Serial Peripheral Interface (SPI)

This chapter describes the serial peripheral interface (SPI) which is a high-speed synchronous serial input and output (I/O) port that allows a serial bit stream of programmed length (one to 16 bits) to be shifted into and out of the device at a programmed bit-transfer rate. The SPI is normally used for communications between the MCU controller and external peripherals or another controller. Typical applications include external I/O or peripheral expansion via devices such as shift registers, display drivers, and analog-to-digital converters (ADCs). Multi-device communications are supported by the master or slave operation of the SPI. The port supports a 16-level, receive and transmit FIFO for reducing CPU servicing overhead.

Topic **Page**

18.1 Introduction

18.1.1 Features

The SPI module features include:

- SPISOMI: SPI slave-output/master-input pin
- SPISIMO: SPI slave-input/master-output pin
- $\overline{\text{SPISTE}}$: SPI slave transmit-enable pin
- SPICLK: SPI serial-clock pin

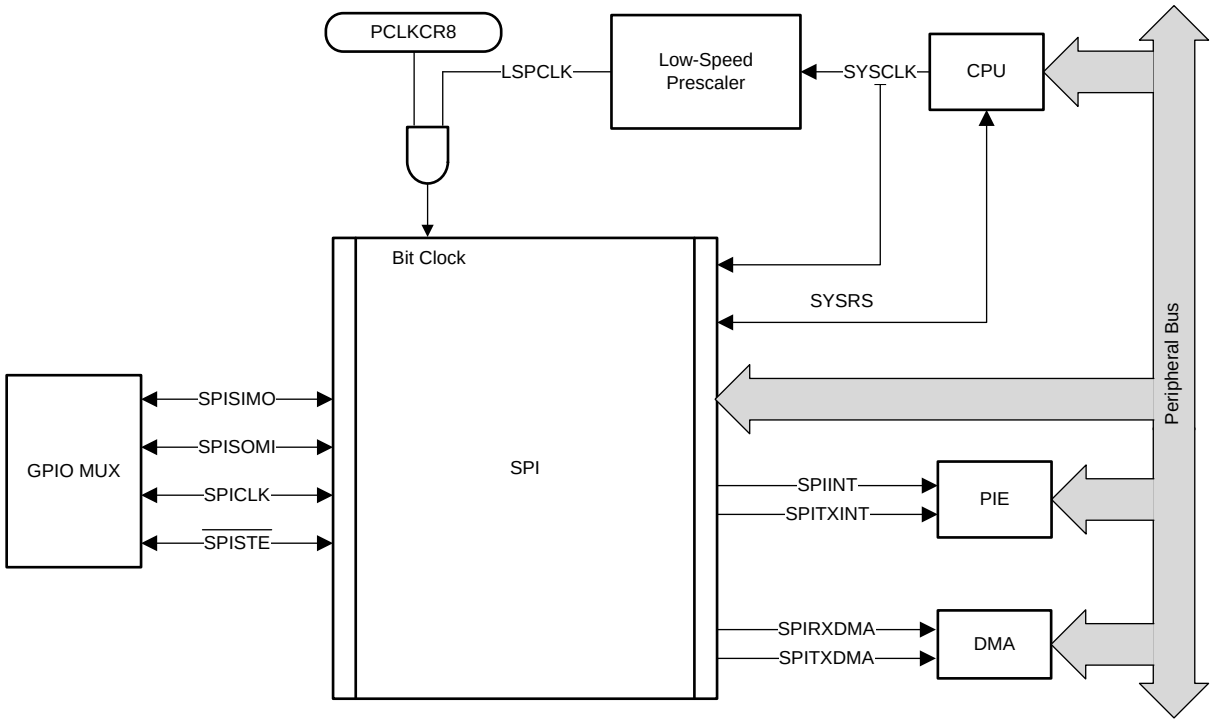
NOTE: All four pins can be used as GPIO if the SPI module is not used.

- Two operational modes: Master and Slave
- Baud rate: 125 different programmable rates. The maximum baud rate that can be employed is limited by the maximum speed of the I/O buffers used on the SPI pins. See the device-specific data manual for more details.
- Data word length: one to sixteen data bits
- Four clocking schemes (controlled by clock polarity and clock phase bits) include:
  - Falling edge without phase delay: SPICLK active-high. SPI transmits data on the falling edge of the SPICLK signal and receives data on the rising edge of the SPICLK signal.
  - Falling edge with phase delay: SPICLK active-high. SPI transmits data one half-cycle ahead of the falling edge of the SPICLK signal and receives data on the falling edge of the SPICLK signal.
  - Rising edge without phase delay: SPICLK inactive-low. SPI transmits data on the rising edge of the SPICLK signal and receives data on the falling edge of the SPICLK signal.
  - Rising edge with phase delay: SPICLK inactive-low. SPI transmits data one half-cycle ahead of the rising edge of the SPICLK signal and receives data on the rising edge of the SPICLK signal.
- Simultaneous receive and transmit operation (transmit function can be disabled in software)
- Transmitter and receiver operations are accomplished through either interrupt- driven or polled algorithm
- 16-level transmit/receive FIFO
- DMA support
- High-speed mode
- Delayed transmit control
- 3-wire SPI mode
- $\overline{\text{SPISTE}}$ inversion for digital audio interface receive mode on devices with two SPI modules

18.1.2 Block Diagram

Figure 18-1 shows the SPI CPU interfaces.

Figure 18-1. SPI CPU Interface



18.2 System-Level Integration

This section describes the various functionality that is applicable to the device integration. These features require configuration of other modules in the device that are not within the scope of this chapter.

18.2.1 SPI Module Signals

Table 18-1 classifies and provides a summary of the SPI module signals.

Table 18-1. SPI Module Signal Summary

Signal Name	Description
External Signals	
SPICLK	SPI clock
SPISIMO	SPI slave in, master out
SPISOMI	SPI slave out, master in
$\overline{\text{SPISTE}}$	SPI slave transmit enable
Control	
SPI Clock Rate	LSPCLK
Interrupt Signals	
SPIINT/SPIRXINT	Transmit interrupt/ Receive Interrupt in non FIFO mode (referred to as SPIINT)
	Receive interrupt in FIFO mode
SPITXINT	Transmit interrupt in FIFO mode
DMA Triggers	
SPITXDMA	Transmit request to DMA
SPIRXDMA	Receive request to DMA

Special Considerations

The $\overline{\text{SPISTE}}$ signal provides the ability to gate any spurious clock and data pulses when the SPI is in slave mode. An active $\overline{\text{SPISTE}}$ will not allow the slave to receive data. This prevents the SPI slave from losing synchronization with the master. It is this reason that TI does not recommend that the $\overline{\text{SPISTE}}$ always be tied to the active state.

If the SPI slave does ever lose synchronization with the master, toggling SPISWRESET will reset internal bit counter as well as the various status flags in the module. By resetting the bit counter, the SPI will interpret the next clock transition as the first bit of a new transmission. The register bit fields which are reset by SPISWRESET can be found in Section 18.5

Configuring a GPIO to emulate $\overline{\text{SPISTE}}$

In many systems, a SPI master may be connected to multiple SPI slaves using multiple instances of $\overline{\text{SPISTE}}$. Though this SPI module does not natively support multiple $\overline{\text{SPISTE}}$ signals, it is possible to emulate this behavior in software using GPIOs. In this configuration, the SPI must be configured as the master. Rather than using the GPIO Mux to select $\overline{\text{SPISTE}}$, the application would configure pins to be GPIO outputs, one GPIO per SPI slave. Before transmitting any data, the application would drive the desired GPIO to the active state. Immediately after the transmission has been completed, the GPIO chip select would be driven to the inactive state. This process can be repeated for many slaves which share the SPICLK, SPISIMO, and SPISOMI lines.

18.2.2 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification should be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pullups can be configured in the GPyPUD register.

See the *GPIO* chapter for more details on GPIO mux and settings.

18.2.2.1 GPIOs Required for High-Speed Mode

The high-speed mode of the SPI is available on the specified GPIO mux options in the device datasheet. To enable the high-speed enhancements, set SPICCR.HS_MODE to 1. Ensure that the capacitive loading on the pin does not exceed the value stated in the device Data Manual.

When not operating in high-speed mode, or if the capacitive loading on the pins exceed the value stated in the device Data Manual, SPICCR.HS_MODE should be set to 0.

18.2.3 SPI Interrupts

This section includes information on the available interrupts present in the SPI module.

The SPI module contains two interrupt lines: SPIINT/SPIRXINT and SPITXINT. When the SPI is operating in non-FIFO mode, all available interrupts are routed together to generate the single SPIINT interrupt. When FIFO mode is used, both SPIRXINT and SPITXINT can be generated.

SPIINT/SPIRXINT

When the SPI is operating in non-FIFO mode, the interrupt generated is called SPIINT. If FIFO enhancements are enabled, the interrupt is called SPIRXINT. These interrupts share the same interrupt vector in the Peripheral Interrupt Expansion (PIE) block.

In non-FIFO mode, two conditions can trigger an interrupt: a transmission is complete (INT_FLAG), or there is overrun in the receiver (OVERRUN_FLAG). Both of these conditions share the same interrupt vector: SPIINT.

The transmission complete flag (INT_FLAG) indicates that the SPI has completed sending or receiving the last bit and is ready to be serviced. At the same time this bit is set, the received character is placed in the receiver buffer (SPIRXBUF). The INT_FLAG will generate an interrupt on the SPIINT vector if the SPIINTENA bit is set.

previous transmission(s) that have been shifted to the left (shown in Example 18-1).

Example 18-1. Transmission of Bit From SPIRXBUF

Conditions:

1. Transmission character length = 1 bit (specified in bits SPICHAR)
2. The current value of SPIDAT = 737Bh

SPIDAT (before transmission)																
	0	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1
SPIDAT (after transmission)																
(TXed) 0 ←	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	x[(1)] ← (RXed)
SPIRXBUF (after transmission)																
	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	x[(1)]

(1) x = 1 if SPISOMI data is high; x = 0 if SPISOMI data is low; master mode is assumed.

18.3.5 Baud Rate Selection

The SPI module supports 125 different baud rates and four different clock schemes. Depending on whether the SPI clock is in slave or master mode, the SPICLK pin can receive an external SPI clock signal or provide the SPI clock signal, respectively.

- In the slave mode, the SPI clock is received on the SPICLK pin from the external source, and can be no greater than the LSPCLK frequency divided by 4.
- In the master mode, the SPI clock is generated by the SPI and is output on the SPICLK pin, and can be no greater than the LSPCLK frequency divided by 4.

NOTE: The baud rate should be configured to not exceed the maximum rated GPIO toggle frequency. Refer to the device Data Manual for the maximum GPIO toggle frequency

Example 18-2 shows how to determine the SPI baud rates.

Example 18-2. Baud Rate Determination

For SPIBRR = 3 to 127:

$$\text{SPI Baud Rate} = \frac{\text{LSPCLK}}{(\text{SPIBRR} + 1)} \quad (6)$$

For SPIBRR = 0, 1, or 2:

$$\text{SPI Baud Rate} = \frac{\text{LSPCLK}}{4} \quad (7)$$

where:

LSPCLK = Low-speed peripheral clock frequency of the device

SPIBRR = Contents of the SPIBRR in the master SPI device

To determine what value to load into SPIBRR, you must know the device system clock (LSPCLK) frequency (which is device-specific) and the baud rate at which you will be operating.

The following example shows how to calculate the baud rate of the SPI module in standard SPI mode (HS_MODE=0).

Example 18-3. Baud Rate Calculation in Non-High Speed Mode (HS_MODE=0)

$$SPI\ Baud\ Rate = \frac{LSPCLK}{SPIBRR + 1'} \quad LSPCLK = 50\ MHz$$
$$= \frac{50 \times 10^6}{3 + 1}$$
$$= 12.5\ Mbps \tag{8}$$

18.3.6 SPI Clocking Schemes

The clock polarity select bit (CLKPOLARITY) and the clock phase select bit (CLK_PHASE) control four different clocking schemes on the SPICLK pin. CLKPOLARITY selects the active edge, either rising or falling, of the clock. CLK_PHASE selects a half-cycle delay of the clock. The four different clocking schemes are as follows:

- Falling Edge Without Delay. The SPI transmits data on the falling edge of the SPICLK and receives data on the rising edge of the SPICLK.
- Falling Edge With Delay. The SPI transmits data one half-cycle ahead of the falling edge of the SPICLK signal and receives data on the falling edge of the SPICLK signal.
- Rising Edge Without Delay. The SPI transmits data on the rising edge of the SPICLK signal and receives data on the falling edge of the SPICLK signal.
- Rising Edge With Delay. The SPI transmits data one half-cycle ahead of the rising edge of the SPICLK signal and receives data on the rising edge of the SPICLK signal.

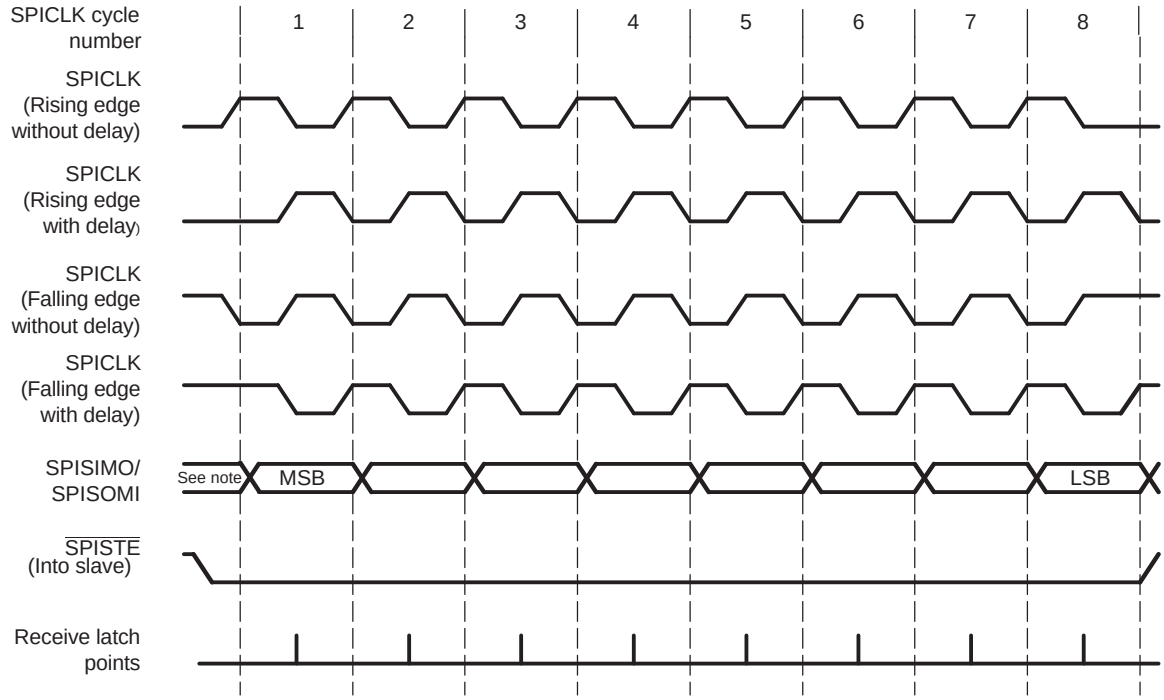
The selection procedure for the SPI clocking scheme is shown in Table 18-3. Examples of these four clocking schemes relative to transmitted and received data are shown in Figure 18-7.

Table 18-3. SPI Clocking Scheme Selection Guide

SPICLK Scheme	CLKPOLARITY	CLK_PHASE [1]
Rising edge without delay	0	0
Rising edge with delay	0	1
Falling edge without delay	1	0
Falling edge with delay	1	1

(1) The description of CLK_PHASE and CLKPOLARITY differs between manufacturers. For proper operation, select the desired waveform to determine the clock phase and clock polarity settings.

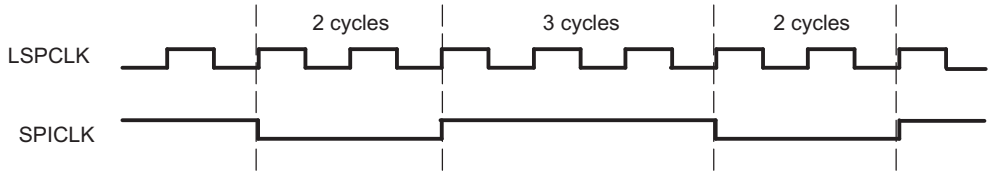
Figure 18-7. SPICLK Signal Options



Note: Previous data bit

SPICLK symmetry is retained only when the result of (SPIBRR+1) is an even value. When (SPIBRR + 1) is an odd value and SPIBRR is greater than 3, SPICLK becomes asymmetrical. The low pulse of SPICLK is one LSPCLK cycle longer than the high pulse when CLKPOLARITY bit is clear (0). When CLKPOLARITY bit is set to 1, the high pulse of the SPICLK is one LSPCLK cycle longer than the low pulse, as shown in Figure 18-8.

Figure 18-8. SPI: SPICLK-LSPCLK Characteristic When (BRR + 1) is Odd, BRR > 3, and CLKPOLARITY = 1



18.3.7 SPI FIFO Description

The following steps explain the FIFO features and help with programming the SPI FIFOs:

1. **Reset.** At reset the SPI powers up in standard SPI mode and the FIFO function is disabled. The FIFO registers SPIFFTX, SPIFFRX and SPIFFCT remain inactive.
2. **Standard SPI.** The standard 28x SPI mode will work with SPIINT/SPIRXINT as the interrupt source.
3. **Mode change.** FIFO mode is enabled by setting the SPIFFENA bit to 1 in the SPIFFTX register. SPIRST can reset the FIFO mode at any stage of its operation.
4. **Active registers.** All the SPI registers and SPI FIFO registers SPIFFTX, SPIFFRX, and SPIFFCT will be active.
5. **Interrupts.** FIFO mode has two interrupts one for the transmit FIFO, SPITXINT and one for the receive FIFO, SPIRXINT. SPIRXINT is the common interrupt for SPI FIFO receive, receive error and receive FIFO overflow conditions. The single SPIINT for both transmit and receive sections of the standard SPI will be disabled and this interrupt will service as SPI receive FIFO interrupt. For more information, refer to Section 18.2.3

18.4 Programming Procedure

This section describes the procedure for configuring the SPI for the various modes of operation.

18.4.1 Initialization Upon Reset

A system reset forces the SPI peripheral into the following default configuration:

- Unit is configured as a slave module (MASTER_SLAVE = 0)
- Transmit capability is disabled (TALK = 0)
- Data is latched at the input on the falling edge of the SPICLK signal
- Character length is assumed to be one bit
- SPI interrupts are disabled
- Data in SPIDAT is reset to 0000h

18.4.2 Configuring the SPI

This section describes the procedure in which to configure the SPI module for operation. To prevent unwanted and unforeseen events from occurring during or as a result of initialization changes, clear the SPISWRESET bit before making initialization changes, and then set this bit after initialization is complete. While the SPI is held in reset (SPISWRESET = 0), configuration may be changed in any order. The following list shows the SPI configuration procedure in a logical order. However, the SPI registers can be written with single 16-bit writes, so the order is not required with the exception of SPISWRESET.

To change the SPI configuration:

Step 1. Clear the SPI Software Reset bit (SPISWRESET) to 0 to force the SPI to the reset state.

Step 2. Configure the SPI as desired:

- Select either master or slave mode (MASTER_SLAVE).
- Choose SPICLK polarity and phase (CLKPOLARITY and CLK_PHASE).
- Set the desired baud rate (SPIBRR).
- Enable high speed mode if desired (HS_MODE).
- Set the SPI character length (SPICHAR).
- Clear the SPI Flags (OVERRUN_FLAG, INT_FLAG).
- Enable $\overline{\text{SPISTE}}$ inversion (STEINV) if needed.
- Enable 3-wire mode (TRIWIRE) if needed.
- If using FIFO enhancements:
  - Enable the FIFO enhancements (SPIFFENA).
  - Clear the FIFO Flags (TXFFINTCLR, RXFFOVFCLR, and RXFFINTCLR).
  - Release transmit and receive FIFO resets (TXFIFO and RXFIFORESET).
  - Release SPI FIFO channels from reset (SPIRST).

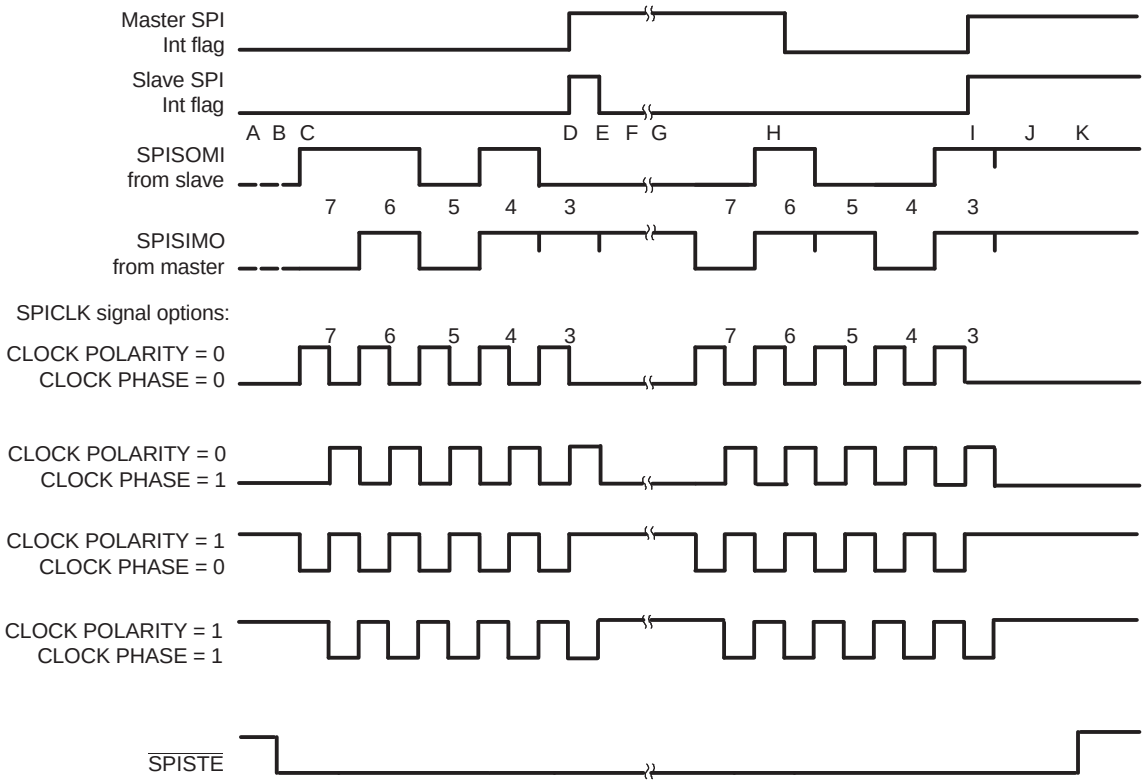
Step 3. If interrupts are used:

- In non-FIFO mode, enable the receiver overrun and/or SPI interrupts (OVERRUNINTENA and SPIINTENA).
- In FIFO mode, set the transmit and receive interrupt levels (TXFFIL and RXFFIL) then enable the interrupts (TXFFIENA and RXFFIENA).

Step 4. Set SPISWRESET to 1 to release the SPI from the reset state.

NOTE: Do not change the SPI configuration when communication is in progress.

Figure 18-11. Five Bits per Character



A Slave writes 0D0h to SPIDAT and waits for the master to shift out the data.

B Master sets the slave $\overline{\text{SPISTE}}$ signal low (active).

C Master writes 058h to SPIDAT, which starts the transmission procedure.

D First byte is finished and sets the interrupt flags.

E Slave reads 0Bh from its SPIRXBUF (right-justified).

F Slave writes 04Ch to SPIDAT and waits for the master to shift out the data.

G Master writes 06Ch to SPIDAT, which starts the transmission procedure.

H Master reads 01Ah from the SPIRXBUF (right−justified).

I Second byte is finished and sets the interrupt flags.

J Master reads 89h and the slave reads 8Dh from their respective SPIRXBUF. After the user's software masks off the unused bits, the master receives 09h and the slave receives 0Dh.

K Master clears the slave $\overline{\text{SPISTE}}$ signal high (inactive).

18.4.5 SPI 3-Wire Mode Code Examples

In addition to the normal SPI initialization, to configure the SPI module for 3-wire mode, the TRIWIRE bit (SPIPRI.0) must be set to 1. After initialization, there are several considerations to take into account when transmitting and receiving data in 3-wire master and slave mode. The following examples demonstrate these considerations.

In 3-wire master mode, SPICLKx, SPISTEx, and SPISIMOx pins must be configured as SPI pins (SPISOMIx pin can be configured as non-SPI pin). When the master transmits, it receives the data it transmits (because SPISIMOx and SPISOMIx are connected internally in 3-wire mode). Therefore, the junk data received must be cleared from the receive buffer every time data is transmitted.

Example 18-4. 3-Wire Master Mode Transmit

```
Uint16 data;
Uint16 dummy;
```

18.5 SPI Registers

This section describes the Serial Peripheral Interface registers. It is important to note that the SPI registers only allow 16-bit accesses.

18.5.1 SPI Base Addresses

Table 18-6. SPI Base Address Table

Device Registers	Register Name	Start Address	End Address
SpiaRegs	SPI_REGS	0x0000_6100	0x0000_610F
SpibRegs	SPI_REGS	0x0000_6110	0x0000_611F
SpicRegs	SPI_REGS	0x0000_6120	0x0000_612F

18.5.2 SPI_REGS Registers

Table 18-7 lists the SPI_REGS registers. All register offset addresses not listed in Table 18-7 should be considered as reserved locations and the register contents should not be modified.

Table 18-7. SPI_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SPICCR	SPI Configuration Control Register		Go
1h	SPICTL	SPI Operation Control Register		Go
2h	SPISTS	SPI Status Register		Go
4h	SPIBRR	SPI Baud Rate Register		Go
6h	SPIRXEMU	SPI Emulation Buffer Register		Go
7h	SPIRXBUF	SPI Serial Input Buffer Register		Go
8h	SPITXBUF	SPI Serial Output Buffer Register		Go
9h	SPIDAT	SPI Serial Data Register		Go
Ah	SPIFFTX	SPI FIFO Transmit Register		Go
Bh	SPIFFRX	SPI FIFO Receive Register		Go
Ch	SPIFFCT	SPI FIFO Control Register		Go
Fh	SPIPRI	SPI Priority Control Register		Go

Complex bit access types are encoded to fit into small table cells. Table 18-8 shows the codes that are used for access types in this section.

Table 18-8. SPI_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
RC	R C	Read to Clear
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

18.5.2.1 SPICCR Register (Offset = 0h) [reset = 0h]

SPICCR is shown in Figure 18-14 and described in Table 18-9.

Return to the Summary Table.

SPICCR controls the setup of the SPI for operation.

Figure 18-14. SPICCR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SPISWRESET	CLKPOLARITY	HS_MODE	SPILBK	SPICHAR			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			

Table 18-9. SPICCR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	SPISWRESET	R/W	0h	SPI Software Reset When changing configuration, you should clear this bit before the changes and set this bit before resuming operation. Reset type: SYSRSn 0h (R/W) = Initializes the SPI operating flags to the reset condition. Specifically, the RECEIVER OVERRUN Flag bit (SPISTS.7), the SPI INT FLAG bit (SPISTS.6), and the TXBUF FULL Flag bit (SPISTS.5) are cleared. SPISTE will become inactive. SPICLK will be immediately driven to 0 regardless of the clock polarity. The SPI configuration remains unchanged. 1h (R/W) = SPI is ready to transmit or receive the next character. When the SPI SW RESET bit is a 0, a character written to the transmitter will not be shifted out when this bit is set. A new character must be written to the serial data register. SPICLK will be returned to its inactive state one SPICLK cycle after this bit is set.
6	CLKPOLARITY	R/W	0h	Shift Clock Polarity This bit controls the polarity of the SPICLK signal. CLOCK POLARITY and POLARITY CLOCK PHASE (SPICTL.3) control four clocking schemes on the SPICLK pin. Reset type: SYSRSn 0h (R/W) = Data is output on rising edge and input on falling edge. When no SPI data is sent, SPICLK is at low level. The data input and output edges depend on the value of the CLOCK PHASE bit (SPICTL.3) as follows: - CLOCK PHASE = 0: Data is output on the rising edge of the SPICLK signal. Input data is latched on the falling edge of the SPICLK signal. - CLOCK PHASE = 1: Data is output one half-cycle before the first rising edge of the SPICLK signal and on subsequent falling edges of the SPICLK signal. Input data is latched on the rising edge of the SPICLK signal. 1h (R/W) = Data is output on falling edge and input on rising edge. When no SPI data is sent, SPICLK is at high level. The data input and output edges depend on the value of the CLOCK PHASE bit (SPICTL.3) as follows: - CLOCK PHASE = 0: Data is output on the falling edge of the SPICLK signal. Input data is latched on the rising edge of the SPICLK signal. - CLOCK PHASE = 1: Data is output one half-cycle before the first falling edge of the SPICLK signal and on subsequent rising edges of the SPICLK signal. Input data is latched on the falling edge of the SPICLK signal.

Table 18-9. SPICCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	HS_MODE	R/W	0h	High Speed Mode Enable Bits This bit determines if the High Speed mode is enabled. The correct GPIOs should be selected in the GPxGMUX/GPxMUX registers. Reset type: SYSRSn 0h (R/W) = SPI High Speed mode disabled. This is the default value after reset. 1h (R/W) = SPI High Speed mode enabled,
4	SPILBK	R/W	0h	SPI Loopback Mode Select Loopback mode allows module validation during device testing. This mode is valid only in master mode of the SPI. Reset type: SYSRSn 0h (R/W) = SPI loopback mode disabled. This is the default value after reset. 1h (R/W) = SPI loopback mode enabled, SIMO/SOMI lines are connected internally. Used for module self-tests.
3-0	SPICHAR	R/W	0h	Character Length Control Bits These four bits determine the number of bits to be shifted in or SPI CHAR0 out as a single character during one shift sequence. SPICHAR = Word length - 1 Reset type: SYSRSn 0h (R/W) = 1-bit word 1h (R/W) = 2-bit word 7h (R/W) = 8-bit word Fh (R/W) = 16-bit word

18.5.2.2 SPICTL Register (Offset = 1h) [reset = 0h]

SPICTL is shown in Figure 18-15 and described in Table 18-10.

Return to the Summary Table.

SPICTL controls data transmission, the SPI's ability to generate interrupts, the SPICLK phase, and the operational mode (slave or master).

Figure 18-15. SPICTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			OVERRUNINTENA	CLK_PHASE	MASTER_SLAVE	TALK	SPIINTENA
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 18-10. SPICTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	OVERRUNINTENA	R/W	0h	Overrun Interrupt Enable Overrun Interrupt Enable. Setting this bit causes an interrupt to be generated when the RECEIVER OVERRUN Flag bit (SPISTS.7) is set by hardware. Interrupts generated by the RECEIVER OVERRUN Flag bit and the SPI INT FLAG bit (SPISTS.6) share the same interrupt vector. Reset type: SYSRSn 0h (R/W) = Disable RECEIVER OVERRUN interrupts. 1h (R/W) = Enable RECEIVER_OVERRUN interrupts.
3	CLK_PHASE	R/W	0h	SPI Clock Phase Select This bit controls the phase of the SPICLK signal. CLOCK PHASE and CLOCK POLARITY (SPICCR.6) make four different clocking schemes possible (see clocking figures in SPI chapter). When operating with CLOCK PHASE high, the SPI (master or slave) makes the first bit of data available after SPIDAT is written and before the first edge of the SPICLK signal, regardless of which SPI mode is being used. Reset type: SYSRSn 0h (R/W) = Normal SPI clocking scheme, depending on the CLOCK POLARITY bit (SPICCR.6). 1h (R/W) = SPICLK signal delayed by one half-cycle. Polarity determined by the CLOCK POLARITY bit.
2	MASTER_SLAVE	R/W	0h	SPI Network Mode Control This bit determines whether the SPI is a network master or slave. SLAVE During reset initialization, the SPI is automatically configured as a network slave. Reset type: SYSRSn 0h (R/W) = SPI is configured as a slave. 1h (R/W) = SPI is configured as a master.

Table 18-10. SPICTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	TALK	R/W	0h	Transmit Enable The TALK bit can disable data transmission (master or slave) by placing the serial data output in the high-impedance state. If this bit is disabled during a transmission, the transmit shift register continues to operate until the previous character is shifted out. When the TALK bit is disabled, the SPI is still able to receive characters and update the status flags. TALK is cleared (disabled) by a system reset. Reset type: SYSRSn 0h (R/W) = Disables transmission: - Slave mode operation: If not previously configured as a general-purpose I/O pin, the SPISOMI pin will be put in the high-impedance state. - Master mode operation: If not previously configured as a general-purpose I/O pin, the SPISIMO pin will be put in the high-impedance state. 1h (R/W) = Enables transmission For the 4-pin option, ensure to enable the receiver's SPISTEn input pin.
0	SPIINTENA	R/W	0h	SPI Interrupt Enable This bit controls the SPI's ability to generate a transmit/receive interrupt. The SPI INT FLAG bit (SPISTS.6) is unaffected by this bit. Reset type: SYSRSn 0h (R/W) = Disables the interrupt. 1h (R/W) = Enables the interrupt.

18.5.2.3 SPISTS Register (Offset = 2h) [reset = 0h]

SPISTS is shown in Figure 18-16 and described in Table 18-11.

Return to the Summary Table.

SPISTS contrains interrupt and status bits.

Figure 18-16. SPISTS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OVERRUN_FLAG	INT_FLAG	BUFFULL_FLAG	RESERVED				
W1C-0h	RC-0h	R-0h	R-0h				

Table 18-11. SPISTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	OVERRUN_FLAG	W1C	0h	SPI Receiver Overrun Flag This bit is a read/clear-only flag. The SPI hardware sets this bit when a receive or transmit operation completes before the previous character has been read from the buffer. The bit is cleared in one of three ways: - Writing a 1 to this bit - Writing a 0 to SPI SW RESET (SPICCR.7) - Resetting the system If the OVERRUN INT ENA bit (SPICTL.4) is set, the SPI requests only one interrupt upon the first occurrence of setting the RECEIVER OVERRUN Flag bit. Subsequent overruns will not request additional interrupts if this flag bit is already set. This means that in order to allow new overrun interrupt requests the user must clear this flag bit by writing a 1 to SPISTS.7 each time an overrun condition occurs. In other words, if the RECEIVER OVERRUN Flag bit is left set (not cleared) by the interrupt service routine, another overrun interrupt will not be immediately re-entered when the interrupt service routine is exited. Reset type: SYSRSn 0h (R/W) = A receive overrun condition has not occurred. 1h (R/W) = The last received character has been overwritten and therefore lost (when the SPIRXBUF was overwritten by the SPI module before the previous character was read by the user application). Writing a '1' will clear this bit. The RECEIVER OVERRUN Flag bit should be cleared during the interrupt service routine because the RECEIVER OVERRUN Flag bit and SPI INT FLAG bit (SPISTS.6) share the same interrupt vector. This will alleviate any possible doubt as to the source of the interrupt when the next byte is received.

Table 18-11. SPISTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	INT_FLAG	RC	0h	SPI Interrupt Flag SPI INT FLAG is a read-only flag. Hardware sets this bit to indicate that the SPI has completed sending or receiving the last bit and is ready to be serviced. This flag causes an interrupt to be requested if the SPI INT ENA bit (SPICTL.0) is set. The received character is placed in the receiver buffer at the same time this bit is set. This bit is cleared in one of three ways: - Reading SPIRXBUF - Writing a 0 to SPI SW RESET (SPICCR.7) - Resetting the system Note: This bit should not be used if FIFO mode is enabled. The internal process of copying the received word from SPIRXBUF to the Receive FIFO will clear this bit. Use the FIFO status, or FIFO interrupt bits for similar functionality. Reset type: SYSRSn 0h (R/W) = No full words have been received or transmitted. 1h (R/W) = Indicates that the SPI has completed sending or receiving the last bit and is ready to be serviced.
5	BUFFULL_FLAG	R	0h	SPI Transmit Buffer Full Flag This read-only bit gets set to 1 when a character is written to the SPI Transmit buffer SPITXBUF. It is cleared when the character is automatically loaded into SPIDAT when the shifting out of a previous character is complete. Reset type: SYSRSn 0h (R/W) = Transmit buffer is not full. 1h (R/W) = Transmit buffer is full.
4-0	RESERVED	R	0h	Reserved

18.5.2.4 SPIBRR Register (Offset = 4h) [reset = 0h]

SPIBRR is shown in Figure 18-17 and described in Table 18-12.

Return to the Summary Table.

SPIBRR contains the bits used for baud-rate selection.

Figure 18-17. SPIBRR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	SPI_BIT_RATE						
R-0h	R/W-0h						

Table 18-12. SPIBRR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6-0	SPI_BIT_RATE	R/W	0h	SPI Baud Rate Control These bits determine the bit transfer rate if the SPI is the network SPI BIT RATE 0 master. There are 125 data-transfer rates (each a function of the CPU clock, LSPCLK) that can be selected. One data bit is shifted per SPICLK cycle. (SPICLK is the baud rate clock output on the SPICLK pin.) If the SPI is a network slave, the module receives a clock on the SPICLK pin from the network master. Therefore, these bits have no effect on the SPICLK signal. The frequency of the input clock from the master should not exceed the slave SPI's LSPCLK signal divided by 4. In master mode, the SPI clock is generated by the SPI and is output on the SPICLK pin. The SPI baud rates are determined by the following formula: For SPIBRR = 3 to 127: SPI Baud Rate = LSPCLK / (SPIBRR + 1) For SPIBRR = 0, 1, or 2: SPI Baud Rate = LSPCLK / 4 Reset type: SYSRSn 3h (R/W) = SPI Baud Rate = LSPCLK/4 4h (R/W) = SPI Baud Rate = LSPCLK/5 7Eh (R/W) = SPI Baud Rate = LSPCLK/127 7Fh (R/W) = SPI Baud Rate = LSPCLK/128

18.5.2.6 SPIRXBUF Register (Offset = 7h) [reset = 0h]

SPIRXBUF is shown in Figure 18-19 and described in Table 18-14.

Return to the Summary Table.

SPIRXBUF contains the received data. Reading SPIRXBUF clears the SPI INT FLAG bit in SPISTS. If FIFO mode is enabled, reading this register will also decrement the RXFFST counter in SPIFFRX.

Figure 18-19. SPIRXBUF Register

15	14	13	12	11	10	9	8
RXBn							
R-0h							
7	6	5	4	3	2	1	0
RXBn							
R-0h							

Table 18-14. SPIRXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RXBn	R	0h	Received Data Once SPIDAT has received the complete character, the character is transferred to SPIRXBUF, where it can be read. At the same time, the SPI INT FLAG bit (SPISTS.6) is set. Since data is shifted into the SPI's most significant bit first, it is stored right-justified in this register. Reset type: SYSRSn

18.5.2.7 SPITXBUF Register (Offset = 8h) [reset = 0h]

SPITXBUF is shown in Figure 18-20 and described in Table 18-15.

Return to the Summary Table.

SPITXBUF stores the next character to be tranmitted. Writing to this register sets the TX BUF FULL Flag bit in SPISTS. When the transmission of the current character is complete, the contents of this register are automatically loaded in SPIDAT and the TX BUF FULL Flag is cleared. If no tranmission is currently active, data written to this register falls through into the SPIDAT register and the TX BUF FULL Flag is not set.

In master mode, if no tranmission is currently active, writing to this register initiates a transmission in the same manner that writing to SPIDAT does.

Figure 18-20. SPITXBUF Register

15	14	13	12	11	10	9	8
TXBn							
R/W-0h							
7	6	5	4	3	2	1	0
TXBn							
R/W-0h							

Table 18-15. SPITXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TXBn	R/W	0h	Transmit Data Buffer This is where the next character to be transmitted is stored. When the transmission of the current character has completed, if the TX BUF FULL Flag bit is set, the contents of this register is automatically transferred to SPIDAT, and the TX BUF FULL Flag is cleared. Writes to SPITXBUF must be left-justified. Reset type: SYSRSn

18.5.2.9 SPIFFTX Register (Offset = Ah) [reset = A000h]

SPIFFTX is shown in Figure 18-22 and described in Table 18-17.

Return to the Summary Table.

SPIFFTX contains both control and status bits related to the output FIFO buffer. This includes FIFO reset control, FIFO interrupt level control, FIFO level status, as well as FIFO interrupt enable and clear bits.

Figure 18-22. SPIFFTX Register

15	14	13	12	11	10	9	8
SPIRST	SPIFFENA	TXFIFO	TXFFST				
R/W-1h	R/W-0h	R/W-1h	R-0h				
7	6	5	4	3	2	1	0
TXFFINT	TXFFINTCLR	TXFFIENA	TXFFIL				
R-0h	W-0h	R/W-0h	R/W-0h				

Table 18-17. SPIFFTX Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SPIRST	R/W	1h	SPI Reset Reset type: SYSRSn 0h (R/W) = Write 0 to reset the SPI transmit and receive channels. The SPI FIFO register configuration bits will be left as is. 1h (R/W) = SPI FIFO can resume transmit or receive. No effect to the SPI registers bits.
14	SPIFFENA	R/W	0h	SPI FIFO Enhancements Enable Reset type: SYSRSn 0h (R/W) = SPI FIFO enhancements are disabled. 1h (R/W) = SPI FIFO enhancements are enabled.
13	TXFIFO	R/W	1h	TX FIFO Reset Reset type: SYSRSn 0h (R/W) = Write 0 to reset the FIFO pointer to zero, and hold in reset. 1h (R/W) = Release transmit FIFO from reset.
12-8	TXFFST	R	0h	Transmit FIFO Status Reset type: SYSRSn 0h (R/W) = Transmit FIFO is empty. 1h (R/W) = Transmit FIFO has 1 word. 2h (R/W) = Transmit FIFO has 2 words. 10h (R/W) = Transmit FIFO has 16 words, which is the maximum. 1Fh (R/W) = Reserved.
7	TXFFINT	R	0h	TX FIFO Interrupt Flag Reset type: SYSRSn 0h (R/W) = TXFIFO interrupt has not occurred, This is a read-only bit. 1h (R/W) = TXFIFO interrupt has occurred, This is a read-only bit.
6	TXFFINTCLR	W	0h	TXFIFO Interrupt Clear Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on TXFIFINT flag bit, Bit reads back a zero. 1h (R/W) = Write 1 to clear SPIFFTX[TXFFINT] flag.
5	TXFFIENA	R/W	0h	TX FIFO Interrupt Enable Reset type: SYSRSn 0h (R/W) = TX FIFO interrupt based on TXFFIL match (less than or equal to) will be disabled. 1h (R/W) = TX FIFO interrupt based on TXFFIL match (less than or equal to) will be enabled.

Table 18-17. SPIFFTX Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	TXFFIL	R/W	0h	Transmit FIFO Interrupt Level Bits Transmit FIFO will generate interrupt when the FIFO status bits (TXFFST4-0) and FIFO level bits (TXFFIL4-0) match (less than or equal to). Reset type: SYSRSn 0h (R/W) = A TX FIFO interrupt request is generated when there are no words remaining in the TX buffer. 1h (R/W) = A TX FIFO interrupt request is generated when there is 1 word or no words remaining in the TX buffer. 2h (R/W) = A TX FIFO interrupt request is generated when there is 2 words or fewer remaining in the TX buffer. 10h (R/W) = A TX FIFO interrupt request is generated when there are 16 words or fewer remaining in the TX buffer. 1Fh (R/W) = Reserved.

18.5.2.10 SPIFFRX Register (Offset = Bh) [reset = 201Fh]

SPIFFRX is shown in Figure 18-23 and described in Table 18-18.

Return to the Summary Table.

SPIFFRX contains both control and status bits related to the input FIFO buffer. This includes FIFO reset control, FIFO interrupt level control, FIFO level status, as well as FIFO interrupt enable and clear bits.

Figure 18-23. SPIFFRX Register

15	14	13	12	11	10	9	8
RXFFOVF	RXFFOVFCLR	RXFIFORESET	RXFFST				
R-0h	W-0h	R/W-1h	R-0h				
7	6	5	4	3	2	1	0
RXFFINT	RXFFINTCLR	RXFFIENA	RXFFIL				
R-0h	W-0h	R/W-0h	R/W-1Fh				

Table 18-18. SPIFFRX Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RXFFOVF	R	0h	Receive FIFO Overflow Flag Reset type: SYSRSn 0h (R/W) = Receive FIFO has not overflowed. This is a read-only bit. 1h (R/W) = Receive FIFO has overflowed, read-only bit. More than 16 words have been received in to the FIFO, and the first received word is lost.
14	RXFFOVFCLR	W	0h	Receive FIFO Overflow Clear Reset type: SYSRSn 0h (R/W) = Write 0 does not affect RXFFOVF flag bit, Bit reads back a zero. 1h (R/W) = Write 1 to clear SPIFFRX[RXFFOVF].
13	RXFIFORESET	R/W	1h	Receive FIFO Reset Reset type: SYSRSn 0h (R/W) = Write 0 to reset the FIFO pointer to zero, and hold in reset. 1h (R/W) = Re-enable receive FIFO operation.
12-8	RXFFST	R	0h	Receive FIFO Status Reset type: SYSRSn 0h (R/W) = Receive FIFO is empty. 1h (R/W) = Receive FIFO has 1 word. 2h (R/W) = Receive FIFO has 2 words. 10h (R/W) = Receive FIFO has 16 words, which is the maximum. 1Fh (R/W) = Reserved.
7	RXFFINT	R	0h	Receive FIFO Interrupt Flag Reset type: SYSRSn 0h (R/W) = RXFIFO interrupt has not occurred. This is a read-only bit. 1h (R/W) = RXFIFO interrupt has occurred. This is a read-only bit.
6	RXFFINTCLR	W	0h	Receive FIFO Interrupt Clear Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on RXFIFINT flag bit, Bit reads back a zero. 1h (R/W) = Write 1 to clear SPIFFRX[RXFFINT] flag

Table 18-18. SPIFFRX Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	RXFFIENA	R/W	0h	RX FIFO Interrupt Enable Reset type: SYSRSn 0h (R/W) = RX FIFO interrupt based on RXFFIL match (greater than or equal to) will be disabled. 1h (R/W) = RX FIFO interrupt based on RXFFIL match (greater than or equal to) will be enabled.
4-0	RXFFIL	R/W	1Fh	Receive FIFO Interrupt Level Bits Receive FIFO generates an interrupt when the FIFO status bits (RXFFST4-0) are greater than or equal to the FIFO level bits (RXFFIL4-0). The default value of these bits after reset is 11111. This avoids frequent interrupts after reset, as the receive FIFO will be empty most of the time. Reset type: SYSRSn 0h (R/W) = A RX FIFO interrupt request is generated when there is 0 or more words in the RX buffer. 1h (R/W) = A RX FIFO interrupt request is generated when there are 1 or more words in the RX buffer. 2h (R/W) = A RX FIFO interrupt request is generated when there are 2 or more words in the RX buffer. 10h (R/W) = A RX FIFO interrupt request is generated when there are 16 words in the RX buffer. 1Fh (R/W) = Reserved.

18.5.2.11 SPIFFCT Register (Offset = Ch) [reset = 0h]

SPIFFCT is shown in Figure 18-24 and described in Table 18-19.

Return to the Summary Table.

SPIFFCT controls the FIFO transmit delay bits.

Figure 18-24. SPIFFCT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TXDLY							
R/W-0h							

Table 18-19. SPIFFCT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	TXDLY	R/W	0h	FIFO Transmit Delay Bits These bits define the delay between every transfer from FIFO transmit buffer to transmit shift register. The delay is defined in number SPI serial clock cycles. The 8-bit register could define a minimum delay of 0 serial clock cycles and a maximum of 255 serial clock cycles. In FIFO mode, the buffer (TXBUF) between the shift register and the FIFO should be filled only after the shift register has completed shifting of the last bit. This is required to pass on the delay between transfers to the data stream. In the FIFO mode TXBUF should not be treated as one additional level of buffer. Reset type: SYSRSn 0h (R/W) = The next word in the TX FIFO buffer is transferred to SPITXBUF immediately upon completion of transmission of the previous word. 1h (R/W) = The next word in the TX FIFO buffer is transferred to SPITXBUF1 serial clock cycle after completion of transmission of the previous word. 2h (R/W) = The next word in the TX FIFO buffer is transferred to SPITXBUF 2 serial clock cycles after completion of transmission of the previous word. FFh (R/W) = The next word in the TX FIFO buffer is transferred to SPITXBUF 255 serial clock cycles after completion of transmission of the previous word.

18.5.2.12 SPIPRI Register (Offset = Fh) [reset = 0h]

SPIPRI is shown in Figure 18-25 and described in Table 18-20.

Return to the Summary Table.

SPIPRI controls auxillary functions for the SPI including emulation control, SPISTE inversion, and 3-wire control.

Figure 18-25. SPIPRI Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	SOFT	FREE	RESERVED		STEINV	TRIWIRE
R-0h		R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h

Table 18-20. SPIPRI Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	SOFT	R/W	0h	Emulation Soft Run This bit only has an effect when the FREE bit is 0. Reset type: SYSRSn 0h (R/W) = Transmission stops midway in the bit stream while TSUSPEND is asserted. Once TSUSPEND is deasserted without a system reset, the remainder of the bits pending in the DATBUF are shifted. Example: If SPIDAT has shifted 3 out of 8 bits, the communication freezes right there. However, if TSUSPEND is later deasserted without resetting the SPI, SPI starts transmitting from where it had stopped (fourth bit in this case) and will transmit 8 bits from that point. 1h (R/W) = If the emulation suspend occurs before the start of a transmission, (that is, before the first SPICLK pulse) then the transmission will not occur. If the emulation suspend occurs after the start of a transmission, then the data will be shifted out to completion. When the start of transmission occurs is dependent on the baud rate used. Standard SPI mode: Stop after transmitting the words in the shift register and buffer. That is, after TXBUF and SPIDAT are empty. In FIFO mode: Stop after transmitting the words in the shift register and buffer. That is, after TX FIFO and SPIDAT are empty.
4	FREE	R/W	0h	Emulation Free Run These bits determine what occurs when an emulation suspend occurs (for example, when the debugger hits a breakpoint). The peripheral can continue whatever it is doing (free-run mode) or, if in stop mode, it can either stop immediately or stop when the current operation (the current receive/transmit sequence) is complete. Reset type: SYSRSn 0h (R/W) = Emulation mode is selected by the SOFT bit 1h (R/W) = Free run, continue SPI operation regardless of suspend or when the suspend occurred.
3-2	RESERVED	R	0h	Reserved

Table 18-20. SPIPRI Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	STEINV	R/W	0h	SPISTEn Inversion Bit On devices with 2 SPI modules, inverting the SPISTE signal on one of the modules allows the device to receive left and right- channel digital audio data. This bit is only applicable to slave mode. Writing to this bit while configured as master (MASTER_SLAVE = 1) has no effect Reset type: SYSRSn 0h (R/W) = SPISTEn is active low (normal) 1h (R/W) = SPISTE is active high (inverted)
0	TRIWIRE	R/W	0h	SPI 3-wire Mode Enable Reset type: SYSRSn 0h (R/W) = Normal 4-wire SPI mode. 1h (R/W) = 3-wire SPI mode enabled. The unused pin becomes a GPIO pin. In master mode, the SPISIMO pin becomes the SPIMOMI (master receive and transmit) pin and SPISOMI is free for non-SPI use. In slave mode, the SPISOMI pin becomes the SPISISO (slave receive and transmit) pin and SPISIMO is free for non-SPI use.