SPI Slave Port Expander

April 2013 **Reference Design RD1168**

Introduction

The Serial Peripheral Interface (SPI) bus provides an industry standard interface between processors and other devices. This reference design provides a programmable solution for serial expansion of processor's general purpose I/O ports. It uses a SPI as the interface between the processor and GPIOs.

Port expanders provide the system host the capability and advantage of expanding its ports thereby reducing pins on the host system. This port expander uses the standard 4 wire SPI bus interface that are available on most host systems to expand into 32 serial input and 32 serial output ports.

The design is implemented in VHDL. The Lattice iCEcube2™ Place and Route tool integrated with the Synopsys Synplify Pro® synthesis tool is used for the implementation of the design. The design can be targeted to other iCE40™ FPGA product family devices.

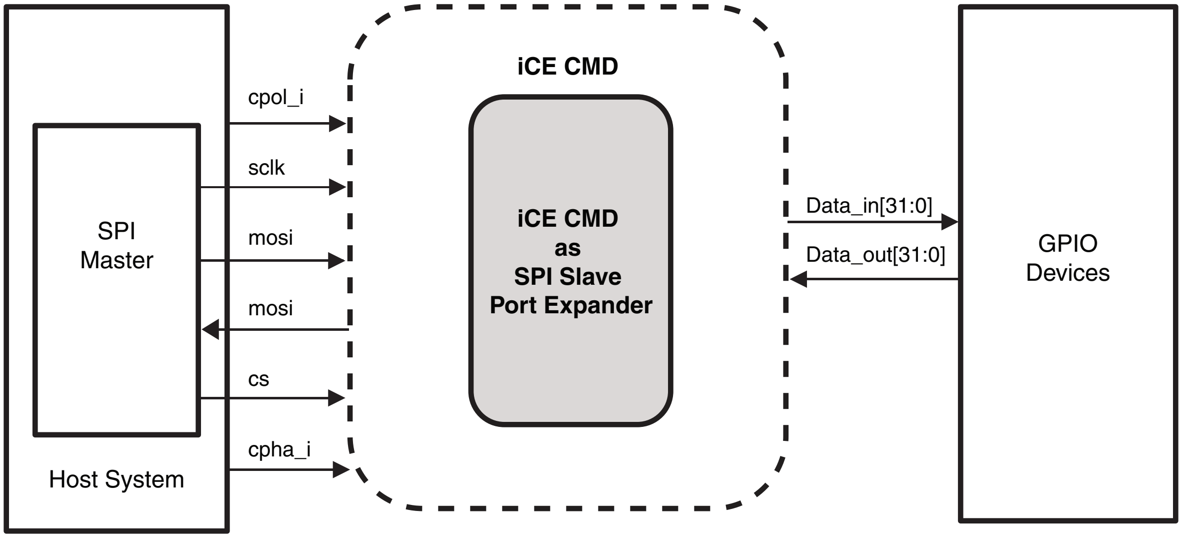
Figure 1 shows the System Block Diagram for the SPI Slave Port Expander.

Features

- Expansion up to 32 serial input and output ports
- All modes of CPOL and CPHA (00/01/10/11)
- Compile time configurable data widths (8, 16 and 32)

System Block Diagram

Figure 1. System Block Diagram



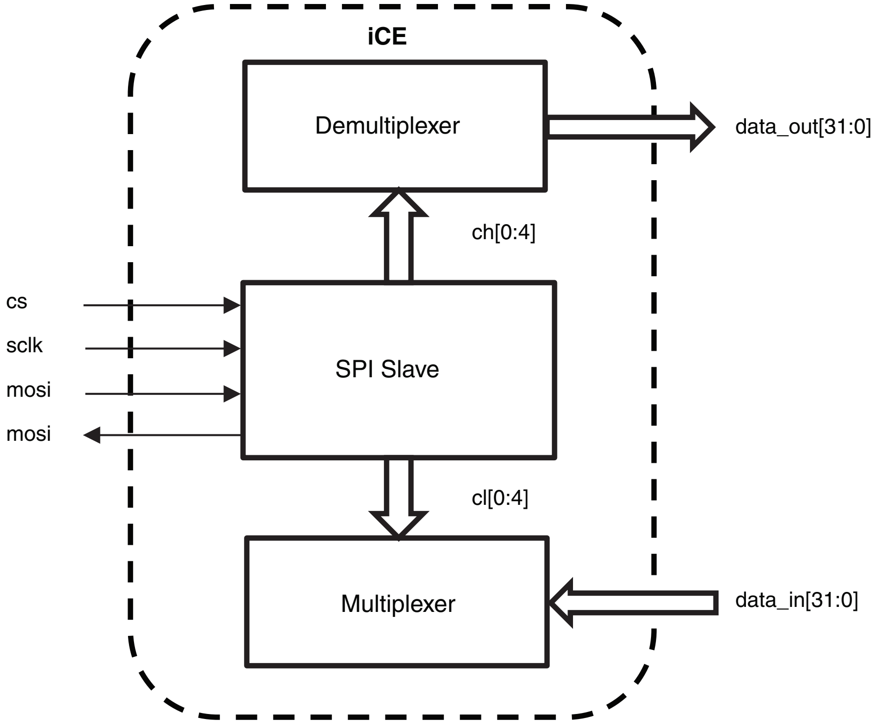
Signal Description

Table 1. Signal Description

Signal	Width	Type	Description
cs	1	Input	Active high chip select(Acts like reset)
Data_in	32	Input	32 expnaded input ports
Data_out	32	Output	32 expanded output ports
mosi	1	Input	Slave input from master
miso	1	Output	Slave output to master
sclk	1	Input	Serial clock from master
cpol_i	1	Input	Clock Polarity
cpha_i	1	Input	Clock Phase

Functional Description

Figure 2. Functional Block Diagram



SPITM, or Serial Peripheral Interface is a popular 4 wire serial interface that is adapted in most systems. It is a Master - Slave system using 4 lines (3 common and 1 exclusive) as follows:

- MOSI (Master Out, Slave In)
- MISO (Master In, Slave Out)
- SCLK (Serial Clock)
- CS (Chip select or Slave select)

The main features of the SPI interface, unlike many other serial interfaces, are:

- It allows for a full duplex serial communication that is possible due to the presence of exclusive lines for data in both directions.

- The availability of another exclusive line used to select a particular Slave, the CS line, reduces the overhead of address decoding in a multi-Slave environment.

These two features combine to add great speeds on the SPI bus (with clock speeds up to 70 MHz).

This design example of the SPI Slave Port Expander is implemented with configuring the iCE FPGA as an SPI Slave. The Slave connects to the system SPI bus and expands to 32 serial inputs and 32 serial outputs.

There are 3 data formats supported in this IP – 32-bit, 16-bit and 8-bit (shown in Table 2, Table 3 and Table 4). By default, it has 16-bit data format. The SPI slave port expander is implemented in the iCE40 FPGA by decoding appropriate bits in the SPI's MOSI data frame to obtain the address of the input and output port the Slave has to expand to. Once this address is decoded in the Slave, the serial data that the Master sends to the Slave on its MOSI line is routed to the appropriate output port on the Slave. At the same time the serial input data that the Slave received on a selected input port is routed on to the MISO line. This selection of the input port is also by means of an address in the SPI Master's MOSI data frame (illustrated in the below tables). This port expander is designed for a 32 bit SPI frame, with the MSB being sent first.

Table 2. 32-Bit Data Frame Format

Input Address[31:27]	Output Address[26:22]	Data[21:0]

Table 3. 16-Bit Data Frame Format

Input Address[31:28]	Output Address[27:24]	Data[23:0]

Table 4. 8-Bit Data Frame Format

Input Address[31:29]	Output Address[28:26]	Data[25:0]

Initialization Conditions

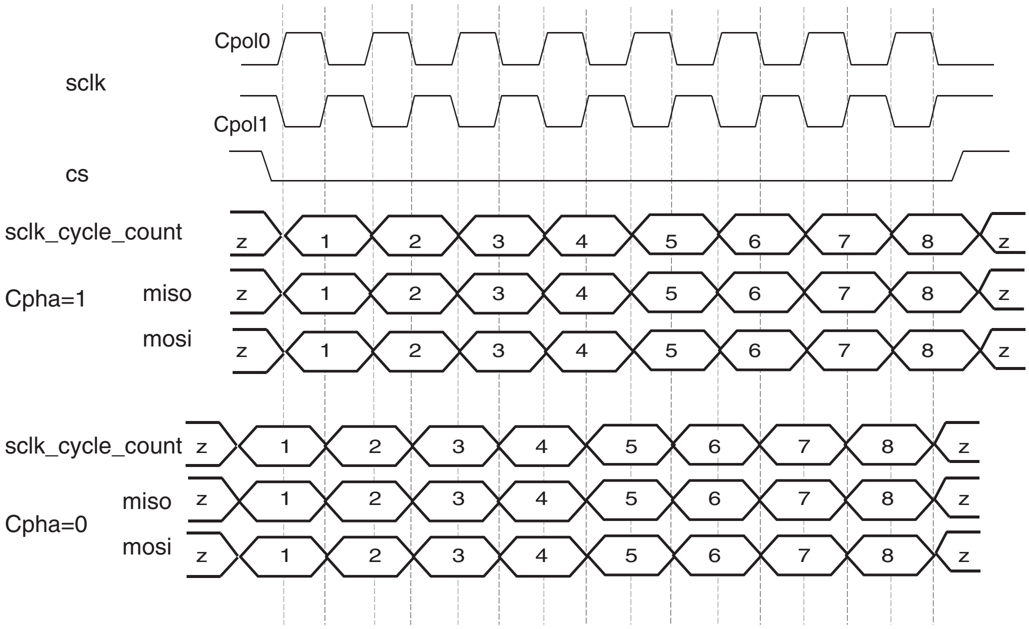
cs acts as Asynchronous active low reset. At reset, data_in and data_out are initialized to x”FFFFFFFF”.

Configurable Parameters

GPIO_WIDTH : This parameter controls the number of expandable input and output ports. Supported values are 8, 16 (default) and 32

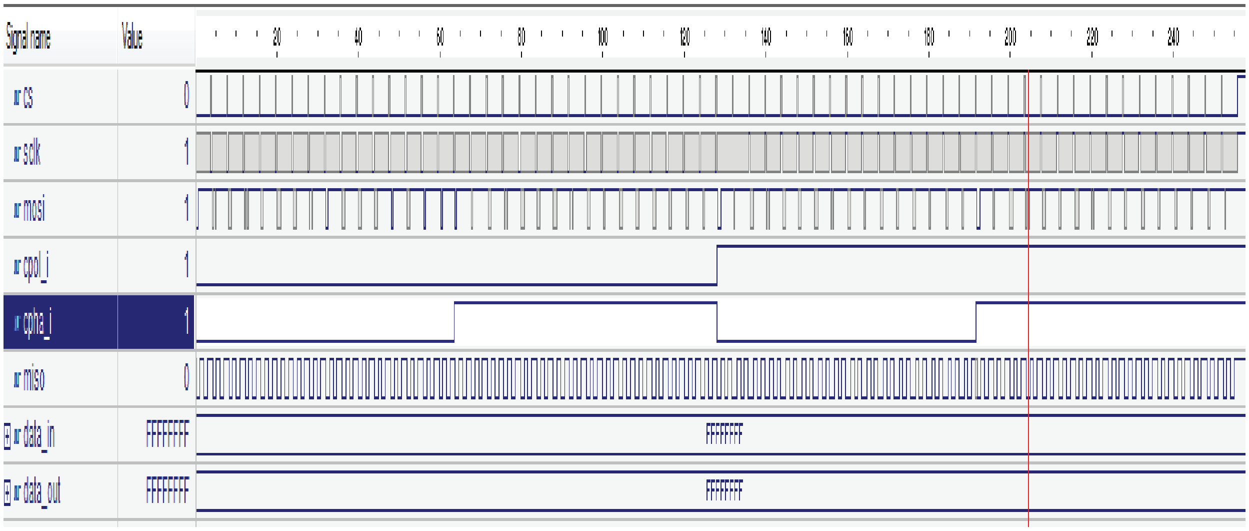
Timing Diagram

Figure 3. Timing Diagram



Simulation Waveforms

Figure 4. Simulation Waveforms



Operation Sequence

1. Make cs = '0', cpol_i = '0' and cpha_i = '0'
2. Write the 32-bit SPI Data frame (MSB first) through the MOSI line with input and output port address as 0, and data as all 1s i.e. “00000000111111111111111111111111”
3. Check if the data at the output port (data_out) is x”FFFFFFFF”
4. Repeat the above steps by changing the input and output address, the polarities of cpol_i and cpha_i, as well as changing the data format to 8-bit and 32-bit

Implementation

This design is implemented in VHDL. When using this design in a different device, density, speed or grade, performance and utilization may vary.

Table 5. Performance and Resource Utilization

Family	Language	Utilization (LUTs)	f_{MAX} (MHz)	I/Os	Architectural Resources
iCE40[1]	VHDL	84	>50	53	(21/160)PLBs

[1] Performance and utilization characteristics are generated using iCE40LP1K-CM121 with iCEcube2 design software.

References

- iCE40 Family Handbook

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
April 2013	01.0	Initial release.