TMS320C5515/14/05/04/VC05/VC04 DSP Serial Peripheral Interface (SPI)

User's Guide

Literature Number: SPRUFO3
September 2009

Preface **6**
1 Serial Peripheral Interface (SPI) **9**
1 Introduction **9**
1.1 Purpose of the peripheral 9
1.2 Features 9
1.3 Functional Block Diagram 9
1.4 Supported Use Case Statement 10
1.5 Industry Standard(s) Compliance Statement 10
2 Serial Peripheral Interface Architecture **11**
2.1 Clock Control 11
2.2 Signal Descriptions 12
2.3 Units of Data: Characters and Frames 12
2.4 Chip Select Control 12
2.5 Clock Polarity and Phase 12
2.6 Data Delay 14
2.7 Data Input and Output 14
2.8 Loopback Mode 15
2.9 Monitoring SPI Activity 15
2.10 Slave Access 15
2.11 Reset Considerations 18
2.12 Initialization 18
2.13 Interrupt Support 19
2.14 DMA Event Support 19
2.15 Power Management 19
2.16 Emulation Considerations 19
3 Interfacing the SPI to an SPI EEPROM **19**
3.1 Operational Description 19
3.2 Hardware Interface 19
3.3 SW Configuration 20
4 Registers **24**
4.1 Clock Divider Register (SPICDR) and Clock Control Register (SPICCR) 25
4.2 Device Configuration Registers (SPIDCR1 and SPIDCR2) 25
4.3 Command Registers (SPICMD1 and SPICMD2) 27
4.4 Status Registers (SPISTAT1 and SPISTAT2) 30
4.5 Data Registers (SPIDAT1 and SPIDAT2) 31

List of Figures

1 Serial Peripheral Interface (SPI) Block Diagram 10
2 Typical SPI Interface 10
3 Clocking Diagram for the SPI 11
4 SPI Mode 0 Transfer (CKP*n* = 0, CKPH*n* = 0) 13
5 SPI Mode 1 Transfer (CKP*n* = 0, CKPH*n* = 1) 13
6 SPI Mode 2 Transfer (CKP*n* = 1, CKPH*n* = 0) 13
7 SPI Mode 3 Transfer (CKP*n* = 1, CKPH*n* = 1) 14
8 Range of Programmable Data Delay 14
9 Data Shift Process 15
10 Flow Diagram for SPI Read or Write 17
11 SPI Access 18
12 Hardware Interface 20
13 Clock Divider Register(SPICDR) 25
14 Clock Control Register (SPICCR) 25
15 Device Configuration Register 1 (SPIDCR1) 26
16 Device Configuration Register 2 (SPIDCR2) 26
17 Command Register 1 (SPICMD1) 28
18 Command Register 2 (SPICMD2) 29
19 Status Register 1 (SPISTAT1) 30
20 Status Register 2 (SPISTAT2) 30
21 Data Register 1 (SPIDAT1) 31
22 Data Register 2 (SPIDAT2) 31

List of Tables

1 Serial Peripheral Interface (SPI) Pins 12
2 Definition of SPI Modes 13
3 SPI Module Status Bits 15
4 SPI Module Registers 24
5 Clock Divider Register (SPICDR) Field Descriptions 25
6 Clock Control Register (SPICCR) Field Descriptions 25
7 Device Configuration Register 1 (SPIDCR1) Field Descriptions 26
8 Device Configuration Register 2 (SPIDCR2) Field Descriptions 27
9 Command Register 1 (SPICMD1) Field Descriptions 28
10 Command Register 2 (SPICMD2) Field Descriptions 29
11 Status Register 1 (SPISTAT1) Field Descriptions 30
12 Status Register 2 (SPISTAT2) Field Descriptions 30
13 Data Register 1 (SPIDAT1) Field Descriptions 31
14 Data Register 2 (SPIDAT2) Field Descriptions 31

Read This First

About This Manual

This document describes the serial peripheral interface (SPI) in the TMS320C5515/14/05/04/VC05/VC04 Digital Signal Processor (DSP).

The SPI is a high-speed synchronous serial input/output port that allows a serial bit stream of programmed length (1 to 32 bits) to be shifted into and out of the device at a programmed bit-transfer rate. The SPI supports multi-chip operation of up to four SPI slave devices. The SPI can operate as a master device only.

The SPI is normally used for communication between the DSP and external peripherals. Typical applications include an interface to external I/O or peripheral expansion via devices such as shift registers, display drivers, SPI EEPROMs, and analog-to-digital converters.

Notational Conventions

This document uses the following conventions.

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

Related Documentation From Texas Instruments

The following documents describe the TMS320C5515/14/05/04 Digital Signal Processor (DSP) Digital Signal Processor (DSP). Copies of these documents are available on the internet at http://www.ti.com.

SWPU073 — TMS320C55x 3.0 CPU Reference Guide. This manual describes the architecture, registers, and operation of the fixed-point TMS320C55x digital signal processor (DSP) CPU.

SPRU652 — TMS320C55x DSP CPU Programmer's Reference Supplement. This document describes functional exceptions to the CPU behavior.

SPRUFO0 — TMS320VC5505/5504 Digital Signal Processor (DSP) Universal Serial Bus 2.0 (USB) User's Guide. This document describes the universal serial bus 2.0 (USB) in the TMS320VC5505/5504 Digital Signal Processor (DSP). The USB controller supports data throughput rates up to 480 Mbps. It provides a mechanism for data transfer between USB devices.

SPRUFO1 — TMS320C5515/14/05/04/VC05/VC04 Digital Signal Processor (DSP) Inter-Integrated Circuit (I2C) Peripheral User's Guide. This document describes the inter-integrated circuit (I2C) peripheral in the TMS320VC5505/5504 Digital Signal Processor (DSP) device. The I2C peripheral provides an interface between the device and other devices compliant with Phillips Semiconductors Inter-IC bus (I2C-bus) specification version 2.1 and connected by way of an I2C-bus. This document assumes the reader is familiar with the I2C-bus specification.

SPRUFO2 — TMS320C5515/14/05/04/VC05/VC04 Digital Signal Processor (DSP) Timer/Watchdog Timer User's Guide. This document provides an overview of the three 32-bit timers in the TMS320VC5505/5504 Digital Signal Processor (DSP) device. The 32-bit timers of the device are software programmable timers that can be configured as general-purpose (GP) timers. Timer 2 can be configured as a GP, a Watchdog (WD), or both simultaneously.

SPRUFO3 — TMS320C5515/14/05/04/VC05/VC04 Digital Signal Processor (DSP) Serial Peripheral Interface (SPI) User's Guide. This document describes the serial peripheral interface (SPI) in the TMS320VC5505/5504 Digital Signal Processor (DSP) device. The SPI is a high-speed synchronous serial input/output port that allows a serial bit stream of programmed length (1 to 32 bits) to be shifted into and out of the device at a programmed bit-transfer rate. The SPI supports multi-chip operation of up to four SPI slave devices. The SPI can operate as a master device only.

SPRUFO4 — TMS320C5515/14/05/04/VC05/VC04 Digital Signal Processor (DSP) General-Purpose Input/Output (GPIO) User's Guide. This document describes the general-purpose input/output (GPIO) on the TMS320VC5505/5504 digital signal processor (DSP). The GPIO peripheral provides dedicated general-purpose pins that can be configured as either inputs or outputs. When configured as an input, you can detect the state of an internal register. When configured as an output you can write to an internal register to control the state driven on the output pin.

SPRUFO5 — TMS320C5515/14/05/04/VC05/VC04 Digital Signal Processor (DSP) Universal Asynchronous Receiver/Transmitter (UART) User's Guide. This document describes the universal asynchronous receiver/transmitter (UART) peripheral in the TMS320VC5505/5504 Digital Signal Processor (DSP) device. The UART performs serial-to-parallel conversions on data received from a peripheral device and parallel-to-serial conversion on data received from the CPU.

SPRUFO6 — TMS320C5515/14/05/04/VC05/VC04 Digital Signal Processor (DSP) Multimedia Card (MMC)/Secure Digital (SD) Card Controller User's Guide. This document describes the Multimedia Card (MMC)/Secure Digital (SD) Card Controller on the TMS320VC5505/5504 Digital Signal Processor (DSP) device. The multimedia card (MMC)/secure digital (SD) card is used in a number of applications to provide removable data storage. The MMC/SD card controller provides an interface to external MMC and SD cards.

SPRUF07 — TMS320VC5505/5504 Digital Signal Processor (DSP) Real-Time Clock (RTC) User's Guide. This document describes the operation of the Real-Time Clock (RTC) module in the TMS320VC5505/5504 Digital Signal Processor (DSP) device. The RTC also has the capability to wake-up the power management and apply power to the rest of the device through an alarm, periodic interrupt, or external WAKEUP signal.

SPRUFO8 — TMS320C5515/14/05/04/VC05/VC04 Digital Signal Processor (DSP) External Memory Interface (EMIF) User's Guide. This document describes the operation of the external memory interface (EMIF) in the TMS320VC5505/5504 Digital Signal Processor (DSP) device. The purpose of the EMIF is to provide a means to connect to a variety of external devices.

SPRUFO9 — TMS320VC5505/5504 Digital Signal Processor (DSP) Direct Memory Access (DMA) Controller User's Guide. This document describes the features and operation of the DMA controller that is available on the TMS320VC5505/5504 Digital Signal Processor (DSP) device. The DMA controller is used to move data among internal memory, external memory, and peripherals without intervention from the CPU and in the background of CPU operation.

SPRUFP0 — TMS320VC5505 Digital Signal Processor (DSP) System User's Guide. This document describes various aspects of the TMS320VC5505/5504 digital signal processor (DSP) including: system memory, device clocking options and operation of the DSP clock generator, power management features, interrupts, and system control.

SPRUGL6 — TMS320VC5504 Digital Signal Processor (DSP) System User's Guide. This document describes various aspects of the TMS320VC5505/5504 digital signal processor (DSP) including: system memory, device clocking options and operation of the DSP clock generator, power management features, interrupts, and system control.

SPRUFP1 — TMS320C5515/05/VC05 Digital Signal Processor (DSP) Successive Approximation (SAR) Analog to Digital Converter (ADC) User's Guide. This document provides an overview of the Successive Approximation (SAR) Analog to Digital Converter (ADC) on the TMS320VC5505/5504 Digital Signal Processor (DSP). The SAR is a 10-bit ADC using a switched capacitor architecture which converts an analog input signal to a digital value.

SPRUFP3 — TMS320C5515/05/VC05 Digital Signal Processor (DSP) Liquid Crystal Display Controller (LCDC) User's Guide. This document describes the liquid crystal display controller (LCDC) in the TMS320VC5505/5504 Digital Signal Processor (DSP) device. The LCD controller includes a LCD Interface Display Driver (LIDD) controller.

SPRUFP4 — TMS320VC5505/5504 Digital Signal Processor (DSP) Inter-IC Sound (I2S) Bus User's Guide. This document describes the features and operation of Inter-IC Sound (I2S) Bus in the TMS320VC5505/5504 Digital Signal Processor (DSP) device. This peripheral allows serial transfer of full duplex streaming data, usually streaming audio, between DSP and an external I2S peripheral device such as an audio codec.

Serial Peripheral Interface (SPI)

1 Introduction

This document describes the serial peripheral interface (SPI) in the TMS320C5515/14/05/04/VC05/VC04 Digital Signal Processor (DSP).

1.1 Purpose of the peripheral

The SPI is a high-speed synchronous serial input/output port that allows a serial bit stream of programmed length (1 to 32 bits) to be shifted into and out of the device at a programmed bit-transfer rate. The SPI supports multi-chip operation of up to four SPI slave devices. The SPI can operate as a master device only.

The SPI is normally used for communication between the DSP and external peripherals. Typical applications include an interface to external I/O or peripheral expansion via devices such as shift registers, display drivers, SPI EEPROMs, and analog-to-digital converters.

1.2 Features

The SPI has the following features:

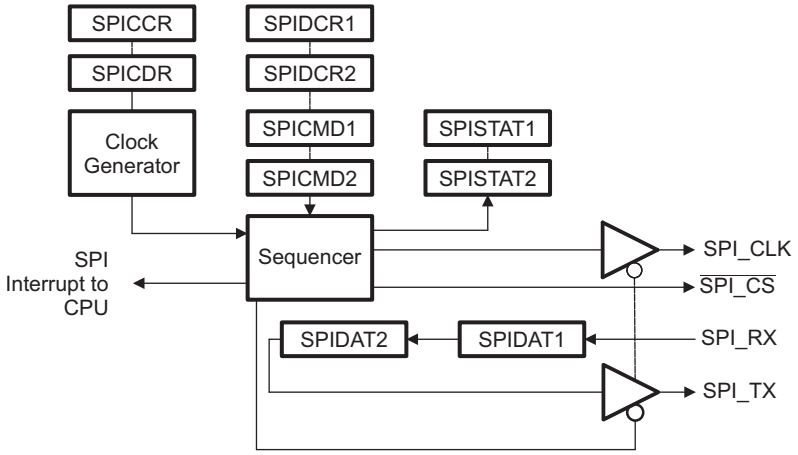
- Programmable divider for serial data clock generation.
- Four pin interface (SPI_CLK, SPI_CS*n*, SPI_TX and SPI_RX).
- Programmable data length (1 to 32 bits).
- 4 external chip select signals.
- Programmable transfer or frame size (1 to 4096 characters).
- Optional interrupt generation on character completion or frame completion.
- Programmable SPI_CS*n* to SPI_TX delay insertion from 0 to 3 SPI_CLK cycles.
- Programmable signal polarities.
- Programmable active clock edge.
- Internal loopback mode for testing.

The SPI can only operate in master mode only, slave mode is not supported.

1.3 Functional Block Diagram

Figure 1 illustrates the main components of the SPI.

Figure 1. Serial Peripheral Interface (SPI) Block Diagram

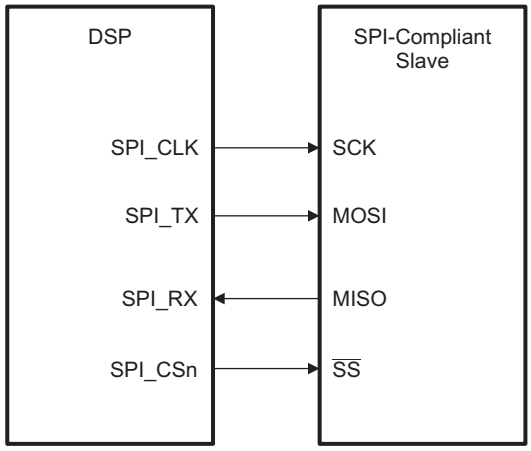


1.4 Supported Use Case Statement

The SPI is intended for communication between the DSP and up to four SPI-complaint slave devices. Typical applications include an interface to external I/O or peripheral expansion via devices such as shift registers, display drivers, SPI EEPROMs, and analog-to-digital converters. The programmable configuration capability of the SPI allows it to interface to a variety of SPI format devices without the need for glue logic.

A typical SPI interface with a single slave device is shown in Figure 2. The DSP controls the flow of communication by providing shift-clock (SPI_CLK) and slave-select signals (SPI_CS*n*).

Figure 2. Typical SPI Interface



1.5 Industry Standard(s) Compliance Statement

The SPI does not conform to a specific industry standard.

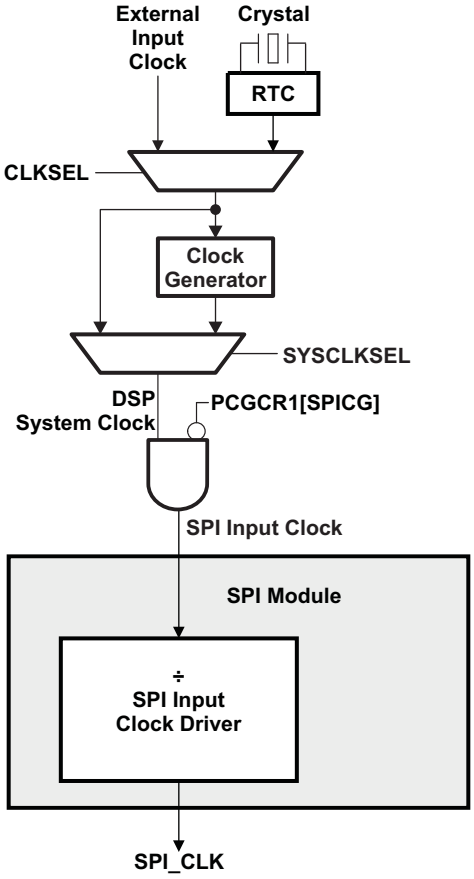
2 Serial Peripheral Interface Architecture

2.1 Clock Control

As shown in Figure 3, the clock generator receives either the real-time clock (RTC) or a signal from an external clock source and produces the DSP system clock. This internal clock is used by the DSP CPU and peripherals. A programmable clock divider in the SPI module divides down the SPI input clock to produce the SPI interface output clock (SPI_CLK). For proper device operation, the frequency of the SPI input clock must be at least four times greater than the frequency of SPI_CLK.

The DSP device includes logic which can be used to gate the clock to its on-chip peripherals, including the SPI. The input clock to the SPI can be enabled and disabled through the peripheral clock gating configuration register 1 (PCGCR1).

Figure 3. Clocking Diagram for the SPI



2.2 Signal Descriptions

Table 1 shows the SPI pins used to interface to external devices. A typical SPI interface with a single slave device is shown in Figure 2.

Table 1. Serial Peripheral Interface (SPI) Pins

Pin	Type	Function
SPI_CLK	Output	Serial clock output pin (also referred to as SCK)
SPI_TX	Output	Serial data output pin (also referred to as Master Output - Slave Input, or MOSI)
SPI_RX	Input	Serial data input pin (also referred to Master Input - Slave Output, or MISO)
SPI_CS0	Output	Slave 0 chip select pin (also referred to as Slave Select, or $\overline{SS}$)
SPI_CS1	Output	Slave 1 chip select pin (also referred to as Slave Select, or $\overline{SS}$)
SPI_CS2	Output	Slave 2 chip select pin (also referred to as Slave Select, or $\overline{SS}$)
SPI_CS3	Output	Slave 3 chip select pin (also referred to as Slave Select, or $\overline{SS}$)

2.3 Units of Data: Characters and Frames

This documentation describes SPI module communication using two terms: characters and frames. Characters are the smallest unit of data that can be transferred by the SPI module. During a transfer, the SPI generates enough clock cycles to send a character of data. The length of the character is specified by the CLEN bits of SPICMD2. The character length can be from 1 to 32 bits and can be of different size each time the SPI initiates a character transfer.

The total number of characters transmitted by the SPI module is referred to as a frame. At the beginning of a frame, the SPI module will assert the chip select pin specified by the chip select bits (CSNUM) of SPICMD2 and transfer a character of data. The SPI module will keep the chip select pin asserted until all the characters in the frame have been transferred. The frame length bits (FLEN) of SPICMD1 define the total number of characters in a frame. A frame can have up to 4096 characters. Please note that you should complete a frame transfer before using a different chip select pin.

The character count bits (CCNT) of SPISTAT2 keep track of the total number of times a character has been transferred by the SPI module. The character count is not affected by the size of the character. For example, a transmission of an 8-bit character followed by a 16-bit character increments CCNT by two.

The frame complete bit (FC) of SPISTAT1 is set after all the requested characters in a frame have been transferred. This bit is reset when SPISTAT1 is read or when a new SPI transfer is initiated via a write of a read or write command to CMD in SPICMD2.

2.4 Chip Select Control

The SPI module initiates a slave access by asserting one of its four chip select pins. The chip select pin remains asserted until an entire frame of data has been transferred. You can specify which chip select pin is activated through the chip select bits (CSNUM) of SPICMD2. Please note that writing to SPICMD2 immediately initiates a slave access; therefore you should only write to CSNUM when you are ready to initiate a slave access. Also, after initiating an access to a particular slave, you must complete the entire frame transfer to that slave before activating a different chip select pin.

The polarity of each of the four chip select pins can be configured through the CSP*n* bits of SPIDCR1 and SPIDCR2. Setting CSP*n* = 0 configures the corresponding SPI_CS*n* pin as active low while setting CSP*n* = 1 configures the SPI_CS*n* pin as active high.

2.5 Clock Polarity and Phase

Before communication between the SPI module and a slave can begin, you must specify the clock polarity and clock phase to be used. Together, the clock polarity and clock phase specify on which clock edge data is shifted in and out as well as the default state of the clock signal.

The configuration of the clock polarity and clock phase is referred to as the SPI mode. There are a total of four SPI modes (see Table 2), all which are supported by the SPI module. The clock polarity and clock phase must be configured correctly for successful communication between the SPI module and slave.

The clock polarity and phase can be specified through the CKP*n* and CKPH*n* bits of the SPI device configuration register (SPIDC). You can program a different clock polarity and phase for each slave.

Table 2. Definition of SPI Modes

SPI Mode	Clock Polarity	Clock Phase
0	Active low (base value of clock is low)	Data shifted out on the falling edge, input captured on the rising edge.
1	Active low (base value of clock is low)	Data shifted out on the rising edge, input captured on the falling edge.
2	Active high (base value of clock is high)	Data shifted out on the rising edge, input captured on the falling edge.
3	Active high (base value of clock is high)	Data shifted out on the falling edge, input captured on the rising edge.

The timing diagrams for the four possible SPI modes are shown in Figure 4 through Figure 7. Please note the following about these figures:

- Although the timing diagrams show an 8-bit character transfer, the character length can be set to 1 through 32 bits. The character length is selected with the CLEN bits SPICMD2.
- The number of characters transferred during one slave access is specified through the FLEN bits of SPICMD1. The figures show the case of FLEN = 0 (1 character).
- The polarity of the chip select pins (SPI_CS*n*) can be configured through the CSP*n* bits of SPIDCR1 and SPIDCR2. The figures show a chip select polarity of active low.
- The SPI module automatically delays the first clock edge with respect to the activation of the SPI_CS*n* pin by half a SPI_CLK cycle plus a system clock cycle. Additional clock delay cycles can be added using the data delay bits (DD*n*) of SPIDCR1 and SPIDCR2. The figures below show the case of DD*n* = 0 (zero data delay) and CLKDV is odd.

Figure 4. SPI Mode 0 Transfer (CKP*n* = 0, CKPH*n* = 0)

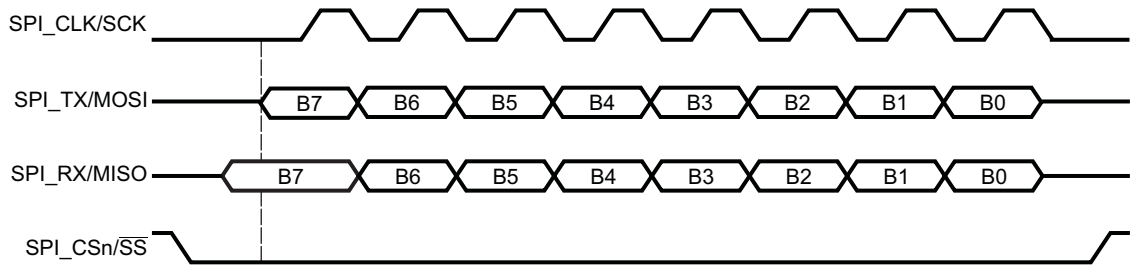


Figure 5. SPI Mode 1 Transfer (CKP*n* = 0, CKPH*n* = 1)

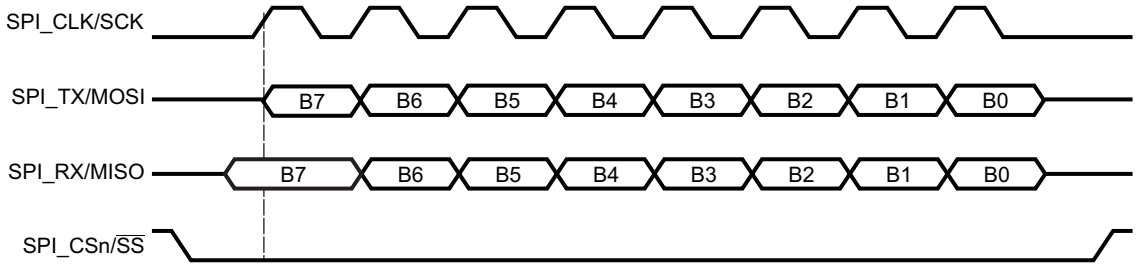


Figure 6. SPI Mode 2 Transfer (CKP*n* = 1, CKPH*n* = 0)

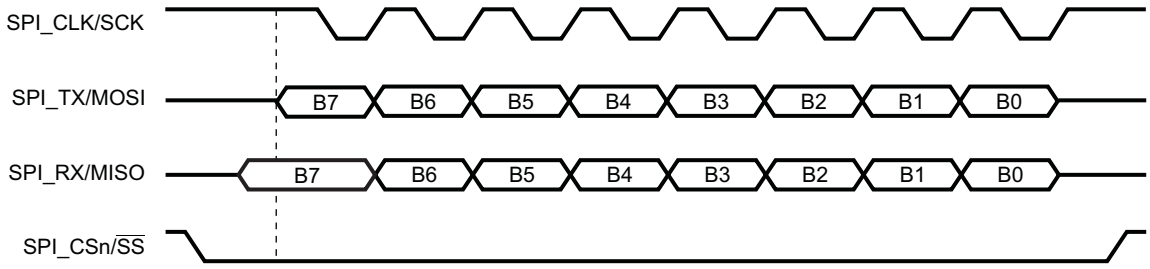
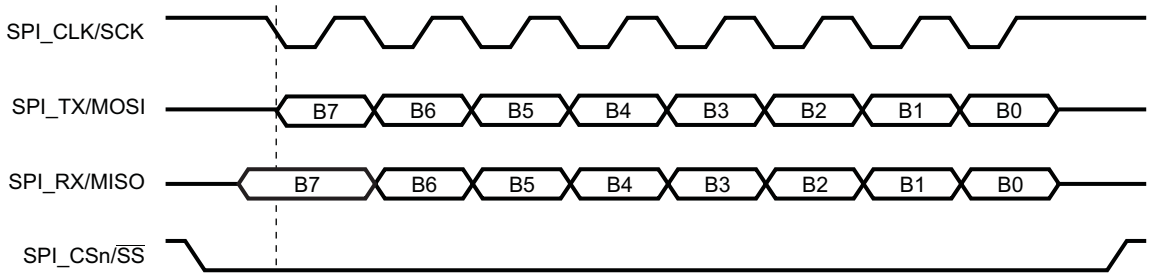


Figure 7. SPI Mode 3 Transfer (CKP*n* = 1, CKPH*n* = 1)

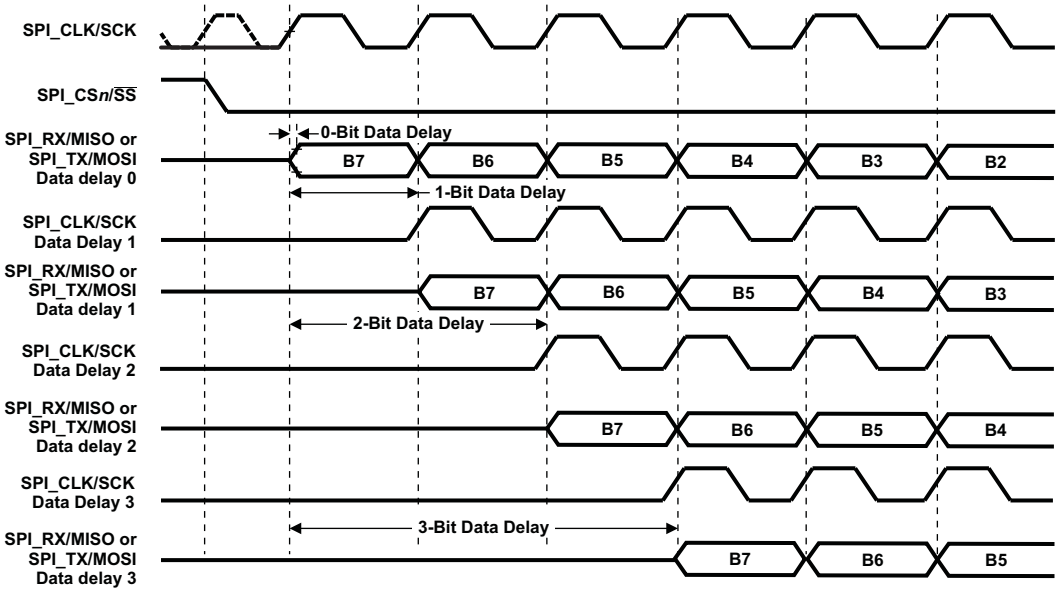


2.6 Data Delay

As described in the previous section, the SPI module automatically delays the first clock edge with respect to the activation of the SPI_CS*n* pin by half a SPI_CLK cycle plus a system clock cycle. You can program the SPI module to insert additional clock delay cycles using the data delay bits (DD*n*) of SPIDCR1 and SPIDCR2 to determine the number of clock delay cycles to insert. The data delay can be specified from zero to three clock cycles (DD*n* = 00b - 11b). Figure 8 below shows the effect of the data delay bits. Please note the following about Figure 8:

- The data transferred is an 8-bit character with bits labeled B7, B6, B5, and so on. The character length can be set to 1 through 32 bits through the CLEN bits of the SPICMD2).
- The polarity of the chip select pins (SPI_CS*n*) can be configured through the CSP*n* bits of SPIDCR1 and SPIDCR2. The figures show a chip select polarity of active low.
- The clock polarity and clock phase can be configured through the CKP*n* and CKPH*n* bits of SPIDCR1 and SPIDCR2. Figure 8 shows an example CKP*n* = 0 and CKPH*n* = 1 (SPI Mode 1).
- The first clock edge is automatically delayed by half a SPI_CLK cycle plus a system clock cycle by the SPI module.

Figure 8. Range of Programmable Data Delay



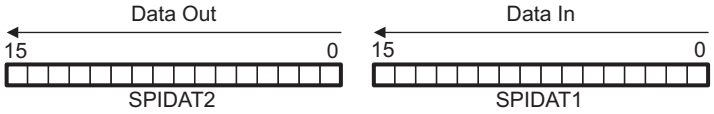
2.7 Data Input and Output

The data registers (SPIDAT1 and SPIDAT2) hold the data that is either shifted in or shifted out during a slave access. The SPIDAT1 and SPIDAT2 registers are treated as a single 32-bit shift register. Data received by the SPI is shifted into the least-significant bit of SPIDAT1 and the contents of both registers are shifted to the left. Similarly, data transferred by the SPI is shifted out of the most-significant bit of SPIDAT2 and the contents of both registers are shifted to the left. This process is illustrated in Figure 9.

The CLEN bits of SPICMD2 determine the length of the character. The character length can be set to any value between 1 and 32 bits.

The data registers are not cleared between reads or writes, and old may exist in the registers. Therefore, you must : (1) clear the data registers prior to initiating a read; and (2) mask off any unneeded data upon completing the read.

Figure 9. Data Shift Process



2.8 Loopback Mode

The SPI includes a loopback mode which can be used for testing purposes. In the loopback mode, the SPI_TX and SPI_RX are internally connected to each other. All SPI modes are usable when loopback mode is enabled.

2.9 Monitoring SPI Activity

The status registers (SPISTAT1 and SPISTAT2) contain indicators that allow you to monitor the progression of a frame transfer. These bits are described in Table 3. Additionally, the SPI module can be configured to generate interrupts after a frame or character has been transferred. These interrupts are described in Section 2.13.

Table 3. SPI Module Status Bits

Status Bit	Register	Description
FC	SPISTAT1	The frame complete bit. This bit is set after all the requested characters in a frame have been transferred. This bit is reset when SPISTAT1 is read or when a new SPI transfer is initiated via a write of a read or write command to CMD in SPICMD2.
CC	SPISTAT1	Character complete bit. This bit is set after each transfer is completed. This bit is reset when SPISTAT1 is read or when a new SPI transfer is initiated via a write of a read or write command to CMD in SPICMD2.
CCNT	SPISTAT2	Character count bits. These bits keep track of the total number of times a character has been transferred by the SPI module. The CCNT bits are not affected by the size of the character. For example, a transmission of an 8-bit character followed by a 16-bit character increments CCNT by two. These bits are reset upon completion of a frame. These bits should only be read after determining CC = 1.
BUSY	SPISTAT1	Busy bit. This bit is set during an active character transfer. Between characters this bit will be cleared to signal that the data registers can be accessed.

2.10 Slave Access

After you have initialized the SPI following the steps outlined in Section 2.12, you can follow these steps to initiate a slave access.

1. Specify the total number of characters you intend to transfer by setting the FLEN bits of SPICMD1. You must finish transferring this number of characters before initiating transfers with a different slave.
2. For writes, load the output data value into SPIDAT1 and SPIDAT2. For reads, clear SPIDAT1 and SPIDAT2. The SPI module will shift data out starting with the most-significant bit of SPIDAT2. Similarly, the SPI will shift data in starting at the least-significant bit of SPIDAT1. The amount of bits the SPI module shifts in or out is determined by the CLEN bits of SPICMD2.
3. Write to the command register (SPICMD2) to start the SPI access. The value you write SPICMD2 must set CSNUM, CLEN, and CMD to the chip select, character length, and command to be used.
4. Poll SPISTAT1 until CC = 1 or wait for the character complete interrupt (CIRQ).
5. Poll SPISTAT1 until BUSY = 0. This ensures that a character is not being transferred.
6. For reads, the data is loaded into SPIDAT1 and SPIDAT2. You must mask off any invalid bits when reading SPIDAT1 and SPIDAT2. The number of valid bits is determined by CLEN in SPICMD2.
7. Repeat steps 2 though 5 until all the characters have been transferred.

Please note the following points:

- The chip select pin specified in SPICMD2 will be activated as soon as the first character transfer is initiated. The chip select pin will remain activated until all the characters specified by FLEN have been transferred.
- SPIDAT1 and SPIDAT2 are treated as a 32-bit shift register where SPIDAT2 holds the most-significant word and SPIDAT1 holds the least-significant word. During writes, data is shifted out starting with the most-significant bit of SPIDAT2. For reads, data is shifted in starting at the least-significant bit of SPIDAT1.
- It is important for you to always load SPIDAT1 and SPIDAT2 with the values you intend to shift out before initiating an SPI write. It is equally important to always mask off invalid bits after reading values from SPIDAT1 and SPIDAT2 after every SPI read.
- You must finish transferring the number of characters specified by FLEN in SPICMD1 before initiating transfers with a different slave.
- SPI_CLK will only be activated while data is being shifted out.

Figure 10 illustrates the steps described above.

Figure 10. Flow Diagram for SPI Read or Write

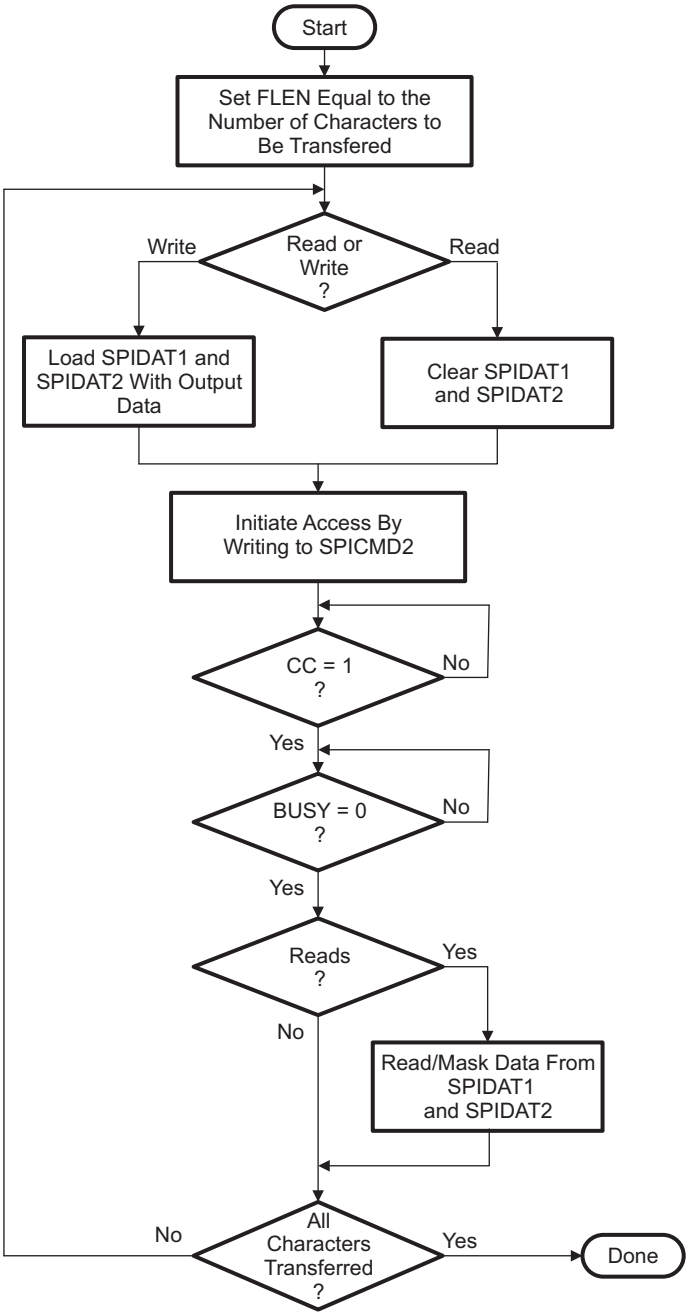
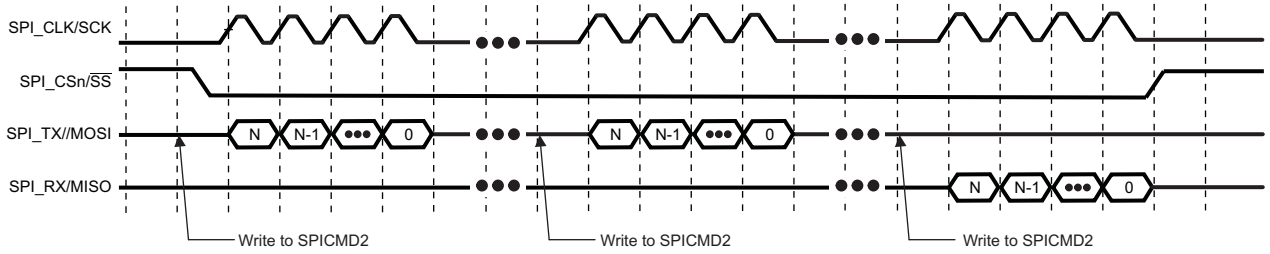


Figure 11 illustrates the activity at the pins of the SPI module.

Figure 11. SPI Access



2.11 Reset Considerations

The SPI module has two reset sources: software reset and hardware reset.

2.11.1 Software Reset Considerations

The SPI module can be reset by software through the RST bit in the clock control register (SPICCR) or through the PG4_RST bit in the peripheral reset control register (PRCR).

When RST is set to 1, the SPI module is reset.

The entire SPI module may also be reset through the PG4_RST bit of PRCR. When this reset is asserted, all SPI registers are set to their default values. The SPI remains inactive until programmed by software. Note that from the SPI perspective, this reset appears as a hardware reset.

NOTE: The PG4_RST bit of PRCR resets other peripherals besides the SPI. For more details on this bit and register, refer to the device-specific data manual.

2.11.2 Hardware Reset Considerations

When the SPI is reset due to a device reset, all the SPI registers are set to their default values. The SPI module remains inactive until programmed by software.

2.12 Initialization

The following initialization procedure describes the basic setup of the SPI module. Please note that the exact configuration will depend on the characteristics of the slave device. For specific examples, please refer to Section 3.

1. Ensure the SPI is out reset by setting SPI_RST = 0 in the peripheral reset control register (PRCR). See the device-specific data manual for more information on PRCR.
2. Enable the SPI input clock by setting SPICG to 0 in the peripheral clock gating configuration register (PCGCR1). See the device-specific data manual for more information on PCGCR1.
3. Set CLKEN = 0 to disable SPI_CLK. The CLKEN bit is part of SPICCR.
4. Set CLKDV in SPICCR to provide the appropriate SPI_CLK frequency. Note that for proper device operation, CLKDV must be greater than or equal to 3. Also, note that the clock frequency selected in this step applies to all slave devices. If a different frequency is required for each slave, you must reprogram the clock register each time you access a different slave.
5. Set CLKEN = 1 to enable SPI_CLK.
6. Program the device configuration registers (SPIDCR1 and SPIDCR2) with the required clock phase, clock polarity, chip select pin polarity, and data delay settings for each slave.
7. Program the SPI module to generate an interrupt after a frame of characters has been transferred (FIRQ = 1) or after a single character has been transferred (CIRQ = 1), If you want to use interrupts. The FIRQ and CIRQ bits are located in SPICMD1.

Enable the SPI pins through the external bus selection register (EBSR). EBSR controls the pin multiplexing options of the device. Refer to the device-specific data manual for more information on EBSR.

2.13 Interrupt Support

2.13.1 Interrupt Events and Requests

The SPI module is capable of interrupting the CPU after every character transfer and after every frame transfer. The SPI interrupts are enabled through the FIRQ and CIRQ bits of the command register 1 (SPICMD1).

2.13.2 Interrupt Multiplexing

The SPI module generates a single interrupt to the CPU. The SPI interrupt is not multiplexed with any other interrupt source.

2.14 DMA Event Support

The SPI module does not generate any DMA events; it must be fully serviced by the CPU. Furthermore, none of the device DMA controllers have access to the SPI memory-mapped registers.

2.15 Power Management

The SPI module can be clock-gated to conserve power during periods of no activity. The SPI input clock can be turned off by using the peripheral clock gating configuration register (PCGCR). For detailed information on PCGCR, see the device-specific data manual.

2.16 Emulation Considerations

The SPI module is not interrupted by emulation events such as an emulation breakpoint.

3 Interfacing the SPI to an SPI EEPROM

The SPI module can be used to communicate with an SPI EEPROM. This section gives an example of interfacing the SPI a 256K-bit SPI EEPROM from Catalyst Semiconductor (CAT25C256). Software sequences for common tasks such as data block reads and writes are also included.

3.1 Operational Description

The SPI module is totally controlled by the CPU. The SPI initiates transfers with SPI devices when the CPU writes to the SPICMD2 register. The SPI can interrupt the CPU when it has completed a character transfer and when it has completed a frame transfer. Alternatively, the CPU can poll the SPI status registers. The SPI cannot generate DMA events; therefore, it is the task of the CPU to move data to and from the SPI data registers.

Communication with SPI EEPROM is carried out completely by the SPI module. The SPI module drives the chip select pin of the memory device and allows the clocks to shift data in and out.

The SPI clock is derived from the chip system clock. The clock divider bits (CLKDV) of SPICCR are used to configure the frequency of the SPI clock (SPI_CLK) as follows:

SPI_CLK frequency = SPI module input clock / (CLKDV + 1)

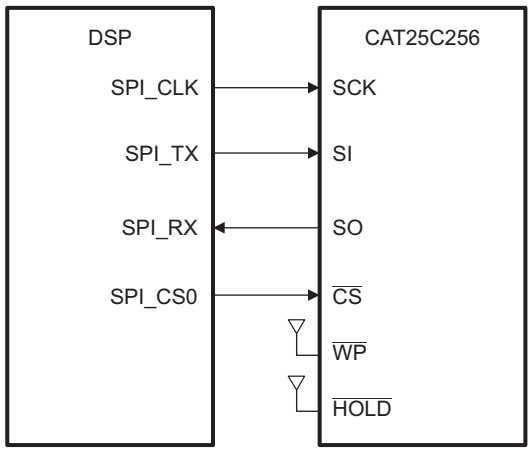
The number of characters sent or received by the SPI module is specified through the FLEN bits of SPICMD1. During communication with a SPI device, the SPI module will assert the chip select pin until all the characters specified through FLEN have been transferred.

3.2 Hardware Interface

Figure 12 shows the required hardware interface. Please note the following:

- The $\overline{WP}$ pin is shown as tied high (write-protect feature disabled); however, you can choose to tie the $\overline{WP}$ pin high or low depending on whether or not you want to use the write-protect feature of the CAT25C256 device.
- The $\overline{HOLD}$ pin is not used in this example and is pulled high.
- The example described in this section assumes chip-select 0 is used; however, you can use any other chip select.

Figure 12. Hardware Interface



3.3 SW Configuration

The following sections describe the software sequence for common tasks such as data block writes and data block reads.

3.3.1 Basic Initialization

The following procedure describes the basic steps required to setup the SPI module for communication with the CAT25C256 device.

1. Ensure the SPI is out reset by setting SPI_RST=0 in the peripheral reset control register (PRCR). See the device-specific data manual for more information on PRCR.
2. Enable the SPI input clock by setting SPICG to 0 in the peripheral clock gating configuration register (PCGCR1). See the device-specific data manual for more information on PCGCR1.
3. Set CLKEN = 0 to disable SPI_CLK. The CLKEN bit is part of SPICCR .
4. Set CLKDV in SPICCR to provide the appropriate SPI_CLK frequency. Note that for proper device operation, CLKDV must be greater than or equal to 3 .
5. Set CLKEN = 1 to enable SPI_CLK.
6. The CAT25C256 device latches input data on the rising edge of the clock and shifts out data on the falling edge of the clock. Therefore, the SPI must be programmed to shift out data on the falling edge of the clock and shift in data on the rising edge of the clock. According to Table 2, this corresponds to SPI mode 0. To enable SPI mode 0, set CKP0 = 0 and CKPH0 = 0 in SPIDCR1. Also, the data delay field should be programmed such that the chip select setup requirement of the CAT25C256 is met. A 1-bit data delay is recommended when running the SPI_CLK at frequencies greater than 2.5 MHz; otherwise a 0-bit data delay can be used.
7. Program the SPI module to generate an interrupt after a frame of characters has been transferred (FIRQ = 1) or after a single character has been transferred (CIRQ = 1), If you want to use interrupts. The FIRQ and CIRQ bits are located in SPICMD1. This example assumes interrupts are not used.

8. Enable the SPI pins through the external bus selection register (EBSR). EBSR controls the pin multiplexing options of the device. The SPI pins are multiplexed with other peripherals on the device and the pin multiplexing configuration you choose will depend on your use-case scenario. Refer to the device-specific data manual for more information on EBSR.

3.3.2 Reading the SPI EEPROM Status Register

The status register of the CAT25C256 device provides useful information including the write status and write protect feature status. The SPI module can be easily used to read the contents of this register by following these steps:

1. Ensure the SPI is not busy with another transfer by polling the CC and BUSY bits in SPISTAT1.
2. Reading the EEPROM status register requires 8 clock cycles for the read status register command (RDSR) and eight cycles for the actual data. Therefore, set FLEN = 1 (2 character transfer) in SPICMD1.
3. Load the opcode for the read status register command (05h) into the upper byte of SPIDAT2.
4. Write to the command register (SPICMD2) to start the SPI write. The value you write SPICMD2 must set CSNUM = 0 (chip-select 0 active), CLEN = 7 (8-bit character length), and CMD = 10b (write).
5. Poll SPISTAT1 until CC = 1 or wait for the character complete interrupt (CIRQ).
6. Poll SPISTAT1 until BUSY = 0. This ensures that a character is not being transferred.
7. Prepare SPIDAT1 and SPIDAT2 for data reception by loading both registers with 0000h.
8. Write to SPICMD2 to start an SPI read. The value you write SPICMD2 must set CSNUM = 0 (chip-select 0 active), CLEN = 7 (8-bit character length), and CMD = 01b (read).
9. Poll SPISTAT1 until CC = 1 or wait for the character complete interrupt (CIRQ).
10. Poll SPISTAT1 until BUSY = 0. This ensures that a character is not being transferred.
11. Read the value shifted out by the SPI EEPROM from the lower byte of SPIDAT1 (mask lower byte in data read from SPIDAT1).

3.3.3 Enabling and Disabling Writes

Before writing data to the CAT25C256 device, writes must be enabled. Similarly, to prevent inadvertent writes, writes should be disabled. Writes can be enabled by sending the write enable command (WREN) and writes can be disabled by sending the write disable command (WRDI). Both of these commands can be easily sent to the SPI EEPROM using the SPI module.

1. Ensure the SPI is not busy with another transfer by polling the CC and BUSY bits in SPISTAT1.
2. Sending the WREN or WRDI command to the EEPROM requires a total of 8 clock cycles. Therefore, set FLEN = 0 (1 character transfer) in SPICMD1.
3. Load the opcode for either WREN (06h) or WRDI (04h) into the upper byte of SPIDAT2.
4. Write to the command register (SPICMD2) to start the SPI write. The value you write SPICMD2 must set CSNUM = 0 (chip-select 0 active), CLEN = 7 (8-bit character length), and CMD = 10b (write).
5. Poll SPISTAT1 until CC = 1 or wait for the character complete interrupt (CIRQ).
6. Poll SPISTAT1 until BUSY = 0. This ensures that a character is not being transferred.

You can read the EEPROM status register to verify the WREN or WRDI commands were received successfully. See Section 3.3.2 for more details on reading the status register of the SPI EEPROM.

3.3.4 Writing a Block of Data to a SPI EEPROM

Once the CAT25C256 device has been configured to accept writes, you can use the write data memory command (WRITE) to program the contents of the device. Please note that the CAT25C256 device allows for a page write sequence in which you are allowed to write up to 64 bytes (a page) while the EEPROM automatically increments its address counter. The only restriction during page writes is that page boundaries must not be crossed. If the edge of a page boundary is reached, the address counter rolls over to the beginning of the page. The following steps outline the procedure for conducting a page write to the SPI EEPROM.

1. Configure the SPI EEPROM to allow writes through the use of the WREN command. See Section 3.3.3 for more details.
2. Ensure the SPI is not busy with another transfer by polling the CC and BUSY bits in SPISTAT1.
3. Writing a page of data to the EEPROM requires 8 clock cycles for the WRITE command, 16 clock cycles for the address, and eight cycles for each data byte. Therefore, set FLEN = 1 + 2 + N – 1, where N is the number of data bytes to be written. For example, if you want to write 64 data bytes, set FLEN = 66.
4. Load the opcode for the WRITE command (02h) into the upper byte of SPIDAT2.
5. Write to the command register (SPICMD2) to start the SPI write. The value you write SPICMD2 must set CSNUM = 0 (chip-select 0 active), CLEN = 7 (8-bit character length), and CMD = 10b (write).
6. Poll SPISTAT1 until CC = 1 or wait for the character complete interrupt (CIRQ).
7. Poll SPISTAT1 until BUSY = 0. This ensures that a character is not being transferred.
8. Load the most-significant byte of the SPI EEPROM address into the upper byte of SPIDAT2.
9. Write to the command register (SPICMD2) to start the SPI write. The value you write SPICMD2 must set CSNUM = 0 (chip-select 0 active), CLEN = 7 (8-bit character length), and CMD = 10b (write).
10. Poll SPISTAT1 until CC = 1 or wait for the character complete interrupt (CIRQ).
11. Poll SPISTAT1 until BUSY = 0. This ensures that a character is not being transferred.
12. Load the least-significant byte of the SPI EEPROM address into the upper byte of SPIDAT2.
13. Write to the command register (SPICMD2) to start the SPI write. The value you write SPICMD2 must set CSNUM = 0 (chip-select 0 active), CLEN = 7 (8-bit character length), and CMD = 10b (write).
14. Poll SPISTAT1 until CC = 1 or wait for the character complete interrupt (CIRQ).
15. Poll SPISTAT1 until BUSY = 0. This ensures that a character is not being transferred.
16. Load the data byte to be written to the upper byte of SPIDAT2. The SPI module will shift data out starting with the most-significant bit of SPIDAT2.
17. Write to SPICMD2 to start an SPI write. The value you write SPICMD2 must set CSNUM = 0 (chip-select 0 active), CLEN = 7 (8-bit character length), and CMD = 10b (write).
18. Poll SPISTAT1 until CC = 1 or wait for the character complete interrupt (CIRQ).
19. Poll SPISTAT1 until BUSY = 0. This ensures that a character is not being transferred.
20. Repeat steps 16 through 19 for the remainder of the bytes.

At the end of this sequence, you can poll the $\overline{\text{RDY}}$ bit in the SPI EEPROM status register (see Section 3.3.2) to determine when the EEPROM has completed the writes. At that time, you can restart this sequence to write more bytes to the EEPROM at a different address.

3.3.5 Reading a Block of Data from a SPI EEPROM

Unlike a page write, the CAT25C256 device allows you to read the entire memory contents without regard for page boundaries. The read data from memory command (READ) is used to read data from the SPI EEPROM. The following steps outline the procedure required to read a block of data from the memory device.

1. Ensure the SPI is not busy with another transfer by polling the CC and BUSY bits in SPISTAT1.
2. Reading a block of data from the EEPROM requires 8 clock cycles for the READ command, 16 clock cycles for the address, and eight cycles for each data byte. Therefore, set FLEN = 1 + 2 + N – 1, where N is the number of bytes to be read. For example, if you want to read 64 bytes, set FLEN = 66.
3. Load the opcode for the READ command (03h) into the upper byte of SPIDAT2.
4. Write to the command register (SPICMD2) to start the SPI write. The value you write SPICMD2 must set CSNUM = 0 (chip-select 0 active), CLEN = 7 (8-bit character length), and CMD = 10b (write).
5. Poll SPISTAT1 until CC = 1 or wait for the character complete interrupt (CIRQ).
6. Poll SPISTAT1 until BUSY = 0. This ensures that a character is not being transferred.
7. Load the most-significant byte of the SPI EEPROM address into the upper byte of SPIDAT2.
8. Write to the command register (SPICMD2) to start the SPI write. The value you write SPICMD2 must set CSNUM = 0 (chip-select 0 active), CLEN = 7 (8-bit character length), and CMD = 10b (write).
9. Poll SPISTAT1 until CC = 1 or wait for the character complete interrupt (CIRQ).
10. Poll SPISTAT1 until BUSY = 0. This ensures that a character is not being transferred.
11. Load the least-significant byte of the SPI EEPROM address into the upper byte of SPIDAT2.
12. Write to the command register (SPICMD2) to start the SPI write. The value you write SPICMD2 must set CSNUM = 0 (chip-select 0 active), CLEN = 7 (8-bit character length), and CMD = 10b (write).
13. Poll SPISTAT1 until CC = 1 or wait for the character complete interrupt (CIRQ).
14. Poll SPISTAT1 until BUSY = 0. This ensures that a character is not being transferred.
15. Prepare SPIDAT1 and SPIDAT2 for data reception by loading both registers with 0000h.
16. Write to SPICMD2 to start an SPI read. The value you write SPICMD2 must set CSNUM = 0 (chip-select 0 active), CLEN = 7 (8-bit character length), and CMD = 01b (read).
17. Poll SPISTAT1 until CC = 1 or wait for the character complete interrupt (CIRQ).
18. Poll SPISTAT1 until BUSY = 0. This ensures that a character is not being transferred.
19. Read the value shifted out by the SPI EEPROM from the lower byte of (mask lower byte in data read from SPIDAT1) SPIDAT1.
20. Repeat steps 15 through 19 for the remainder of the bytes.

4 Registers

Table 4 lists the memory-mapped registers associated with the SPI module of the device DSP. The SPI registers can be accessed by the CPU at the 16-bit addresses specified in Table 4. Note that the CPU accesses all peripheral registers through its I/O space. All other register addresses not listed in Table 4 should be considered as reserved locations and the register contents should not be modified.

Table 4. SPI Module Registers

CPU Word Address	Acronym	Register Description	Section
3000h	SPICDR	Clock Divider Register	Section 4.1
3001h	SPICCR	Clock Control Register	Section 4.1
3002h	SPIDCR1	Device Configuration Register 1	Section 4.2
3003h	SPIDCR2	Device Configuration Register 2	Section 4.2
3004h	SPICMD1	Command Register 1	Section 4.3
3005h	SPICMD2	Command Register 2	Section 4.3
3006h	SPISTAT1	Status Register 1	Section 4.4
3007h	SPISTAT2	Status Register 2	Section 4.4
3008h	SPIDAT1	Data Register 1	Section 4.5
3009h	SPIDAT2	Data Register 2	Section 4.5

4.1 Clock Divider Register (SPICDR) and Clock Control Register (SPICCR)

The SPI includes two registers for controlling the SPI interface clock (SPI_CLK). The SPI input clock is divided down to generate a SPI_CLK. The clock divider register (SPICDR) specifies the clock divider value for the SPI input clock. The clock control register (SPICCR) is used to enable the clock output and to initiate a soft reset to the SPI module.

Figure 13. Clock Divider Register(SPICDR)

15	0
CLKDV	
RW-0	

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

Table 5. Clock Divider Register (SPICDR) Field Descriptions

Bit	Field	Value	Description
15-0	CLKDV	0-3FFFh	Clock divider bits. The SPI input clock is divided down to generate the SPI interface clock (SPI_CLK). You can specify the divider value through the CLKDV bits. The frequency SPI_CLK is: SPI_CLK frequency = (SPI input clock frequency) / (CLKDV + 1) The duty cycle of SPI_CLK depends on the value of CLKDV + 1. When CLKDIV+1 is even, the duty cycle is 50%. When CLKDIV+1 is odd, the clock high and low times depend on the clock polarity. Low clock polarity: %high = (CLKDIV/2)/(CLK_DIV+1) %low = ((CLK_DIV/2)+1)/(CLK_DIV+1) High clock polarity %low = (CLKDIV/2)/(CLK_DIV+1) %high = ((CLK_DIV/2)+1)/(CLK_DIV+1) **NOTE:** For proper device operation CLKDV must be greater than or equal to 3. See Section 2.5 for more information on the SPI modes.

Figure 14. Clock Control Register (SPICCR)

15	14	13	0
CLKEN	RST	Reserved	
RW-0	W-0	R-xxxx	

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

Table 6. Clock Control Register (SPICCR) Field Descriptions

Bit	Field	Value	Description
15	CLKEN		Clock enable bit. This bit is used to enable and disable the SPI interface clock (SPI_CLK).
		0	SPI_CLK is disabled and held at the logic value specified by the clock polarity bit of SPIDC.
		1	SPI_CLK is enabled.
14	RST		Soft reset bit. Writing a 1 to this bit will reset the SPI module. This bit is self-clearing and will deactivate the soft reset on the next SPI input clock cycle after this bit is set to 1.
		0	Soft reset is released.
		1	Soft reset is asserted.
13-0	Reserved	0	Reserved.

4.2 Device Configuration Registers (SPIDCR1 and SPIDCR2)

The device configuration registers (SPIDCR1 and SPIDCR2) are used to specify the clock phase, clock polarity, and data delay for each SPI slave connected to the SPI chip select pins. Together, the clock phase and clock polarity determine the SPI mode (refer to Section 2.5 for more details). The loopback mode of the SPI can also be enabled through SPIDCR2.

Figure 15. Device Configuration Register 1 (SPIDCR1)

15 13	12 11	10	9	8	7 5	4 3	2	1	0
Reserved	DD1	CKPH1	CSP1	CKP1	Reserved	DD0	CKPH0	CSP0	CKP0
R-0	RW-0	RW-0	RW-0	RW-0	R-0	RW-0	RW-0	RW-0	RW-0

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

Table 7. Device Configuration Register 1 (SPIDCR1) Field Descriptions

Bit	Field	Value	Description
15-13	Reserved	0	Reserved.
12-11	DD1	0-3h	Data delay for chip select 1 pin (SPI_CS1).
		0	First SPI_CLK edge is delayed 0.5 clock cycles from SPI_CS1 assertion.
		1h	First SPI_CLK edge is delayed 1.5 SPI clock cycles from SPI_CS1 assertion.
		2h	First SPI_CLK edge is delayed 2.5 SPI clock cycles from SPI_CS1 assertion.
		3h	First SPI_CLK edge is delayed 3.5 SPI clock cycles from SPI_CS1 assertion.
10	CKPH1		Clock phase for chip select 1 pin (SPI_CS1). The clock phase bit, in conjunction with the clock polarity bit (CKP1), controls the clock-data relationship between master and slave.
		0	When CKP1 = 0, data shifted out on falling edge, input captured on rising edge. When CKP1 = 1, data shifted out on rising edge, input captured on falling edge.
		1	When CKP1 = 0, data shifted out on rising edge, input captured on falling edge. When CKP1 = 1, data shifted out on falling edge, input captured on rising edge.
9	CSP1		Polarity for chip select 1 pin (SPI_CS1).
		0	Active low.
		1	Active high.
8	CKP1		Clock polarity inactive state for the clock pin during accesses to chip select 1.
		0	When data is not being transferred, a steady state low value is produced at the SPI_CLK pin.
		1	When data is not being transferred, a steady state high value is produced at the SPI_CLK pin.
7-5	Reserved	0	Reserved.
4-3	DD0	0-3h	Data delay for chip select 0 pin (SPI_CS0).
		0	First SPI_CLK edge is delayed 0.5 clock cycles from SPI_CS0 assertion.
		1h	First SPI_CLK edge is delayed 1.5 clock cycles from SPI_CS0 assertion.
		2h	First SPI_CLK edge is delayed 2.5 clock cycles from SPI_CS0 assertion.
		3h	First SPI_CLK edge is delayed 3.5 clock cycles from SPI_CS0 assertion.
2	CKPH0		Clock phase for chip select 0 pin (SPI_CS0). The clock phase bit, in conjunction with the clock polarity bit (CKP0), controls the clock-data relationship between master and slave.
		0	When CKP0 = 0, data shifted out on falling edge, input captured on rising edge. When CKP0 = 1, data shifted out on rising edge, input captured on falling edge.
		1	When CKP0 = 0, data shifted out on rising edge, input captured on falling edge. When CKP0 = 1, data shifted out on falling edge, input captured on rising edge.
1	CSP0		Polarity for chip select 0 pin (SPI_CS0).
		0	Active low.
		1	Active high.
0	CKP0		Clock polarity inactive state for the clock pin during accesses to chip select 0.
		0	When data is not being transferred, a steady state low value is produced at the SPI_CLK pin.
		1	When data is not being transferred, a steady state high value is produced at the SPI_CLK pin.

Figure 16. Device Configuration Register 2 (SPIDCR2)

15	14 13	12 11	10	9	8	7 5	4 3	2	1	0
LPBK	Reserved	DD3	CKPH3	CSP3	CKP3	Reserved	DD2	CKPH2	CSP2	CKP2
RW-0	R-0	RW-0	RW-0	RW-0	RW-0	R-0	RW-0	RW-0	RW-0	RW-0

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

Table 8. Device Configuration Register 2 (SPIDCR2) Field Descriptions

Bit	Field	Value	Description
15	LPBK		Loopback mode.
		0	Loopback mode is disabled.
		1	Loopback mode is enabled.
14-13	Reserved	0	Reserved
12-11	DD3	0-3h	Data delay for chip select 3 pin (SPI_CS3).
		0	First SPI_CLK edge is delayed 0.5 clock cycles from SPI_CS3 assertion.
		1h	First SPI_CLK edge is delayed 1.5 clock cycles from SPI_CS3 assertion.
		2h	First SPI_CLK edge is delayed 2.5 clock cycles from SPI_CS3 assertion.
		3h	First SPI_CLK edge is delayed 3.5 clock cycles from SPI_CS3 assertion.
10	CKPH3		Clock phase for chip select 3 pin (SPI_CS3). The clock phase bit, in conjunction with the clock polarity bit (CKP3), controls the clock-data relationship between master and slave.
		0	When CKP3 = 0, data shifted out on falling edge, input captured on rising edge. When CKP3 = 1, data shifted out on rising edge, input captured on falling edge.
		1	When CKP3 = 0, data shifted out on rising edge, input captured on falling edge. When CKP3 = 1, data shifted out on falling edge, input captured on rising edge.
9	CSP3		Polarity for chip select 3 pin (SPI_CS3).
		0	Active low.
		1	Active high.
8	CKP3		Clock polarity inactive state for the clock pin during accesses to chip select 3.
		0	When data is not being transferred, a steady state low value is produced at the SPI_CLK pin.
		1	When data is not being transferred, a steady state high value is produced at the SPI_CLK pin.
7-5	Reserved	0	Reserved
4-3	DD2	0-3h	Data delay for chip select 2 pin (SPI_CS2).
		0	First SPI_CLK edge is delayed 0.5 clock cycles from SPI_CS2 assertion.
		1h	First SPI_CLK edge is delayed 1.5 clock cycles from SPI_CS2 assertion.
		2h	First SPI_CLK edge is delayed 2.5 clock cycles from SPI_CS2 assertion.
		3h	First SPI_CLK edge is delayed 3.5 clock cycles from SPI_CS2 assertion.
2	CKPH2		Clock phase for chip select 2 pin (SPI_CS2). The clock phase bit, in conjunction with the clock polarity bit (CKP2), controls the clock-data relationship between master and slave.
		0	When CKP2 = 0, data shifted out on falling edge, input captured on rising edge. When CKP2 = 1, data shifted out on rising edge, input captured on falling edge.
		1	When CKP2 = 0, data shifted out on rising edge, input captured on falling edge. When CKP2 = 1, data shifted out on falling edge, input captured on rising edge.
1	CSP2		Polarity for chip select 2 pin (SPI_CS2).
		0	Active low.
		1	Active high.
0	CKP2		Clock polarity inactive state for the clock pin during accesses to chip select 2.
		0	When data is not being transferred, a steady state low value is produced at the SPI_CLK pin.
		1	When data is not being transferred, a steady state high value is produced at the SPI_CLK pin.

4.3 Command Registers (SPICMD1 and SPICMD2)

The command registers (SPICMD1 and SPICMD2) are used to control several aspects of an SPI access. You can use SPICMD1 to enable character and frame complete interrupts and to specify the number of characters in a frame. The command to use (read or write), character length, and chip select pin used during SPI transfers are specified through SPICMD2. Writing to SPICMD2 will cause the SPI to execute the command specified by the transfer command bits (CMD).

Figure 17. Command Register 1 (SPICMD1)

15	14	13	12	11	0
FIRQ	CIRQ	Reserved		FLEN	
RW-0	RW-0	R-0		RW-0	

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

Table 9. Command Register 1 (SPICMD1) Field Descriptions

Bit	Field	Value	Description
15	FIRQ		Frame count interrupt enable.
		0	No interrupt generated at the end of the frame count.
		1	Interrupt generated at the end of the frame count.
14	CIRQ		Character interrupt enable.
		0	No interrupt generated at the end of the character transfer.
		1	Interrupt generated at the end of the character transfer.
13-12	Reserved	0	Reserved.
11-0	FLEN	0-FFFh	Frame length bits. These bits are used to specify the length of entire transfer.
			The total number of characters transferred equals FLEN + 1.
			For example, if FLEN = 63, a frame consists of a total of 64 characters.

Figure 18. Command Register 2 (SPICMD2)

15 14	13 12	11 8	7 3	2	1 0
Reserved	CSNUM	Reserved	CLEN	RSV	CMD
R-0	RW-0	R-0	RW-0	R-0	RW-0

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

Table 10. Command Register 2 (SPICMD2) Field Descriptions

Bit	Field	Value	Description
15-14	Reserved	0	Reserved.
13-12	CSNUM	0-3h	Device select. Sets the active chip select for the transfer.
		0	Chip select 0 is active.
		1h	Chip select 1 is active.
		2h	Chip select 2 is active.
		3h	Chip select 3 is active.
11-8	Reserved	0	Reserved.
7-3	CLEN	0-1Fh	Character length. Sets the transfer size of the individual transfer elements from 1 to 32 bits.
			The character length is set to CLEN + 1.
			For example, if CLEN = 7, the character length is set to 8 bits.
2	Reserved	0	Reserved.
1-0	CMD	0-3h	Transfer command bits. These bits specify the type of transaction being used.
		0	Reserved.
		1h	Read.
		2h	Write.
		3h	Reserved.

4.4 Status Registers (SPISTAT1 and SPISTAT2)

The status registers (SPISTAT1 and SPISTAT2) contain indicators to allow monitoring of the progression of a frame transfer. The character complete (CC) and frame complete (FC) bits are used as stimulus for generating interrupts. Setting the corresponding interrupt enable bits in the SPICMD1 command register allows these events to generate an interrupt. The CC and FC bits are reset every time (1) SPISTAT1 is read, or (2) a new SPI transfer is initiated via a write of a read or write command to CMD in the SPICMD2 command register.

Figure 19. Status Register 1 (SPISTAT1)

15	3	2	1	0
Reserved		FC	CC	BUSY
R-0		R-0	R-0	R-0

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

Table 11. Status Register 1 (SPISTAT1) Field Descriptions

Bit	Field	Value	Description
15-3	Reserved	0	Reserved
2	FC		Frame complete. This bit is set after all the requested characters have been transferred. This bit is reset when SPISTAT1 is read or when a new SPI transfer is initiated via a write of a read or write command to CMD in SPICMD2.
		0	Transfer is not complete.
		1	All characters have been transferred.
1	CC		Character complete. This bit is set after each character transfer is completed. This bit is reset when SPISTAT1 is read or when a new SPI transfer is initiated via a write of a read or write command to CMD in SPICMD2.
		0	Character transfer is not complete.
		1	Character transfer is complete.
0	BUSY		Busy bit. This bit is set during an active character transfer. Between characters this bit will be cleared to signal that the data registers can be accessed.
		0	Idle, no character transfers in progress.
		1	Active, a character transfer is in progress.

Figure 20. Status Register 2 (SPISTAT2)

15	12	11	0
Reserved		CCNT	
R–0		R-0	

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

Table 12. Status Register 2 (SPISTAT2) Field Descriptions

Bit	Field	Value	Description
15-12	Reserved	0	Reserved.
12-0	CCNT	0-FFFh	Character count. These bits reflect the total number of characters transferred. For example, when CCNT = 64, a total of 64 characters have been transferred. These bits are reset upon completion of a frame. and should only be read after determining CC = 1.

4.5 Data Registers (SPIDAT1 and SPIDAT2)

The data registers (SPIDAT1 and SPIDAT2) are treated as a 32-bit shift register. Data received by the SPI is shifted into the least-significant bit of SPIDAT1 and the contents of both registers are shifted to the left. Similarly, data transferred by the SPI is shifted out of the most-significant bit of SPIDAT2 and the contents of both registers are shifted to the left. This process is illustrated in Figure 9.

The data registers are not cleared between reads or writes. Old data may exist in the register and it is the responsibility of the user to mask off any unneeded data on a read.

Figure 21. Data Register 1 (SPIDAT1)

15		0
DATA		
RW-0		

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

Table 13. Data Register 1 (SPIDAT1) Field Descriptions

Bit	Field	Value	Description
15-0	DATA	0-FFFFh	Low part of data during read and write operations.

Figure 22. Data Register 2 (SPIDAT2)

15		0
DATA		
RW-0		

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

Table 14. Data Register 2 (SPIDAT2) Field Descriptions

Bit	Field	Value	Description
15-0	DATA	0-FFFFh	High part of data during read and write operations.

IMPORTANT NOTICE

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

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

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

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

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

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

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

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

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

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

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

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