SPI_W Serial Peripheral Interface Controller

Summary

This document provides detailed reference information with respect to the SPI Controller device.

Core Reference
CR0153 (v2.0) March 11, 2008

The SPI_W Controller provides an SPI Master interface, enabling a host processor to communicate with a slave SPI peripheral device which resides outside of the physical FPGA device to which the design is targeted, such as the audio DAC and serial Flash RAM devices located on the NanoBoard.

The SPI_W can be used with any of the Wishbone-compliant processors available in Altium Designer.

Features

- Full Duplex - capable of simultaneous transmission and reception
- Serial clock signal configurable for polarity, phase and frequency
- Wishbone-compliant

Available Devices

The SPI_W device can be found in the FPGA Peripherals integrated library (`FPGA Peripherals.IntLib`), located in the `\Library\Fpga` folder of the installation.

Functional Description

Symbol

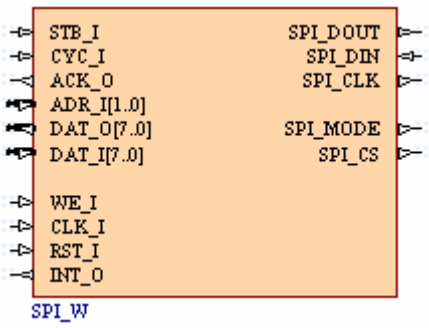

Figure 1. SPI_W Symbol

Pin Description

Table 1. SPI_W pin description

Name	Type	Polarity/Bus size	Description
Control Signals			
CLK_I	I	Rise	External system clock
RST_I	I	High	External system reset. If this cycle is active for at least one full cycle of the external clock signal (CLK_I), all internal registers will be reset
Microcontroller Interface Signals			
STB_I	I	High	Strobe signal. When asserted, indicates the start of a valid Wishbone data transfer cycle
CYC_I	I	High	Cycle signal. When asserted, indicates the start of a valid Wishbone cycle
ACK_O	O	High	Standard Wishbone-device acknowledgement signal. When this signal goes high, the SPI Controller (Wishbone Slave) has finished execution of the requested action and the current bus cycle is terminated
ADR_I	I	2	Standard Wishbone address bus, used to select an internal register of the Wishbone slave device for writing to/reading from.
DAT_O	O	8	Data to be sent to an external Wishbone master device (e.g. host microcontroller).
DAT_I	I	8	Data received from an external Wishbone master device (e.g. host microcontroller).
WE_I	I	Level	Write enable signal. Used to indicate whether the current local bus cycle is a Read or Write cycle. 0 – Read 1 - Write
INT_O	O	High	Interrupt signal. Note that this signal is currently grounded internally
SPI Interface Signals			
SPI_DOUT	O	-	Serial Data Out. This is data sent from the MCU to the target SPI peripheral
SPI_DIN	I	-	Serial Data In. This is data received from the target SPI peripheral, to be sent to the MCU

Name	Type	Polarity/Bus size	Description
SPI_CLK	O	-	Serial Clock. This signal is generated by the Controller and is used to clock data sent from the target SPI device to the Controller on the SPI_DIN line
SPI_MODE	O	Level	Serial Mode. This signal is, in fact, connected directly to bit 2 of the Control/Status register (CSR.2) and can be used for application-specific purposes
SPI_CS	O	Level	Serial Chip Select. This signal is, in fact, connected directly to bit 1 of the Control/Status register (CSR.1) and can be used for application-specific purposes. For example, this signal can be used to enable a target SPI peripheral device for serial communications with the Controller, by setting CSR.1 to '0' and wiring the SPI_CS output to the corresponding chip select pin for the peripheral

Hardware Description

Block Diagram

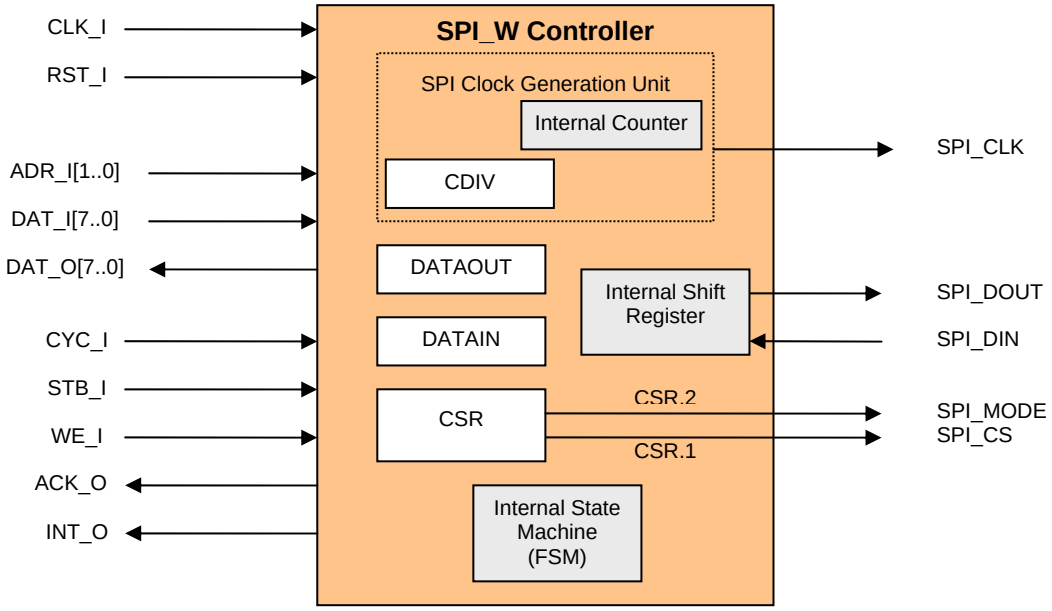

Figure 2. SPI_W block diagram

Internal Registers

The SPI Controller contains four internal registers that are accessible by software, as detailed in the following sections. Unless specified otherwise, registers can be both written to and read from.

Control/Status Register (CSR)

This 8-bit register is used to control aspects of the SPI Controller's operation and to determine the current state of the Controller.

After a reset, this register is initialized to 00h.

Table 2. The CSR register

MSB LSB

busy	-	-	cpha	cpol	mode	cs	txen

Table 3. The CSR register bit functions

Bit	Symbol	Function
CSR.7	busy	Controller Busy Status flag. This bit is set High whenever the Controller has started transmitting data to the target SPI peripheral device (i.e. FSM has left the IDLE state)
CSR.6	-	Not used
CSR.5	-	Not used
CSR.4	cpha	SPI Clock Phase Control bit. Determines the phase of the SPI_CLK signal, in relation to the transmitted data on the SPI_DOUT line: 0 – the first edge of the SPI_CLK signal is generated half an SPI_CLK period after the MSB of the data to be transmitted is made available on the SPI_DOUT line. Data will be latched on the leading edge and changed on the trailing edge 1 – the first edge of the SPI_CLK signal is generated in conjunction with the MSB of the data to be transmitted being made available on the SPI_DOUT line. Data will be changed on the leading edge and latched on the trailing edge

Bit	Symbol	Function
CSR.3	cpol	SPI Clock Polarity Control bit. Determines the idle (inactive) state for the SPI_CLK signal: 0 – SPI_CLK signal is inactive Low 1 – SPI_CLK signal is inactive High
CSR.2	mode	SPI_MODE Control bit. This bit is directly linked to the Controller's SPI_MODE output pin and can be used to control this signal as required
CSR.1	cs	SPI_CS Control bit. This bit is directly linked to the Controller's SPI_CS output pin and can be used to control this signal as required. When this signal is connected to the corresponding chip select input of a peripheral SPI device, taking the signal Low will enable that device for serial communications with the Controller. When the target SPI device is not selected (i.e. CSR.1 is 1), it must tristate its data output
CSR.0	txen	Transfer Enable Control bit. Used to control state machine operation: 0 – Initiate serial communications transfer 1 – Inhibit serial communications

Note: Bits 6 and 5 are ignored when writing to the register and return '0' when read. Bit 7 is Read-only.

Clock Divider Register (CDIV)

This 8-bit register is used to store a divisor for use in generation of the SPI_CLK signal, based on the external clock signal (CLK_I). As part of the SPI Clock Generation Unit, an internal counter is used to count up to the value written to the CDIV register. The next edge of SPI_CLK will only be generated when the internal counter reaches this divisor value.

The value written to the Clock Divider register can be anywhere in the valid range 0 to 2^8 – 1.

After a reset, this register is initialized to 00h.

Table 4. The CDIV register

MSB LSB

div7	div6	div5	div4	div3	div2	div1	div0

Table 5. The CDIV register bit functions

Bit	Symbol	Function
CDIV.7	div7	Clock Divisor bit 7
CDIV.6	div6	Clock Divisor bit 6
CDIV.5	div5	Clock Divisor bit 5
CDIV.4	div4	Clock Divisor bit 4
CDIV.3	div3	Clock Divisor bit 3
CDIV.2	div2	Clock Divisor bit 2
CDIV.1	div1	Clock Divisor bit 1
CDIV.0	div0	Clock Divisor bit 0

Parallel to Serial Data Register (DATAOUT)

This 8-bit write-only register is used to store the data to be transmitted to the target SPI peripheral device.

The value written to this register can be anywhere in the valid range 0 to 2^8 – 1.

After a reset, this register is initialized to 00h.

Table 6. The DATAOUT register

MSB LSB

data7	data6	data5	data4	data3	data2	data1	data0

Table 7. The DATAOUT register bit functions

Bit	Symbol	Function
DATAOUT.7	data7	Transmit data bit 7
DATAOUT.6	data6	Transmit data bit 6
DATAOUT.5	data5	Transmit data bit 5
DATAOUT.4	data4	Transmit data bit 4
DATAOUT.3	data3	Transmit data bit 3
DATAOUT.2	data2	Transmit data bit 2
DATAOUT.1	data1	Transmit data bit 1
DATAOUT.0	data0	Transmit data bit 0

Serial to Parallel Data Register (DATAIN)

This 8-bit read-only register is used to store the data received from the target SPI peripheral device.

The value in this register can be anywhere in the valid range 0 to $2^8 - 1$.

After a reset, this register is initialized to 00h.

Table 8. The DATAIN register

MSB LSB

data7	data6	data5	data4	data3	data2	data1	data0

Table 9. The DATAIN register bit functions

Bit	Symbol	Function
DATAIN.7	data7	Received data bit 7
DATAIN.6	data6	Received data bit 6
DATAIN.5	data5	Received data bit 5
DATAIN.4	data4	Received data bit 4
DATAIN.3	data3	Received data bit 3
DATAIN.2	data2	Received data bit 2
DATAIN.1	data1	Received data bit 1
DATAIN.0	data0	Received data bit 0

Host to Controller Communications

Communications between the host microcontroller and the SPI Controller is carried out over the standard Wishbone bus.

The host microcontroller can read/write (as applicable) any of the SPI_W's four software-accessible internal registers. Selection of a particular register is achieved by supplying the 2-bit binary ID address code of the register. This code is sent to the SPI_W and appears at its ADR_I input. Table 10 shows the address IDs associated with each of the addressable registers.

Table 10. Internal register address IDs

Register	Register Address ID
DATAOUT	00
DATAIN	00
CSR	01
CDIV	10

Note: DATAOUT and DATAIN registers use the same address. Provided you are performing a write (WE_I input High), you will access the DATAOUT register. When performing a read (WE_I input Low), you will access the DATAIN register.

Writing to an Internal Register

Data is written from the host microcontroller to an internal register in the SPI_W, in accordance with the standard Wishbone data transfer handshaking protocol. The write operation occurs on the rising edge of the CLK_I input and can be summarized as follows:

- The host presents the 2-bit address ID for the register to be written on its ADR_O output and a valid byte of data on its DAT_O output. It then asserts its WE_O signal, to specify a Write cycle.
- The SPI_W receives the address ID on its ADR_I input and prepares to receive data into the selected register.
- The host asserts its STB_O and CYC_O outputs, indicating that the transfer is to begin. The SPI_W, which monitors its STB_I and CYC_I inputs on each rising edge of the CLK_I signal, reacts to this assertion by latching the byte of data appearing at its DAT_I input, into the specified target register, and asserting its ACK_O signal – to indicate to the host that the data has been received.
- The host, which monitors its ACK_I input on each rising edge of the CLK_I signal, responds by negating the STB_O and CYC_O signals. At the same time, the SPI_W negates the ACK_O signal and the data transfer cycle is naturally terminated.

Reading from an Internal Register

Data is read from one of the SPI_W's internal registers, in accordance with the standard Wishbone data transfer handshaking protocol. This data transfer cycle can be summarized as follows:

- The host presents the 2-bit address ID for the register to be read on its ADR_O output. It then negates its WE_O signal, to specify a Read cycle
- The SPI_W receives the address ID on its ADR_I input and prepares to transmit data from the selected register
- The host asserts its STB_O and CYC_O outputs, indicating that the transfer is to begin. The SPI_W, which monitors its STB_I and CYC_I inputs on each rising edge of the CLK_I signal, reacts to this assertion by presenting the valid byte of data at its DAT_O output and asserting its ACK_O signal – to indicate to the host that valid data is present
- The host, which monitors its ACK_I input on each rising edge of the CLK_I signal, responds by latching the byte of data appearing at its DATA_I input and negating the STB_O and CYC_O signals. At the same time, the SPI_W negates the ACK_O signal and the data transfer cycle is naturally terminated.

Interfacing to a 32-bit Processor

Figure 3 shows an example of how an SPI_W Controller can be wired into a design that uses a TSK3000A 32-bit RISC processor. The target SPI device in this example is the Audio Codec located on the NanoBoard.

A configurable Wishbone Interconnect device (WB_INTERCON) is used to simplify connection and also handle the word addressing – taking the 24-bit address line from the processor and mapping it to the 2-bit address line used to drive the peripheral.

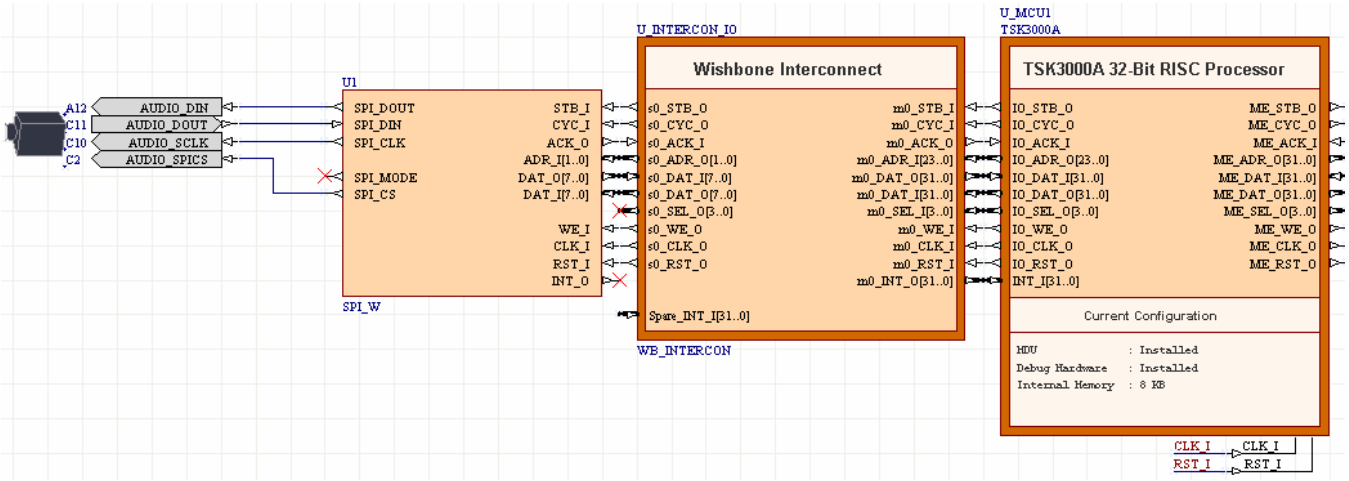

Figure 3. Example interfacing between a 32-bit processor (TSK3000A) and an SPI_W Controller

Internal SPI registers are accessed directly by adding the 2-bit address for the required register to the 24-bit base address of the SPI_W device. This base address is specified as part of the peripheral's definition when adding it as a slave to the Wishbone Interconnect. For example, if the base address entered for the device is 100000h (mapping it to address FF10_0000h in the processor's address space), and you want to write to the Command/Status register (CSR) with binary address 01 (or 1h), the value entered on the processor's 24-bit IO_ADR_O line would be:

`100000h + 1h = 100001h`.

For further information on the Wishbone Interconnect peripheral, refer to the *WB_INTERCON Configurable Wishbone Interconnect* core reference.

Operation

The following steps outline the basic procedure in order to initiate serial communications with the target SPI peripheral device:

Initialization

You will need to re-initialize the Controller after each external reset. To initialize the Controller:

- Write to the CSR register, setting the following bits as required:
  - SPI_CLK polarity and phase – CSR.3 and CSR.4 respectively
  - SPI_MODE – CSR.2
  - SPI_CS – CSR.1(by default this bit will be cleared after a reset is issued to the register and therefore in the correct state for enabling the target SPI device)
  - txen – CSR.0 (by default this bit will be cleared after a reset and therefore in the correct state to enable transmission upon reception of data in the Parallel to Serial Data register (DATAOUT). Should you wish to inhibit transmission, set this bit to 1)
- Write to the CDIV register with the required value for division of the CLK_I signal, to achieve the desired SPI_CLK frequency.

Transmission

In order to start the Controller's internal state machine – and hence transmission of data to/reception of data from the target SPI device – simply write the data to be transmitted into the Parallel to Serial Data register (DATAOUT) and ensure that the txen bit in the Control/Status register is '0'.

The Controller's state machine will generate the serial clock (SPI_CLK) and manage the data flow as follows:

- The byte of data to be transmitted will be copied into an internal shift register.
- The MSB of this data will be shifted out onto the SPI_DOUT line (to the target SPI device). As the state machine is no longer in the IDLE state, the busy flag is set (High) in the Control/Status register (CSR.7), indicating that the Controller is transmitting
- At the same time, a bit of data is received from the target SPI device on the Controller's SPI_DIN line and shifted into bit 0 of the internal shift register.
- An internal bit counter keeps track of the transmission, which proceeds until all 8 bits of the data to be transmitted have been sent. At this time, the internal shift register holds an 8-bit data value received from the SPI device.
- The valid byte of data is loaded from the internal shift register into the Serial to Parallel Data register (DATAIN), ready to be read by the host processor.

Revision History

Date	Version No.	Revision
13-Dec-2004	1.0	New release
27-May-2005	1.1	Updated for Altium Designer SP4
10-Oct-2005	1.2	Correction made for busy bit of CSR register. This is bit 7 and not bit 6. DAC register renamed DATAOUT ADC register renamed DATAIN
12-Dec-2005	1.3	Path references updated for Altium Designer 6
20-Sep-2006	1.4	Updated for Altium Designer 6.6.
11-Mar-2008	2.0	Updated for Altium Designer Summer 08