

Overview

Serial Peripheral Interface(SPI) is a popular 4 wire serial interface that is adapted in most low speed systems. It is a Master - Slave system, makes use of following 4 lines - MOSI (Master Out Slave In), MISO (Master In Slave Out), SCLK (Serial Clock), SS (Slave Select).

This design example illustrates the implementation of SPI Master on SiliconBlue's low power iCE65 FPGAs. This SPI Master design supports all modes of CPOL and CPHA - 00, 01, 10, 11. SPI Master data sampling from MISO line depends upon current bit count and CPOL/CPHA mode. A shift register on receive data path converts serial to parallel conversion. Similar parallel to serial conversion takes place in transmit data path.

Features

- 4 SPI Slave select lines based on address
- Provision for easy integration of any processor interface.
- Run time configurable features
 - CPOL, CPHA Modes - 00, 01, 10, 11
 - Configurable SCLK period
 - Configurable setup, hold and time interval between two SPI transactions
- Parameterizable data width
- User configurable Read and Write Data FIFO
- IP-XACT version 1.2 compliant

Features not supported

- Multi-master mode

Resource Utilization

Table 1: Resource Utilization

LUTs	Registers	Memory	I/Os	GBs
343	194	2	0	0

Note: Resource Utilization is based on iCEcube 2010.12.14671 release.

System Block Diagram

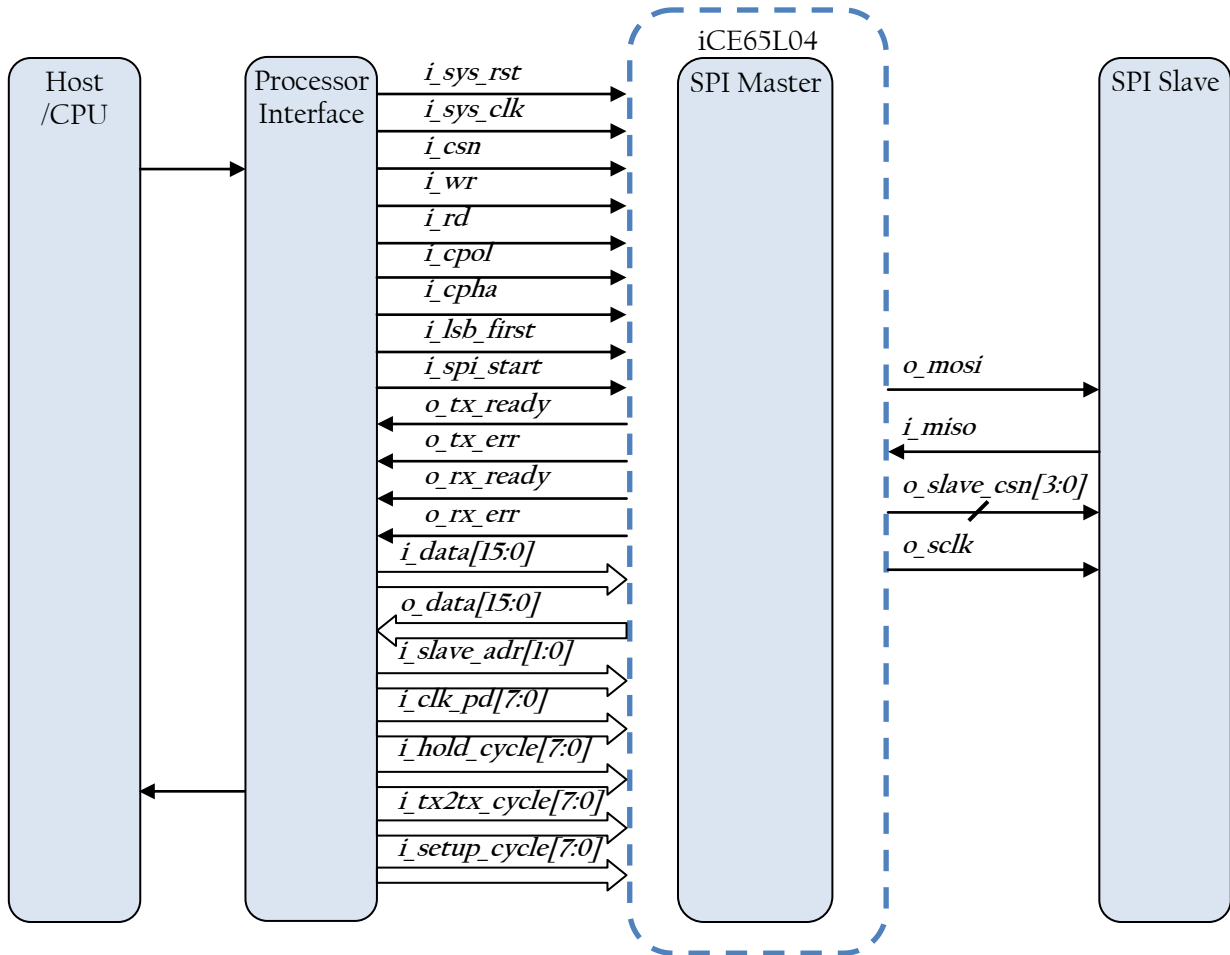


Figure 1: System Block Diagram

Functional Block Diagram

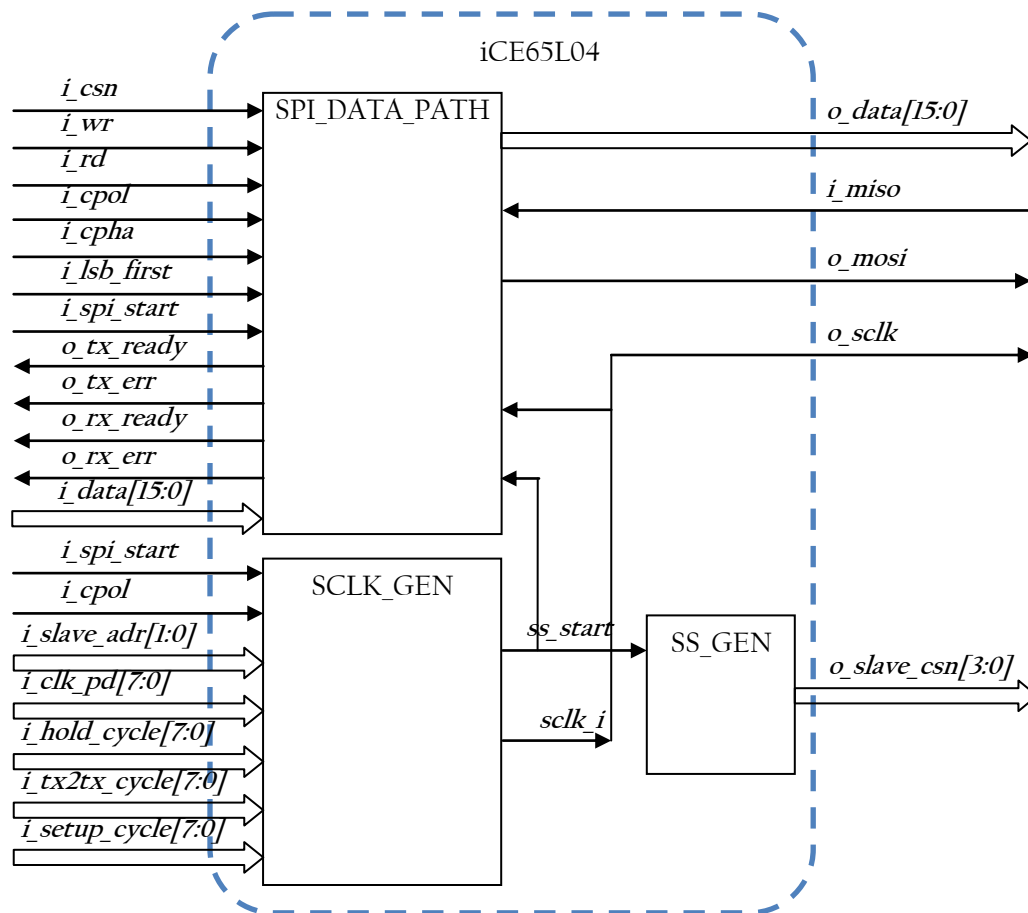


Figure 2: Functional Block Diagram

Design Interface

Table 2: Pin Description

Signal Name	Pin Type	Signal Description
i_sys_rst	Input	Asynchronous Active Low reset.
i_sys_clk	Input	System clock.
i_csn	Input	Active low chip select.
i_data[15:0]	Input	Input data from processor interface
i_wr	Input	Active low write enable
i_rd	Input	Active high read enable
o_data[15:0]	Output	Output data to the processor interface
o_tx_ready	Output	Transmitter ready - Indicates another data can be written.
o_rx_ready	Output	Receiver ready - Indicates another data can be read.
o_tx_error	Output	Indicates error in transmission of data.
o_rx_error	Output	Indicates error in received data.
i_cpol	Input	Polarity of the clock
i_cpha	Input	Phase of the clock
i_lsb_first	Input	LSB sent first when '1' and MSB goes first when '0'.
i_slave_addr[1:0]	Input	Slave address to select a slave device.
i_spi_start	Input	Start SPI Master Transactions
i_setup_cycles	Input	SPIM setup time in terms of i_sys_clk
i_hold_cycles	Input	SPIM hold time in terms of i_sys_clk
i_tx2tx_cycles	Input	SPIM interval between data transactions in terms of i_sys_clk
i_clk_period	Input	SCLK clock period in terms of i_sys_clk
o_mosi	Output	Serial data output from Master
i_miso	Input	Serial data input to the Master
o_slave_csn[3:0]	Output	Slave select from Master
o_sclk	Output	Serial Clock is generated by the Master.

Configurable Parameters

- FIFO_REQ – This Boolean parameter configures the FIFO usage.
 - If “TRUE”, two 16x16 FIFOs are inferred
 - If “FALSE”, no FIFOs are inferred
- DATA_SIZE – This parameter configures the data width of the SPI transaction. It can take 2 values.
 - 16 (default value)
 - 8 (In this case the higher 8 bits of the data in the FIFO are filled with zeroes)

Register Map

This design does not have any user accessible registers or memory.

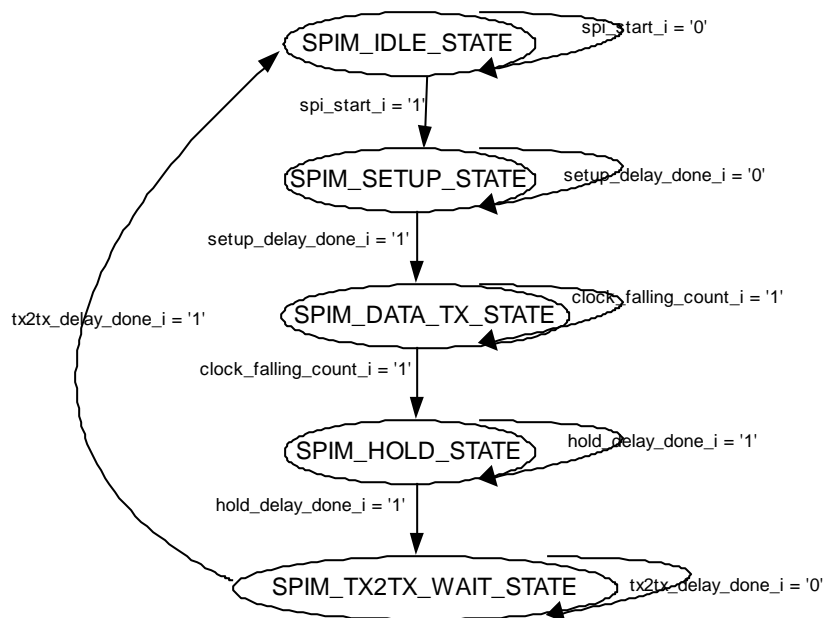
Design Details

To begin a communication, the master first configures the clock, using a frequency less than or equal to the maximum frequency the slave device supports. Such frequencies are commonly in the range of 1-70 MHz.

The master then pulls the slave select low of the desired slave based on slave address and starts issuing the clock pulses.

SCLK Generation Module :

SCLK generation module generates SPI Master clock, SCLK based on input control signals - clock period, setup, hold time and SPI transaction to transaction interval, all expressed in terms of system clock cycle counts. This module comprises of an simple FSM which is enables clock generation, holding SCLK at LOW or HIGH based on CPOL = 0 or CPOL = 1. It is also responsible for generating chip select for SPI Slave adhering to setup and hold time as well as wait time between two transactions.



SPI Data Path

SPI Data Path samples MISO line and drives MOSI line based on CPOL/CPHA modes as follows:

- At CPOL=0, the base value of the clock is zero
 - For CPHA=0, data is read on clock's rising edge and data is changed on a falling edge.
 - For CPHA=1, data is read on clock's falling edge and data is changed on a rising edge.
- At CPOL=1, the base value of the clock is one (inversion of CPOL=0)
 - For CPHA=0, data is read on clock's falling edge and data is changed on a rising edge.
 - For CPHA=1, data is read on clock's rising edge and data is changed on a falling edge.

SPI Master module supports 4 modes, viz., {CPOL, CPHA} = {00, 01, 10, 11}. This module is responsible for sampling miso line and shift the data based on the current bit count of SPI transaction as well as CPOL and CPHA modes. This module is also responsible for pushing bits of data on MOSI line from a shift register based on bit count and CPOL/CPHA modes.

During each SPI clock cycle, a full duplex data transmission occurs:

- the master sends a bit on the MOSI line; the slave reads it from that same line
- the slave sends a bit on the MISO line; the master reads it from that same line

Not all transmissions require all four of these operations to be meaningful but they do happen.

Write FIFO :

This 16x16 FIFO is compile time configurable which stores data written by the processor. Before the SPI transactions take place, the data to be written to the slave device is written to the FIFO by the processor. SPI Master then reads data from this FIFO and sends it to the slave.

Read FIFO :

This 16x16 FIFO is compile time configurable which stores data written by the slave device. The processor can read data from this FIFO after a SPI transaction.

Initialization Conditions

No user specific initialization conditions, except that the `i_sys_rst` must be held low initially to bring-up the design in a correct operating state.

Timing Diagram

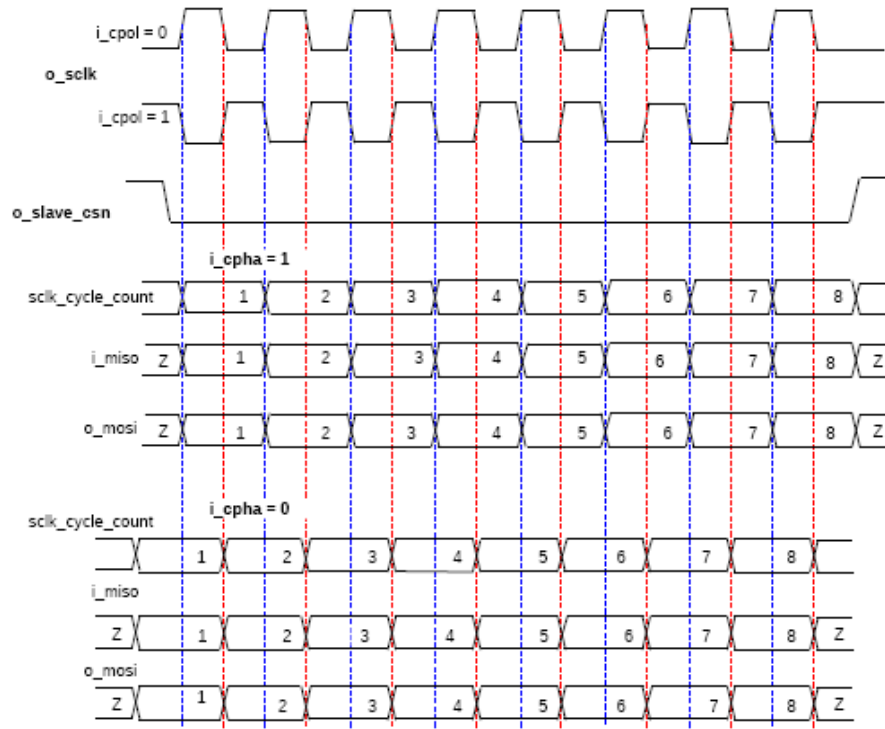
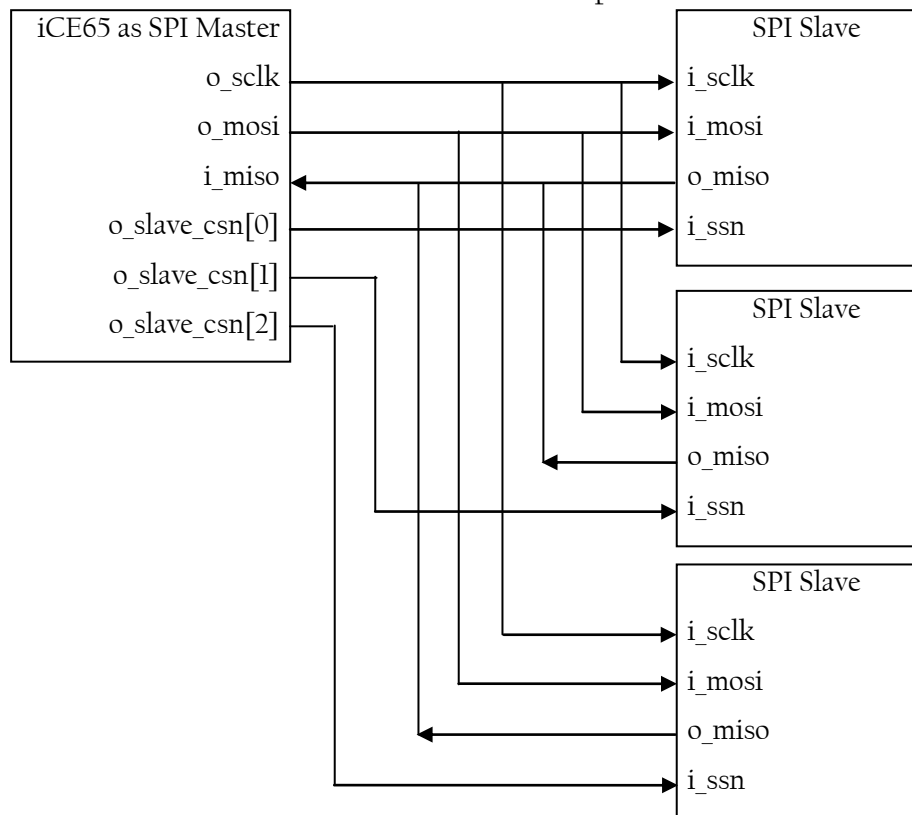


Figure 3: SPI Timing diagram

Usage Examples

Example #1

Figure below illustrates the usage of SPI Master to communicate with three slaves. The Master generates appropriate slave select lines based on slave address and enables that particular SPI Slave.



System Designer Flow

SPI Master is compatible with System Designer/IP-XACT 1.2. Following parameter can be configured in the System Designer environment.

- FIFO_REQ – This Boolean parameter configures the FIFO usage.
 - If “TRUE”, two 16x16 FIFOs are inferred
 - If “FALSE”, no FIFOs are inferred
- DATA_SIZE – This parameter configures the data width of the SPI transaction. It can take 2 values.
 - 16 (default value)
 - 8 (In this case the higher 8 bits of the data in the FIFO are filled with zeroes)

The System Designer flow is as follows,

1. Launch the System Designer from Synplify Pro using menu 'Import -> Launch System Designer'.
2. Create a new project (open an existing old project, as necessary) and import the IP-XACT XML file
3. Drag and place the component from the 'Library' pane to the 'Design' pane
4. To change the parameters, FIFO_REQ and DATA_SIZE, right-click on the component instance, and click on “Open Configuration”. Go to “Edit Instance Parameters” tab, change the “DATA_SIZE” and “FIFO_REQ” parameters. Click on the “Apply” button, and then close it.
5. Click on the “Generate Files” button, which generates the necessary files required for synthesis and simulation.

- Go to Synplify Pro and click on the “Run” button to synthesize the System Designer generated files. Synplify Pro generates all the necessary files for P&R in iCECube.

References

The following references were used in the creation of this design:

- SiliconBlue Technologies, Inc. “[iCE65 Ultra Low-Power mobileFPGA Family](#)” datasheet (26-MAY-2010).
- Wikipedia : http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus

Revision History

Version	Date	Description
1.0	16-SEP-2010	Initial Draft Document
1.1	08-DEC-2010	IP-XACT format Update

Disclaimer

Copyright © 2007–2009 by SiliconBlue Technologies LTD. All rights reserved. SiliconBlue is a registered trademark of SiliconBlue Technologies LTD in the United States. Specific device designations, and all other words and logos that are identified as trademarks are, unless noted otherwise, the trademarks of SiliconBlue Technologies LTD. All other product or service names are the property of their respective holders. SiliconBlue products are protected under numerous United States and foreign patents and pending applications, maskwork rights, and copyrights. SiliconBlue warrants performance of its semiconductor products to current specifications in accordance with SiliconBlue's standard warranty, but reserves the right to make changes to any products and services at any time without notice. SiliconBlue assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by SiliconBlue Technologies LTD. SiliconBlue customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.



SiliconBlue Technologies Corporation

3255 Scott Blvd.,
Building 7, Suite 101
Santa Clara, CA 95054

Tel: 408-727-6101
Fax: 408-727-6085

www.SiliconBlueTech.com