MANIPAL ACADEMY *of* HIGHER EDUCATION

(Deemed to be University under Section 3 of the UGC Act, 1956)

MANIPAL SCHOOL OF INFORMATION SCIENCES

(A Constituent unit of MAHE, Manipal)

Design and Simulation of SPI Controller using Verilog HDL

Reg. Number	Name	Branch
221039026	Nikhilkumar Agasar	Embedded Systems
221039020	Rahul V	Embedded Systems

Under the guidance of

Dr. Mohan Kumar Jayasubramanian
Associate Professor,
Manipal School of Information Sciences,
MAHE, MANIPAL

10/11/2022

MANIPAL SCHOOL OF INFORMATION SCIENCES
MANIPAL
(A constituent unit of MAHE, Manipal)

Contents

LIST OF FIGURES

ABBREVIATIONS

SPI	Serial Peripheral Interface
MOSI	Master Out Slave In
MISO	Master In Slave Out
CS/SS	Chip/Slave Select
BIST	Built In Self Test
FSM	Finite State Machine
FPGA	Field Programmable Gate Array
UART	Universal Asynchronous Reception Transmission
CPOL	Clock Polarity
CPHA	Clock Phase

1. Introduction

Communication between electronic devices is very much essential in today’s scenario as it needs different devices to interact and required to transmit data from device to device. Both sides need to speak the same language. In electronics, these languages are called communication protocols.

There are two types of communication when we refer to embedded systems and microcontrollers. They are Serial Communication and Parallel Communication. In serial communication the data bits are transmitted serially one by one i.e., bit by bit on single communication line, It requires only one communication line rather than n lines to transmit data from sender to receiver. Thus, all the bits of data are transmitted on single lines in serial fashion. In parallel communication, all the bits of data are transmitted simultaneously on separate communication lines. In order to transmit n bit, n wires or lines are used.

Serial communication uses two methods: Asynchronous and Synchronous. Asynchronous, this type transfers single byte at a time, it does not need of clock signal. UART (universal asynchronous receiver transmitter) is asynchronous. Synchronous, this type of protocol transfers a block of data (characters) at a time. It requires clock signal. SPI (serial peripheral interface), I2C (inter integrated circuit) are synchronous serial communication protocol. In the view of Data Transmission, again we can divide it into further like, if the data can be transmitted and received, it is a duplex transmission. If the data can be transmitted in only one direction it is Simplex i.e. from TX to RX only one TX and one RX only. In half duplex, Data is transmitted in two directions but only one way at a time i.e. two TX's, two RX’s and one line. In full duplex, Data is transmitted both ways at the same time i.e. two TX's, two RX’s and two lines.

SPI, I2C, and UART are quite a bit slower than protocols like USB, Ethernet, Bluetooth, and Wi-Fi, but they’re a lot simpler and use less hardware and system resources. SPI, I2C, and UART are ideal for communication between microcontrollers and between microcontrollers and sensors where large amounts of high-speed data don’t need to be transferred.

In our project we are using SPI protocol which is synchronous and full duplex type.

SPI is a common communication protocol used by many different devices. For example, SD card modules, RFID card reader modules, and 2.4 GHz wireless transmitter/receivers all use SPI to communicate with microcontrollers. One unique benefit of SPI is the fact that data can be transferred without interruption. Any number of bits can be sent or received in a continuous stream. With I2C and UART, data is sent in packets, limited to a specific number of bits. Start and stop conditions define the beginning and end of each packet, so the data is interrupted during transmission. Devices communicating via SPI are in a master-slave relationship. The master is the controlling device (usually a microcontroller), while the slave (usually a sensor, display, or memory chip) takes instruction from the master. The simplest configuration of SPI is a single master, single slave system, but one master can control more than one slave.

The SPI (Serial Peripheral Interface) protocol uses the four wires for the communication MOSI, MISO, SCL, SS/CS. MOSI is the master out slave in line. This is used to send data from the master to the slave when the communication takes place. MISO is the master in slave out line. This is used to send data from the slave to the master when the communication takes place. SCL is the serial clock line, which the master generates, and goes to the slaves. SS/CS is the Slave Select or Chip Select line which is used to select the slave to communicate.

The master can choose which slave it wants to talk to by setting the slave’s CS/SS line to a low voltage level. In the idle, non-transmitting state, the slave select line is kept at a high voltage level. Multiple CS/SS pins may be available on the master, which allows for multiple slaves to be wired in parallel. If only one CS/SS pin is present, multiple slaves can be wired to the master by daisy-chaining. multiple slaves SPI can be set up to operate with a single master and a single slave, and it can be set up with multiple slaves controlled by a single master. There are two ways to connect multiple slaves to the master.

The current working of SPI is, for communicating with multiple slaves, master has to select multiple slaves using multiple slave select or chip select lines. That means, for n number of slaves, n number of CS/SS lines are used.

2. Literature Survey

1. Design and Implementation of SPI Bus Protocol with Built-In-Self-Test Capability over FPGA:
   The authors of this paper Shumit Saha, Md. Ashikur Rahman, Amit Thakur have explained and implemented SPI using BIST technology over FPGA. This BIST technology which is very much required in today's scenario helps in self testing. It gave us an idea about the architecture of the SPI module. It talks about the SPI pin diagrams and the modules used in the design. The comparator used to verify the bit format using some value which gave us idea about using comparator. The waveforms which are given in the paper made us to understand the working of different modules and we could figure out the nature of the waveform in what way we should get that.

2. Design and Simulation of SPI Master / Slave Using Verilog HDL:
   This paper which is written by the authors, T. Durga Prasad, B. Ramesh Babu mainly aims about the working of SPI protocol with master and slave. This aims at the data transmission from master to slave. It also talks about the master mode operation and slave mode operation by using master control bit. Master control bit is set means, it is writing data to the shift register in the master and if Master control bit is clear means, it is operating in slave mode. This paper gives the total structure about Master and Slave verification of Testing and Design along with functional verification.

3. SPI to I2C Protocol Conversion Using Verilog:
   This paper which is written by authors Dvijen Trivedi, Aniruddha Khade, Kashish Jain, Ruchira Jadhav describes about conversion of SPI protocol to I2C protocol using Verilog The purpose of this paper is to design and simulate a Protocol Conversion Unit (PCU) for seamless communication between the two widely accepted serial communication protocols SPI and I2C. Design given in this paper takes data from a sender device working on SPI protocol and sends it to a receiver device working on I2C protocol, which otherwise without such design would not be possible. This also gives an extract about how to design an SPI FSM for the simulation purpose a top-level design SPItoI2C was implemented.

4. Design and Implementation of High-Speed Serial Peripheral Interface:

   This paper which is written by Anand N, George Joseph, Suwin Sam Oommen, and R Dhanabal, gives a great idea about the implementation about the high-speed serial peripheral interface which is having a control register to implement the work of Master mode or slave mode. The control register can be programmed by the user to initiate the data transfer between the master and the slaves. By appropriately programming the control register, the SPI module can be made to operate in master mode or slave mode. The status register gives information on the current position of the data transfer operation. From the SPI status register, information on whether the data transfer has completed or not can be inferred. We also came to know about the detailed SPI protocol architecture from this paper which was functionally implemented and logically synthesised in Xilinx.

5. An Introduction to I2C and SPI Protocols:

   The author of this paper, Frédéric Leens has explained detailed history of development of both SPI and I2C protocols and the purpose of those. The paper also explains details about the working mode of the I2C and SPI protocols. This paper has helped us in implementing the block diagrams for our project and understanding the entire protocols in the detailed manner. It also gives brief comparison between I2c and SPI with frame formats and timing diagrams respectively. It also mentions about different concepts like clock stretching in I2C.

3. Objective(s)

- To study and understand the SPI protocol and the controller aspects.
- To design SPI protocol using Verilog HDL.
- To simulate the waveforms of the SPI protocol and show its working.
- To experiment and try to reduce number of salves if possible.

4. Block diagram

SPI is called as a 4-wire bus as it requires four wires for its communication as shown below. In the case of single slave communications we need only 3 wires, as slave select

(SS) is not required. So SPI requires more communication lines in contrast to UART, I^{2}C, USB etc.

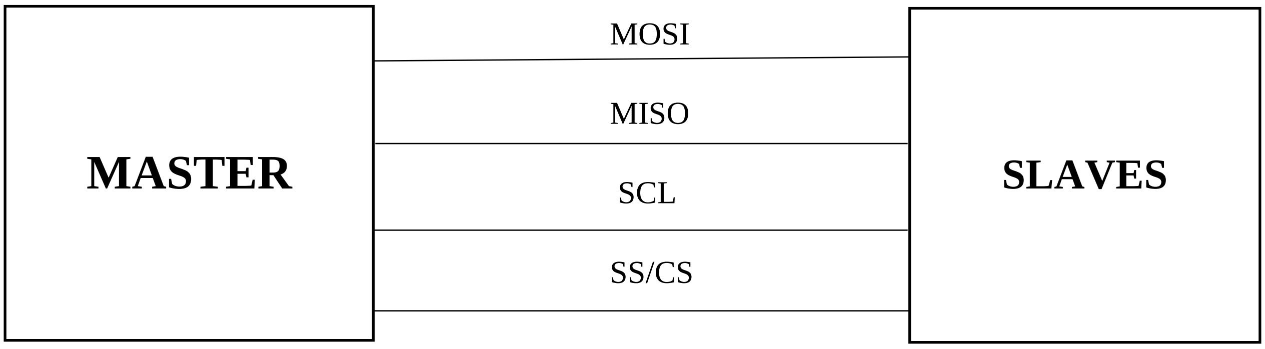

Fig 4.1: Block Diagram of SPI

MOSI: Master Out Slave In

MISO: Master In Slave Out

SCL: Clock Signal

SS/CS: Chip Select

Schematic Diagram:

Master Internal Architecture

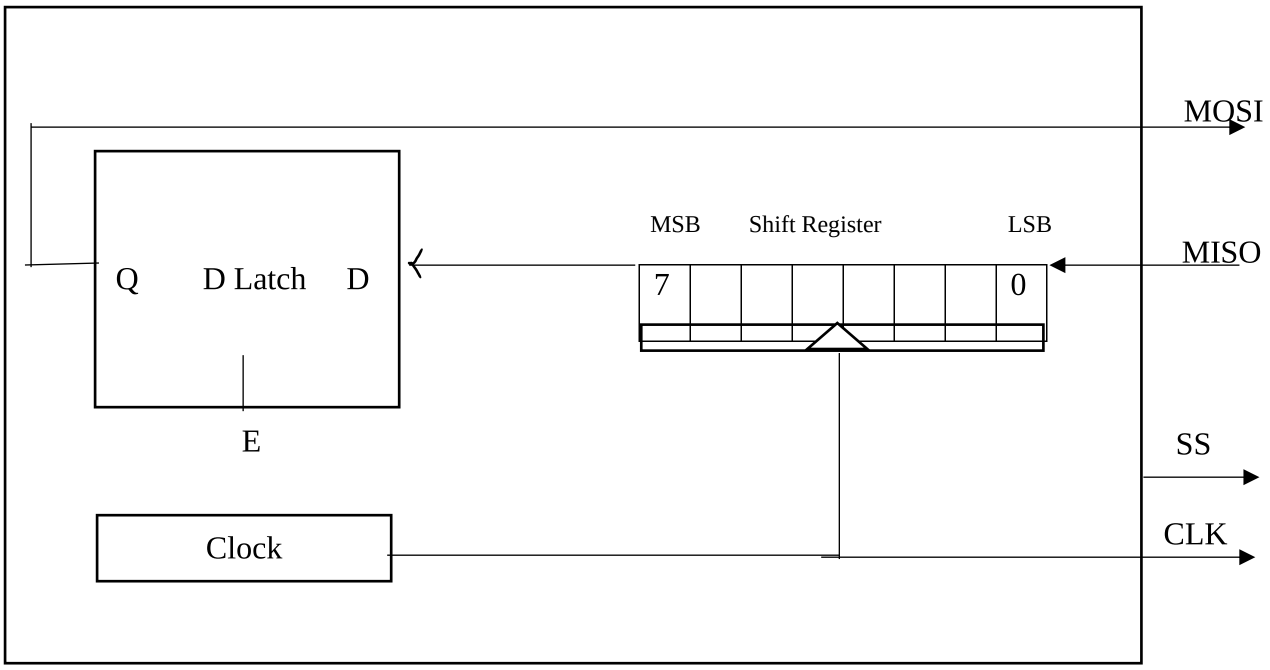

Figure4.2 Master Internal Architecture

- From the Block Diagram, we can know about the module of SPI Protocol working, which contains Master and slave.

- In which the it has signals of MOSI, MISO and Slave Select along with Clock signals.

Schematic Diagrams:

- From the Figure 4.2, the Master device consists of a Shift Register, a data latch and a clock generator. MOSI signal is going out from D latch and MISO signal is coming from slave through shift register and to D latch as input.

- From the Figure 4.3, the Slave device consists of a Shift Register, a data latch and a clock signal which is in-turn connected to clock from master. MOSI is coming from Master to slave through shift register and MISO is going out of slave from D latch to master.

- The slave consists of similar hardware: a shift register and a data latch. Both the shift registers are connected to form a loop.

- Usually, the size of the register is 8 – bits but higher size registers of 16 – bits are also common.

Slave Internal Architecture

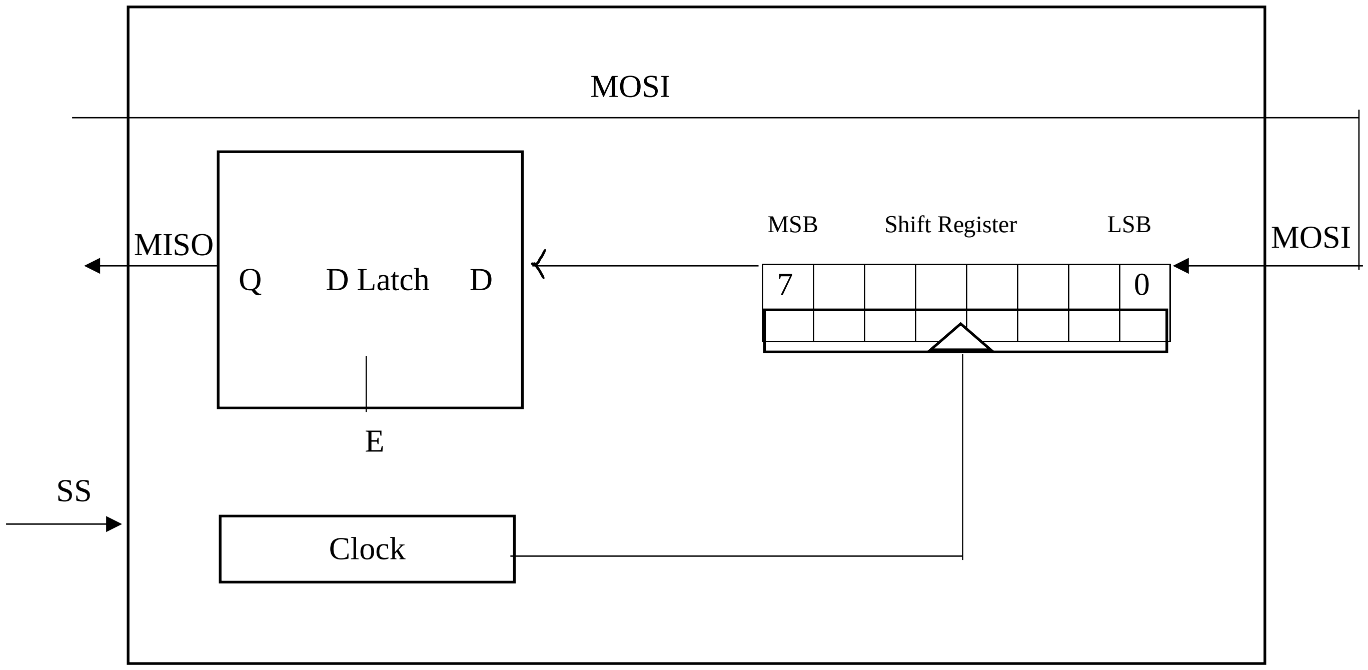

Figure 4.3 Slave Internal Architecture

5. Proposed work/Discussion

1. Analysing the whole controller aspects of SPI
2. Analysing the signals which are should be given to SPI master to communicate with slave.
3. Analysing the required hardware architecture inside the Master and Slave blocks to implement and communicate.
4. Developing Verilog code for SPI protocol to know the controller aspects.
5. Analysing separate Verilog code for Master part
6. Analysing separate Verilog code for Slave part

Working of SPI Protocol

In SPI protocol, there can be only one master but many slave devices. The SPI bus consists of 4 signals or pins. They are

- Master – Out / Slave – In (MOSI)
- Master – In / Slave – Out (MISO)
- Serial Clock (SCLK) and
- Chip Select (CS) or Slave Select (SS)
- **Master – Out / Slave – In** or MOSI, as the name suggests, is the data generated by the Master and received by the Slave. Hence, MOSI pins on both the master and slave are connected together. Master – In / Slave – Out or MISO is the data generated by Slave and must be transmitted to Master.

- **MISO** pins on both the master and slave are ties together. Even though the Signal in MISO is produced by the Slave, the line is controlled by the Master. The Master generates a clock signal at SCLK and is supplied to the clock input of the slave. Chip Select (CS) or Slave Select (SS) is used to select a particular slave by the master.

- Since the clock is generated by the Master, the flow of data is controlled by the master. For every clock cycle, one bit of data is transmitted from master to slave and one bit of data is transmitted from slave to master.

- This process happen simultaneously and after 8 clock cycles, a byte of data is transmitted in both directions and hence, SPI is a full – duplex communication.

- If the data has to be transmitted by only one device, then the other device has to send something (even garbage or junk data) and it is up to the device whether the transmitted data is actual data or not.

- This means that for every bit transmitted by one device, the other device has to send one bit data i.e. the Master simultaneously transmits data on MOSI line and receive data from slave on MISO line.

- If the slave wants to transmit the data, the master has to generate the clock signal accordingly by knowing when the slave wants to send the data in advance.

From the Figure 1 and 2, the Master device consists of a Shift Register, a data latch and a clock generator. The slave consists of similar hardware: a shift register and a data latch. Both the shift registers are connected to form a loop. Usually, the size of the register is 8 – bits but higher size registers of 16 – bits are also common.

During the positive edge of the clock signal, both the devices (master and slave) read input bit into LSB of the register. During the negative cycle of the clock signal, both the master and slave places a bit on its corresponding output from the MSB of the shift register.

Hence, for each clock cycle, a bit of data is transferred in each direction i.e. from master to slave and slave to master. So, for a byte of data to be transmitted from each device, it will take 8 clock cycles.

We have already seen that it is the job of the Master device to generate the clock signal and distribute it to the slave in order to synchronise the data between master and slave. The work of master doesn't end at generating clock signal at a particular frequency.

In fact, the master and slave have to agree on certain synchronization protocols. For this, two features of the clock i.e. the Clock Polarity (CPOL or CKP) and Clock Phase (CPHA) come in to picture.

Clock Polarity determines the state of the clock. When CPOL is LOW, the clock generated by the Master i.e. SCK is LOW when idle and toggles to HIGH during active state (during a transfer). Similarly, when CPOL is HIGH, SCK is HIGH during idle and LOW during active state.

Clock Phase determines the clock transition i.e. rising (LOW to HIGH) or falling (HIGH to LOW), at which the data is transmitted. When CPHA is 0, the data is transmitted on the rising edge of the clock. Data is transmitted on the falling edge when CPHA is 1.

Depending on the values of Clock Polarity (CPOL) and Clock Phase (CPHA), there are 4 modes of operation of SPI: Modes 0 through 3.

Mode 0:

Mode 0 occurs when Clock Polarity is LOW and Clock Phase is 0 (CPOL = 0 and CPHA = 0). During Mode 0, data transmission occurs during rising edge of the clock.

Mode 1:

Mode 1 occurs when Clock Polarity is LOW and Clock Phase is 1 (CPOL = 0 and CPHA = 1). During Mode 1, data transmission occurs during falling edge of the clock.

Mode 2:

Mode 2 occurs when Clock Polarity is HIGH and Clock Phase is 0 (CPOL = 1 and CPHA = 0). During Mode 2, data transmission occurs during rising edge of the clock.

Mode 3:

Mode 3 occurs when Clock Polarity is HIGH and Clock Phase is 1 (CPOL = 1 and CPHA = 1). During Mode 3, data transmission occurs during rising edge of the clock.

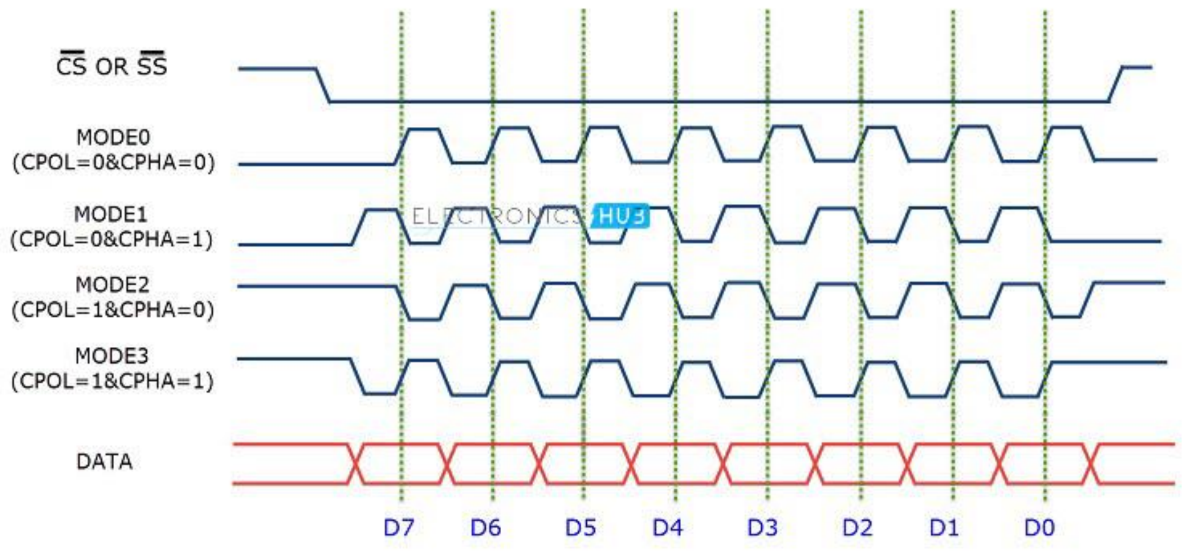

Figure 1 Timing Diagram of Different Modes of SPI

6. Conclusions

1. Analysing and implementing SPI controller aspects using Verilog HDL.

2. Analysing of SPI using Verilog HDL code for Master and Slave separately.

7. Scope for further work

- After analysing SPI protocol, we can come to know that the one challenge in SPI is about SlaveSelect or ChipSelect lines.
- For n different Slaves there are n different CS lines. This will increase the Hardware size and make the circuit or system more complex.
- Hence, the future scope lies here to modify this protocol to use single select line for multiple slaves. A single slave select line is used, which goes to all slaves and hence space can be saved and adding a new slave becomes easier.

8. References

1. S. Saha, M. A. Rahman and A. Thakur, "Design and implementation of SPI bus protocol with Built-in-self-test capability over FPGA," 2014 International Conference on Electrical Engineering and Information & Communication Technology, 2014, pp. 1-6, doi: 10.1109/ICEEICT.2014.6919076.
2. Prasad, Tadi Durga and B. Ramesh Babu. "Design and Simulation of SPI Master / Slave Using Verilog HDL." (2014). Paper ID: 02015624 International Journal of Science and Research (IJSR) ISSN (Online)
3. D. Trivedi, A. Khade, K. Jain and R. Jadhav, "SPI to I2C Protocol Conversion Using Verilog," 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), 2018, pp. 1-4, doi: 10.1109/ICCUBEA.2018.8697415.
4. Anand N, G. Joseph, S. S. Oommen and R. Dhanabal, "Design and implementation of a high speed Serial Peripheral Interface," 2014 International Conference on Advances in Electrical Engineering (ICAEE), 2014, pp. 1-3, doi: 10.1109/ICAEE.2014.6838431.
5. F. Leens, "An introduction to I2C and SPI protocols," in IEEE Instrumentation & Measurement Magazine, vol. 12, no. 1, pp. 8-13, February 2009, doi: 10.1109/MIM.2009.4762946.