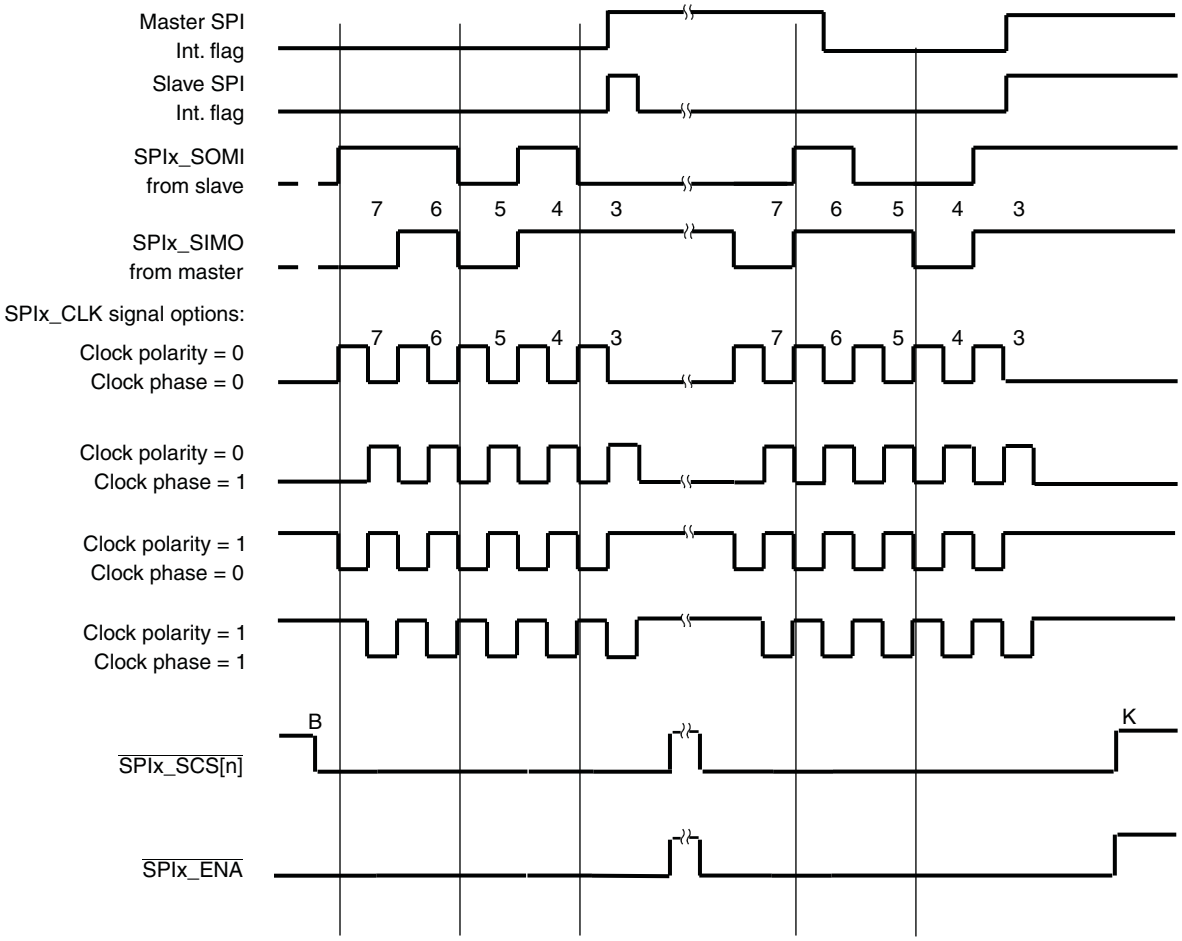


Clock phase = 1 (SPIx_CLK with delay)
- Data is output one-half cycle before the first falling edge of SPIx_CLK and on the subsequent rising edges of SPIx_CLK
- Input data is latched on the falling edge of SPIx_CLK

2.11.4 SPI Data Transfer Example

Figure 12 illustrates an SPI data transfer between two devices using a character length of five bits.

Figure 12. Five Bits per Character (5-Pin Option)



2.12 Interrupt Support

The SPI interrupt system is controlled by three registers:

- The SPI interrupt level register (SPILVL) controls the interrrupt level. The interrupt level must be set to select the level one interrupt (INT1).
- The SPI interrupt register (SPIINT) contains bits to selectively enable/disable each interrupt event.
- The SPI flag register (SPIFLG) contains flags indicating the interrupt conditions that have occurred.

To identify the interrupt source in the SPI peripheral, the CPU reads the SPI flag status register (SPIFLG) or the INTVECT1 code in the SPI interrupt vector register 1 (INTVEC1).

Check your device-specific data manual for details on the exact CPU interrupt numbers assigned to the SPI interrupts.

2.13 DMA Events Support

If handling the SPI message traffic on a character-by-character basis requires too much CPU overhead, then the CPU can configure the system DMA to handle the SPI data transfer.

The SPI module has two DMA synchronization event outputs for receive (REVT) and transmit (XEVT), allowing DMA transfers to be triggered by SPI read receive and write transmit events. The SPI module enables DMA requests by enabling the DMA request enable (DMAREQEN) bit in the SPI interrupt register (SPIINT0).

When a character is to be transmitted the SPI module signals the DMA via the XEVT signal. The DMA controller then transfers the data from the source buffer into the SPI transmit data register (SPIDAT1). When a character is received, the SPI module signals the DMA via the REVT signal. The DMA controller then reads the data from the SPI receive buffer register (SPIBUF) and transfers it to a destination buffer for ready access.

In most cases, if the DMA is being used to service received data from the SPI, the receive interrupt enable (RXINTEN) bit in SPIINT0 should be cleared to 0. This prevents the CPU from responding to the received data in addition to the DMA. For specific SPI synchronization event number assignments and detailed DMA features, see your device-specific data manual.

2.14 Robustness Features

The SPI module includes many features to make the SPI communication link robust. A internal loopback test mode can be used to facilitate a power on self test routine. Additionally, the SPI master continually monitors the bus for faults on its data line. The handshaking between master and slave can be monitored as well, and appropriate actions can be taken (interrupt, timeout) when the handshake breaks down. The following sections describe these robustness features in more detail.

2.14.1 SPI Internal Loopback Test Mode (Master Only)

CAUTION

The internal loop-back self-test mode should not be entered during a normal data transaction or unpredictable operation may occur.

To select the loopback mode, the SPIx_CLK, SPIx_SOMI, SPIx_SIMO pins should be configured as functional pins by configuring the SPI pin control register 0 (SPIPC0) and by setting the LOOPBACK bit in the SPI global control register 1 (SPIGCR1). The $\overline{\text{SPIx\_ENA}}$ and $\overline{\text{SPIx\_SCS[n]}}$ pins can be used as general-purpose I/O pins by configuring the SPIPC1 through SPIPC5 registers. The internal loop-back self-test mode can be utilized to test the SPI transmit path and receive path including the transmit and receive buffers. In this mode, the transmit signal is internally fed back to the receiver and the SPIx_SIMO, SPIx_SOMI, and SPIx_CLK pins are in a high-impedance state. This mode allows the CPU to write into the transmit buffer, and check that the receive buffer contains the correct transmit data. If an error occurs the corresponding error is set within the status field.

2.14.2 SPI Transmission Continuous Self-Test

During a data transfer, the SPI inputs the value from its data output pin on the appropriate SPIx_CLK edge. This value is compared against the expected value and any difference indicates a fault on the SPI bus. If a fault is detected, then the BITERR bit in the SPI receive buffer register (SPIBUF) and the BITERRFLG bit in the SPI flag register (SPIFLG) are set and an error interrupt is generated if enabled. The SPI continuous self-test mode is not available in SPI loopback mode.

2.14.3 SPI Detection of Slave Desynchronization

In the 4-pin with enable and 5-pin modes, the SPI master can monitor the slave $\overline{\text{SPIx\_ENA}}$ activity to detect a desynchronization event.

Some conditions that may cause a desynchronization event are:

- Master or slave device being reset during a transmission.
- Asserting a software reset of the SPI module during transmission.
- Having an incorrect SPI pin configuration, causing the $\overline{\text{SPIx\_ENA}}$ pin to behave incorrectly.
- Signal integrity problem causing additional clocks to be recognized by the slave.

The master can detect two desynchronization error conditions on the $\overline{\text{SPIx\_ENA}}$ pin:

1. Slave deasserts $\overline{\text{SPIx\_ENA}}$ after a transmission has begun, but before it completes.
2. Slave fails to deassert $\overline{\text{SPIx\_ENA}}$ within a certain time period after the completion of the last bit of the transmission.

The first error condition is straightforward to detect. To detect the second error condition, the SPI module includes an eight-bit counter with a timeout count that can be configured through the T2EDELAY field in the SPI delay register (SPIDELAY).

When a desynchronization event is detected, the DESYNC bit in the SPI receive buffer register (SPIBUF) and the DESYNCFLG bit in the SPI flag register (SPIFLG) are set and a desynchronization error interrupt is asserted if enabled.

NOTE: Remember that even though the desynchronization is detected by the master device, the problem causing the desynchronization event can be on either the master or the slave device.

The T2EDELAY period begins once the T2CDELAY period terminates or after the data shifting period in case the T2CDELAY is disabled. It defines the maximum time for the slave to deassert the $\overline{\text{SPIx\_ENA}}$ signal. If the slave device does not deassert the $\overline{\text{SPIx\_ENA}}$ signal before the T2EDELAY timeout value expires, the SPIFLG.DESYNC flag is set and a desynchronization interrupt is asserted if enabled. The T2E delay period does not always complete, sometimes it is skipped or terminated early. The T2E delay period terminates immediately after the $\overline{\text{SPIx\_ENA}}$ input is sampled (using the SPI module clock at intervals of SPIFMT*n*.PRESCALE + 2) as deasserted. However, assuming the T2E period completes its duration is specified by:

Maximum duration of T2EDELAY period = SPIDELAY.T2EDELAY + SPIFMTn.PRESCALE + 2 (SPI module clock cycles)

The T2EDELAY period is enabled only when the $\overline{\text{SPIx\_ENA}}$ is asserted at the beginning of the T2E delay period, the SPIDELAY.T2EDELAY field has a non-zero value, and SPIFMT*n*.WAITENA bit is set to 1.

2.14.4 $\overline{\text{SPIx\_ENA}}$ Signal Time-Out

In 5-pin mode, in addition to the slave desynchronization detection, the master can also detect whether the slave fails to respond to the $\overline{\text{SPIx\_SCS[n]}}$ signal by asserting $\overline{\text{SPIx\_ENA}}$ in a timely manner.

This condition could be the result of a serious error, or it could simply be the result of the slave device taking too long to service its SPI.

To detect this condition, the C2EDELAY field in the SPI delay register (SPIDELAY) is used. The C2EDELAY period begins once the C2TDELAY period terminates or when the master asserts $\overline{\text{SPIx\_SCS[n]}}$ (if C2TDELAY is disabled). It defines the maximum time for the addressed slave to respond by activating the $\overline{\text{SPIx\_ENA}}$ signal. If the slave does not respond with the $\overline{\text{SPIx\_ENA}}$ signal before the timeout value expires, then the TIMEOUT bit in the SPI receive buffer register (SPIBUF) and the TIMEOUTFLG bit in the SPI flag register (SPIFLG) are set, an interrupt is asserted if enabled, and the current transfer is terminated. The C2E delay period does not always complete, sometimes it is skipped or terminated early. The C2E delay period terminates immediately after the $\overline{\text{SPIx\_ENA}}$ input is sampled (using the SPI module clock at intervals of SPIFMT*n*.PRESCALE + 2) as asserted. However, assuming the C2E period completes its duration is specified by:

Maximum duration of C2EDELAY period = SPIDELAY.C2EDELAY + SPIFMTn.PRESCALE + 2 (SPI module clock cycles)

The C2EDELAY period is enabled only when the $\overline{\text{SPIx\_ENA}}$ is deasserted at the beginning of the C2E delay period and SPIFMT*n*.WAITENA bit is set to 1. If SPIFMT*n*.WAITENA bit is set to 1 and C2EDELAY is cleared to 0, then the master waits indefinitely for the slave to assert $\overline{\text{SPIx\_ENA}}$.

2.14.5 SPI Data Length Error

An SPI can generate an error flag by detecting any mismatch in length of received/transmitted data with the programmed character length under certain conditions.

Master Mode: During a data transfer, if the SPI detects a deassertion of the $\overline{\text{SPIx\_ENA}}$ pin (by the slave) while the character counter is not overflowed, then an error flag is set indicating the data length error. This can be caused by a slave receiving extra clocks (because of noise on the SPIx_CLK line).

NOTE: In SPI master mode, the data length error will be generated only if the $\overline{\text{SPIx\_ENA}}$ pin is used as a functional pin.

Slave Mode: During a transfer, if the SPI detects a deassertion of the $\overline{\text{SPIx\_SCS[n]}}$ pin before its character length counter overflows, then an error flag is set indicating the data length error. If the slave SPI misses one or more SPIx_CLK pulses from the master, this situation can occur. This error in slave mode would mean that both the transmitted and received data were not complete.

NOTE: In SPI slave mode, the data length error flag will be generated only if the $\overline{\text{SPIx\_SCS[n]}}$ pin is configured as a functional pin.

2.15 Reset Considerations

This section describes the software and hardware reset considerations.

2.15.1 Software Reset Considerations

The SPI module contains a software reset (RESET) bit in the SPI global control register 0 (SPIGCR0) that is used to reset the SPI module. As a result of a reset, the SPI module register values go to their reset state. The RESET bit must be set before any operation on the SPI is done.

2.15.2 Hardware Reset Considerations

In the event of a hardware reset, the SPI module register values go to their reset state and the application software needs to reprogram the registers to the desired values.

2.16 Power Management

The SPI module can be put in either local or global low-power mode. Global low-power mode is asserted by the system and is not controlled by the SPI. During global low-power mode, all clocks to the SPI are turned off so the module is completely inactive.

The SPI local low-power mode is asserted by setting the POWERDOWN bit in the SPI global control register 1 (SPIGCR1). Setting this bit stops the clocks to the SPI internal logic and the SPI registers. Setting the POWERDOWN bit causes the SPI to enter local low-power mode and clearing the POWERDOWN bit causes SPI to exit from local low-power mode. All the registers are accessible during local power-down mode as any register access enables the clock to SPI for that particular access alone.

Since entering a low-power mode has the effect of suspending all state machine activities, care must be taken when entering such modes to ensure that a valid state is entered when low-power mode is active. As a result, application software must ensure that a low-power mode is not entered during a transmission or reception of data.

2.17 General-Purpose I/O Pin

Each of the SPI pins may be programmed via the SPI pin control registers (SPIPC0 to SPIPC5) to be a general-purpose I/O (GPIO) pin.

When the SPI pins are not used as functional pins, they may be programmed to be either general input or general output pins by configuring SPIPC0. For example, in 3-pin mode, SPIx_SOMI, SPIx_SIMO, and SPIx_CLK must be configured as SPI pins, while the $\overline{\text{SPIx\_SCS[n]}}$ and $\overline{\text{SPIx\_ENA}}$ pins should be configured as GPIO pins. The direction is controlled by configuring SPIPC1.

If configured as a general-purpose output, then SPIPC3 controls the output value. There is also a write 1 to set (SPIPC4) and a write 1 to clear (SPIPC5) for the data out value. These registers allow different tasks running on the CPU to manipulate the SPI I/O pins without read-modify-write hazards.

SPIPC2 reflects the current value on the pin when the particular pin is configured as a functional or general-purpose input pin. When the pin is configured as a functional or general-purpose output pin, SPIPC2 indicates the value that is attempted to be driven on the pin.

2.18 Emulation Considerations

CAUTION

Viewing or otherwise reading the following SPI registers: SPIBUF, SPIFLG, and INTVEC1 through the JTAG debugger causes their contents to change, possibly invalidating the results of the debug session. Be sure to set up the debugger to avoid reading these registers.

The SPI module does not support soft or hard stop during emulation breakpoints. The SPI module will continue to run if an emulation breakpoint is encountered.

In addition, any status registers that are cleared after reading will be affected if viewed in a memory or watch window of the debugger; since the emulator will read these registers to update the value displayed in the window.

2.19 Initialization

Perform the following procedure for initializing the SPI:

1. Reset the SPI by clearing the RESET bit in the SPI global control register 0 (SPIGCR0) to 0.
2. Take the SPI out of reset by setting SPIGCR0.RESET to 1.
3. Configure the SPI for master or slave mode by configuring the CLKMOD and MASTER bits in the SPI global control register 1 (SPIGCR1).
4. Configure the SPI for 3-pin, 4-pin with chip select, 4-pin with enable, or 5-pin mode by configuring the SPI pin control register 0 (SPIPC0).
5. Chose the SPI data format register *n* (SPIFMT*n*) to be used by configuring the DFSEL bit in the SPI transmit data register (SPIDAT1). In slave mode, only SPIFMT0 is supported.
6. Configure the SPI data rate, character length, shift direction, phase, polarity and other format options using SPIFMT*n* selected in step 5.
7. If SPI master, then configure the master delay options using the SPI delay register (SPIDELAY). In slave mode, SPIDELAY is not relevant.
8. Select the error interrupt notifications by configuring the SPI interrupt register (SPIINT0) and the SPI interrupt level register (SPILVL).
9. Enable the SPI communication by setting the SPIGCR1.ENABLE to 1.
10. Setup and enable the DMA for SPI data handling and then enable the DMA servicing for the SPI data requests by setting the SPIINT0.DMAREQEN to 1.
11. Handle SPI data transfer requests using DMA and service any SPI error conditions using the interrupt service routine.

3 Registers

This section describes the SPI control, data, and pin registers. The offset is relative to the associated base address of the module. See your device-specific data manual for the memory address of these registers.

Table 8. SPI Registers

Offset Address	Acronym	Register Description	Section
0h	SPIGCR0	SPI Global Control Register 0	Section 3.1
4h	SPIGCR1	SPI Global Control Register 1	Section 3.2
8h	SPIINT0	SPI Interrupt Register	Section 3.3
Ch	SPILVL	SPI Interrupt Level Register	Section 3.4
10h	SPIFLG	SPI Flag Register	Section 3.5
14h	SPIPC0	SPI Pin Control Register 0 (Function)	Section 3.6
18h	SPIPC1	SPI Pin Control Register 1 (Direction)	Section 3.7
1Ch	SPIPC2	SPI Pin Control Register 2 (Input)	Section 3.8
20h	SPIPC3	SPI Pin Control Register 3 (Output)	Section 3.9
24h	SPIPC4	SPI Pin Control Register 4 (Set SPIPC3)	Section 3.10
28h	SPIPC5	SPI Pin Control Register 5 (Clear SPIPC3)	Section 3.11
38h	SPIDAT0	SPI Data Transmit Register 0	Section 3.12
3Ch	SPIDAT1	SPI Data Transmit Register 1 (Data Transmit and Format Select)	Section 3.13
40h	SPIBUF	SPI Receive Buffer Register	Section 3.14
44h	SPIEMU	SPI Receive Emulation Register	Section 3.15
48h	SPIDELAY	SPI Delay Register	Section 3.16
4Ch	SPIDEF	SPI Default Chip Select Register	Section 3.17
50h	SPIFMT0	SPI Data Format Register 0	Section 3.18
54h	SPIFMT1	SPI Data Format Register 1	Section 3.18
58h	SPIFMT2	SPI Data Format Register 2	Section 3.18
5Ch	SPIFMT3	SPI Data Format Register 3	Section 3.18
64h	INTVEC1	SPI Interrupt Vector Register 1	Section 3.19

3.1 SPI Global Control Register 0 (SPIGCR0)

The SPI global control register 0 (SPIGCR0) is shown in Figure 13 and described in Table 9.

Figure 13. SPI Global Control Register 0 (SPIGCR0)

31	16
Reserved	
R-0	

15	1	0
Reserved		RESET
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; *-n* = value after reset

Table 9. SPI Global Control Register 0 (SPIGCR0) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reads return zero and writes have no effect.
0	RESET		Reset bit for the module. This bit needs to be set to 1 before any operation on SPI can be done.
		0	SPI is in reset state.
		1	SPI is out of reset state.

3.2 SPI Global Control Register 1 (SPIGCR1)

The SPI global control register 1 (SPIGCR1) is shown in Figure 14 and described in Table 10.

Figure 14. SPI Global Control Register 1 (SPIGCR1)

31	25	24
Reserved		ENABLE
R-0		R/W-0
23	**17**	**16**
Reserved		LOOPBACK
R-0		R/W-0
15	**9**	**8**
Reserved		POWERDOWN
R-0		R/W-0

7	2	1	0
Reserved		CLKMOD	MASTER
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 10. SPI Global Control Register 1 (SPIGCR1) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return zero and writes have no effect.
24	ENABLE		SPI enable. This bit enables the SPI transfers. The other SPI configuration registers except SPIINT0.DMAREQEN should be configured before writing a 1 to this bit. This will prevent the SPI from responding to bus operations erroneously while it is in the process of being configured. The SPIINT0.DMAREQEN should be enabled after setting ENABLE. If SPIINT0.DMAREQEN is enabled before setting ENABLE then the first DMA request that occurs before the SPI is ready for data transfer may get dropped. When ENABLE bit is cleared to 0, the following SPI registers get forced to their default states (to 0s except for RXEMPTY bit in SPIBUF): • Both TX and RX shift registers • The TXDATA fields of SPIDAT0 and SPIDAT1 registers • All the fields of the SPIFLG register • Contents of SPIBUF and the internal RXBUF registers
		0	SPI is not activated for transfers.
		1	Activates SPI.
23-17	Reserved	0	Reads return zero and writes have no effect.
16	LOOPBACK		Internal loop-back test mode. The internal self-test option can be enabled by setting this bit. If the SPIx_SIMO and SPIx_SOMI pins are configured with SPI functionality, then the SPIx_SIMO pin is internally connected to the SPIx_SOMI pin. The transmit data is looped back as receive data and is stored in the receive field of the concerned buffer. Externally, during loop-back operation, the SPIx_CLK pin outputs an inactive value, SPIx_SIMO and SPIx_SOMI pins remain in high-impedance state. The SPI has to be initialized in master mode before the loop-back can be selected. If the SPI is initialized in slave mode or a data transfer is ongoing, errors may result.
		0	Internal loop-back test mode disabled.
		1	Internal loop-back test mode enabled.
15-9	Reserved	0	Reads return zero and writes have no effect.
8	POWERDOWN		When active, the SPI state machine enters a power-down state.
		0	The SPI is in active mode.
		1	The SPI is in power-down mode.
7-2	Reserved	0	Reads return zero and writes have no effect.

Table 10. SPI Global Control Register 1 (SPIGCR1) Field Descriptions (continued)

Bit	Field	Value	Description
1-0	CLKMOD,MASTER	0-3h	These two bits (CLKMOD,MASTER) determine whether the SPI operates in master or slave mode.
		0	SLAVE MODE. SPIx_CLK is an input from the master who initiates the transfers. Data is transmitted on the SPIx_SOMI pin and received on the SPIx_SIMO pin. The $\overline{\text{SPIx\_SCS[n]}}$ pin is an input pin if configured as SPI slave chip select. The $\overline{\text{SPIx\_ENA}}$ pin is an output pin if configured as the SPI enable pin.
		1h-2h	Reserved
		3h	MASTER MODE. SPIx_CLK is an output and the SPI initiates transfers. Data is transmitted on the SPIx_SIMO pin and received on the SPIx_SOMI pin. The $\overline{\text{SPIx\_SCS[n]}}$ pin is an output pin if configured as SPI slave chip select. The $\overline{\text{SPIx\_ENA}}$ pin is an input pin if configured as the SPI enable pin.

3.3 SPI Interrupt Register (SPIINT0)

The SPI interrupt register (SPIINT0) is shown in Figure 15 and described in Table 11.

Figure 15. SPI Interrupt Register (SPIINT0)

31						25	24
Reserved							ENABLEHIGHZ
R-0							R/W-0
23						17	16
Reserved							DMAREQEN
R-0							R/W-0
15						10	9	8
Reserved							TXINTENA	RXINTENA
R-0							R/W-0	R/W-0

7	6	5	4	3	2	1	0
Reserved	OVRNINTENA	Reserved	BITERRENA	DESYNCENA	PARERRENA	TIMEOUTENA	DLENERRENA
R-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 11. SPI Interrupt Register (SPIINT0) Field Descriptions

Bit	Field	Value	Description
31-25	Reserved	0	Reads return zero and writes have no effect.
24	ENABLEHIGHZ		$\overline{\text{SPIx\_ENA}}$ pin high-impedance enable. If ENABLEHIGHZ is enabled, the $\overline{\text{SPIx\_ENA}}$ pin (when it is configured as a WAIT functional output signal in a slave SPI) is forced to place it is output in high-impedance when not driving a low signal. If ENABLEHIGHZ is disabled, then the pin will output both a high and a low signal.
		0	$\overline{\text{SPIx\_ENA}}$ pin is pulled high when not active.
		1	$\overline{\text{SPIx\_ENA}}$ pin remains in high-impedance when not active.
23-17	Reserved	0	Reads return zero and writes have no effect.
16	DMAREQEN		DMA request enable. Enables the DMA request signal to be generated for both receive and transmit channels. Set DMAREQEN only after setting the SPIGCR1.ENABLE bit to 1.
		0	DMA is not used.
		1	DMA requests will be generated.
			Note: A transmit DMA request will be generated each time a transmit data is copied to the shift register either from TXBUF or directly from SPIDAT0/SPIDAT1.
			Note: A receive DMA request will be generated each time a received data is copied to SPIBUF register either from RXBUF or directly from the shift register.
15-10	Reserved	0	Reads return zero and writes have no effect.
9	TXINTENA		An interrupt is to be generated every time data is written to the shift register, so that a new data can be written to TXBUF. Setting this bit will generate an interrupt if the SPIFLG.TXINTFLG bit is set to 1.
		0	No interrupt will be generated upon SPIFLG.TXINTFLG being set to 1.
		1	Interrupt will be generated upon SPIFLG.TXINTFLG being set to 1.
8	RXINTENA		Receive interrupt enable. An interrupt is to be generated when the SPIFLG.RXINTFLAG bit is set.
		0	Interrupt will not be generated.
		1	Interrupt will be generated.
7	Reserved	0	Reads return zero and writes have no effect.
6	OVRNINTENA		Overrun interrupt enable. An interrupt is to be generated when the SPIFLG.OVRNINTFLG bit is set. The overrun interrupt is not useful if receive data is serviced with CPU interrupts because the overrun and receive events share a common level interrupt signal.
		0	Overrun interrupt will not be generated.
		1	Overrun interrupt will be generated.
5	Reserved	0	Reads return zero and writes have no effect.

Table 11. SPI Interrupt Register (SPIINT0) Field Descriptions (continued)

Bit	Field	Value	Description
4	BITERRENA		Enables interrupt on bit error. An interrupt is to be generated when the SPIFLG.BITERRFLG is set.
		0	No interrupt asserted upon bit error.
		1	Enables an interrupt on a bit error.
3	DESYNCENA		Enables interrupt on desynchronized slave. DESYNCENA is used in master mode only. The desynchronization monitor is active in master mode for the 4-pin with enable and 5-pin options. An interrupt is to be generated when the SPIFLG.DESYNCFLG is set.
		0	No interrupt asserted upon desynchronization error.
		1	Enables an interrupt on desynchronization of the slave.
2	PARERRENA		Enables interrupt on parity error. An interrupt is to be generated when the SPIFLG.PARERRFLG is set.
		0	No interrupt asserted upon parity error.
		1	Enables an interrupt on a parity error.
1	TIMEOUTENA		Enables interrupt on $\overline{\text{SPIx\_ENA}}$ signal time-out. An interrupt is to be generated when SPIFLG.TIMEOUTFLG is set.
		0	No interrupt asserted upon $\overline{\text{SPIx\_ENA}}$ signal time-out.
		1	Enables an interrupt on a time-out of the $\overline{\text{SPIx\_ENA}}$ signal.
0	DLENERRENA		Data length error interrupt enable. A data length error occurs under the following conditions. Master: In a 4-pin with $\overline{\text{SPIx\_ENA}}$ mode or 5-pin mode, if the $\overline{\text{SPIx\_ENA}}$ pin from the slave is deasserted before the master has completed its transfer, the data length error is set. That is, if the character length counter has not overflowed while $\overline{\text{SPIx\_ENA}}$ deassertion is detected, then it means that the slave has neither received full data from the master nor has it transmitted complete data. Slave: In a 4-pin with chip select mode or 5-pin mode, if the incoming valid $\overline{\text{SPIx\_SCS[n]}}$ pin is deactivated before the character length counter overflows, then data length error is set.
		0	No interrupt is generated upon data length error.
		1	Enables an interrupt when data length error occurs.

3.4 SPI Interrupt Level Register (SPILVL)

The SPI interrupt level register (SPILVL) is shown in Figure 16 and described in Table 12.

Figure 16. SPI Interrupt Level Register (SPILVL)

31							16
Reserved							
R-0							
15					10	9	8
Reserved						TXINTLVL	RXINTLVL
R-0						R/W-0	R/W-0
7	6	5	4	3	2	1	0
Reserved	OVRNINTLVL	Reserved	BITERRLVL	DESYNCLVL	PARERRLVL	TIMEOUTLVL	DLENERRLVL
R-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 12. SPI Interrupt Level Register (SPILVL) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	0	Reads return zero and writes have no effect.
9	TXINTLVL		Transmit interrupt level.
		0	Reserved
		1	Transmit interrupt is mapped to interrupt line INT1.
8	RXINTLVL		Receive interrupt level.
		0	Reserved
		1	Receive interrupt is mapped to interrupt line INT1.
7	Reserved	0	Reads return zero and writes have no effect.
6	OVRNINTLVL		Receive overrun interrupt level. The overrun interrupt is not useful if receive data is serviced with CPU interrupts because the overrun and receive events share a common level interrupt signal.
		0	Reserved
		1	Receive overrun interrupt is mapped to interrupt line INT1.
5	Reserved	0	Reads return zero and writes have no effect.
4	BITERRLVL		Bit error interrupt level.
		0	Reserved
		1	Bit error interrupt is mapped to interrupt line INT1.
3	DESYNCLVL		Desynchronized slave interrupt level. DESYNCLVL is used in master mode only.
		0	Reserved
		1	An interrupt due to desynchronization of the slave is mapped to interrupt line INT1.
2	PARERRLVL		Parity error interrupt level.
		0	Reserved
		1	A parity error interrupt is mapped to interrupt line INT1.
1	TIMEOUTLVL		$\overline{\text{SPIx\_ENA}}$ signal time-out interrupt level.
		0	Reserved
		1	An interrupt on a time-out of the $\overline{\text{SPIx\_ENA}}$ signal is mapped to interrupt line INT1.
0	DLENERRLVL		Data length error interrupt enable level.
		0	Reserved
		1	An interrupt on data length error is mapped to interrupt line INT1.

3.5 SPI Flag Register (SPIFLG)

The SPI flag register (SPIFLG) is shown in Figure 17 and described in Table 13.

Figure 17. SPI Flag Register (SPIFLG)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-100h															

15	14	13	12	11	10	9	8
Reserved						TXINTFLG	RXINTFLG
R-0						R-0	R/WC-0

7	6	5	4	3	2	1	0
Reserved	OVRNINTFLG	Reserved	BITERRFLG	DESYNCFLG	PARERRFLG	TIMEOUTFLG	DLENERRFLG
R-0	R/WC-0	R-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0	R/WC-0

LEGEND: R/W = Read/Write; R = Read only; WC = Write/Clear; -*n* = value after reset

Table 13. SPI Flag Register (SPIFLG) Field Descriptions

Bit	Field	Value	Description
31-10	Reserved	4000h	Reads return default value and writes have no effect.
9	TXINTFLG		Transmitter empty interrupt flag. Serves as an interrupt flag indicating that the transmit buffer (TXBUF) is empty and a new data can be written to it. This flag is set when a data is copied to the shift register either directly or from the TXBUF register. This bit is cleared by one of following ways: • Writing a new data to either SPIDAT0 or SPIDAT1 • Writing a 0 to SPIGCR1.ENABLE
		0	Transmit buffer is now full. No interrupt pending for transmitter empty.
		1	Transmit buffer is empty. An interrupt is pending to fill the transmitter.
8	RXINTFLG		Receiver full interrupt flag. This flag is set when a word is received and copied into the buffer register (SPIBUF). This bit is cleared under the following ways: • Reading the SPIBUF register. During emulation mode, however, a read to the emulation register (SPIEMU) does not clear this flag bit. • Reading INTVEC1 register when there is a receive buffer full interrupt • Writing a 1 to this bit • Writing a 0 to SPIGCR1.ENABLE • System reset
		0	No new received data pending. Receive buffer is empty.
		1	A newly received data is ready to be read. Receive buffer is full. **Note:** Clearing RXINTFLG bit by writing a 1 before reading the SPIBUF sets the RXEMPTY bit of the SPIBUF register too. This way, one can ignore a received data. However, if the internal RXBUF is already full, the data from RXBUF will be copied to SPIBUF and the RXEMPTY bit will be cleared again. The SPIBUF contents should be read first if this situation needs to be avoided.
7	Reserved	0	Reads return zero and writes have no effect.
6	OVRNINTFLG		Receiver overrun flag. The bit is set when a receive operation completes before the previous character has been read from the receive buffer. The bit indicates that the last received character has been overwritten and therefore lost. This bit is cleared under the following conditions: • Reading INTVEC1 register when there is a receive buffer overrun interrupt • Writing a 1 to this bit
		0	Overrun condition did not occur.
		1	Overrun condition has occurred. **Note:** Reading SPIBUF register does not clear the OVRNINTFLG bit. If an overrun interrupt is detected, then the SPIBUF may need to be read twice to get to the overrun buffer. This is due to the fact that the overrun will always occur to the internal RXBUF. Each read to the SPIBUF will result in RXBUF contents (if it is full) getting copied to SPIBUF. **Note:** A special condition under which OVRNINTFLG flag gets set. If both SPIBUF and RXBUF are already full and while another buffer receive is underway, if any errors like TIMEOUT, BITERR and DLENERR occur, then OVRNINTFLG will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a normal receiver overrun.

Table 13. SPI Flag Register (SPIFLG) Field Descriptions (continued)

Bit	Field	Value	Description
5	Reserved	0	Reads return zero and writes have no effect.
4	BITERRFLG		This bit is set when a mismatch of internal transmit data and transmitted data is detected. The SPI samples the signal of the transmit pin (master: SPIx_SIMO, slave: SPIx_SOMI) at the receive point (half clock cycle after transmit point). If the sampled value differs from the transmitted value a bit error is detected and the flag is set. A possible reason for a bit error can be a too high bit rate/capacitive load or another master/slave trying to transmit at the same time. This flag can be cleared by one of the following ways: • Write a 1 to this bit. • Set SPIGCR1.ENABLE bit to 0.
		0	No bit error occurred.
		1	A bit error occurred.
3	DESYNCFLG		Desynchronization of slave device. Desynchronization monitor is active in master mode only. The master monitors the $\overline{\text{SPIx\_ENA}}$ signal coming from the slave device and sets the DESYNCFLG bit if the $\overline{\text{SPIx\_ENA}}$ signal is not deasserted after the last bit is transmitted plus $t_{T2EDELAY}$. Desynchronization can occur if a slave device misses a clock edge coming from the master. This flag can be cleared by one of the following ways: • Write a 1 to this bit. • Set SPIGCR1.ENABLE bit to 0.
		0	No slave desynchronization detected.
		1	Slave is desynchronized
			Note: Inconsistency of DESYNCFLG in SPI. Due to the nature of this error, under some circumstances it is possible for a desynchronized error detected for the previous buffer to be visible in the current buffer. This is due to the fact that receive completion flag/interrupt will be generated when the buffer transfer is completed. But decence will be detected after the buffer transfer is completed. So, if CPU/DMA reads the received data quickly when an receive interrupt is detected, then the status flag may not reflect the correct decence condition.
2	PARERRFLG		Calculated parity differs from received parity bit. If the parity generator is enabled an even or odd parity bit is added at the end of a data word . During reception of the data word the parity generator calculates the reference parity and compares it to the received parity bit. In the event of a mismatch the PARERRFLG flag is set. This flag can be cleared by one of the following ways: • Write a 1 to this bit. • Set SPIGCR1.ENABLE bit to 0.
		0	No parity error detected.
		1	A parity error occurred.
1	TIMEOUTFLG		Time-out due to non-activation of $\overline{\text{SPIx\_ENA}}$ signal. This flag is applicable only for the master mode. The SPI generates a time-out because the slave hasn't responded in time by activating the $\overline{\text{SPIx\_ENA}}$ signal after the chip select signal has been activated. If a time-out condition is detected the corresponding chip select is deactivated immediately and the TIMEOUTFLG flag is set. This flag can be cleared by one of the following ways: • Write a 1 to this bit. • Set SPIGCR1.ENABLE bit to 0.
		0	No $\overline{\text{SPIx\_ENA}}$ signal time-out occurred.
		1	An $\overline{\text{SPIx\_ENA}}$ signal time-out occurred.
0	DLENERRFLG		Data length error flag. This flag can be cleared by one of the following ways: • Write a 1 to this bit. • Set SPIGCR1.ENABLE bit to 0.
		0	No data length error has occurred.
		1	A data length error has occurred.

3.6 SPI Pin Control Register 0 (SPIPC0)

The SPI pin control register 0 (SPIPC0) is shown in Figure 18 and described in Table 14.

Figure 18. SPI Pin Control Register 0 (SPIPC0)

31	16
Reserved	
R-0	

15	12	11	10	9	8
Reserved		SOMIFUN	SIMOFUN	CLKFUN	ENAFUN
R-0		R/W-0	R/W-0	R/W-0	R/W-0

7	0
SCS0FUN[*n*](1)	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

(1) Not all devices support multiple slave chip select ($\overline{\text{SPIx\_SCS[n]}}$) I/O pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.

Table 14. SPI Pin Control Register 0 (SPIPC0) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return zero and writes have no effect.
11	SOMIFUN		Slave out, master in pin function. This bit determines whether the SPIx_SOMI pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	SPIx_SOMI pin is a GPIO pin.
		1	SPIx_SOMI pin is a SPI functional pin.
10	SIMOFUN		Slave in, master out pin function. This bit determines whether the SPIx_SIMO pin is to be used as a general-purpose I/O pin or as a SPI functional pin.
		0	SPIx_SIMO pin is a GPIO pin.
		1	SPIx_SIMO pin is a SPI functional pin.
9	CLKFUN		SPI clock pin function. This bit determines whether the SPIx_CLK pin is to be used as a general-purpose I/O pin, or as a SPI functional pin.
		0	SPIx_CLK pin is a GPIO pin.
		1	SPIx_CLK pin is a SPI functional pin.
8	ENAFUN		SPI enable pin function. This bit determines whether the $\overline{\text{SPIx\_ENA}}$ pin is to be used as a general-purpose I/O pin, or as a SPI functional pin.
		0	$\overline{\text{SPIx\_ENA}}$ pin is a GPIO pin.
		1	$\overline{\text{SPIx\_ENA}}$ pin is a SPI functional pin.
7-0	SCS0FUN[*n*]		SPI chip select pin *n* function. This bit determines whether the $\overline{\text{SPIx\_SCS[n]}}$ pin is to be used as a general-purpose I/O pin, or as a SPI functional pin. Not all devices support multiple $\overline{\text{SPIx\_SCS[n]}}$ pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.
		0	$\overline{\text{SPIx\_SCS[n]}}$ pin is a GPIO pin.
		1	$\overline{\text{SPIx\_SCS[n]}}$ pin is a SPI functional pin.

3.7 SPI Pin Control Register 1 (SPIPC1)

The SPI pin control register 1 (SPIPC1) is shown in Figure 19 and described in Table 15.

Figure 19. SPI Pin Control Register 1 (SPIPC1)

31					16
Reserved					
R-0					
15	**12**	**11**	**10**	**9**	**8**
Reserved		SOMIDIR	SIMODIR	CLKDIR	ENADIR
R-0		R/W-0	R/W-0	R/W-0	R/W-0
7					**0**
SCS0DIR[*n*](1)					
R/W-0					

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

(1) Not all devices support multiple slave chip select ($\overline{\text{SPIx\_SCS[n]}}$) I/O pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.

Table 15. SPI Pin Control Register 1 (SPIPC1) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return zero and writes have no effect.
11	SOMIDIR		SPIx_SOMI pin direction. Controls the direction of the SPIx_SOMI pin when it is used as a general-purpose I/O pin. If the SPIx_SOMI pin is used as a SPI functional pin, the I/O direction is determined by whether the SPI is configured as master or slave.
		0	SPIx_SOMI pin is an input.
		1	SPIx_SOMI pin is an output.
10	SIMODIR		SPIx_SIMO pin direction. Controls the direction of the SPIx_SIMO pin when it is used as a general-purpose I/O pin. If the SPIx_SIMO pin is used as a SPI functional pin, the I/O direction is determined by whether the SPI is configured as master or slave.
		0	SPIx_SIMO pin is an input.
		1	SPIx_SIMO pin is an output.
9	CLKDIR		SPIx_CLK pin direction. Controls the direction of the SPIx_CLK pin when it is used as a general-purpose I/O pin. If the SPIx_CLK pin is used as a SPI functional pin, the I/O direction is determined by whether the SPI is configured as master or slave.
		0	SPIx_CLK pin is an input.
		1	SPIx_CLK pin is an output.
8	ENADIR		$\overline{\text{SPIx\_ENA}}$ pin direction. Controls the direction of the $\overline{\text{SPIx\_ENA}}$ pin when it is used as a general-purpose I/O pin. If the $\overline{\text{SPIx\_ENA}}$ pin is used as a SPI functional pin, then the I/O direction is determined by whether the SPI is configured as master or slave.
		0	$\overline{\text{SPIx\_ENA}}$ pin is an input.
		1	$\overline{\text{SPIx\_ENA}}$ pin is an output.
7-0	SCS0DIR[*n*]		$\overline{\text{SPIx\_SCS[n]}}$ pin direction. Controls the direction of the $\overline{\text{SPIx\_SCS[n]}}$ pin when it is used as a general-purpose I/O pin. If the $\overline{\text{SPIx\_SCS[n]}}$ pin is used as a SPI functional pin, then the I/O direction is determined by whether the SPI is configured as master or slave. Not all devices support multiple $\overline{\text{SPIx\_SCS[n]}}$ pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.
		0	$\overline{\text{SPIx\_SCS[n]}}$ pin is an input.
		1	$\overline{\text{SPIx\_SCS[n]}}$ pin is an output.

3.8 SPI Pin Control Register 2 (SPIPC2)

The SPI pin control register 2 (SPIPC2) is shown in Figure 20 and described in Table 16.

Figure 20. SPI Pin Control Register 2 (SPIPC2)

31	16
Reserved	
R-0	

15	12	11	10	9	8
Reserved		SOMIDIN	SIMODIN	CLKDIN	ENADIN
R-0		R-U	R-U	R-U	R-U

7	0
SCS0DIN[*n*](1)	
R-U	

LEGEND: R/W = Read/Write; R = Read only; U = Undefined; -*n* = value after reset

(1) Not all devices support multiple slave chip select ($\overline{\text{SPIx\_SCS[n]}}$) I/O pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.

Table 16. SPI Pin Control Register 2 (SPIPC2) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return zero and writes have no effect.
11	SOMIDIN		SPIx_SOMI data in. This bit reflects the value of the SPIx_SOMI pin.
		0	Current value of SPIx_SOMI pin is logic 0.
		1	Current value of SPIx_SOMI pin is logic 1.
10	SIMODIN		SPIx_SIMO data in. This bit reflects the value of the SPIx_SIMO pin.
		0	Current value of SPIx_SIMO pin is logic 0.
		1	Current value of SPIx_SIMO pin is logic 1.
9	CLKDIN		Clock data in. This bit reflects the value of the SPIx_CLK pin.
		0	Current value of SPIx_CLK pin is logic 0.
		1	Current value of SPIx_CLK pin is logic 1.
8	ENADIN		$\overline{\text{SPIx\_ENA}}$ data in. This bit reflects the value of the $\overline{\text{SPIx\_ENA}}$ pin.
		0	Current value of $\overline{\text{SPIx\_ENA}}$ pin is logic 0.
		1	Current value of $\overline{\text{SPIx\_ENA}}$ pin is logic 1.
7-0	SCS0DIN[*n*]		$\overline{\text{SPIx\_SCS[n]}}$ data in. This bit reflects the value of the $\overline{\text{SPIx\_SCS[n]}}$ pin. Not all devices support multiple $\overline{\text{SPIx\_SCS[n]}}$ pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.
		0	Current value of $\overline{\text{SPIx\_SCS[n]}}$ pin is logic 0.
		1	Current value of $\overline{\text{SPIx\_SCS[n]}}$ pin is logic 1.

3.9 SPI Pin Control Register 3 (SPIPC3)

The SPI pin control register 3 (SPIPC3) is shown in Figure 21 and described in Table 17.

Figure 21. SPI Pin Control Register 3 (SPIPC3)

31	16
Reserved	
R-0	

15	12	11	10	9	8
Reserved		SOMIDOUT	SIMODOUT	CLKDOUT	ENADOUT
R-0		R/W-0	R/W-0	R/W-0	R/W-0

7	0
SCS0DOUT[*n*](1)	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

(1) Not all devices support multiple slave chip select ($\overline{\text{SPIx\_SCS[n]}}$) I/O pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.

Table 17. SPI Pin Control Register 3 (SPIPC3) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return zero and writes have no effect.
11	SOMIDOUT		SPIx_SOMI data out write. This bit is only active when the SPIx_SOMI pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value of SPIx_SOMI pin is logic 0.
		1	Current value of SPIx_SOMI pin is logic 1.
10	SIMODOUT		SPIx_SIMO data out write. This bit is only active when the SPIx_SIMO pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value of SPIx_SIMO pin is logic 0.
		1	Current value of SPIx_SIMO pin is logic 1.
9	CLKDOUT		SPIx_CLK data out write. This bit is only active when the SPIx_CLK pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value of SPIx_CLK pin is logic 0.
		1	Current value of SPIx_CLK pin is logic 1.
8	ENADOUT		$\overline{\text{SPIx\_ENA}}$ data out write. Only active when the $\overline{\text{SPIx\_ENA}}$ pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin.
		0	Current value of $\overline{\text{SPIx\_ENA}}$ pin is logic 0.
		1	Current value of $\overline{\text{SPIx\_ENA}}$ pin is logic 1.
7-0	SCS0DOUT[*n*]		$\overline{\text{SPIx\_SCS[n]}}$ data out write. Only active when the $\overline{\text{SPIx\_SCS[n]}}$ pin is configured as a general-purpose I/O pin and configured as an output pin. The value of this bit indicates the value sent to the pin *n*. Not all devices support multiple $\overline{\text{SPIx\_SCS[n]}}$ pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.
		0	Current value of $\overline{\text{SPIx\_SCS[n]}}$ pin is logic 0.
		1	Current value of $\overline{\text{SPIx\_SCS[n]}}$ pin is logic 1.

3.10 SPI Pin Control Register 4 (SPIPC4)

The SPI pin control register 4 (SPIPC4) is shown in Figure 22 and described in Table 18.

Figure 22. SPI Pin Control Register 4 (SPIPC4)

31	16
Reserved	
R-0	

15	12	11	10	9	8
Reserved		SOMISET	SIMOSET	CLKSET	ENASET
R-0		R/W-0	R/W-0	R/W-0	R/W-0

7	0
SCS0SET[*n*](1)	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

(1) Not all devices support multiple slave chip select ($\overline{\text{SPIx\_SCS[n]}}$) I/O pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.

Table 18. SPI Pin Control Register 4 (SPIPC4) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return zero and writes have no effect.
11	SOMISET		SPIx_SOMI data out set. This bit is only active when the SPIx_SOMI pin is configured as a general-purpose output pin. Reads return the value of the SPIx_SOMI pin.
		Write 0	No effect
		Write 1	SPIPC3.SOMIDOUT is set to 1.
10	SIMOSET		SPIx_SIMO data out set. This bit is only active when the SPIx_SIMO pin is configured as a general-purpose output pin. Reads return the value of the SPIx_SIMO pin.
		Write 0	No effect
		Write 1	SPIPC3.SIMODOUT is set to 1.
9	CLKSET		SPIx_CLK data out set. This bit is only active when the SPIx_CLK pin is configured as a general-purpose output pin. Reads return the value of the SPIx_CLK pin.
		Write 0	No effect
		Write 1	SPIPC3.CLKDOUT is set to 1.
8	ENASET		$\overline{\text{SPIx\_ENA}}$ data out set. This bit is only active when the $\overline{\text{SPIx\_ENA}}$ pin is configured as a general-purpose output pin. Reads return the value of the $\overline{\text{SPIx\_ENA}}$ pin.
		Write 0	No effect.
		Write 1	SPIPC3.ENADOUT is set to 1.
7-0	SCS0SET[*n*]		$\overline{\text{SPIx\_SCS[n]}}$ data out set. This bit is only active when the $\overline{\text{SPIx\_SCS[n]}}$ pin is configured as a general-purpose output pin. Reads return the value of the $\overline{\text{SPIx\_SCS[n]}}$ pin. Not all devices support multiple $\overline{\text{SPIx\_SCS[n]}}$ pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.
		Write 0	No effect
		Write 1	SPIPC3.SCS0DOUT is set to 1.

3.11 SPI Pin Control Register 5 (SPIPC5)

The SPI pin control register 5 (SPIPC5) is shown in Figure 23 and described in Table 19.

Figure 23. SPI Pin Control Register 5 (SPIPC5)

31	16
Reserved	
R-0	

15	12	11	10	9	8
Reserved		SOMICLR	SIMOCLR	CLKCLR	ENACLR
R-0		R/W-0	R/W-0	R/W-0	R/W-0

7	0
SCS0CLR[[*n*] (1)	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

(1) Not all devices support multiple slave chip select ($\overline{\text{SPIx\_SCS[n]}}$) I/O pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.

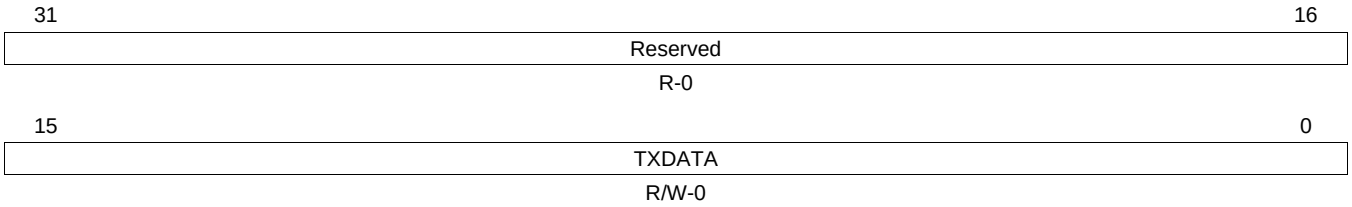
Table 19. SPI Pin Control Register 5 (SPIPC5) Field Descriptions

Bit	Field	Value	Description
31-12	Reserved	0	Reads return zero and writes have no effect.
11	SOMICLR		SPIx_SOMI data out clear. This bit is only active when the SPIx_SOMI pin is configured as a general-purpose output pin. Reads return the value of the SPIx_SOMI pin.
		Write 0	No effect.
		Write 1	SPIPC3.SOMIDOUT is cleared to 0.
10	SIMOCLR		SPIx_SIMO data out clear. This bit is only active when the SPIx_SIMO pin is configured as a general-purpose output pin. Reads return the value of the SPIx_SIMO pin.
		Write 0	No effect.
		Write 1	SPIPC3.SIMODOUT is cleared to 0.
9	CLKCLR		SPIx_CLK data out clear. This bit is only active when the SPIx_CLK pin is configured as a general-purpose output pin. Reads return the value of the SPIx_CLK pin.
		Write 0	No effect.
		Write 1	SPIPC3.CLKDOUT is cleared to 0.
8	ENACLR		$\overline{\text{SPIx\_ENA}}$ data out clear. This bit is only active when the $\overline{\text{SPIx\_ENA}}$ pin is configured as a general-purpose output pin. Reads return the value of the $\overline{\text{SPIx\_ENA}}$ pin.
		Write 0	No effect.
		Write 1	SPIPC3.ENADOUT is cleared to 0.
7-0	SCS0CLR[*n*]		$\overline{\text{SPIx\_SCS[n]}}$ data out clear. This bit is only active when the $\overline{\text{SPIx\_SCS[n]}}$ pin is configured as a general-purpose output pin. Reads return the value of the $\overline{\text{SPIx\_SCS[n]}}$ pin. Not all devices support multiple $\overline{\text{SPIx\_SCS[n]}}$ pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.
		Write 0	No effect.
		Write 1	SPIPC3.SCS0DOUT is cleared to 0.

3.12 SPI Transmit Data Register 0 (SPIDAT0)

The SPI transmit data register 0 (SPIDAT0) is shown in Figure 24 and described in Table 20.

Figure 24. SPI Data Register 0 (SPIDAT0)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20. SPI Data Register 0 (SPIDAT0) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return zero and writes have no effect.
15-0	TXDATA	0-FFFFh	SPI transmit data. When written, these bits will be copied to the shift register if it is empty. If the shift register is not empty, the TXBUF will hold the written values. SPIGCR1.ENABLE must be set to 1 before this register can be written to. Writing a 0 to the SPIGCR1.ENABLE forces the TXDATA[15:0] field to 0. **Note:** Irrespective of the character length, the transmit data should be right-justified before writing to SPIDAT0 register. **Note:** The default data format control register for SPIDAT0 is SPIFMT0. However, it is possible to reprogram the DFSEL field of SPIDAT1 before using SPIDAT0, to select a different SPIFMT*n* register.

3.13 SPI Transmit Data Register 1 (SPIDAT1)

The SPI transmit data register (SPIDAT1) is shown in Figure 25 and described in Table 21.

Figure 25. SPI Data Register 1 (SPIDAT1)

31	29	28	27	26	25	24
Reserved		CSHOLD	Reserved	WDEL	DFSEL	
R-0		R/W-0	R-0	R/W-0	R/W-0	

23	16
CSNR[*n*](1)	
R/W-0	

15	0
TXDATA	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

(1) Not all devices support multiple slave chip select ($\overline{\text{SPIx\_SCS[n]}}$) I/O pins, see your device-specific data manual for supported pins.

Table 21. SPI Data Register 1 (SPIDAT1) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads return zero and writes have no effect.
28	CSHOLD		Chip select hold mode. The CSHOLD bit is supported in master mode only. In slave mode, this bit is ignored. CSHOLD defines the behavior of the chip select line at the end of a data transfer.
		0	The chip select signal is deactivated at the end of a transfer after the T2CDELAY time has passed.
		1	The chip select signal is held active at the end of a transfer until a control field with new data and control information is loaded into SPIDAT1. If the new chip select hold information equals the previous one, the active chip select signal is extended until the end of transfer with CSHOLD cleared.
27	Reserved	0	Reads return zero and writes have no effect.
26	WDEL		Enable the delay counter at the end of the current transaction. The WDEL bit is supported in master mode only. In slave mode, this bit is ignored.
		0	No delay will be inserted. However, $\overline{\text{SPIx\_SCS[n]}}$ pin will still be deactivated for at least 2 SPI module clock cycles if CSHOLD = 0.
		1	After a transaction, SPIFMT*n*.WDELAY of the selected data format will be loaded into the delay counter. No transaction will be performed until the SPIFMT*n*.WDELAY counter overflows. The $\overline{\text{SPIx\_SCS[n]}}$ pin will be deactivated for at least (WDELAY + 2) × SPI module clock period.
25-24	DFSEL	0-3h	Data word format select
		0	Data word format 0 is selected
		1h	Data word format 1 is selected
		2h	Data word format 2 is selected
		3h	Data word format 3 is selected
			Note: Preselecting a Format Register. Writing to just the control field (using byte writes) does not initiate any SPI transfer in master mode. This feature can be used to set up SPIx_CLK phase or polarity before actually starting the transfer by just updating the DFSEL fields in the control field to select the required phase/polarity combination.
23-16	CSNR[*n*]		Chip select number. The CSNR field defines the state of the $\overline{\text{SPIx\_SCS[n]}}$ pins during a master data transfer. The value of the CSNR field is driven directly on the $\overline{\text{SPIx\_SCS[n]}}$ pins. Each bit in the CSNR field corresponds to an $\overline{\text{SPIx\_SCS[n]}}$ pin, for example, CSNR[0] corresponds to $\overline{\text{SPIx\_SCS[0]}}$ (see your device-specific data manual to determine how many SPI pins are available on your device).
			The state of the chip select pins when no transmissions are active is specified through the CSDEF field in the SPI default chip select register (SPIDEF). The chip select pins can remain in their active state by setting the CSHOLD bit to 1. When the SPI is configured in slave mode, this field must be written as 00h.
		0	$\overline{\text{SPIx\_SCS[n]}}$ pin is driven low.
		1	$\overline{\text{SPIx\_SCS[n]}}$ pin is driven high.

Table 21. SPI Data Register 1 (SPIDAT1) Field Descriptions (continued)

Bit	Field	Value	Description
15-0	TXDATA	0-FFFFh	Transfer data. When written, these bits will be copied to the shift register if it is empty. If the shift register is not empty, the TXBUF will hold the written values. SPIGCR1.ENABLE must be set to 1 before this register can be written to. Writing a 0 to the SPIGCR1.ENABLE forces the lower 16 bits of the SPIDAT1 to 0. **Note:** Irrespective of the character length, the transmit data should be right-justified before writing to SPIDAT1.

3.14 SPI Receive Buffer Register (SPIBUF)

The SPI receive buffer register (SPIBUF) is shown in Figure 26 and described in Table 22.

Figure 26. SPI Buffer Register (SPIBUF)

31	30	29	28	27	26	25	24
RXEMPTY	RXOVR	TXFULL	BITERR	DESYNC	PARERR	TIMEOUT	DLENERR
RS-1	RC-0	R-0	RC-0	RC-0	RC-0	RC-0	RC-0

23	16
Reserved	
R-0	

15	0
RXDATA	
R-0	

LEGEND: R/W = Read/Write; R = Read only; C = Clear; S = Set; -*n* = value after reset

Table 22. SPI Buffer Register (SPIBUF) Field Descriptions

Bit	Field	Value	Description
31	RXEMPTY		Receive data buffer empty. When host reads the SPIBUF field or the whole SPIBUF register this will automatically set the RXEMPTY flag. When a data transfer is completed, the received data is copied into SPIBUF, the RXEMPTY flag is cleared. This flag gets set to 1 under following conditions: • Reading the RXDATA portion of the SPIBUF register. • Writing 1 to clear the RXINTFLG bit in SPIFLG register.
		0	New data has been received and copied into the SPIBUF register.
		1	No data received since last reading of SPIBUF register.
			Write-Clearing the SPIFLG.RXINTFLG bit before reading the SPIBUF indicates the received data is being ignored. Conversely, SPIFLG.RXINTFLG can be cleared by reading the RXDATA portion of the SPIBUF register or the entire SPIBUF register.
30	RXOVR		Receive data buffer overrun. When a data transfer is completed and the received data is copied into the RXBUF while it is already full, RXOVR is set. An overrun always occurs to the RXBUF, and SPIBUF contents never get overwritten until after it is read by the CPU/DMA. Reading SPIBUF register does not clear the RXOVR bit. If an overrun interrupt is detected, then the SPIBUF may need to be read twice to get to the overrun buffer. This is due to the fact that the overrun will always occur to the internal RXBUF. Each read to the SPIBUF will result in RXBUF contents (if it is full) getting copied to SPIBUF. **Note:** A special condition under which RXOVR flag gets set. If both SPIBUF and RXBUF are already full and while another buffer receive is underway, if any errors like TIMEOUT, BITERR and DLENERR occur, then RXOVR will be set to indicate that the status flags are getting overwritten by the new transfer. This overrun should be treated like a normal receiver overrun.
		0	No receive data overrun condition occurred since last time reading the data field.
		1	A receive data overrun condition occurred since last time reading the data field.

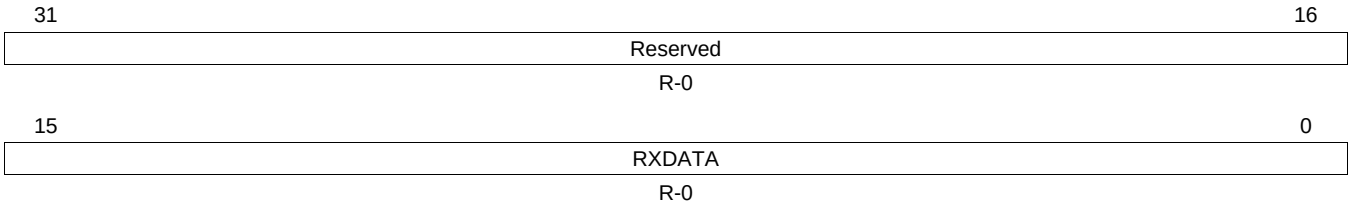
Table 22. SPI Buffer Register (SPIBUF) Field Descriptions (continued)

Bit	Field	Value	Description
29	TXFULL		Transmit data buffer full. This flag is a read-only flag. Writing into SPIDAT0 or SPIDAT1 field while the TX shift register is full will automatically set the TXFULL flag. Once the data is copied to the shift register, the TXFULL flag will be cleared. Writing to the SPIDAT0/SPIDAT1 register when both TXBUF and the TX shift register are empty does not set the TXFULL flag.
		0	The transmit buffer is empty; SPIDAT0/SPIDAT1 is ready to accept a new data.
		1	The transmit buffer is full; SPIDAT0/SPIDAT1 is not ready to accept new data.
28	BITERR		Bit error. There was a mismatch of internal transmit data and transmitted data. The SPI samples the signal of the transmit pin (master: SIMO, slave: SOMI) at the receive point (half clock cycle after transmit point). If the sampled value differs from the transmitted value, a bit error is detected and the flag BITERR is set. A possible reason for a bit error can be noise, a too-high bit rate/capacitive load, or another master/slave trying to transmit at the same time.
			Note: This flag is cleared to 0 when RXDATA portion of the SPIBUF register is read.
		0	No bit error occurred.
		1	A bit error occurred.
27	DESYNC		Desynchronization of slave device. This bit is active in master mode only. The master monitors the $\overline{\text{SPIx\_ENA}}$ signal coming from the slave device and sets the DESYNC flag if $\overline{\text{SPIx\_ENA}}$ is deactivated before the last reception point or after the last bit is transmitted plus $t_{T2EDELAY}$. If DESYNCENA is set, an interrupt is asserted. Desynchronization can occur if a slave device misses a clock edge coming from the master.
			Note: Possible inconsistency of DESYNC flag in SPI. Because of the nature of this error, under some circumstances it is possible for a desync error detected for the previous buffer to be visible in the current buffer. This is because the receive completion flag/interrupt will be generated when the buffer transfer is completed. But desync will be detected after the buffer transfer is completed. So, if CPU/DMA reads the received data quickly when an RXINT is detected, then the status flag may not reflect the correct desync condition.
			Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.
		0	No slave de-synchronization detected.
		1	A slave device is desynchronized.
26	PARERR		Parity error. The calculated parity differs from received parity bit. If the parity generator is enabled an even or odd parity bit is added at the end of a data word. During reception of the data word ,the parity generator calculates the reference parity and compares it to the received parity bit. If a mismatch is detected, the PARERR flag is set.
			Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.
		0	No parity error detected.
		1	A parity error occurred.
25	TIMEOUT		Time-out because of non-activation of $\overline{\text{SPIx\_ENA}}$ pin. This bit is valid in master mode only. The SPI generates a time-out because the slave hasn't responded in time by activating the $\overline{\text{SPIx\_ENA}}$ signal after the chip select signal has been activated. If a time-out condition is detected, the corresponding chip select is deactivated immediately and the TIMEOUT flag is set.
			Note: This flag is cleared to 0 when RXDATA portion of the SPIBUF register is read.
		0	No $\overline{\text{SPIx\_ENA}}$ pin time-out occurred.
		1	An $\overline{\text{SPIx\_ENA}}$ signal time-out occurred.
24	DLENERR		Data length error flag.
			Note: This flag is cleared to 0 when the RXDATA portion of the SPIBUF register is read.
		0	No data length error has occurred.
		1	A data length error has occurred.
23-16	Reserved	0	Reads return zero and writes have no effect.
15-0	RXDATA	0-FFFFh	SPI receive data. This is the received data, transferred from the receive shift-register at the end of a transfer completion. Irrespective of the programmed character length and the direction of shifting, the received data is stored right-justified in the register.

3.15 SPI Emulation Register (SPIEMU)

The SPI emulation register (SPIEMU) is shown in Figure 27 and described in Table 23.

Figure 27. SPI Emulation Register (SPIEMU)



LEGEND: R = Read only; -*n* = value after reset

Table 23. SPI Emulation Register (SPIEMU) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reads return zero and writes have no effect.
15-0	RXDATA	0-FFFFh	SPI receive data. SPI emulation is a mirror of the SPIBUF register. The only difference between SPIEMU and SPIBUF is that a read from SPIEMU does not clear any of the status flags.

3.16 SPI Delay Register (SPIDELAY)

The SPI delay register (SPIDELAY) is shown in Figure 28 and described in Table 24.

Figure 28. SPI Delay Register (SPIDELAY)

31	24	23	16
C2TDELAY		T2CDELAY	
R/W-0		R/W-0	

15	8	7	0
T2EDELAY		C2EDELAY	
R/W-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 24. SPI Delay Register (SPIDELAY) Field Descriptions

Bit	Field	Value	Description
31-24	C2TDELAY	0-FFh	Chip-select-active-to-transmit-start-delay. C2TDELAY is used in master mode only. It defines a setup time for the slave device that delays the data transmission from the chip select active edge by a multiple of SPI module clock cycles. C2TDELAY can be configured between 3 and 257 SPI module clock cycles. See Figure 29. The setup time value is calculated as follows: $t_{C2TDELAY}$ = (C2TDELAY + 2) × SPI module clock period **Note:** If C2TDELAY = 0, then $t_{C2TDELAY}$ = 0. Example: SPI module clock = 25 MHz -> SPI module clock period = 40 ns; C2TDELAY = 06h; > $t_{C2TDELAY}$ = 320 ns; When the chip select signal becomes active, the slave has to prepare for data transfer within 320 ns. **Note:** If phase = 1, the delay between $\overline{\text{SPIx\_SCS[n]}}$ falling edge to the first edge of SPIx_CLK will have an additional 0.5 SPIx_CLK period delay. This delay is as per the SPI protocol.
23-16	T2CDELAY	0-FFh	Transmit-end-to-chip-select-inactive-delay. T2CDELAY is used in master mode only. It defines a hold time for the slave device that delays the chip select deactivation by a multiple of SPI module clock cycles after the last bit is transferred. T2CDELAY can be configured between 2 and 256 SPI module clock cycles. See Figure 30. The hold time value is calculated as follows: $t_{T2CDELAY}$ = (T2CDELAY + 1) × SPI module clock period **Note:** If T2CDELAY = 0, then $t_{T2CDELAY}$ = 0 Example: VBUSPCLK = 25 MHz -> VBUSPCLK period = 40 ns; T2CDELAY = 03h; > $t_{T2CDELAY}$ = 160 ns; After the last data bit (or parity bit) is being transferred the chip select signal is held active for 160 ns. **Note:** If phase = 0, then between the last edge of SPIx_CLK and rise-edge of $\overline{\text{SPIx\_SCS[n]}}$ there will be an additional delay of 0.5 SPIx_CLK period. This is as per the SPI protocol. Both C2TDELAY and T2CDELAY counters will not have any dependancy on the $\overline{\text{SPIx\_ENA}}$ pin value. Even if the $\overline{\text{SPIx\_ENA}}$ pin is asserted by the slave, the master will continue to delay the start of SPIx_CLK until the C2TDELAY counter overflows. Similarly, even if the $\overline{\text{SPIx\_ENA}}$ pin is deasserted by the slave, the master will continue to hold the $\overline{\text{SPIx\_SCS[n]}}$ pins active until the T2CDELAY counter overflows. This way, it is assured that the setup/hold times of the $\overline{\text{SPIx\_SCS[n]}}$ pins are determined by the delay timers alone. To achieve better throughput, it should be ensured that these two timers are kept at the minimum possible values.

Table 24. SPI Delay Register (SPIDELAY) Field Descriptions (continued)

Bit	Field	Value	Description
15-8	T2EDELAY	0-FFh	Transmit-data-finished-to-$\overline{\text{SPIx\_ENA}}$-pin-inactive-time-out. T2EDELAY is used in master mode only. It defines a time-out value as a multiple of SPI clock before the $\overline{\text{SPIx\_ENA}}$ signal has to become inactive and after the CS becomes inactive. The SPI clock depends on which data format is selected. If the slave device is missing one or more clock edges, it is becoming desynchronized. Although the master has finished the data transfer the slave is still waiting for the missed clock pulses and the $\overline{\text{SPIx\_ENA}}$ signal is not disabled. The T2EDELAY defines a time-out value that triggers the DESYNC flag, if the $\overline{\text{SPIx\_ENA}}$ signal is not deactivated in time. The DESYNC flag is set to indicate that the slave device did not deassert its $\overline{\text{SPIx\_ENA}}$ pin in time to acknowledge that it has received all the bits of the sent character. The DESYNC flag is also set if the SPI detects a deassertion of the $\overline{\text{SPIx\_ENA}}$ pin even before the end of the transmission. See Figure 31. The time-out value is calculated as follows: $t_{T2EDELAY}$ = T2EDELAY/SPIclock Example: SPIclock = 8 Mbit/s; T2EDELAY = 10h; > $t_{T2EDELAY}$ = 2 µs; The slave device has to disable the $\overline{\text{SPIx\_ENA}}$ signal within 2 µs; otherwise, the DESYNC flag in SPIFLG is set and an interrupt is asserted if enabled.
7-0	C2EDELAY	0-FFh	Chip-select-active-to-$\overline{\text{SPIx\_ENA}}$-signal-active-time-out. C2EDELAY is used only in master mode and it applies only if the addressed slave generates an $\overline{\text{SPIx\_ENA}}$ signal as a hardware handshake response. C2EDELAY defines the maximum time between the SPI activates the chip select signal and the addressed slave has to respond by activating the $\overline{\text{SPIx\_ENA}}$ signal. C2EDELAY defines a time-out value as a multiple of SPI clocks. See Figure 32. **Note:** If the slave device is not responding with the $\overline{\text{SPIx\_ENA}}$ signal before the time-out value is reached, the TIMEOUT flag in SPIFLG is set and an interrupt is asserted if enabled. The timeout value is calculated as follows: $t_{C2EDELAY}$ = C2EDELAY/SPIclock Example: SPIclock = 8 Mbit/s; C2EDELAY = 30h; > $t_{C2EDELAY}$ = 6 µs; The slave device has to activate the $\overline{\text{SPIx\_ENA}}$ signal within 6 µs after the SPI has activated the chip select signal ($\overline{\text{SPIx\_SCS[n]}}$); otherwise, the TIMEOUT flag in SPIFLG is set and an interrupt is asserted if enabled.

Figure 29. Example: $t_{C2TDELAY}$ = 8 SPI Module Clock Cycles

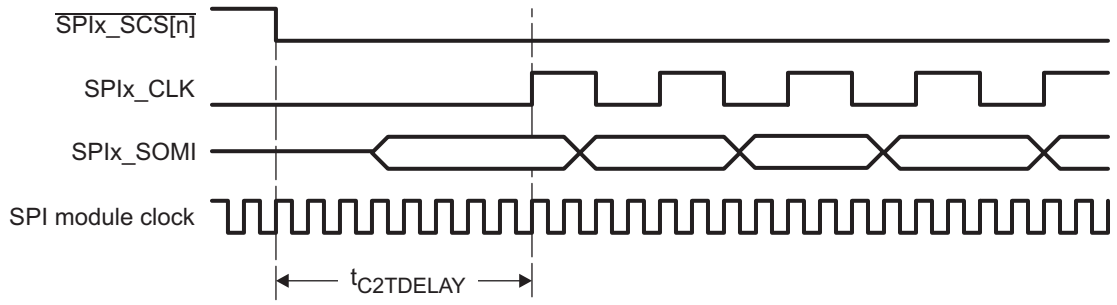


Figure 30. Example: $t_{T2CDELAY}$ = 4 SPI Module Clock Cycles

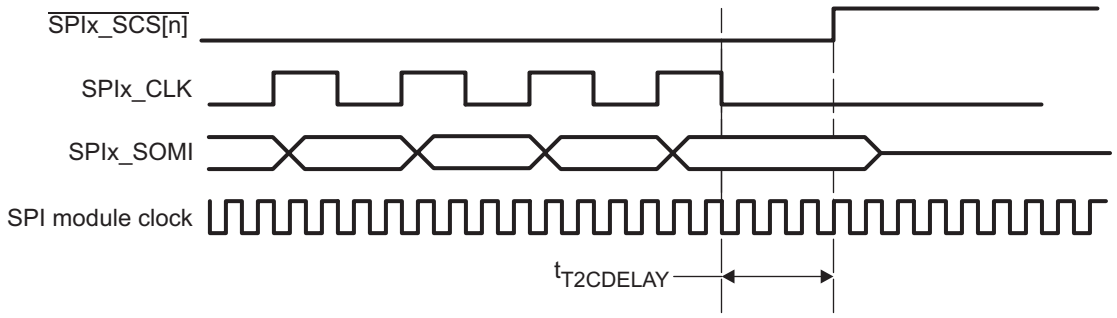


Figure 31. Transmit-Data-Finished-to-SPIx_ENA-Inactive-Timeout

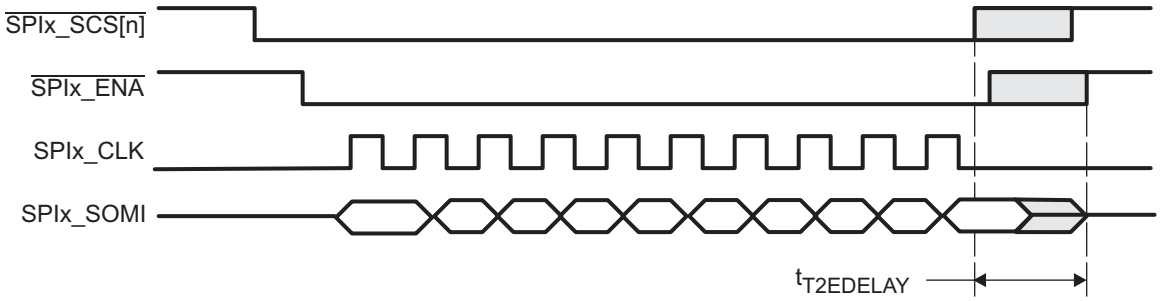
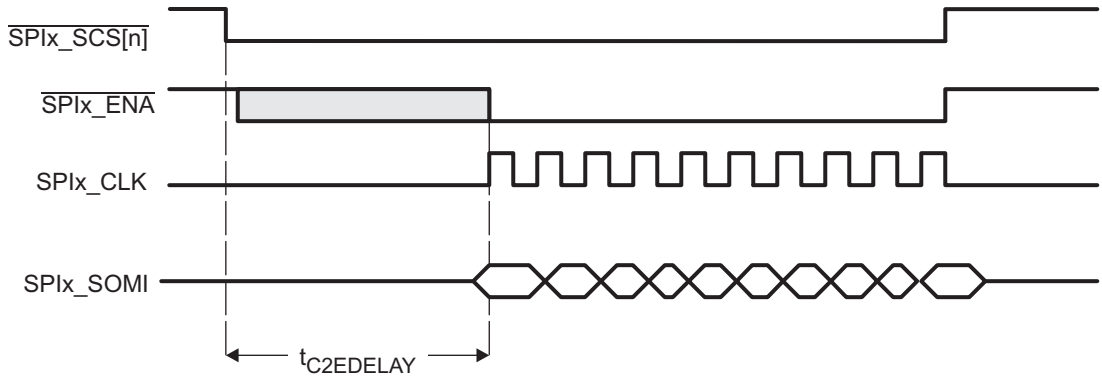


Figure 32. Chip-Select-Active-to-SPIx_ENA-Signal-Active-Timeout



3.17 SPI Default Chip Select Register (SPIDEF)

The SPI default chip select register (SPIDEF) is shown in Figure 33 and described in Table 25.

Figure 33. SPI Default Chip Select Register (SPIDEF)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		CSDEF[*n*](1)	
R-0		R/W-FFh	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

(1) Not all devices support multiple slave chip select ($\overline{\text{SPIx\_SCS[n]}}$) I/O pins, see your device-specific data manual for supported pins. If the pins are not available, the corresponding bit is reserved and should be cleared to 0.

Table 25. SPI Default Chip Select Register (SPIDEF) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reads return zero and writes have no effect.
7-0	CSDEF[*n*]		Chip select default pattern. The CSDEF field defines the state of the the $\overline{\text{SPIx\_SCS[n]}}$ pins when no transmissions are performed. The value of the CSDEF field is driven directly on the $\overline{\text{SPIx\_SCS[n]}}$ pins. Each bit in the CSDEF field corresponds to an $\overline{\text{SPIx\_SCS[n]}}$ pin, for example, CSDEF[0] corresponds to $\overline{\text{SPIx\_SCS[0]}}$ (see your device-specific data manual to determine how many SPI pins are available on your device). The state of the chip select pins during a transmission is specified through the CSNR field in the SPI transmit data register (SPIDAT1). The chip select pins can remain in their active state by setting the CSHOLD bit in SPIDAT1 to 1. In slave mode, the CSDEF field should be set to FFh.
		0	$\overline{\text{SPIx\_SCS[n]}}$ pin is driven low.
		1	$\overline{\text{SPIx\_SCS[n]}}$ pin is driven high.

3.18 SPI Data Format Registers (SPIFMT*n*)

The SPI data format registers (SPIFMT0, SPIFMT1, SPIFMT2, and SPIFMT3) are shown in Figure 34 and described in Table 26.

Figure 34. SPI Data Format Register (SPIFMT*n*)

31	30	29	24
Reserved		WDELAY	
R-0		R/W-0	

23	22	21	20	19	18	17	16
PARPOL	PARENA	WAITENA	SHIFTDIR	Reserved	DISCSTIMERS	POLARITY	PHASE
R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0

15	8
PRESCALE	
R/W-0	

7	5	4	0
Reserved		CHARLEN	
R-0		R/W-0	

LEGEND: R/W = Read/Write; R = Read only; WP = Write in privilege mode only; -*n* = value after reset

Table 26. SPI Data Format Register (SPIFMT*n*) Field Descriptions

Bit	Field	Value	Description
31-30	Reserved	0	Reads return zero and writes have no effect.
29-24	WDELAY	0-3Fh	Delay in between transmissions. Idle time that will be applied at the end of the current transmission if the bit WDEL is set in the current buffer. The delay to be applied is equal to: WDELAY × $P_{SPI\ module\ clock}$ + 2 × $P_{SPI\ module\ clock}$ $P_{SPI\ module\ clock}$ -> Period of SPI module clock
23	PARPOL		Parity polarity: even or odd. PARPOL can be modified in privilege mode only.
		0	An even parity flag is added at the end of the transmit data stream.
		1	An odd parity flag is added at the end of the transmit data stream.
22	PARENA		Parity enable.
		0	No parity generation/ verification is performed.
		1	A parity is transmitted at the end of each transmit data stream. At the end of a transfer the parity generator compares the received parity bit with the locally calculated parity flag. If the parity bits do not match the PARERR flag is set in the corresponding control field. The parity type (even or odd) can be selected via the PARPOL bit.
21	WAITENA		The master waits for $\overline{\text{SPIx\_ENA}}$ signal from slave. WAITENA is considered in master mode only. In slave mode this bit has no meaning. WAITENA enables a flexible SPI network where slaves with $\overline{\text{SPIx\_ENA}}$ signal and slaves without $\overline{\text{SPIx\_ENA}}$ signal can be mixed.
		0	The SPI does not wait for the $\overline{\text{SPIx\_ENA}}$ signal from the slave and directly starts the transfer.
		1	Before the SPI starts the data transfer it waits for the $\overline{\text{SPIx\_ENA}}$ signal to become low. If the $\overline{\text{SPIx\_ENA}}$ signal is not pulled down by the addressed slave before the internal time-out counter (C2EDELAY) overflows, then the master aborts the transfer and sets the TIMEOUT error flag.
20	SHIFTDIR		Shift direction.
		0	Most significant bit is shifted out first.
		1	Least significant bit is shifted out first.
19	Reserved	0	Reads return zero and writes have no effect.
18	DISCSTIMERS		Disable chip select timers for this format register. The C2TDELAY and T2CDELAY timers are by default enabled for all the data format registers. Using this bit, these timers can be disabled for a particular data format if not required. When a master is handling multiple slaves, with varied set-up hold requirement, the application can selectively choose to include or not include the chip select delay timers for any slaves.
		0	Both C2TDELAY and T2CDELAY counts are inserted for the chip selects.
		1	No C2TDELAY or T2CDELAY is inserted in the chip select timings.

Table 26. SPI Data Format Register (SPIFMT*n*) Field Descriptions (continued)

Bit	Field	Value	Description
17	POLARITY		SPI clock polarity.
		0	SPI clock signal is low-inactive (before and after data transfer the clock signal is low).
		1	SPI clock signal is high-inactive (before and after data transfer the clock signal is high).
16	PHASE		SPI clock delay.
		0	SPI clock signal is not delayed versus the transmit/receive data stream. The first data bit is transmitted with the first clock edge and the first bit is received with the second (inverse) clock edge.
		1	SPI clock signal is delayed by a half SPI clock cycle versus the transmit/receive data stream. The first transmit bit has to output prior to the first clock edge. The master and slave receive the first bit with the first edge.
15-8	PRESCALE	2h-FFh	SPI prescaler. It determines the bit transfer rate if the SPI is the network master and is directly derived from the SPI module clock. If the SPI is configured as slave, PRESCALE needs to be configured to a valid value, but PRESCALE is ignored. The clock rate can be calculated as: SPI clock frequency = SPI module clock/(PRESCALE + 1) **Note:** PRESCALE values less than 2h are not supported.
7-5	Reserved	0	Reads return zero and writes have no effect.
4-0	CHARLEN	0-1Fh	SPI data word length. Legal values are 2h (data word length = 2 bit) to 10h (data word length = 16). Illegal values, such as 0 or 1Fh are not detected and their effect is indeterminate.

3.19 SPI Interrupt Vector Register 1 (INTVEC1)

The SPI interrupt vector register 1 (INTVEC1) is shown in Figure 35 and described in Table 27.

Figure 35. SPI Interrupt Vector Register 1 (INTVEC1)

31	16
Reserved	
R-0	

15	6	5	1	0
Reserved		INTVECT1		Rsvd
R-0		R-0		R-0

LEGEND: R = Read only; -*n* = value after reset

Table 27. SPI Interrupt Vector Register 1 (INTVEC1) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reads return zero and writes have no effect.
5-1	INTVECT1	0-1Fh	Interrupt vector for interrupt line INT1. INTVECT1 returns the vector of the pending interrupt at interrupt line INT1. If more than one interrupt is pending, INTVECT1 always references the highest priority interrupt source first. The interrupts available for SPI in the descending order of their priorities are as given below. • Transmission error Interrupt • Receive buffer overrun interrupt • Receive buffer full interrupt • Transmit buffer empty interrupt The INTVECT1 field just reflects the status of SPIFLG in a vectorized format. So, any updates to SPIFLG will automatically reflect in the vector value in this register. Vectors for each of these interrupts will be reflected on the INTVECT1 bits, when they occur. Reading the vectors for the receive buffer overrun and receive buffer full interrupts will automatically clear the respective flags in the SPIFLG. Reading the vector register when transmitter empty is indicated does not clear the TXINTFLG in SPIFLG. Writing a new data to SPIDAT0/SPIDAT1 clears the transmitter empty interrupt. On reading the INTVECT1 bits, the vector of the next highest priority interrupt (if any) will be then reflected on the INTVECT1 bits. If two or more interrupts occur simultaneously, the vector for the highest priority interrupt will be reflected on the INTVECT1 bits. The following are the SPI interrupt vectors for line INT1:
		0	No interrupt pending
		1h-10h	Reserved
		11h	Error interrupt pending. Refer to lower halfword of SPIINT0 to determine more details about the type of error.
		12h	The pending interrupt is receive buffer full interrupt.
		13h	The pending interrupt is receive buffer overrun interrupt.
		14h	The pending interrupt is transmit buffer empty interrupt.
		15h-1Fh	Reserved
0	Reserved	0	Reads return zero and writes have no effect.

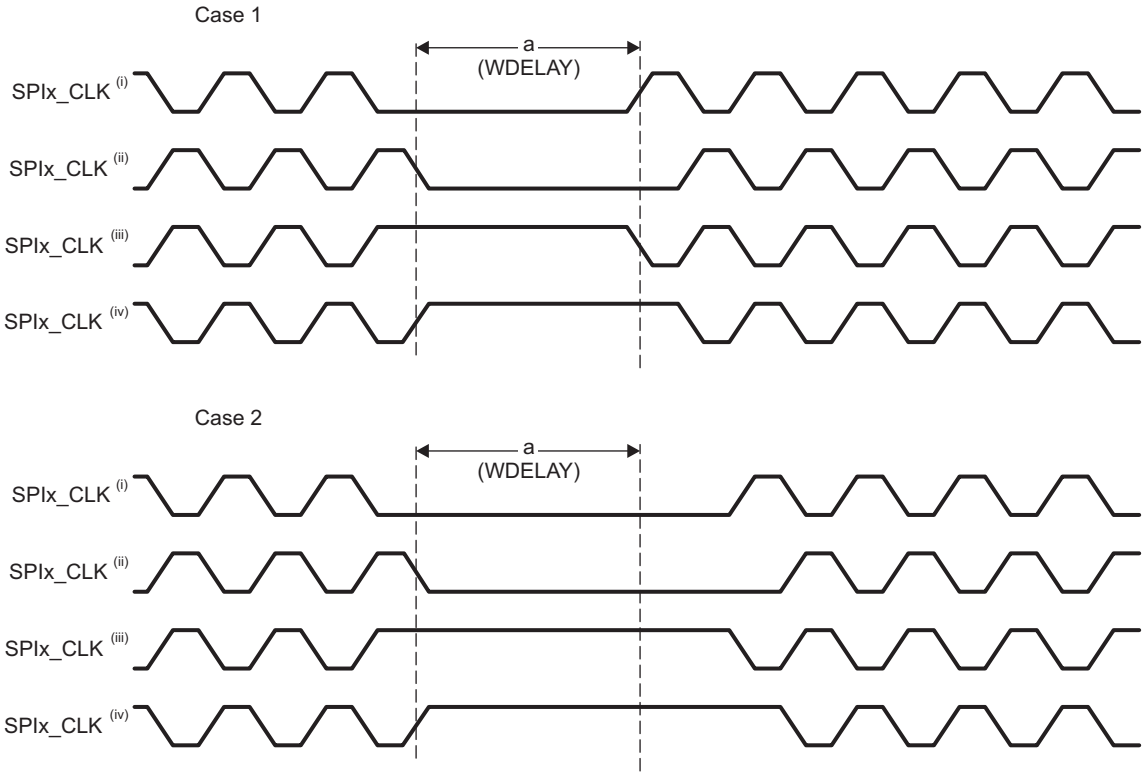
Appendix A Timing Diagrams

This appendix contains timing diagrams illustrating the C2TDELAY, C2EDELAY, T2CDELAY, T2EDELAY, and WDELAY delays and their interaction with the SPIx_SCS[n] and SPIx_ENA pins for all SPI modes.

A.1 SPI 3-Pin Mode

Figure 36 illustrates the WDELAY option in SPI 3-pin master mode. This is the only delay available in this mode. In CASE1, a new transfer is initiated during the WDELAY period and the transfer begins immediately after the WDELAY period ends. In CASE2, while WDELAY has completed, a new transfer will not begin until SPIDAT0/SPIDAT1 have been written with new data.

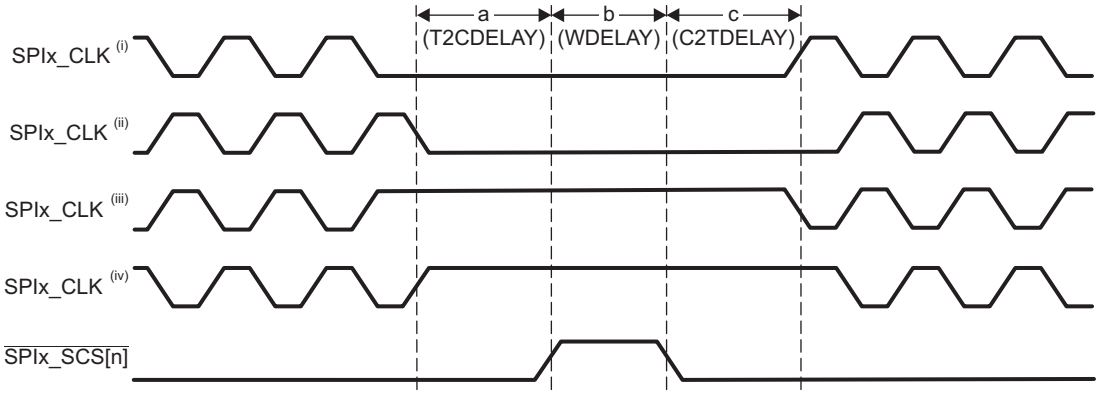
Figure 36. SPI 3-Pin Master Mode with WDELAY



A.2 SPI 4-Pin with SPIx_SCS[n] Mode

Figure 37 illustrates the T2CDELAY, WDELAY and C2TDELAY delays in SPI 4-pin with SPIx_SCS[n] master mode. C2EDELAY and T2EDELAY are not available in this mode. All the three delay periods T2CDELAY, WDELAY, and C2TDELAY proceed to completion when enabled.

Figure 37. SPI 4-Pin with SPIx_SCS[n] Mode with T2CDELAY, WDELAY, and C2TDELAY



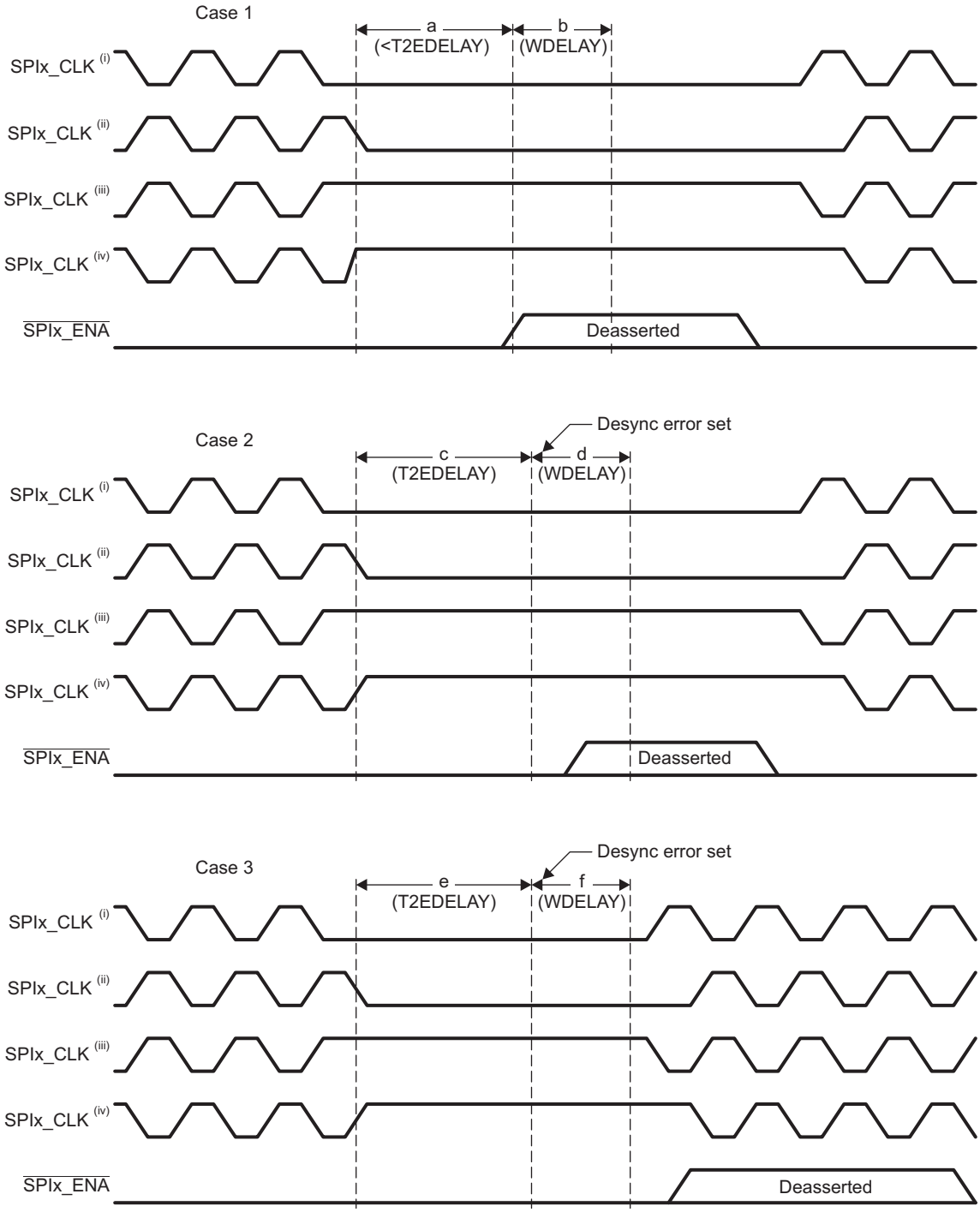
A.3 SPI 4-Pin with SPIx_ENA Mode

Figure 38 shows the T2EDELAY and WDELAY delays in SPI 4-pin with SPIx_ENA master mode. T2CDELAY, C2TDELAY, and C2EDELAY are not available in this mode.

- In CASE1, the SPIx_ENA is deasserted during the T2EDELAY period. Consequently the T2EDELAY period is terminated early (a) and the WDELAY period begins immediately (b) if enabled. The next transfer is initiated as soon as the slave asserts SPIx_ENA again.
- In CASE2, the T2EDELAY period (c) completes before the SPIx_ENA is deasserted. As a result the DESYNC error is set. However since the SPIx_ENA is deasserted during the WDELAY period (d), the master delays the next transfer until the SPIx_ENA is asserted again.
- In CASE3, the T2EDELAY (e) and WDELAY (f) period (if enabled) both expire before the SPIx_ENA input is deasserted. The DESYNC error is set at the end of the T2EDELAY period (e). However in this case the master begins the next transfer immediately after it is initiated and ignores the SPIx_ENA during the transfer even if it is subsequently deasserted.

  If the T2EDELAY delay period is disabled then the DESYNC error is not set. The SPI master behavior in this case depends on whether the SPIx_ENA gets deasserted during the WDELAY period (CASE2) or SPIx_ENA gets deasserted after the WDELAY period completes (CASE3).

Figure 38. SPI 4-Pin with SPIx_ENA Mode Demonstrating T2EDELAY and WDELAY



A.4 SPI 5-Pin Mode

Figure 39 shows the T2CDELAY, T2EDELAY, and WDELAY delays in SPI 5-pin master mode.

- In CASE1, the SPIx_ENA is deasserted during the T2CDELAY period. However the T2CDELAY period proceeds to completion(a), the T2EDELAY period is skipped (if enabled) and the WDELAY period begins immediately (b) (if enabled). The next transfer is initiated as soon as the slave asserts SPIx_ENA again.
- In CASE2, the SPIx_ENA signal is deasserted by the slave during the T2EDELAY period (d) which begins upon the completion of the T2CDELAY period (c). The deassertion of the SPIx_ENA causes the T2EDELAY period to terminate early and the WDELAY period (e) begins immediately (if enabled) after the T2EDELAY period terminates. The next transfer is initiated as soon as the slave asserts SPIx_ENA again.
- In CASE3, the SPIx_ENA signal is deasserted by the slave during the WDELAY period (h) which begins upon the completion of the T2CDELAY period (f) and T2EDELAY period (g). As a result the DESYNC error is set at the end of the T2EDELAY period (g). However since the SPIx_ENA is deasserted during the WDELAY period (h), the master delays the next transfer until the SPIx_ENA is asserted again.
- In CASE4, the SPIx_ENA signal is not deasserted until after the completion of the T2CDELAY (j), T2EDELAY (k) and WDELAY (m) (if enabled) periods. The DESYNC error is set at the end of the T2EDELAY period (k). However in this case the master begins the next transfer immediately after it is initiated and ignores the SPIx_ENA during the transfer even if it is subsequently deasserted.

  If the T2EDELAY delay period is disabled then the DESYNC error is not set. The SPI master behavior in this case depends on whether the SPIx_ENA gets deasserted during the T2CDELAY period (CASE1), WDELAY period (CASE3) or after the WDELAY period completes (CASE4).

  If the slave deasserts the SPIx_ENA signal before the completion of the configured master delays (T2CDELAY, T2EDELAY, WDELAY) then the master delays the next transfer until the slave asserts the SPIx_ENA again. However if the slave delays the SPIx_ENA deassertion until after the completion of the configured master delays then the master begins the next transfer immediately after it is initiated and ignores the SPIx_ENA during the transfer even if it is subsequently deasserted.

Figure 40 shows the C2TDELAY and C2EDELAY in SPI 5-pin master mode.

- In CASE1, the SPIx_ENA signal is asserted during the C2TDELAY period (a). However the C2TDELAY period proceeds to completion(a), the C2EDELAY period is skipped (if enabled) and the master begins generating the SPI clock for transmission.
- In CASE2, the SPIx_ENA signal is asserted during the C2EDELAY period (d) which begins upon the completion of C2TDELAY period (c). The assertion of the SPIx_ENA causes the C2EDELAY period to terminate early and the master begins generating the SPI clock for transmission.
- In CASE3, the SPIx_ENA signal is not asserted until after the completion of the C2TDELAY (f) and C2EDELAY (g) periods. The TIMEOUT error is set at the end of the C2EDELAY period (g). The master deasserts the SPIx_SCS[n] signal immediately and clears the current transmit request.

  If the C2EDELAY delay period is disabled then the SPI master behavior depends on whether the SPIx_ENA gets asserted during the C2TDELAY period (CASE1) or after the C2TDELAY period completes (CASE2). In latter case there is no limit on how long the master will wait for the slave to respond with SPIx_ENA asserted and hence there is no limit on period 'd' shown in CASE2. Thus when C2EDELAY period is disabled the TIMEOUT error is not set.

Figure 39. SPI 5-Pin Mode Demonstrating T2CDELAY, T2EDELAY, and WDELAY

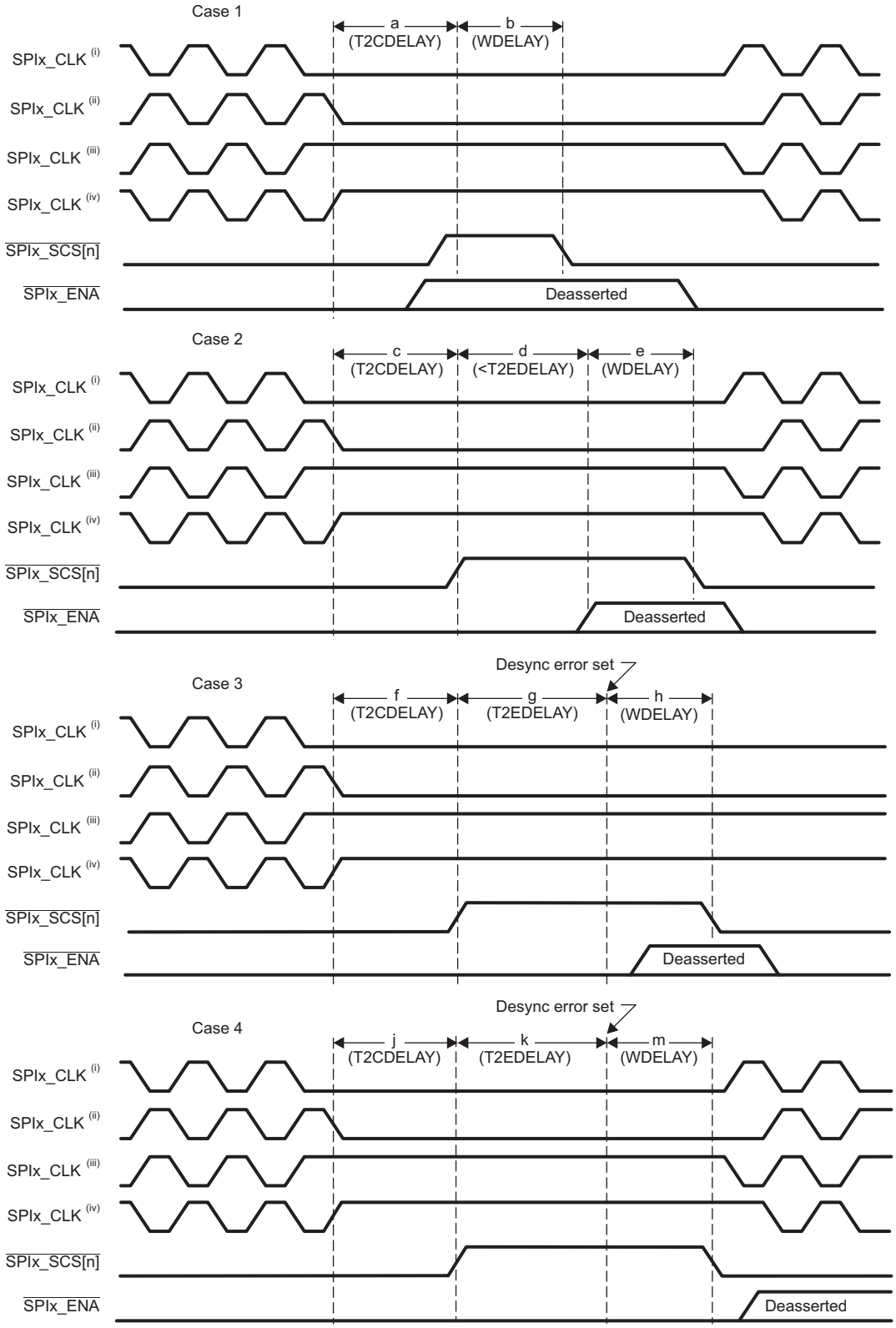
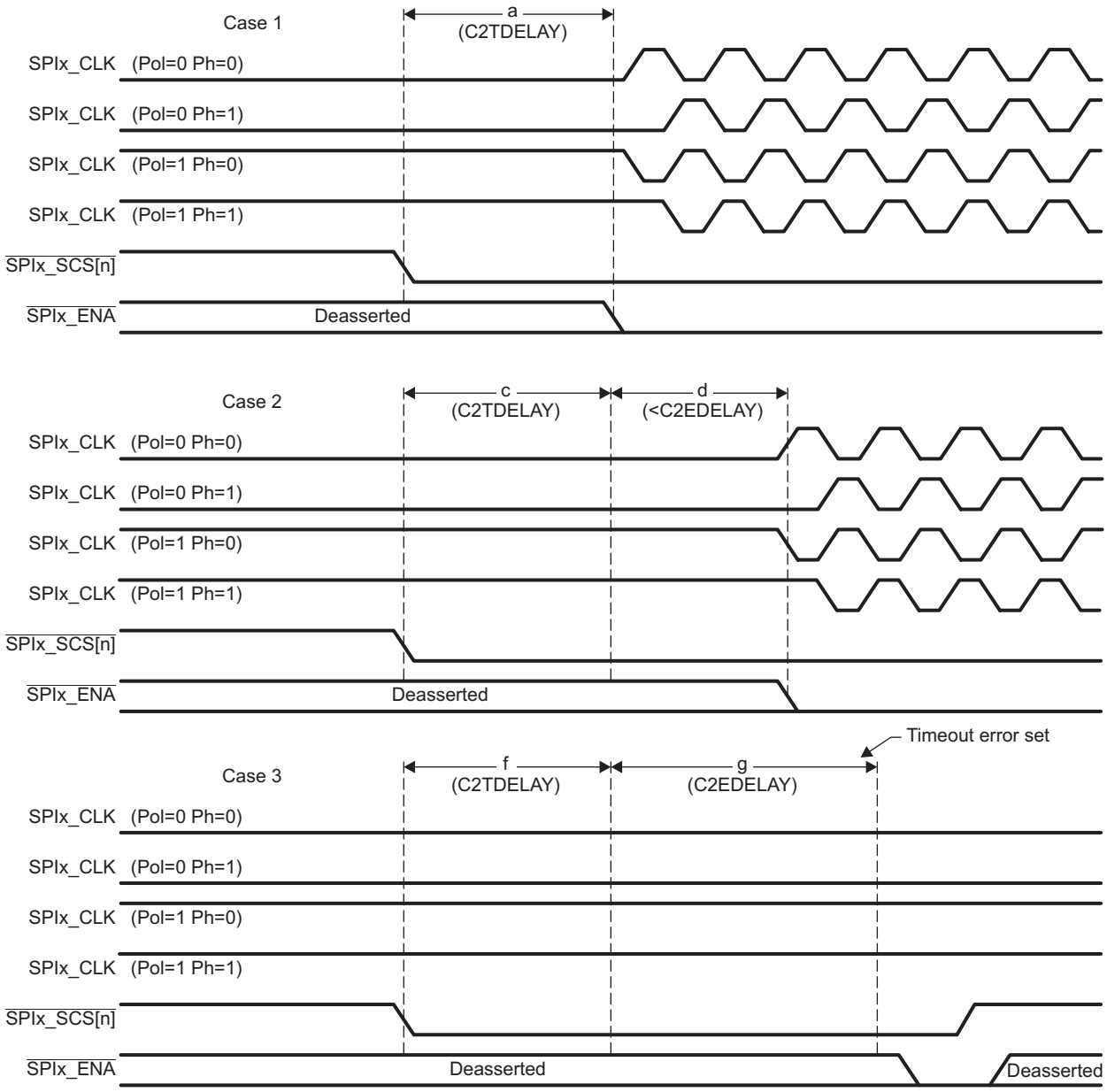


Figure 40. SPI 5-Pin Mode Demonstrating C2TDELAY and C2EDELAY



Appendix B Revision History

Table 28 lists the changes made since the previous version of this document.

Table 28. Document Revision History

Reference	Additions/Modifications/Deletions
Table 11	Changed Description of OVRNINTENA bit.
Table 12	Changed Description of OVRNINTLVL bit.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Audio	www.ti.com/audio	Communications and Telecom	www.ti.com/communications
Amplifiers	amplifier.ti.com	Computers and Peripherals	www.ti.com/computers
Data Converters	dataconverter.ti.com	Consumer Electronics	www.ti.com/consumer-apps
DLP® Products	www.dlp.com	Energy and Lighting	www.ti.com/energy
DSP	dsp.ti.com	Industrial	www.ti.com/industrial
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Security	www.ti.com/security
Logic	logic.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Power Mgmt	power.ti.com	Transportation and Automotive	www.ti.com/automotive
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com	Wireless	www.ti.com/wireless-apps
RF/IF and ZigBee® Solutions	www.ti.com/lprf		
	TI E2E Community Home Page		e2e.ti.com

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265