International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Special Issue: 1 Volume No.7 April – 2023
DOI:10.46647/ijetms.2023.v07si01.009 ISSN: 2581-4621

BIST Embedded Master Slave Communication Design Using SPI Protocol

[1]G. Anitha Rani, [2]N Yaswanth Reddy, [3]C Govinda Pavan Kumar, [4]R Tejaswari, [5]V Yaswanth

[1]*Assistant Professor, Department of ECE, Annamacharya Institute of Technology and Sciences, Tirupati, A.P.*

[2,3,4,5]*B. Tech Student, Department of ECE, Annamacharya Institute of Technology and Sciences, Tirupati, A.P.*

Abstract—The Serial-Peripheral Interface(SPI) Protocol is an interface that enables the serial(one bit at a time) exchange of data between two devices, one called a master and the other called a slave. An SPI operates in full duplex mode. It is also called as synchronous serial interface specification is used for communication between single master and single/multiple slaves. A self-testability feature for SPI modules is required because of the rise in the number of slaves, which increases circuit complexity and calls for testing for fault-free circuits. The solution for self-test in circuits is built-in self-test (BIST), which also lowers the cost of testing and maintenance. This paper introduces the design of a BIST embedded SPI module with a single master and single slave configuration. 8-bit data is sent over the module, and the circuit under test (CUT) is self-tested using the BIST feature to ensure that it is working properly. For applications like Application Specific Integrated Circuits (ASICs) or System on Chip, this SPI module was created using the Verilog Hardware Description Language (HDL) on an EDA playground platform (SOC).

Keywords— SPI, BIST, MISO, MOSI, TPG, ORA, Full duplex.

I. INTRODUCTION

In order to replace parallel interfaces and demonstrate fast data transfer across modules, Motorola created the Serial Peripheral Interface (SPI) protocol in the middle of the 1980s. SPI has become the most popular serial communication protocol due to its simple interface and quick transfer. When sending and receiving data between them, SPI uses full duplex, master-slave communication that synchronizes on the rising or falling edge of the clock. Both the master and the slave can transmit data simultaneously. There are essentially two different types of SPI interfaces: 3-wired and 4-wired.The primary focus of this study is the well-known 4-wired SPI interface. Even embedded devices like SoC processors and microcontrollers like the Programmable Interface Controller (PIC) and Advanced Virtual RISC adhere to the SPI Protocol (AVR). These chips' internal SPI controllers let them function as either master or slave blocks. SPI's unique features include master/slave operation, double buffered data registers for transmission and reception, polarity and phase synchronization with serial clock, interrupt capability of CPU with fault deduction, and high-speed data transfer requirements. These applications are where SPI is most commonly used. Their recognition relies on the functional registers of SPI. The polarity and phase of the synchronous clock are controlled by the clock polarity (CPOL) and clock phase (CPHA) bits of SPI. There are many different communication protocols available for both long distance and close proximity communication. We have the Universal Serial Bus (USB), Serial Advanced Technology Attachment (SATA), EXPRESS, and ETHERNET for long-distance communication. SPI and the Inter Integrated Circuit (I2C) protocol are used for local communication. SPI is easy to use and has a high transmission speed when compared to other protocols. When a data exchange has been completed, the master generates the primary SPI clock for the SPI module. Processing data is frequently referred to as the "small" protocol communication for on-board communication. SPI is a protocol for interfaces. The term "interface" can be explained as a shared boundary that two distinct computer hardware or software components must share in order to exchange information. Instead, "protocols' ' are largely defined as a collection of rules that have nothing to do with communications of any kind; rather, interfacing protocol refers to the exchanging of information between blocks so that they can communicate and acknowledge one another. Making the slaves generate interrupts from an interrupt pin allows for the creation of a priority-based master-slave communication. This strategy works well when the linked slave must have the highest priority in communication from the master. A situation that occurs in real time is one in which a microcontroller serves as the master and is connected to two slaves, an alarm and a sensor. This specific system adheres to the SPI protocol, where the temperature is continuously monitored until it reaches its maximum level, at which point an alert should sound. As a result, with the scenario as it is, the master must interact with the slaves first. Testing of circuits became particularly challenging to carry out as a result of the rapid rise in circuit complexity and the development of new technology. In this work, the SPI module's self-testability feature has been established. A CUT is introduced within the SPI slave, and the designed CUT has the capability of doing self-tests. As a result, fewer circuits will fail because the self-testability feature contributes to the creation of fault-free circuits.

II. LITERATURE SURVEY

A configurable Intellectual Property (IP) module for the Serial Peripheral Interface (SPI) protocol was proposed by S. Choudhury et al. in 2014. Using Verilog HDL and System Verilog, the IP module was created and tested.

Using test bench architecture, Vineeth B and Dr. B. Bala Tripura Sundari (2018) suggested a functional verification technique for the SPI protocol. Thanks to the meticulous use of legitimate test cases, the SPI model has been fully functionally covered. The proposed UVM testbench generates random results and utilizes a scoreboard to compare them to the anticipated results in order to verify its operation. It is advised that this work be utilized for the verification of SoCs that include the SPI protocol because the functional and cross coverage report of this work clearly shows that the whole operation of SPI is exercised correctly.

A SPI paradigm with a single master-single slave using an independent slave configuration and a single master-multiple slave using a daisy chain configuration was presented by Pallavi Polsani in 2020. In addition to being implemented in Spartan 3E, the proposed SPI model has been built using Verilog HDL for synthesis and confirmed using System Verilog.

By creating an IP core for SPI, Muhammad Hafeez and Azilah Saparon (2019) offered a model for how APB will interface with the latter. The SPI in the work that has been described can send or receive data from a connected slave and APB-SPI controller at a very excellent frequency of 16 MHz with a high degree of data flexibility. To create the gdsii file for tape out, the suggested work is synthesized in Quartus lite 16 and simulated in Model Sim. The modeled design in this work is a straightforward interface that may be used to communicate with APB and has very basic connections to the master's IO port.

A BIST capability integrated FPGA implementation of the SPI protocol was presented by Shumit Saha in 2014. The complete module was created using Verilog HDL, and the Xilinx Spartan-2 FPGA was used to implement the design. In terms of speed, flexibility, stability, and cost, the proposed SPI model is superior to the standard SPI model. The demonstrated work can help the fabrication business because it can be tested with simply a switch click. saving a great deal of time and money on testing.

According to Deepika and Jayanthi K Murthy (2020), the SPI protocol uses priority-based master-slave interactions for communication purposes, with the arbiter block actually determining the priority of each slave. Priority-based data flow from a single master to numerous connected slaves is the major focus of the concept. When the interrupt pin becomes high, the master chooses which slaves to speak with first based on priority. In this case, the suggested architecture utilizes less electricity since it uses fewer resources than interrupt-based approaches.

III. EXISTING METHOD

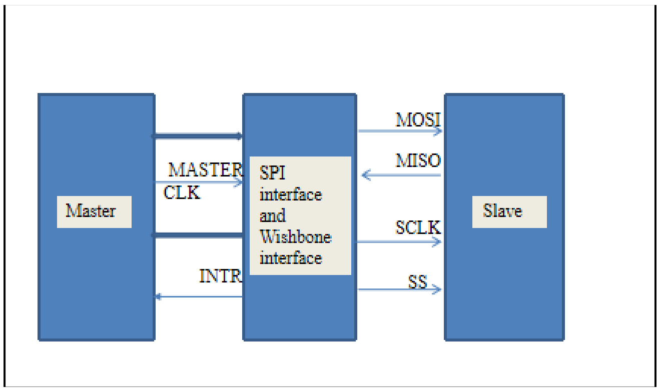

Fig. 1.Master Slave Design for Existing Method

There are two ways to use the current SPI protocol to construct the priority-based master-slave communication system. Using the existing pin MISO, which the slave can utilize to ask the master for communication using a wishbone interface, is one method of generating the interrupt signal. The drawback of this approach is that the system can no longer be referred to as full duplex because MISO becomes dedicated to sending commands whenever the slave initiates any communication. Another strategy is to create a new "interrupt pin" connection between the master and slave.. This can be used by the slave to send a request to the master, and depending on the mode and condition of operation, the master may send an interrupt grant response in response to the request to establish a connection for data transmission. In this instance, the master operates in two modes, Idle and Wait for Interrupt, as well as the condition of An arbiter module is defined to assign slaves a priority, with each slave receiving a different priority from highest to lowest. The user's requirements determine how the priority is assigned, and this module can be customized. The slave with the highest priority is treated first, and then the other slaves are treated in order of priority. If a master receives an interruption from a slave during a communication, it first determines the priority of the slave. The master creates communication with the slave if it has a higher priority. If not, it completes the current data transfer and then determines whether the slave with the second priority has any data that has to be transferred. Fig. illustrates the use of the SPI interface for communication between the master and slave using different registers and signals. The interrupt signal INTR, which sends the interrupt request signal from slave to master, is described in the diagram.

- Generally speaking, the following things happen when data is transferred using SPI:

1. To choose the slave, the SS line is muted.
2.The SCLK line then begins to tick as soon as the SS shuts off.
3.Next, the address and the first positive edge of the SCLK line read-write bit are transmitted over the line. Prior to the SCLK line firing up, the MOSI lines are null. If a slave sends some data, the MISO line is used.
4.The data bits are then transmitted one by one..

Priority must be taken into consideration while designing the interface for master and slave communication. Verilog code was used to create the RTL code for each module and test bench, and the Xilinx Vivado tool was used for simulation and synthesis. The next several actions must be taken in order to create and manage interrupts and priorities. Design of a priority controller that gives different slaves different priorities.

IV. SPI ARCHITECTURE

Many kinds of devices typically use SPI as a communication protocol. One key advantage of SPI is the ability to send any quantity of data uninterruptedly and continuously. Through the SPI protocol, all device communication is master-slave in nature. The device known as the "master" contributes to the creation of the primary SPI clock that is used to synchronize the data. Compared to other serial

protocols, the SPI supports substantially greater frequencies. SPI protocol only allows for one master, however it allows for connecting a single master to several slaves in a slave configuration.
The connection between a standard SPI Master and SPI Slave block is shown in Fig. 2. Here, we can see that a connected slave to the master block is selected using an active low signal from the master, also known as a chip choose signal. when there are several parallel connections between the slave blocks and the master block. The chip select signal for each slave block will then be unique, and when a slave block wants to disconnect from the master block, the signal will be turned high. Master Out Slave In-signal from Master to Slave (MOSI) and Master In Slave Out (MISO) are the two data lines for SPI block (Master In Slave Out-signal from slave to master).

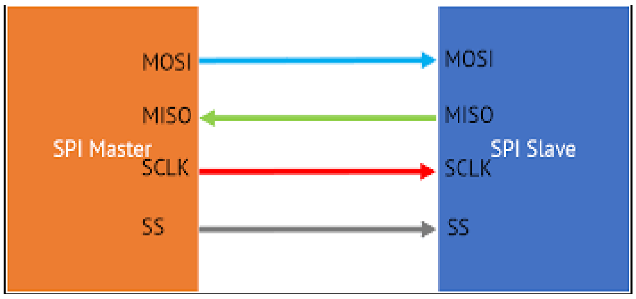

Fig 2: SPI with Master Slave Communication

A. Transfer Operation in SPI :

• The SPI clock, which is produced by the master, is used to synchronize the data supplied and received.

• Then the master block lowers the voltage on the CS pin, which chooses the slave.

• In the SPI module, there are two 8 bit shift registers.

The data is shifted from the MSB of the master shift register to the other one in the slave block in one of the two the LSB of the slave shift register.

•The Master then transmits 8-bit data, one bit at a time, along the MOSI line to the Slave. The slave bitwise reads the transmitted data as it is received.

•If the slave has to send data in order to communicate, it will send one bit at a time from its MSB shift register to the master's LSB shift register through the MISO pin.

V. BIST ARCHITECTURE

Built-In-Self Test (BIST) is a method used as a self-testability feature where all testing functions are located externally to the CUT and the CUT has to go through testing to check any defects present in it.
The BIST architecture has namely three important basic
blocks connected to the CUT:

• Test Pattern Generator(TPG) block
• Output Response Analyzer(ORA) block
• Test Controller

The test pattern generator block generates a random pattern for the CUT, as seen in Fig. 2. A counter, linear feedback shift registers, and memory with stored patterns are a few of the pattern generators (LFSR). A compactor and comparator block make up the output response analyzer block. Whereas the responses arriving from the CUT are compressed using the MISR (Multiple Input Serial Register) approach by the compaction block. The golden signature kept in a ROM, which directs in CUT correction, is then compared to these condensed responses, also known as output signatures. All planned tests are activated by a test controller block, which is also in charge of analyzing the results.

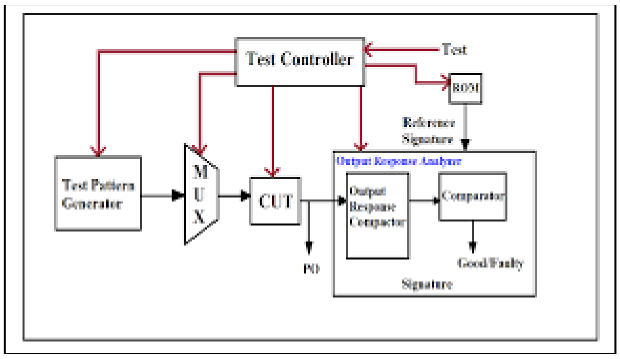

Fig 3 : BIST Architecture Block Diagram

VI. PROPOSED ALGORITHM

An SPI module with self-testing capabilities has been introduced in this paper. The fundamental elements of the BIST architecture have been introduced in the block diagram of the SPI architecture, where the intended CUT can self-test. The test patterns are generated by the TPG block and sent to the Slave via the MOSI pin of the Master block in the block diagram for the BIST embedded SPI protocol (fig. 3).Where slave reads the data sent from the master and segregates the received data and then finally passes to the CUT designed within the SPI slave block for performing the operation. when the arithmetic process is finished. Using the MISO pin, the CUT results are transmitted from the Slave block to the Master block. The ORA block created within the SPI Master is then fed the obtained ALU results in order to verify the accuracy of the CUT.

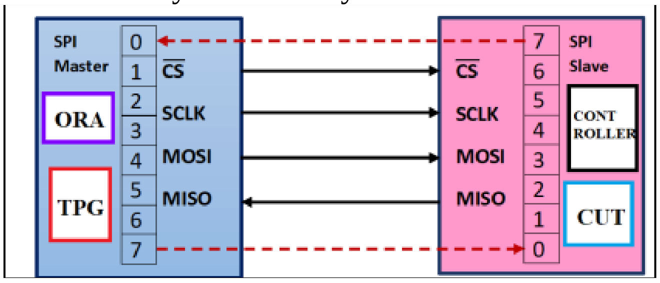

Fig 4 :BIST embedded SPI Master-Slave block

A. TPG:

The TPG block that is being presented here generates test patterns using the LFSR (linear feedback shift register) approach. The term "pseudo-random pattern generator" also applies to linear feedback shift registers. The maximum sequence length for an n-bit LFSR is 2n-1. Let's use the n=4 bit LFSR as an example. The fig4 can be used to show how test pattern generation works.

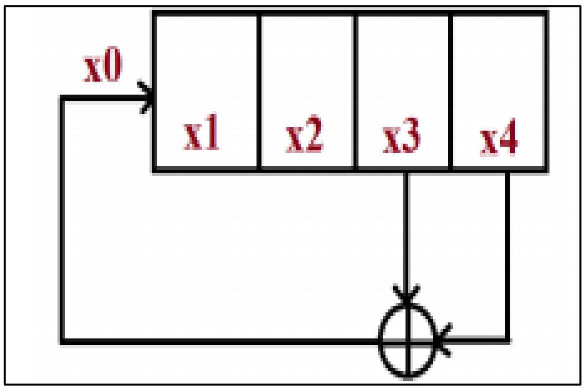

Fig 5 : 4 bit Linear Feedback Shift Register

A 4 bit shift register and an exclusive OR gate make up the feedback path of this 4 bit LFSR. The third and fourth bits of the data stream are exposed and provided as serial input to the first input, as can be seen in the image. A simple polynomial (F(x)=x4+x3+1) is used to determine the third and fourth bits for exploration. Now, we extract the fourth and third bits from the polynomial for exponentiation, and the result is sent to x0 (i.e. 1). An LFSR's characteristic polynomial known as a primitive polynomial determines the maximum length of the sequence generation. Hence, the maximum number of patterns that may be formed for a 4-bit LFSR is 15. (2^4-1).

Table 1: Pseudo pattern generation for 4-bit LFSR

X1	X2	X3	X4	X3^X4=X0	Clock cycle
1	1	1	1	0	1
0	1	1	1	0	2
0	0	1	1	0	3
0	0	0	1	1	4
1	0	0	0	0	5
0	1	0	0	0	6
0	0	1	0	1	7
1	0	0	1	1	8
1	1	0	0	0	9
0	1	1	0	1	10
1	0	1	1	0	11
0	1	0	1	1	12
1	0	1	0	1	13
1	1	0	1	1	14
1	1	1	0	1	15
1	1	1	1	0	16 (cycle repeats)

The table plainly shows that the LFSR explores every possible combination of codes, bar one (in this case it is 0000). This is referred to as a locked state, and because of its locking character, such a condition is discarded.
The suggested approach uses an 8 bit LFSR that creates 255 pseudo-random patterns (2^8-1).

B. ORA:
The MISR (Multiple Input-Serial-Register) approach is used by the ORA block described in this paper to condense the replies obtained from the CUT. The golden signature is then used to compare the compressed response and evaluate whether or not a problem is present. Fig. 5 can be used to explain MISR operation.

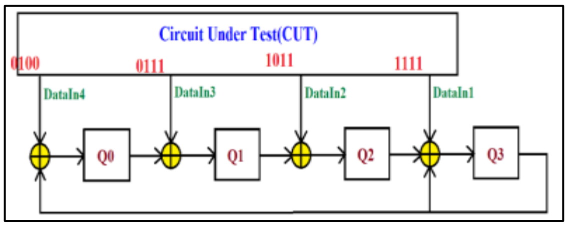

Fig 6 : MISR Block

Here, we can see that the MISR block receives four inputs from the CUT block for the compaction of the received replies. Four ex-or blocks make up the MISR block, which comes before each D flip flop. where each byte of the data input from the cut is processed individually Each data input stream's rightmost bit is output first. Here too, it is clear that Q3's output serves as feedback for the first and last exponents, which also use the same basic polynomial.

Table2: Signature analysis for the given inputs

Clock cycle	Q0	Q1	Q2	Q3
0	0	0	0	0
1	0	1	1	1
2	1	1	0	1
3	0	0	1	0
4	0	0	1	0

The table demonstrates how the CUT's generated inputs are condensed into a single output signature.

VI. Results And discussions :

Fig. 7 depicts the suggested design's schematic layout. Data is transferred between the slave and the master in this manner.

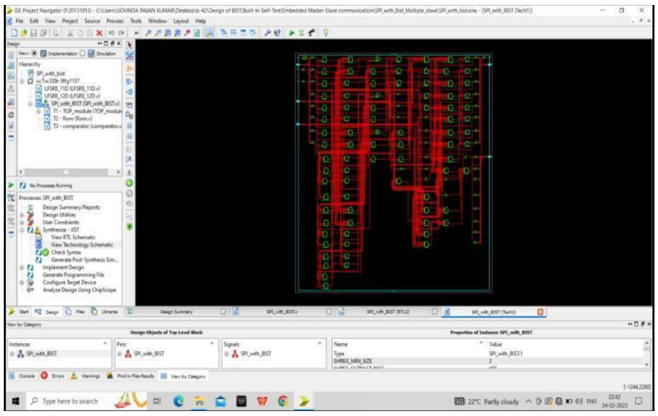

Fig 7. Schematic layout of the Master-slave communication

Fig. 7 shows the synthesized design of the master slave Communication.

With the aid of both master and slave communication using the XILINX in EDA playground platform, the resultant waveform is shown in Fig. 8.

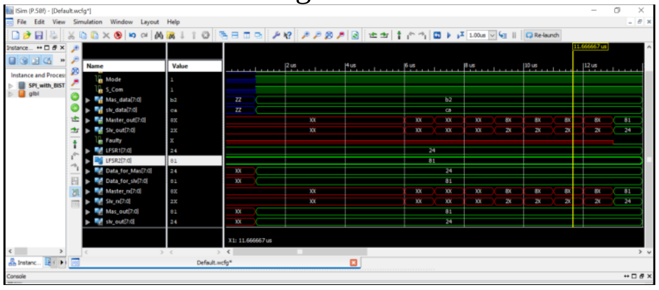

Fig 8 : Simulation Output of BIST SPI Protocol

VII.CONCLUSION AND FUTURE SCOPE

This study describes the successful design of a BIST embedded SPI protocol with master slave configuration using Verilog HDL and simulation using Cadence Xilinx 20.09 on EDA Playground Platform. The proposed Model makes good use of the BIST's self-testability capability. The planned CUT employs the SPI protocol to transport data and has the ability to self-test to ensure that the circuit being tested is functioning properly.

VIII. REFERENCES

[1] Choudhury, G.K.Singh, R.M. Mehra, "Design and Verification Serial Peripheral Interface (SPI) Protocol for Low Power Applications",International Journal of Innovative Research in Science,Engineering and Technology, Vol. 3, Issue 10, October 2014

[2] Vineeth B, Dr. B. Bala Tripura Sundari, "UVM Based Test-bench Architecture for Coverage Driven Functional Verification of SPI Protocol", 2018 International Conference on Advances in Computing,
Communications and Informatics (ICACCI)

[3] PallaviPolsani, V. Priyanka B., Y. Padma Sai, "Design & Verification of Serial Peripheral Interface (SPI) Protocol", International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8 Issue-6, March 2020

[4] Muhammad Hafeez, Azilah Saparon, " IP Core of Serial Peripheral Interface (SPI) with AMBA APB Interface",IEEE 9th Symposium on Computer Applications & Industrial Electronics (ISCAIE),2019
[5] Sumit Saha, Md. Ashikur Rahman, Amit Thakur, "Design and Implementation of SPI Bus Protocol with Built-In-Self-Test Capability over FPGA" International Conference on Electrical Engineering and
Information & Communication Technology (ICEEICT) 2014 .
[6] Deepika, Jayanthi K Murthy "Interrupt Enabled Priority Based Master Slave Communication using SPI Protocol", International Journal of Innovative Technology and Exploring Engineering (IJITEE) , ISSN:
2278-3075, Volume-9 Issue-9, July 2020
[7]Xingchun Liu,Yandan Liu "Multi-functional Serial Communication Interface Design Based on FPGA", 3rd IEEE International Conference on Computer and Communications,2017.