International Journal of VLSI System Design and Communication Systems

ISSN 2322-0929
Vol.02,Issue.03,
May-2014,
Pages:0225-0228

ww.semargroup.org
www.ijvdcs.org

Implementation of Low Power SPI Protocol with Clock Domain Crossing

MANISH KUNDU[1], ABHIJEET KUMAR[2]

[1]PG Scholar, Dept of ECE, M.M University, Mullana, India, Email: manishkundu@gmail.com.
[2]Asst Prof, Dept of ECE, M.M University, Mullana, India, Email: abhijeet.kumar@mmumullana.org.

Abstract: The SPI serial bus was originally developed by Motorola. Today, it is one of the most common communication buses used by IC (Integrated Circuit) manufactures. In this paper, an approach is proposed for the design and implementation of a low power serial peripheral interface with clock domain crossing using Field Programmable Gate Array (FPGA). The focus of this research was to develop an effective low power Serial Peripheral Interface which is a full-duplex, three-wire synchronous transfer, serial data link that enables communication between a host (processor/controller) and peripherals in a SoC (System on chip) application. Low power techniques are being implemented at system level. Here, approaches related to front-end HDL based design styles, which can reduce power consumption, have been mentioned. The cross-clock domain crossing (CDC) signals pose a unique and challenging issue for verification. Traditional functional Simulation is inadequate to verify clock domain crossings. Whereas static timing analysis (STA) is an integral part of the timing closure solution, little attention has been paid to addressing proper clock domain implementation.

Keywords: Serial Peripheral Interface (SPI), Hardware Description Language (HDL), Serial Clock (SCK), Clock Domain Crossing (CDC).

I. INTRODUCTION

In communication protocols, SPI is considered as little Communication protocol compared to other protocols. Le SPI, is used for communications between integrated circuits for high/low/medium data transfer speed with on-board peripherals. SPI is a serial bus standard developed by Motorola. Its silicon products are fabricated by various manufacturers. SPI interfaces are available on famous communication processors and microcontrollers. SPI is a serial interface protocol which stands for Serial peripheral Interface, it has high transmission speed, simple to use and little pins advantages. Usually, the devices which based on SPI protocol are divided into master-device and slave device for transmitting the data as well as receiving the data.

The chip select signal and clock signal will be generated by the master-device only. The Communication process will be initiated by the master-device only. The standard SPI communication is a single-master communication, that means all the communications are only have one master-device. The power factor is also a major issue while designing communication protocols. With ongoing technologies, there is a requirement for reducing power. Power consumption can be minimized at various levels like system level, RTL level, circuit level, physical design and technology level. There is considerable effort put in, throughout the world, to reduce power consumption in VLSI circuits. Apart this while designing synchronous serial transfer protocols cross-clock domain crossing issues generally comes into limelight. These issues mainly occur due to bad clock routing, clock phase and clock frequency relationships.

II. SPI PROTOCOL

The Serial Peripheral Interface (SPI) is a full-duplex (Signals carrying data in both directions simultaneously), synchronous, serial communication link that is standard across many microprocessors, microcontrollers, and peripherals. It provides Communication between microprocessors and peripherals and/or inter-processor communication. The SPI system is flexible enough to interface directly with numerous commercially available peripherals. The SPI bus consists of four wires, Serial Clock (SCK), Master out Slave in (MOSI), Master in Slave out (MISO), and Slave Select/Chip Select (SS), which carry information between the devices connected to the bus.

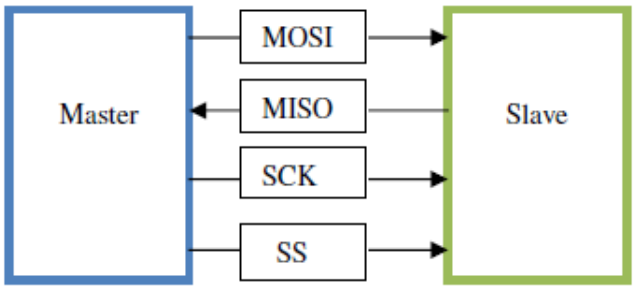

Fig.1. Master/Slave block diagram of SPI.

SPI Master is driving the SCK line and regulates the flow of data bits. The master can transmit data at a variety of frequencies. The master selects one of four different modes for SCK. Data is shifted on one edge of SCK and is sampled on the opposite edge when the data is stable. The communication will be initiated by the master all the time. The SS pin must be low to select a slave. This signal can come from any pin on the master, including its SS pin when it is configured as an output.

III. SPI DATA TRANSFER

The clock phase and polarity must be configured for SPI data transfer. Generally there are four modes of operation mode0 – mode3. These four modes directly depend on the Clock Polarity (CPOL) and Clock Phase (CPHA). SCK will always be HIGH in normal conditions. CPOL is used for selecting the active edge of SCK whereas CPHA is used for sampling of data at respective edge. On the whole clock plays an important role in this communication.

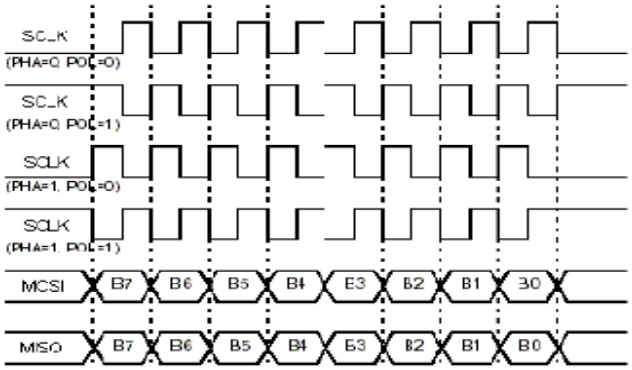

Fig.2. Timing Diagram of Data Transfer with different modes [3]

The data transfer clock typically operates at rates from 1.0 MHz to several MHz's depending upon the system clock.

IV. LOW POWER DESIGN

Increasing clock frequency and a continuous increase in The number of transistors on chip has made implementing low power techniques in the design compulsory. The different low power techniques used in our designs are:

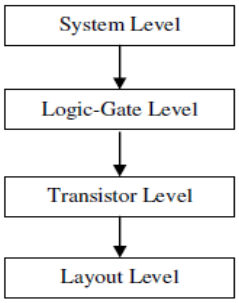

Fig.3. Levels of Optimization Techniques.

State machine encoding, Resource sharing (no redundant logic elements), Control over counters, Minimizing data transitions on bus, Clock gating, reduced supply voltages, Resizing of the transistor, Allow synthesis optimization. Generally low power optimization techniques can be achieved at multiple levels of abstraction.

V. CLOCK DOMAIN CROSSING

Clocks which have a known phase and frequency relationship between them are known as synchronous clocks. These are essentially the clocks originating from the same clock-root. A clock crossing between such clocks is known as a synchronous clock domain crossing. If the sources of potential errors are not addressed and verified early on, designs can end up with Functional errors that are only detected late in the design cycle or even worse, during post-silicon verification. CDC issues can be verified at device level implementation. CDC is also prone to glitch problem. A clock domain crossing occurs whenever data is Transferred from a flop driven by one clock to a flop driven by another clock.

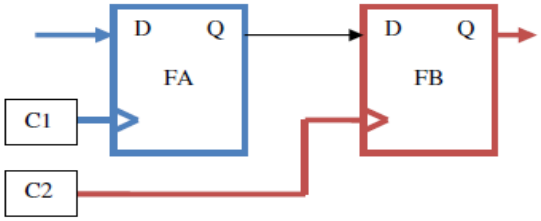

Fig.4. Clock Domain Crossing

VI. RESULTS

The whole design code is implemented in Verilog 2001 HDL. The design is simulated and synthesized in Xilinx ISE 9.2i (Complete package tool) and ModelSim 6.3 (Simulation Tool). The low power techniques are applied at the RTL level using different styles of coding. The Synthesis design is done at technology level as well as RTL level with Timing report analysis. The Design is being implemented on FPGA Xilinx Spartan XC3S50. During Implementation all the clock domain crossing issues can be observed.

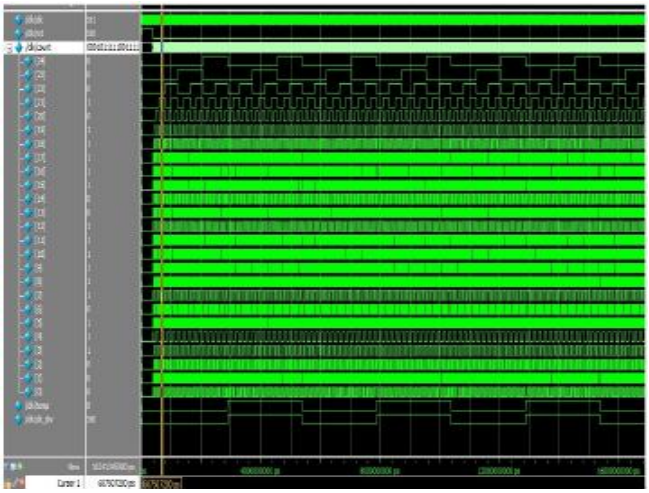

Fig.5. Simulation of Clock Generator.

The clock is divided according to the requirement of the User. Likewise the above mentioned diagram is the simulation result of divide by 2 frequencies. The system clock is fed to a T- Flip flop Circuit and from the output of T- Flip flop a divide by 2 frequency clock will be achieved. The FIFO (First in First out) design also plays vital role in the field of communication protocol. As the SPI protocol is synchronous serial data link so a synchronous FIFO is required to Store the data. The read and write clock to the FIFO will be provided by the clock generator. The whole work of FIFO is fully dependent on the control circuitry and clock domain. The CDC issues can be verified with the FIFO as FIFO have two different pointers one for read and other for write, the phase and frequency relation for both the pointers can be different depending on the requirement.

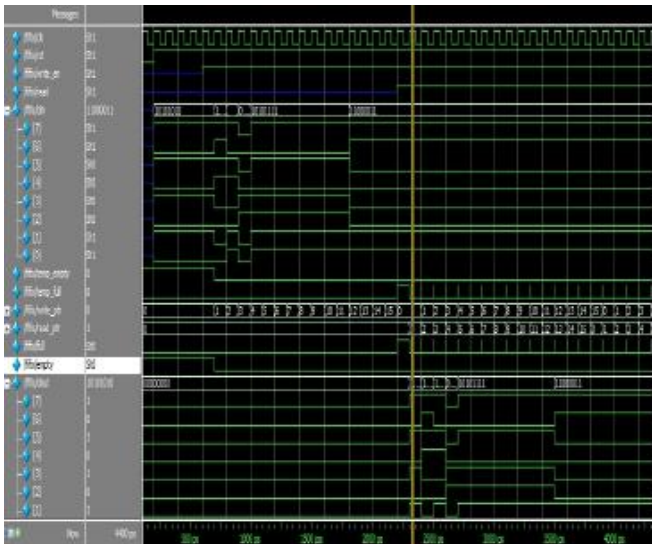

Fig.6. Simulation of FIFO.

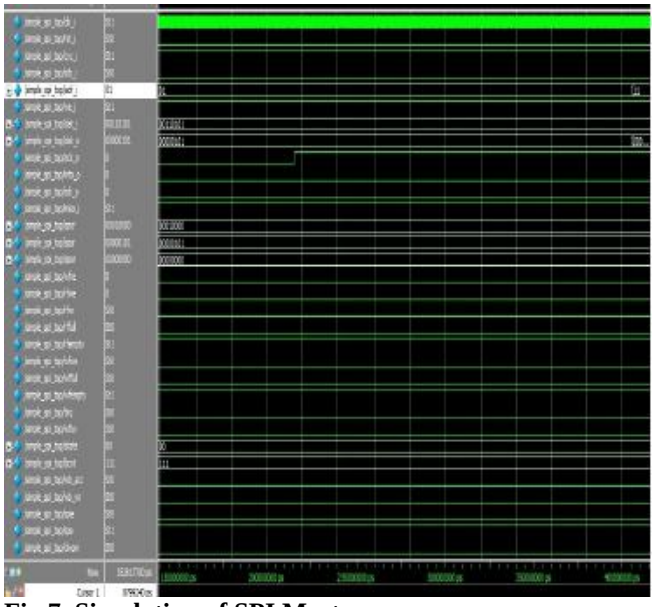

Fig.7. Simulation of SPI Master.

The simulation shows the data is transferred from the master. This shows the successful storage of data transmitted by the SPI Master on a particular address location. The transmission will be done using MOSI pin. The low power design technique is applied while we transfer the data from Master to Slave. This can be done by Minimizing data transitions on bus.

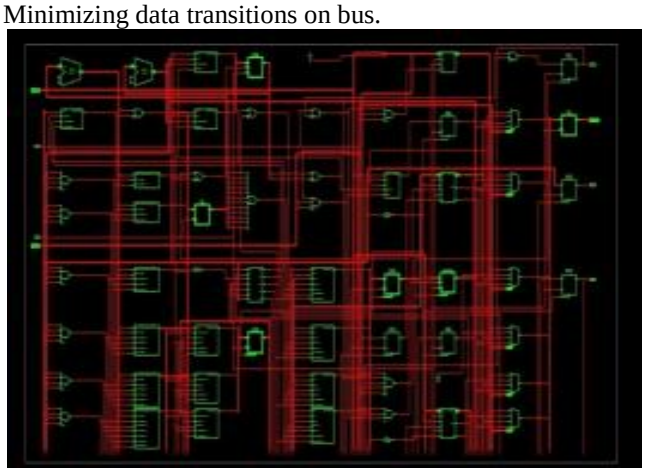

Fig.8. Synthesized RTL view of Master.

Hardware generated after synthesizing the code of SPI Master.

Fig.9. Synthesis Report of SPI Master

The Report shows the utilization of Flip-flops, LUT's (Look up table), IOBs (input/output Bonds) and GCLK's (Global Clocks).

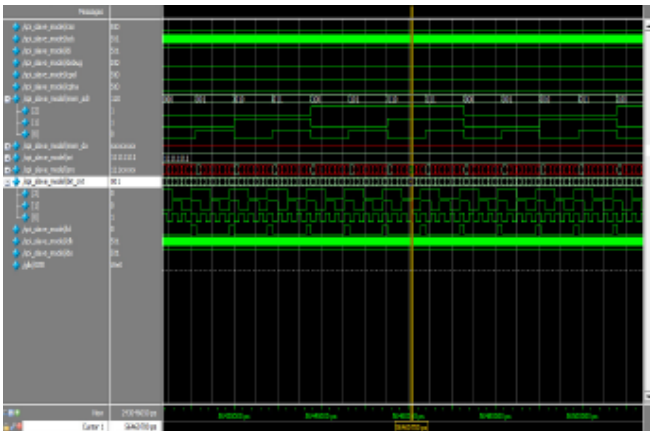

Fig.10. Simulation of SPI Slave.

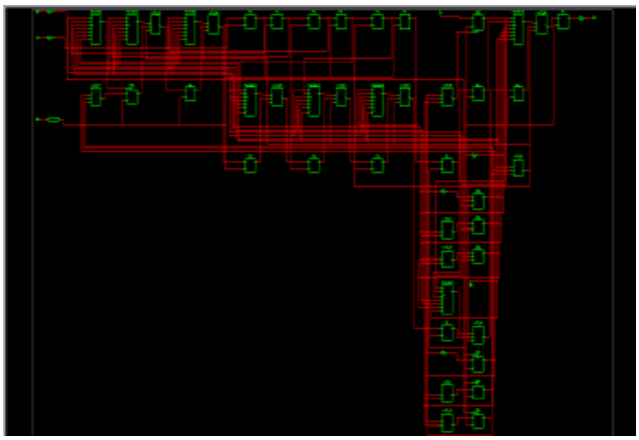

Fig.11. Synthesized RTL view of Slave.

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	14	768	1%
Number of Slice Flip Flops	23	1536	1%
Number of 4 input LUTs	25	1536	1%
Number of bonded IOBs	4	124	3%
Number of GCLKs	1	8	12%

Fig.12. Synthesis Report of SPI Slave.

VII. CONCLUSION

The design of Serial Peripheral Interface (SPI) with Single Master and Single Slave on FPGA using Verilog HDL gives a flexible vision to the designer to test the functional simulation and implementation of SPI Master-Slave Protocol using various IC (Integrated Chip) Industry EDA (Electronic Design Automation) Tools. The Design Successfully showing that it operates in Full Duplex mode. Data has been successfully transferred from SPI Master to SPI Slave using minimal utilization of resources (resource sharing). The overall Low power Designed protocol is implemented on FPGA where its clock domain crossing issues are fixed. Meanwhile CDC issues are fixed at RTL level but it is verified while implementation on FPGA.

VIII. REFERENCES

[1] Harish Sharma , Charu Rana, "Designing of 8-bit Synchronous FIFO Memory using Register File," International Journal of Computer Applications, vol. 63, no.16, pp. 23-26, Feb. 2013.

[2] Veda Patil, Vijay Dahake, Dharmesh Verma, Elton Pinto, "Implementation of SPI Protocol in FPGA," International Journal of Computational Engineering Research, vol. 3, no. 1, pp. 142-147, Jan. 2013.

[3] Trupti D. Shingare, R. T. Patil, "SPI Implementation on FPGA," International Journal of Innovative Tech. and Exploring Engineering, vol. 2, no. 2, pp. 7-9, Jan 2013.

[4] K.Aditya, M.Sivakumar, Fazal Noorbasha, T.Praveen Blessington, "Design and Functional Verification of A SPI Master Slave Core Using System Verilog," International Journal of Soft Computing and Engineering, vol. 2, no. 2, pp. 558-563, May 2012.

[5] Kaushal Buch, "HDL Design Methods for Low-Power Implementation," EInfochips, Dec. 2009.

[6] Saurabh Verma, Ashima S. Dabare, "Understanding Clock Domain Crossing Issues," Atrenta EDA DesignLine, Dec. 2007.

[7] Clifford E. Cummings, "Clock Domain Crossing (CDC) Design & Verification Techniques Using SystemVerilog," Sunburst Design, Inc., SNUG Boston 2008.

[8] Saurabh Verma, Ashima S. Dabare, "Understanding clock domain crossing issues," EE Times-India, Dec. 2007.

[9] Motorola, "SPI Block Guide V03.06," Motorola, Inc., doc. no. S12SPIV3/D, Feb.2003.