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 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.

Resale of TI products or services with statements different form 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.

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

<table>
<thead>
<tr>
<th>Products</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amplifiers</td>
<td>Audio</td>
</tr>
<tr>
<td>Data Converters</td>
<td>Automotive</td>
</tr>
<tr>
<td>DSP</td>
<td>Broadband</td>
</tr>
<tr>
<td>Interface</td>
<td>Digital Control</td>
</tr>
<tr>
<td>Logic</td>
<td>Military</td>
</tr>
<tr>
<td>Power Mgmt</td>
<td>Optical Networking</td>
</tr>
<tr>
<td>Microcontrollers</td>
<td>Security</td>
</tr>
<tr>
<td></td>
<td>Telephony</td>
</tr>
<tr>
<td></td>
<td>Video &amp; Imaging</td>
</tr>
<tr>
<td></td>
<td>Wireless</td>
</tr>
</tbody>
</table>

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

Copyright © 2005, Texas Instruments Incorporated
<table>
<thead>
<tr>
<th>REVISION</th>
<th>DATE</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>8/05</td>
<td>Page 16, Baud Rate Limitations section added</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Page 24, information on slave prescale baud rate added</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Page 27, note on clearing SPIBUF added</td>
</tr>
</tbody>
</table>
Contents

1 Overview .................................................. 2
2 SPI Operation Modes ........................................ 3
  2.1 SPI Internal Registers .................................... 4
  2.2 SPI Operation; Three-Pin Option ......................... 5
  2.3 SPI Operation; Four-Pin Option ........................ 7
  2.4 SPI Operation; Five-Pin Option (Hardware Handshaking) 9
  2.5 Data Format .............................................. 11
  2.6 Clocking Modes .......................................... 12
  2.7 Data Transfer Example .................................... 15
  2.8 Baud Rate Limitations .................................... 16
3 General Purpose I/O ........................................... 17
4 Low Power Mode ............................................... 18
5 DMA Interface ................................................. 19
6 Control Registers ............................................ 20
  6.1 SPI Control Register 1 (SPICTRL1) ..................... 24
  6.2 SPI Control Register 2 (SPICTRL2) ..................... 26
  6.3 SPI Control Register 3 (SPICTRL3) ..................... 28
  6.4 SPI Shift Register 0 (SPIDAT0) ........................ 30
  6.5 SPI Shift Register 1 (SPIDAT1) ........................ 31
  6.6 SPI Buffer Register (SPIBUF) .......................... 32
  6.7 SPI Emulation Register (SPIEMU) ..................... 34
  6.8 SPI Pin Control Register 1 (SPIPC1) ................. 35
  6.9 SPI Pin Control Register 2 (SPIPC2) ................. 37
  6.10 SPI Pin Control Register 3 (SPIPC3) ................. 39
  6.11 SPI Pin Control Register 4 (SPIPC4) ................. 41
  6.12 SPI Pin Control Register 5 (SPIPC5) ................. 43
  6.13 SPI Pin Control Register 6 (SPIPC6) ................. 45
Figures

1. SPI Module Block Diagram (Five Pin Mode Shown) ........................................... 3
2. SPI Three Pin Option .......................................................................................... 5
3. SPI Four Pin Option with SPISCS ....................................................................... 7
4. SPI Four Pin Option with SPIENA ................................................................. 8
5. SPI Five-Pin Option with SPIENA and SPISCS ........................................... 10
6. Clock Mode with POLARITY = 0 and PHASE = 0 ......................................... 13
7. Clock Mode with POLARITY = 0 and PHASE = 1 ......................................... 13
8. Clock Mode with POLARITY = 1 and PHASE = 0 .......................................... 14
9. Clock Mode with POLARITY = 1 and PHASE = 1 .......................................... 14
10. Five Bits per Character (5-Pin Option) ......................................................... 15
Serial Peripheral Interface (SPI)

This reference guide provides the specifications for a 16-bit configurable synchronous serial peripheral interface (SPI). The SPI is in effect a programmable-length shift register used for high speed communication between external peripherals or other microcontrollers.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Overview</td>
<td>2</td>
</tr>
<tr>
<td>2 SPI Operation Modes</td>
<td>3</td>
</tr>
<tr>
<td>3 General Purpose I/O</td>
<td>17</td>
</tr>
<tr>
<td>4 Low Power Mode</td>
<td>18</td>
</tr>
<tr>
<td>5 DMA Interface</td>
<td>19</td>
</tr>
<tr>
<td>6 Control Registers</td>
<td>20</td>
</tr>
</tbody>
</table>
Overview

1 Overview

The SPI is a high-speed synchronous serial input/output port that allows a serial bit stream of programmed length (3 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 microcontroller and external peripherals or another microcontroller. Typical applications include interface to external I/O or peripheral expansion via devices such as shift registers, display drivers, and analog-to-digital converters.

The SPI is available with three, four or five pins. The pins SPICLK, SPISIMO and SPISOMI are used in all SPI pin modes. The pins SPIENA and SPISCS are optional and may be used if the pin are present on a given device.

The SPI has the following attributes:

- 16-bit shift register
- Receive buffer register
- 8-bit baud clock generator
- Serial clock (SPICLK) I/O pin
- Slave in, master out (SPISIMO) I/O pin
- Slave out, master in (SPISOMI) I/O pin
- SPI enable (SPIENA) I/O pin (4-or 5-pin mode only)
- Slave chip select (SPISCS) I/O pin (4- or 5-pin mode only)

The SPI allows software to program the following options:

- SPISOMI/SPISIMO pin direction configuration
- SPICLK pin source (external/internal)
- SPICLK frequency (interface clock [ICLK] /2 through /256)
- SPI pins as functional or digital I/O pins
- Character length (3 to 16 bits)
- Phase (delay/no delay)
- Polarity (high or low).

**Note: Maximum Input Frequency**

The maximum input frequency on the SPICLK pin when in slave mode is the ICLK frequency /2.
2 SPI Operation Modes

The SPI operates in a master or slave mode. The MASTER bit (SPICTRL2.3) selects the configuration of the SPISIMO and SPISOMI pins and the CLKMOD bit (SPICTRL2.5) determines whether an internal or external clock source will be used.

The slave chip select (SPISCS) pin is used when communicating with multiple slave devices. When the master (SPI sending out the clock stream) writes to SPIDAT1, the SPISCS pin is automatically driven low to select the slave connected to that signal. Writing to SPIDAT0 will not drive SPISCS low, thus allowing the master to communicate with all slave devices connected to the same SPI bus.

In addition, a handshaking mechanism, provided by the SPIENA pin, enables the slave to delay the generation of the clock signal supplied by the master as long as it is not prepared for the next exchange of data.

Figure 1. SPI Module Block Diagram (Five Pin Mode Shown)
2.1 SPI Internal Registers

A general representation of the SPI internal registers is shown in Table 1. The page column provides a cross reference to additional information on the individual registers. For more information regarding individual bytes, see Table , on page 20.

<table>
<thead>
<tr>
<th>Offset Address†</th>
<th>Mnemonic</th>
<th>Name</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>SPICRTL1</td>
<td>SPI Control Register 1</td>
<td>Sets transfer rate and character length</td>
<td>24</td>
</tr>
<tr>
<td>0x04</td>
<td>SPICRTL2</td>
<td>SPI Control Register 2</td>
<td>Controls SPI clock</td>
<td>26</td>
</tr>
<tr>
<td>0x08</td>
<td>SPICRTL3</td>
<td>SPI Control Register 3</td>
<td>Controls system interface</td>
<td>28</td>
</tr>
<tr>
<td>0x0C</td>
<td>SPIDAT0</td>
<td>SPI Shift Register 0</td>
<td>Main shift register</td>
<td>30</td>
</tr>
<tr>
<td>0x10</td>
<td>SPIDAT1</td>
<td>SPI Shift Register 1</td>
<td>Shift register used in automatic slave chip select mode only</td>
<td>31</td>
</tr>
<tr>
<td>0x14</td>
<td>SPIBUF</td>
<td>SPI Buffer Register</td>
<td>Holds received word</td>
<td>32</td>
</tr>
<tr>
<td>0x18</td>
<td>SPIEMU</td>
<td>SPI Emulation Register</td>
<td>Mirror of SPIBUF. Read does not clear flags</td>
<td>34</td>
</tr>
<tr>
<td>0x1C</td>
<td>SPIPC1</td>
<td>SPI Pin Control Register 1</td>
<td>Controls the direction of data on the I/O pins</td>
<td>35</td>
</tr>
<tr>
<td>0x20</td>
<td>SPIPC2</td>
<td>SPI Pin Control Register 2</td>
<td>Reflects the values on the I/O pins</td>
<td>37</td>
</tr>
<tr>
<td>0x24</td>
<td>SPIPC3</td>
<td>SPI Pin Control Register 3</td>
<td>Controls the values sent to the I/O pins</td>
<td>39</td>
</tr>
<tr>
<td>0x28</td>
<td>SPIPC4</td>
<td>SPI Pin Control Register 4</td>
<td>Sets data values in the SPIPC3 register</td>
<td>41</td>
</tr>
<tr>
<td>0x2C</td>
<td>SPIPC5</td>
<td>SPI Pin Control Register 5</td>
<td>Clears values in the SPIPC3 register</td>
<td>43</td>
</tr>
<tr>
<td>0x30</td>
<td>SPIPC6</td>
<td>SPI Pin Control Register 6</td>
<td>Determines if pins will operate as general I/O or SPI functional pin.</td>
<td>45</td>
</tr>
</tbody>
</table>

† The actual address of these registers is device specific and CPU specific. See the specific device data sheet to verify the SPI register addresses.
2.2 SPI Operation; Three-Pin Option

In master mode configuration (MASTER = 1 (SPICTRL2.3) and CLKMOD = 1 (SPICTRL2.5)), the SPI provides the serial clock on the SPICLK pin for the entire serial communications network. Data is output on the SPISIMO pin and latched in from the SPISOMI pin (see Figure 2).

![Figure 2: SPI Three Pin Option](image)

Data written to the shift register (SPIDAT0) initiates data transmission on the SPISIMO pin, most significant bit (MSB) first. Simultaneously, received data is shifted through the SPISOMI pin into the least significant bit (LSB) of the SPIDAT0 register. When the selected number of bits has been transmitted, the data is transferred to the SPIBUF register for the CPU to read. Data is stored right-justified in SPIBUF.

When the specified number of bits has been shifted through the SPIDAT0 register, the following events occur:

- The RXINTFLAG bit (SPICTRL3.0) is set to 1
- The SPIDAT0 register contents transfer to the SPIBUF register
- An interrupt is asserted if the RXINTEN bit (SPICTRL3.1) is set to 1

In slave mode configuration (MASTER = 0 and CLKMOD = 0), data shifts out on the SPISOMI pin and in on the SPISIMO pin. The SPICLK pin is used as
the input for the serial shift clock, which is supplied from the external network master. The transfer rate is defined by this clock.

Data written to the SPIDAT0 register is transmitted to the network when the SPICLK signal is received from the network master. To receive data, the SPI waits for the network master to send the SPICLK signal and then shifts data on the SPISIMO pin into the SPIDAT0 register. If data is to be transmitted by the slave simultaneously, it must be written to the SPIDAT0 register before the beginning of the SPICLK signal.
2.3 SPI Operation; Four-Pin Option

The three-pin option and the four-pin options of the SPI are identical in the master mode (CLKMOD = 1), except that the four-pin option uses either SPIENA or SPISCS pin. The I/O direction of these pins is determined by the CLKMOD control bit as SPI not general purpose I/O.

**4-pin option with SPISCS**

To use the SPISCS as an automatic chip select pin, the SPISCS pin must be configured to be functional (SPIPC6.4 = 1). In this mode, the master will drive this signal low when data has been written to SPIDAT1 and then drive the pin high again after a character transmission has completed. If data is written to SPIDAT0, SPISCS remains high (see Figure 3).

Figure 3. **SPI Four Pin Option with SPISCS**

To use the SPISCS as a chip select, the slave SPISCS pin must be configured as SPI functional (SPIPC6.4 = 1). In this mode, an active low signal on the SPISCS pin will allow the slave SPI to transfer data to the serial data line. An inactive high signal will put the slave SPI’s serial output pin in a high-impedance state. Therefore many slave devices can be tied together on the network, but only one slave at a time is allowed to talk. While the slave is not selected, no shifting or interrupts will occur.
**4-pin option with SPIENA**

To use the SPIENA as a WAIT signal pin, the SPIENA pin must be configured to be functional (SPIPC6.0 = 1). In this mode, an active low signal on the SPIENA pin will allow the master SPI to drive the clock pulse stream; otherwise, the master will hold the clock signal.

To use the SPIENA as a WAIT signal pin, the slave SPIENA pin must be configured as functional (SPIPC6.0 = 1). If the SPIENA pin is in high-z mode (ENABLE_HIGHZ = 1), the slave will put SPIENA into the high-impedance once it receives a new character. If the SPIENA pin is in push-pull mode (ENABLE_HIGHZ = 0), the slave will drive SPIENA high once it receives a new character. The slave will drive SPIENA low again after new data is written to the slave shift register (SPIDAT0).

**Figure 4. SPI Four Pin Option with SPIENA**

![Diagram showing SPI four pin option](image-url)

Write to SPIDAT0 (SLAVE)

**Write to SPIDAT0 (Master)**
2.4 SPI Operation; Five-Pin Option (Hardware Handshaking)

To use the hardware handshaking mechanism, both the SPIENA pin and SPISCS pin must be configured as functional pins.

In the master SPI (CLKMOD = 1), if the SPIENA pin is configured as a functional input. If configured as a slave SPI, the SPIENA pin is configured as a functional output. If the SPIENA pin is in high-z mode (ENABLE_HIGHZ = 1), the slave SPI will put this signal into the high-impedance state if it receives a new character from the master or if the slave becomes de-selected by the master (SPISCS goes high). The slave will drive the signal low when new data is written to the slave shift register (SPIDAT0) and the slave has been selected by the master (SPISCS is low).

If the SPIENA pin is in push-pull mode (ENABLE_HIGHZ = 0), the slave will drive SPIENA high only if there is new data in the buffer register and the slave is selected by the master (SPISCS is low). The slave SPI will drive the SPIENA signal low when new data is written to the slave shift register (SPIDAT0) and the slave is selected by the master (SPISCS is low). If the slave is de-selected by the master (SPISCS goes high), the slave SPIENA signal is driven low, allowing the master SPI to communicate with other slave SPIs.

In the master SPI (CLKMOD = 1), the SPISCS pin is configured as a functional output. If configured as a slave SPI (CLKMOD = 0), the SPISCS pin is configured as a functional input. A write to the master’s SPIDAT1 shift register will automatically drive the SPISCS signal low. The master will drive the SPISCS signal high again after transmitting the new character. If the new data is written to the master’s SPIDAT0 shift register, the SPISCS signal will NOT be driven low.
**Figure 5.** SPI Five-Pin Option with SPIENA and SPISCS

Master
(Master = 1 ; CLKMOD = 1)

Slave
(Master = 0 ; CLKMOD = 0)

Write to SPIDAT1

Write to SPIDAT0

Write to SPIDAT1 (MASTER)

SPISCS

Write to SPIDAT0 (SLAVE)

SPIENA

SPICLK

SPISIMO

SPISOMI
2.5 Data Format

The data formats for the three, four and five pin options are identical.

CHARLEN[4:0] (SPICTRL1.4-0) specifies the number of bits (3 to 16) in the data word. The CHARLEN[4:0] value directs the state control logic to count the number of bits received or transmitted to determine when a complete word is processed.

The following conditions apply for words with fewer than 16 bits:
- Data must be left-justified when it is written to the SPI for transmission
- Data is right-justified when read back from the receive register

The buffer contains the most recently received word, right-justified, plus any bits that are left over from previous transmissions that have been shifted to the left. The diagram below shows how a 14-bit word is stored in the buffer once it is received.

<table>
<thead>
<tr>
<th>Bits</th>
<th>D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X X 1 0 1 0 1 0 1 0 1 0 1 0 1 0</td>
</tr>
</tbody>
</table>

In transmit mode, the SPIBUF register contains the most recently transmitted word, left-justified. The diagram below shows how a 14-bit word needs to be written to the buffer in order to be transmitted correctly.

<table>
<thead>
<tr>
<th>Bits</th>
<th>D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 0 1 0 1 0 1 0 1 0 1 0 1 0 X X</td>
</tr>
</tbody>
</table>

To allow for the efficient transmission of byte-sized words, if a character length is programmed for 8 bits or less, the SDPDAT[7] bit instead of SDPDAT[15] is the source of the serial out data. This prevents the need to further add 8 justification bits.
2.6 Clocking Modes

There are four clock modes in which SPICLK may operate, depending on the choice of the phase (delay/no delay) and the polarity (rising edge / falling edge) of the clock. When operating with PHASE active, the SPI makes the first bit of data available after the SPIDAT0 register is written and before the first edge of SPICLK. The data input and output edges depend on the values of both POLARITY and PHASE as shown in Table 2.

Table 2. Clocking Modes

<table>
<thead>
<tr>
<th>POLARITY</th>
<th>PHASE</th>
<th>ACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Data is output on the rising edge of SPICLK. Input data is latched on the falling edge.</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Data is output one half-cycle before the first rising edge of SPICLK and on subsequent falling edges. Input data is latched on the rising edge of SPICLK.</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Data is output on the falling edge of SPICLK. Input data is latched on the rising edge.</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Data is output one half-cycle before the first falling edge of SPICLK and on subsequent rising edges. Input data is latched on the falling edge of SPICLK.</td>
</tr>
</tbody>
</table>

Figure 6 to Figure 9 illustrate the four possible signals of SPICLK corresponding to each mode. Having four signal options allows the SPI to interface with different types of serial devices. Also shown are the SPICLK control bit polarity and phase values corresponding to each signal.
Figure 6. Clock Mode with POLARITY = 0 and PHASE = 0

Clock polarity = 0, Clock phase = 0

Write SPIDAT

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

Figure 7. Clock Mode with POLARITY = 0 and PHASE = 1

Clock polarity = 0, Clock phase = 1

Write SPIDAT

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

**Figure 8. Clock Mode with POLARITY = 1 and PHASE = 0**

Clock polarity = 1, Clock phase = 0

- Data is output on the falling edge of SPICLK
- Input data is latched on the rising edge of SPICLK
- A write to the SPIDAT register starts SPICLK

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

Clock polarity = 1, Clock phase = 1

- Data is output one-half cycle before the first falling edge of SPICLK and on the subsequent rising edges of SPICLK
- Input data is latched on the falling edge of SPICLK
2.7 Data Transfer Example

The following timing diagram illustrates an SPI data transfer between two devices using a character length of five bits.

*Figure 10. Five Bits per Character (5-Pin Option)*
2.8 Baud Rate Limitations

It is recommended to operate the master and slave SPIs at the same baud rate. However, when this is not possible the SPICLK ranges specified in Table 3 must be followed to ensure proper data transfer. The SPICLK rate is set by adjusting the PRESCALE value in the SPICTRL1 register.

Table 3. SPICLK Ranges

<table>
<thead>
<tr>
<th>POLARITY</th>
<th>PHASE</th>
<th>SPICLK RATIO</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>0</td>
<td>( \frac{\text{MasterSPICLK}}{2} \leq \text{SlaveSPICLK} \leq (\text{MasterSPICLK} + 1%) )</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td>( \frac{\text{MasterSPICLK}}{2} \leq \text{SlaveSPICLK} \leq (\text{MasterSPICLK} \times 2) )</td>
</tr>
</tbody>
</table>

In all clocking mode configurations, the slave SPICLK must never be less than half the speed of the master SPICLK. Doing so may allow the master to start a new SPI transmission before the slave is ready. When operating with PHASE = 0, the slave SPICLK must not be more than 1% faster than the master SPICLK. When operating with PHASE = 1 the slave SPICLK must not be more than two times faster than the master SPICLK. If the slave SPICLK exceeds the master SPICLK by more than 1% when PHASE = 0 or by 2x when PHASE = 1 there is a possibility that the slave will move data from the input shift register to SPIBUF before the master is finished transferring data.
3 General Purpose I/O

Each of the SPI pins may be programmed via the SPI Pin Control Registers (SPIPC1, SPIPC2, SPIPC3, SPIPC4, SPIPC5, SPIPC6) to be a general-purpose I/O pin.

When the SPI module is not used, the SPI pins may be programmed to be either general input or general output pins. The direction is controlled in the SPIPC1 register. Note that each pin can be programmed to be either a SPI pin or a GPIO pin through register SPIPC6.

If the SPI function is to be used, application software must ensure that each pin is configured as a SPI pin and not a GPIO pin, or else unexpected behavior may result.

**Note: Unused SPI Pins**

If there are four or five SPI pins available and only the three- or four-pin configuration is desired, the remaining pin(s) can be configured and used as general-purpose input/output (GIO) pins.
4 Low Power Mode

The SPI module has two means to be placed in a low-power mode: a global low-power mode from the system and a local low-power mode via the POWERDOWN bit (SPICTRL2.2). The net effect on the SPI is the same, independent of the source.

A low-power mode in effect shuts down all the clocks to the module. During a global low-power mode, no registers are visible to the software; nothing can be written to or read from any register. A local low-power mode has the same effect when both the local POWERDOWN bit and the system level PPWNOVR bit are set. If only the local POWERDOWN bit is set, then the SPI logic is not clocked, but the registers continue to be clocked.

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

If handling the SPI message traffic on a character-by-character basis requires too much CPU overhead and if the particular device is equipped with the DMA controller, the SPI may use the DMA controller to receive or transmit data directly to memory. The SPI module contains one DMA request enable bit (DMA REQ EN).

When a character is being transmitted or received, the SPI will signal the DMA via a DMA request signal. The DMA controller will then perform the needed data manipulation.

For DMA-based transmissions, all characters are assembled in RAM, and DMA transfers move the message, word-by-word, from RAM into the SPIDAT0 register. (See the DMA controller specification). Data is then read from SPIBUF, clearing RXINTFLAG (SPICTRL3.0).

For efficient behavior, during DMA operations, the receive interrupt enable flag RXINTEN (SPICTRL3.1) should be cleared to 0. For specific DMA features, refer to the DMA controller specification.
6 Control Registers

This section describes the SPI control, data and pin registers. The registers support 16-bit and 32-bit writes.
**Table 4. SPI Registers**

| Offset Address† | Register | 31  | 15 | 14  | 13  | 29  | 28  | 27  | 26  | 25  | 24  | 23  | 22  | 21  | 20  | 19  | 18  | 17  | 16  | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----------------|----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 0x00            | SPICTRL1 |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | Reserved |     |     |     |     |     |     |     |     |     |     |     |
|                 |          | Reserved |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | Reserved |     |     |     |     |     |     |     |     |     |     |     | CHARLEN.4:0 |
| 0x04            | SPICTRL2 |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | Reserved |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|                 |          | Reserved |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | Reserved |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 0x08            | SPICTRL3 |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | Reserved |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|                 |          | Reserved |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | Reserved |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 0x0C            | SPIDAT0  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | Reserved |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|                 |          |          |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | SPIDAT0.15:0 |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 0x10            | SPIDAT1  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | Reserved |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|                 |          |          |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     | SPIDAT1.15:0 |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |

†The address is offset from the SPI module base address.
| Offset Address† | Register | 31 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 0x14          | SPIBUF   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|               |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|               |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 0x18          | SPIEMU   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|               |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|               |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 0x1C          | SPIPC1   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|               |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|               |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 0x20          | SPIPC2   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|               |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|               |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 0x24          | SPIPC3   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|               |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|               |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| 0x28          | SPIPC4   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|               |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|               |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
### Table 4. SPI Registers (Continued)

<table>
<thead>
<tr>
<th>Offset Address†</th>
<th>Register</th>
<th>31</th>
<th>30</th>
<th>29</th>
<th>28</th>
<th>27</th>
<th>26</th>
<th>25</th>
<th>24</th>
<th>23</th>
<th>22</th>
<th>21</th>
<th>20</th>
<th>19</th>
<th>18</th>
<th>17</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x2C SPIPC5</td>
<td>0x0</td>
<td>Reserved</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x0</td>
<td>Reserved</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x0</td>
<td>Reserved</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x0</td>
<td>Reserved</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

† The actual addresses of these registers are device specific. See the specific device data sheet to verify the SPI register addresses.
‡ The SPIBUF is a 32 bit register. Two bits in the upper 16 bits are used for control, all 16 lower bits are data buffers.
Control Registers

6.1 SPI Control Register 1 (SPICRTL1)

<table>
<thead>
<tr>
<th>Bits</th>
<th>31</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>Reserved</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bits</th>
<th>15</th>
<th>13</th>
<th>12</th>
<th>5</th>
<th>4</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
<td>PRESCALE</td>
<td>CHARLEN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R = Read, W = Write, U = Undefined; = Value after reset

**Bits 31:13**  
Reserved.

Reads are undefined and writes have no effect.

**Bits 12:5**  
PRESCALE  
Determines the bit transfer rate if the SPI is the network master. There are 255 data transfer rates (each a function of the interface clock) that can be selected. One data bit is shifted per SPICLK cycle.

SPI Baud Rate for PRESCALE = 1 to 255

\[
SPI_{BaudRate} = \frac{ICLK}{(PRESCALE + 1)}
\]

SPI Baud Rate for PRESCALE = 0

\[
SPI_{BaudRate} = \frac{ICLK}{2}
\]

If the SPI is a network slave, the module receives a clock signal on the SPICLK pin from the network master. However, the slave's PRESCALE baud rate (Slave SPICLK) must also conform to the following specifications:

<table>
<thead>
<tr>
<th>POLARITY</th>
<th>PHASE</th>
<th>SPICLK RATIO</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>0</td>
<td>( \frac{MasterSPICLK}{2} \leq SlaveSPICLK \leq (MasterSPICLK + 1) % )</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td>( \frac{MasterSPICLK}{2} \leq SlaveSPICLK \leq (MasterSPICLK \times 2) )</td>
</tr>
</tbody>
</table>
**Control Registers**

**Bits 4:0**  
**CHARLEN**  
Controls how many times the SPI shifts per character transmitted or the number of bits per character. The binary value of the bit length must be programmed into this register. Legal values are 0x03 to 0x10. Illegal values, such as 0x00 or 0x1F are not detected and their effect is indeterminate.

**Note: CHARLEN Bits Must Be Initialized**

CHARLEN.4:0 must be initialized to the desired character length before the SPIEN bit is set. Otherwise, the first character may be shifted with an incorrect length.
## Control Registers

### 6.2 SPI Control Register 2 (SPICTRL2)

<table>
<thead>
<tr>
<th>Bits</th>
<th>31</th>
<th>30</th>
<th>29</th>
<th>28</th>
<th>27</th>
<th>26</th>
<th>25</th>
<th>24</th>
<th>23</th>
<th>22</th>
<th>21</th>
<th>20</th>
<th>19</th>
<th>18</th>
<th>17</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x04</td>
<td>U</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Reserved**

<table>
<thead>
<tr>
<th>Bits</th>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**Reserved CLK MOD SPI EN MASTER PWR DN POLARITY PHASE**

<table>
<thead>
<tr>
<th></th>
<th>RW-0</th>
<th>RW-0</th>
<th>RW-0</th>
<th>RW-0</th>
<th>RW-0</th>
<th>RW-0</th>
<th>RW-0</th>
</tr>
</thead>
<tbody>
<tr>
<td>U</td>
<td>U</td>
<td>U</td>
<td>U</td>
<td>U</td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
</tbody>
</table>

**R** = read; **W** = write; **U** = undefined; **-n** = value after reset

#### Bits 31:6

**Reserved.**

Reads are undefined and writes have no effect.

#### Bit 5

**CLKMOD.** Clock mode

Selects either an internal or external clock source. This bit also determines the I/O direction of the SPIENA and SPISCS pins in functional mode.

\[
\begin{align*}
0 & \quad \text{Clock is external} \\
1 & \quad \text{Clock is internal}
\end{align*}
\]

#### Bit 4

**SPIEN.** SPI enable

Holds the SPI in a reset state after a chip reset. The SPI is enabled only after a 1 is written to this bit. This bit must be set to 1 after all other SPI configuration bits have been written. This prevents an invalid operation of the SPI while the clock polarity is being changed. When this bit is 0, the SPI shift registers (SPIDAT0 and SPIDAT1) are held in reset mode and forced to 0x0000.

The RXINTFLAG (SPICTRL3.0) and RCVROVRN (SPICTRL3.2) bits are also held in reset mode and forced to 0 when this bit is 0. SPICLK is disabled when this bit is 0.

\[
\begin{align*}
0 & \quad \text{SPI is in reset} \\
1 & \quad \text{Activates SPI}
\end{align*}
\]
Control Registers

Note: Clearing SPIBUF

Clearing and then setting the SPIEN bit does not clear an internal flag that indicates that there is valid data in the SPI data register. This could lead to an inadvertent overrun error. The software should do a dummy read of SPIBUF after setting the SPIEN bit to clear the internal flag.

Bit 3  MASTER: SPISIMO/SPISOMI pin direction determination.

Determines the direction of the SPISIMO and SPISOMI pins.

0 = SPISIMO pin an input, SPISOMI pin an output
1 = SPISOMI pin an input, SPISIMO pin an output

Bit 2  POWERDOWN.

When active, the SPI state machines enter a powerdown state.

0 = SPI in active mode
1 = SPI in powerdown mode

Bit 1  POLARITY.

Controls the polarity of the SPICLK. Clock polarity and clock phase (SPICTRL2.0) controls four clocking schemes on the SPICLK pin. See Figure 6 to Figure 9, page 13 for waveform diagrams of the SPI clocking schemes.

Bit 0  PHASE.

Data is sent or latched in-phase with the clock signal. When PHASE = 1, SPICLK is delayed by one-half cycle from when data is output. Polarity is determined by the POLARITY bit (SPICTRL2.1). POLARITY and PHASE make four different clocking schemes possible. For information on the use of the Polarity and Phase bits, see section 2.5, Data Format, on page 11

Note: Register Configuration Bits

Since there are configuration bits in this register, two write operations must occur when setting these bits. One write to set the configuration bits and one to set the SPIEN bit.
## 6.3 SPI Control Register 3 (SPICTRL3)

<table>
<thead>
<tr>
<th>Bits</th>
<th>31</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x08</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Reserved

<table>
<thead>
<tr>
<th>Bits</th>
<th>15</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>U</td>
<td>RW-0</td>
<td>RW-0</td>
<td>RW-0</td>
<td>RC-0</td>
<td>RW-0</td>
<td>RC-0</td>
<td></td>
</tr>
</tbody>
</table>

R = Read, W = Write, C = Clear, U = Undefined; = Value after reset

**Bits 31:6** Reserved.

Reads are undefined and writes have no effect.

**Bit 5** ENABLE HIGHZ. SPIENA pin high-z enable.

When active, the SPIENA pin (when it is configured as a WAIT functional output signal in a slave SPI) is forced to place it's output in high-z when not driving a low signal. If inactive, then the pin will output both a high and a low signal.

- 0 = SPIENA pin is a value
- 1 = SPIENA pin is in high-z

**Bit 4** DMA REQ EN. DMA request enable.

Enables the DMA request signal to be generated for both receive and transmit channels.

- 0 = DMA is not used
- 1 = DMA is used

**Bit 3** OVRNINTEN. Overrun interrupt enable.

An interrupt is to be generated when the RCVR OVRN flag bit (SPICTRL3.2) is set by hardware. Otherwise, no interrupt will be generated.

- 0 = Overrun interrupt will not be generated
- 1 = Overrun interrupt will be generated
Bit 2  **RCVR OVRN.** Receiver overrun flag.

This bit is a read/clear only flag. The SPI hardware sets this bit when an operation completes before the previous character has been read from the buffer. The bit indicates that the last received character has been overwritten and therefore lost. The SPI will generate an interrupt request if this bit is set and the OVRN INTEN bit (SPICTRL3.3) is set high.

This bit is cleared in one of four ways:

- Reading the SPIBUF register
- Writing a 1 to this bit
- Writing a 0 to SPIEN (SPICTRL2.4)
- System reset

0 = Overrun condition did not occur  
1 = Overrun condition has occurred

Bit 1  **RXINTEN.**

An interrupt is to be generated when the RXINTFLAG bit (SPICTRL3.0) is set by hardware. Otherwise, no interrupt will be generated.

0 = Interrupt will not be generated  
1 = Interrupt will be generated

Bit 0  **RXINTFLAG.** Serves as the SPI interrupt flag.

This flag is set when a word is received and copied into the buffer register (SPIBUF). If RXINTEN is enabled, an interrupt is also generated. During emulation mode, however, a read to the emulation register (SPIEMU) does not clear this flag bit. This bit is cleared in one of four ways:

- Reading the SPIBUF register
- Writing a 1 to this bit
- Writing a 0 to SPIEN (SPICTRL2.4)
- System reset

0 = Interrupt condition did not occur  
1 = Interrupt condition did occur
6.4 SPI Shift Register 0 (SPIDAT0)

<table>
<thead>
<tr>
<th>Bits 31:16</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bits 15:0</td>
<td>SPIDAT0 SPI shift data 0.</td>
</tr>
</tbody>
</table>

These bits make up the SPI shift register 0. Data is shifted out of the MSB (bit 15) and into the LSB (bit 0).

SPIEN (SPICTRL2.4) must be set to 1 before this register can be written to. Writing a 0 to the SPIEN register forces the lower 16 bits of the SPIDAT0 register to 0x00.

When data is read from this register, the value is indeterminate because of the shift operation. The value in the buffer register (SPIBUF) should be read after the shift operation is complete to determine what data was shifted into the SPIDAT0 register.

When transmitting data, input data is automatically clocked in at the receive side. As the data is shifted from the MSB, the LSB of the received data is shifted in. Similarly, when the shift register is used as a receiver, the shift register continues to send data out as it receives new data on each input clock cycle. This allows the concurrent transmission and reception of data. The application software must determine whether the data transferred is valid.

For word sizes of 8 bits or less (as determined by CHARLEN) (SPICTRL1.4:0), the shift register is tapped at SPIDAT7. As a result, data of 8 bits does not need to be justified at all. For data of less than 8 bits, the data should be justified as if it is an 8-bit register.
6.5 SPI Shift Register 1 (SPIDAT1)

**Bits 31:16**  **Reserved**

Reads are undefined and writes have no effect.

**Bits 15:0**  **SPIDAT1 SPI shift data 1.**

These bits make up the SPI shift register 1. Data is shifted out of the MSB (bit 15) and into the LSB (bit 0).

SPIEN must be set to 1 before this register can be written to. Writing a 0 to the SPIEN register forces the lower 16 bits of the SPIDAT1 register to 0x00.

Write to this register ONLY when using the automatic Slave Chip Select feature. See section 2, SPI Operation Modes, on page 3. A write to this register will drive the SPISCS signal low.

When data is read from this register, the value is indeterminate because of the shift operation. The value in the buffer register (SPIBUF) should be read after the shift operation is complete to determine what data was shifted into the SPIDAT1 register.
### 6.6 SPI Buffer Register (SPIBUF)

<table>
<thead>
<tr>
<th>Bits</th>
<th>31</th>
<th>18</th>
<th>17</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x14</td>
<td>Reserved</td>
<td>RCVR OVRN IMG</td>
<td>INT FLAG IMG</td>
<td></td>
</tr>
<tr>
<td></td>
<td>U</td>
<td>RC-0</td>
<td>RC-0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bits</th>
<th>15</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SPIBUF</td>
<td>R-U</td>
</tr>
</tbody>
</table>

**R** = Read, **C** = Clear, **U** = Undefined;  
**n** = Value after reset

**Bits 31:18**  
Reserved.  
Reads are undefined and writes have no effect

**Bit 17**  
**RCVR OVRN IMG.**  
SPI receiver overrun flag image.  
This is a mirror bit of the RCVROVRN flag bit (SPICTRL3.2) and is used to reduce the interrupt latency and execution time.

This bit is cleared in one of four ways.

- Reading the SPIBUF register
- Writing a 1 to this bit
- Writing a 0 to SPIEN (SPICTRL2.4)
- System reset

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Overrun condition did not occur</td>
</tr>
<tr>
<td>1</td>
<td>Overrun condition has occurred</td>
</tr>
</tbody>
</table>

**Note: The SPIBUF Register**

The SPIBUF is a 32 bit register. Two bits in the upper 16 bits are used for control, all 16 lower bits are data buffers
Bit 16  **RXINTFLAG IMG.** SPI interrupt flag image.

This is a mirror bit of the RXINTFLAG bit (SPICtrl3.0).

This bit is cleared in one of four ways.

- Reading the SPIBUF register
- Writing a 1 to this bit
- Writing a 0 to SPIEN (SPICtrl2.4)
- System reset

0 = Interrupt condition did not occur
1 = Interrupt condition did occur

**Bits 15:0 SPIBUF: SPI buffer.**

The data in this register is the data transferred from the shift-register (SPIDAT). Since the data is shifted into the SPI most significant bit first, for word lengths less than 16, the data is stored right-justified in the register.

---

**Note: SPI Buffer**

Reading the SPIBUF register clears the RCVROVRN (SPICtrl3.2), RXINTFLAG (SPICtrl3.0), RCVR OVRN IMG (SPIBUF.17), and the RXINTFLAG IMG (SPIBUF.16) bits.
6.7 SPI Emulation Register (SPIEMU)

**Bits 31:16**  
Reserved.  
Reads are undefined and writes have no effect

**Bits 15:0**  
SPIEMU: SPI emulation.  
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 the RCVR OVRN (SPICTRL3.2) or RXINTFLAG (SPICTRL3.0) bits.
6.8 SPI Pin Control Register 1 (SPIPC1)

| Bits | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 0x1C |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |

Reserved

### Bits 31:5

**Reserved.**

Reads are undefined and writes have no effect.

### Bit 4

**SCS DIR: SPISC$\text{S}$ direction.**

Controls the direction of the SPISC$\text{S}$ pin when it is used as a general-purpose I/O pin. If the SPISC$\text{S}$ is used as a SPI functional pin, the I/O direction is determined by the CLKMOD bit (SPICTRL2.5).

- **0** = SPISC$\text{S}$ pin is an input
- **1** = SPISC$\text{S}$ pin is an output

### Bit 3

**SOMI DIR: SPISOMI direction.**

Controls the direction of the SPISOMI pin when it is used as a general-purpose I/O pin. If the SPISOMI pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit (SPICTRL2.3).

- **0** = SPISOMI pin is an input
- **1** = SPISOMI pin is an output

### Bit 2

**SIMODIR: SPISIMO direction.**

Controls the direction of the SPISIMO pin when it is used as a general-purpose I/O pin. If the SPISIMO pin is used as a SPI functional pin, the I/O direction is determined by the MASTER bit (SPICTRL2.3).

- **0** = SPISIMO pin is an input
- **1** = SPISIMO pin is an output
**Bit 1  **CLKDIR: SPICLK direction.

Controls the direction of the SPICLK pin when it is used as a general-purpose I/O pin. In functional mode, the I/O direction is determined by the CLKMOD bit (SPICTRL2.5).

0 = SPICLK pin is an input
1 = SPICLK pin is an output

**Bit 0  **ENA DIR: SPIENA direction.

Controls the direction of the SPIENA pin when it is used as a general-purpose I/O. If the SPIENA pin is used as a functional pin, then the I/O direction is determined by the CLKMOD bit (SPICTRL2.5).

0 = SPIENA pin is an input
1 = SPIENA pin is an output
### 6.9 SPI Pin Control Register 2 (SPIPC2)

<table>
<thead>
<tr>
<th>Bits</th>
<th>31</th>
<th>30</th>
<th>29</th>
<th>28</th>
<th>27</th>
<th>26</th>
<th>25</th>
<th>24</th>
<th>23</th>
<th>22</th>
<th>21</th>
<th>20</th>
<th>19</th>
<th>18</th>
<th>17</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
<td>R-U</td>
</tr>
</tbody>
</table>

- **Bits 31:5**  
  - **Reserved.**
  - **Write:** Has no effect  
  - **Read:** Value is indeterminate  
  - **Reset:** Undefined

- **Bit 4**  
  - **SCS DIN:** SPISCS data in.  
  - Reflects the value of the SPISCS pin.  
    - 0 = Current value on SPISCS pin is logic 0.  
    - 1 = Current value on SPISCS pin is logic 1

- **Bit 3**  
  - **SOMI DIN:** SPISOMI data in.  
  - Reflects the value of the SPISOMI pin.  
    - 0 = Current value on SPISOMI pin is logic 0.  
    - 1 = Current value on SPISOMI pin is logic 1

- **Bit 2**  
  - **SIMO DIN:** SPISIMO data in.  
  - Reflects the value of the SPISIMO pin.  
    - 0 = Current value on SPISIMO pin is logic 0.  
    - 1 = Current value on SPISIMO pin is logic 1

- **Bit 1**  
  - **CLK DIN:** Clock data in.  
  - Reflects the value of the SPICLK pin.  
    - 0 = Current value on SPICLK pin is logic 0.  
    - 1 = Current value on SPICLK pin is logic 1
**Control Registers**

**Bit 0  ENA DIN: SPIENA data in.**

Reflects the value of the SPIENA pin.

- **0** = Current value on SPIENA pin is logic 0.
- **1** = Current value on SPIENA pin is logic 1
6.10 SPI Pin Control Register 3 (SPIPC3)

**Bits 31:5**  
Reserved.  
Reads are undefined and writes have no effect.

**Bit 4**  
**SCS DOUT: SPISCS dataout write.**  
Only active when the **SPISCS** 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.

\[ \begin{array}{c}
0 = \text{Current value on SPISCS pin is logic 0.} \\
1 = \text{Current value on SPISCS pin is logic 1}
\end{array} \]

**Bit 3**  
**SOMI DOUT: SPISSOMI dataout write.**  
Only active when the **SPISSOMI** 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.

\[ \begin{array}{c}
0 = \text{Current value on SPISSOMI pin is logic 0.} \\
1 = \text{Current value on SPISSOMI pin is logic 1}
\end{array} \]

**Bit 2**  
**SIMO DOUT: SPISSIMO dataout write.**  
Only active when the **SPISSIMO** 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.

\[ \begin{array}{c}
0 = \text{Current value on SPISSIMO pin is logic 0.} \\
1 = \text{Current value on SPISSIMO pin is logic 1}
\end{array} \]
Control Registers

Bit 1  **CLK DOUT: SPICLK dataout write.**

Only active when the SPICLK 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.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Current value on SPICLK pin is logic 0.</td>
</tr>
<tr>
<td>1</td>
<td>Current value on SPICLK pin is logic 1.</td>
</tr>
</tbody>
</table>

Bit 0  **ENA DOUT: SPIENA dataout write.**

Only active when the SPIENA 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.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Current value on SPIENA pin is logic 0.</td>
</tr>
<tr>
<td>1</td>
<td>Current value on SPIENA pin is logic 1.</td>
</tr>
</tbody>
</table>
### 6.11 SPI Pin Control Register 4 (SPIPC4)

| Bits | 31 | 30 | 29 | 28 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 0x28 |     |     |     |     |     |     | Reserved |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|      | U  | Reserved |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| Bits | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| U | Reserved | SCS DSET | SOMI DSET | SIMO DSET | CLK DSET | ENA DSET | RW-0 | RW-0 | RW-0 | RW-0 | RW-0 | RW-0 | RW-0 | RW-0 | RW-0 | RW-0 | RW-0 | RW-0 | RW-0 | RW-0 | RW-0 | RW-0 |

R = Read, W = Write, U = Undefined; \(-n\) = Value after reset

**Bits 31:5**  
Reserved.

Reads are undefined and writes have no effect

**Bit 4**  
**SCS DOUT SET: SPISCS dataout set.**

Only active when the SPISCS pin is configured as a general-purpose output pin. A value of one written to this bit sets the corresponding SCS DOUT bit (SPIPC3.4) to one.

Write:  
0 = Has no effect  
1 = Logic 1 placed on SPISCS pin

Read:  
0 = Current value on SPISCS pin is logic 0.  
1 = Current value on SPISCS pin is logic 1

**Bit 3**  
**SOMI DSET: SPISOMI dataout set.**

Only active when the SPISOMI pin is configured as a general-purpose output pin. A value of one written to this bit sets the corresponding SPISOMIDOUT bit (SPIPC3.3) to one.

Write:  
0 = Has no effect  
1 = Logic 1 placed on SPISOMI pin

Read:  
0 = Current value on SPISOMI pin is logic 0.  
1 = Current value on SPISOMI pin is logic 1
Bit 2 SIMO DSET: SPISIMO dataout set.

Only active when the SPISIMO pin is configured as a general-purpose output pin. A value of one written to this bit sets the corresponding SPISIMODOUT bit (SPIPC3.2) to one.

Write:
0 = Has no effect
1 = Logic 1 placed on SPISIMO pin

Read:
0 = Current value on SPISIMO pin is logic 0.
1 = Current value on SPISIMO pin is logic 1

Bit 1 CLK DSET: SPICLK dataout set.

Only active when the SPICLK pin is configured as a general-purpose output pin. A value of one written to this bit sets the corresponding CLKDOUT bit (SPIPC3.1) to one.

Write:
0 = Has no effect
1 = Logic 1 placed on SPICLK pin

Read:
0 = Current value on SPICLK pin is logic 0.
1 = Current value on SPICLK pin is logic 1

Bit 0 ENA DSET: SPIENA dataout set.

Only active when the SPIENA pin is configured as a general-purpose output pin. A value of one written to this bit sets the corresponding ENABLEDOUT bit (SPIPC3.0) to one.

Write:
0 = Has no effect
1 = Logic 1 placed on SPIENA pin

Read:
0 = Current value on SPIENA pin is logic 0.
1 = Current value on SPIENA pin is logic 1

Note: Register Read

A read to this register gives the corresponding value of the SPIPC3 register.
6.12 SPI Pin Control Register 5 (SPIPC5)

Bits 31:5 Reserved.
Reads are undefined and writes have no effect.

Bit 4 SCS DCLR: SPISCS dataout clear.
Only active when the SPISCS pin is configured as a general-purpose output pin. A value of one written to this bit clears the corresponding SCSDOUT bit (SPIPC3.4) to zero.

Write:
0 = Has no effect
1 = Logic 0 placed on SPISCS pin

Read:
0 = Current value on SPISCS pin is logic 0.
1 = Current value on SPISCS pin is logic 1

Bit 3 SOMI DCLR: SPISOMI dataout clear.
Only active when the SPISOMI pin is configured as a general-purpose output pin. A value of one written to this bit clears the corresponding SPISOMIDOUT bit (SPIPC3.3) to zero.

Write:
0 = Has no effect
1 = Logic 0 placed on SPISOMI pin

Read:
0 = Current value on SPISOMI pin is logic 0.
1 = Current value on SPISOMI pin is logic 1
### Control Registers

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Active Condition</th>
<th>Write</th>
<th>Read</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>SIMO DCLR: SPISIMO dataout clear.</td>
<td>Only active when the SPISIMO pin is configured as a general-purpose output pin. A value of one written to this bit clears the corresponding SPISIMODOUT bit (SPIPC3.2) to zero.</td>
<td>0 = Has no effect 1 = Logic 0 placed on SPISIMO pin</td>
<td>0 = Current value on SPISIMO pin is logic 0. 1 = Current value on SPISIMO pin is logic 1</td>
</tr>
<tr>
<td>1</td>
<td>CLK DCLR: SPICLK dataout clear.</td>
<td>Only active when the SPICLK pin is configured as a general-purpose output pin. A value of one written to this bit clears the corresponding CLKDOUT bit (SPIPC3.1) to zero.</td>
<td>0 = Has no effect 1 = Logic 0 placed on SPICLK pin</td>
<td>0 = Current value on SPICLK pin is logic 0. 1 = Current value on SPICLK pin is logic 1</td>
</tr>
<tr>
<td>0</td>
<td>ENA DCLR: SPIENA dataout clear.</td>
<td>Only active when the SPIENA pin is configured as a general-purpose output pin. A value of one written to this bit clears the corresponding ENABLEDOUT bit (SPIPC3.0) to zero.</td>
<td>0 = Has no effect 1 = Logic 0 placed on SPIENA pin</td>
<td>0 = Current value on SPIENA pin is logic 0. 1 = Current value on SPIENA pin is logic 1</td>
</tr>
</tbody>
</table>

**Note:** Register Read

A read to this register gives the corresponding value of the SPIPC3 register.
### 6.13 SPI Pin Control Register 6 (SPIPC6)

<table>
<thead>
<tr>
<th>Bits</th>
<th>31</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0x30</td>
</tr>
</tbody>
</table>

R = Read, W = Write, U = Undefined; -n = Value after reset

#### Bits 31:5
**Reserved.**

Reads are undefined and writes have no effect

#### Bit 4
**SCS FUN: SPISCS function.**

Determines whether the SPISCS pin is to be used as a general-purpose I/O pin or as a SPI functional pin. If the slave SPISCS pin is in functional mode and receives an inactive high signal, the slave SPI will place it’s output in high-z and disable shifting.

- 0 = SPISCS pin is a GPIO
- 1 = SPISCS pin is a SPI functional pin

#### Bit 3
**SOMI FUN: Slave out, master in function.**

Determines whether the SPISOMI pin is to be used as a general-purpose I/O pin or as a SPI functional pin.

- 0 = SPISOMI pin is a GPIO
- 1 = SPISOMI pin is a SPI functional pin

#### Bit 2
**SIMO FUN: Slave in, master out function.**

Determines whether the SPISIMO pin is to be used as a general-purpose I/O pin, or as a SPI functional pin.

- 0 = SPISIMO pin is a GPIO
- 1 = SPISIMO pin is a SPI functional pin

#### Bit 1
**CLK FUN: SPI clock function.**

Determines whether the SPICLK pin is to be used as a general-purpose I/O pin, or as a SPI functional pin.

- 0 = SPICLK pin is a GPIO
- 1 = SPICLK pin is a SPI functional pin
Control Registers

**Bit 0  ENA FUN: SPIENA function.**

Determines whether the SPIENA pin is to be used as a general-purpose I/O pin, or as a SPI functional pin.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SPIENA pin is a GPIO</td>
</tr>
<tr>
<td>1</td>
<td>SPIENA pin is a SPI functional pin</td>
</tr>
</tbody>
</table>