Design of SPI flash controller for Spartan6 FPGA

Bhavana.B.M
B.M.S.College of engineering, Bangalore, Karnataka, India

Shreela Dattawadkar
B.M.S.College of engineering, Bangalore, Karnataka, India

Ajaykumar D
B.M.S.College of engineering, Bangalore, Karnataka, India

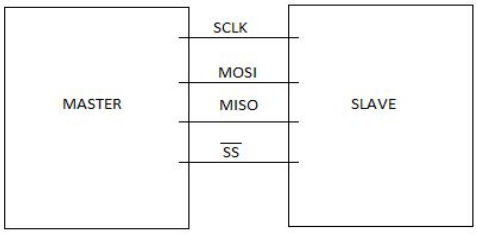
Abstract- In this paper, SPI protocol is demonstrated by using SPI flash memory present on Spartan6 board. SPI flash controller program is written in VHDL and is used to write into or read from the flash memory. The contents of the flash are then displayed using LEDs. The Linux (Ubuntu) as an operating system is used along with Xilinx to develop this project.

Keywords – serial peripheral interface, MOSI, MISO, write status register, read status register.

I. INTRODUCTION

In the world of communication protocols, SPI is often considered as "little" communication protocol compared to Ethernet, USB and others. Ethernet, USB are meant for "outside the box communications" and data exchanges between whole systems. While SPI, as well as I2C are well suited for communications between integrated circuits for low/medium data transfer speed with on-board peripherals. SPI allows a full duplex, synchronous, serial communication between the MCU and peripheral devices. SPI is often referred to as SSI (Synchronous Serial Interface). It is to be noted that unlike I²C, it is hard to find a formal separate 'specification' of the SPI bus – for a detailed 'official' description, one has to read the microcontrollers data sheets and associated application notes.

II. SERIAL PHERIPHERAL INTERFACE

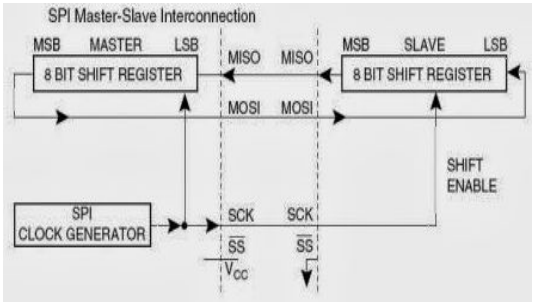


Serial peripheral interface is a synchronous serial data link which operates in full duplex mode. Devices communicate in master slave configuration where the master device initiates the clock and data frame. Multiple slave devices are allowed with individual slave select lines.SPI is a single-master communication protocol. This means that one central device initiates all the communications with the slaves The four basic SPI signals are:

1. SCLK (SCK) - the clock signal used for synchronizing data transfers. It is generated by the master.
2. MISO - Master In Slave Out. It is the line used for sending data from a slave to the master, with the most significant bit sent first.
3. MOSI - Master out Slave In. it is the line used for sending data from the master to a slave, with the most significant bit sent first.

4. SS - Slave Select. It is used to select a slave device. It has to be low prior to data transactions and must stay low for the duration of the transaction.

When the SPI master wishes to send data to a slave and/or request information from it, it selects slave by pulling the corresponding SS line low and it activates the clock (SCLK/SCK) signal at a clock frequency usable by the master and the slave. The master generates information onto MOSI line while it samples the MISO line.



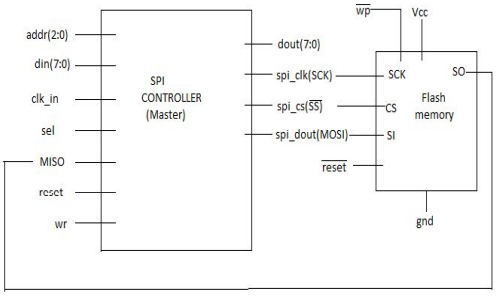
Hence, during each SPI clock cycle, a full duplex data transmission occurs:

- 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

Transmissions normally involve two shift registers of some given word size, such as eight bits, one in the master and one in the slave. The main element of the SPI system is the SPI Data Register. The 8-bit data register in the master and the 8-bit data register in the slave are linked by the MOSI and MISO pins to form a distributed 16-bit register.
When a data transfer operation is performed, this 16-bit register is serially shifted eight bit positions by the SCLK from the master, so data is exchanged between the master and the slave. Data written to the master SPI Data Register becomes the output data for the slave, and data read from the master SPI Data Register after a transfer operation is the input data from the slave. If there is more data to exchange, the shift registers are loaded with new data and the process repeats.

Four communication modes are available (MODE 0, 1, 2, 3) – that basically define the SCLK edge on which the MOSI line toggles, the SCLK edge on which the master samples the MISO line and the SCLK signal steady level (that is the clock level, high or low, when the clock is not active). Each mode is formally defined with a pair of parameters called ‘clock polarity’ (CPOL) and ‘clock phase’ (CPHA). Therefore, in addition to setting the clock frequency, the master must also configure the clock polarity and phase with respect to the data.

Block diagram:



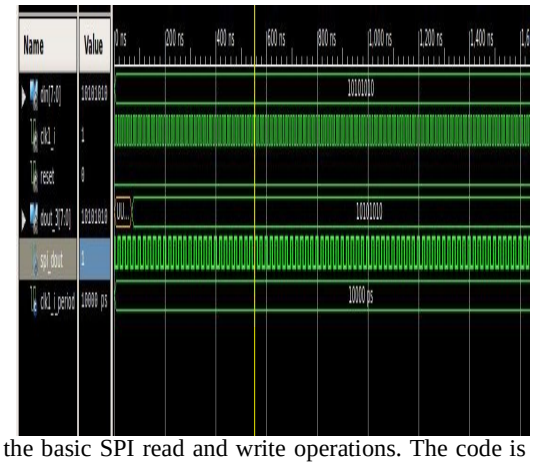
AT45DB161D is the Flash Memory on Spartan6 board. It is a 16Mbit flash and is compatible with SPI Modes 0 and 3.

Basic WRITE and READ Operations:

WRSR-Write Status Register
Most SPI flash memories have a write status register command that writes one or two bytes of data. To write to the status register, the SPI host first enables the slave select line for the current device. The master then outputs the appropriate instruction followed by data bytes that define the intended status register contents. Since the transaction does not need to return any data, the slave device keeps the MISO line in a high impedance state and the master masks any incoming data. Finally, Slave Select is de-asserted to complete the transaction.

RDSR- Read Status Register
A status register read transaction would be similar to the write transaction, but now takes advantage of data returned from the slave as shown in Figure 8. After sending the read status register instruction, the slave begins transmitting data on the MISO line at a rate of one byte per eight clock cycles. The host receives the bitstream and completes the transaction by de-asserting Slave Select.

III. RESULT



The simulation is verified for the basic SPI read and write operations. The code is also implemented on Spartan 6 FPGA present in UTLP kit.

IV.CONCLUSION

In this paper SPI protocol has been implemented in VHDL. The Design of Serial Peripheral Interface (SPI) with Single Master and Single Slave configuration has been done successfully showing that it operates in Full Duplex Mode. The SPI bus is straightforward and versatile, enabling simple and fast communication with a variety of peripherals. The full-duplex capability makes SPI very simple and efficient for single master/single slave applications. This SPI Master is a flexible programmable logic component that accommodates communication with a variety of slaves via single parallel interface. In this paper, we have more focused on data transmission between master and slave modules. We have verified that the data in slave device is same as the data in the master device.

REFERENCES

[1] Trupti D. Shingare, R. T. Patil,"SPI implementation on fpga", International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-2, Issue-2, January 2013
[2] Veda Patil, Vijay Dahake, Dharmesh Verma, Elton Pinto, "Implementation of SPI protocol on FPGA", International Journal Of Computational Engineering Research (ijceronline.com) Vol. 3 Issue. 1
[3] Ananthula Srinivas, M.Kiran Kumar, Jugal Kishore Bhandari, " Design and verification of serial peripheral interface", International journal of engineering development and research(IJEDR), ISSN: 2321-9939
[4] T. Durga Prasad, B. Ramesh Babu," Design and Simulation of SPI Master / Slave Using Verilog HDL", International Journal of Science and Research (IJSR) ISSN (Online): 2319-7064.
[5] Motorola Inc., "SPI Block Guide V03.06," February 2003.
[6] MC68HC11A8, Serial Peripheral Interface, Freescale Semiconductor, Inc.