International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 05 | May 2020 www.irjet.net p-ISSN: 2395-0072

A Review on BIST Embedded I2C and SPI using Verilog HDL

Sumanth Sajjanar[1], Mrs. Deepika Prabhakar[2]

[1]Department of Electronics and Communication Engineering, RV College of Engineering, Bengaluru-59
[2]Assistant Professor, Department of Electronics and Communication Engineering, RV College of Engineering, Bengaluru – 59

Abstract - *This paper reviews I2C (Inter-Integrated Circuit) and SPI (Serial-Peripheral Interface) protocols with Built-in-self-test (BIST) mode. Both the protocols are used to connect two devices for exchanging data with each other in a quick fashion without any kind of data losses. Built-in-self-test (BIST) is a methodology that addresses the issue of self-testability with a powerful arrangement over costly circuit testing setup. The need of programming for setting up a system with two gadgets is not, at this point required in this designed framework. So as to accomplish minimal, steady and reliable information transmission, the I2C and SPI are structured with Verilog HDL.*

Key Words: Built-in-self-test, Inter-Integrated Circuit, Serial Peripheral Interface, Verilog Hardware Descriptive Language, Clock, Master, Slave, Data, System-on-chip

1. INTRODUCTION

Inter-Integrated Circuit (I2C) and Serial Peripheral Interface (SPI) are protocols which enable faster devices to interact with slower devices with no information misfortune. These are considered to be the best interfaces in a system that involves a number of devices connected to each other. The development of System-on-chip (SoC) poses a requirement of testing the interfaces of the embedded cores. Built-in-self-test (BIST) seems to be an efficient solution to the requirement.

Philips Semiconductors invented I2C in January 2000 [1] whereas SPI was invented by Motorola [3]. In [4] an efficient and a modern approach to design I2C and SPI with Built-in-self-test (BIST) is proposed and simulated it using Verilog HDL.[5] focuses on the advantages of BIST and its implementation is discussed. BIST gives the predetermined testability necessities and best execution with least cost. The quantity of modules and blocks used to structure these are generally less so that there is decrease in the testing unpredictability. This framework can be manufactured into a single chip. [6] and [7] show that the coding for this system is done using Verilog HDL & design, testing and evaluation are done using the ISE 6.0 tool of Xilinx and Verilogger Pro 6.5. The framework requisites of low bit error rate, high integration and low cost can be fulfilled by using I2C and SPI as seen in the results of [6] and [7].

2. ARCHITECTURES

The normal architectures of I2C and SPI protocols discussed in this section are understood from the papers [1], [2], [6] and [7].

2.1 I2C Protocol Architecture

Fig. 1 shows the I2C protocol architecture. [1] shows that the I2C protocol uses two main buses: data bus and clock. Data bus is a bidirectional pathway which can move information from master to slave and the other way around Clock is a unidirectional pathway which goes from the master to the slave gadgets. The clock is first generated by master device and initiates a data transfer. Slave at that point gives back an acknowledgement signal. [6] shows that the exchange of data mainly involves three components:

1) *Slave Address*: By this address, the slave device address to which the data must be transmitted is determined by the master. It is 8 bit long.

2) *Word Address*: By this address, the address of the data is conveyed by the master to the slave. It is also 8 bit long

3) *Data Value*: The master transmits the information to the slave in the form of data value. Data values consist of 8 bits.

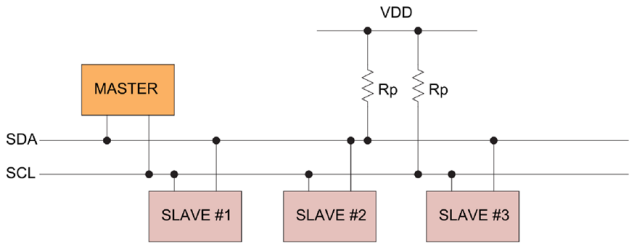

Fig -1: I2C Architecture [1]

2.1 SPI Protocol Architecture

[2] shows that SPI protocol consists of four different logic signals: SS (Slave Select) MISO (Master Input-Slave Output), MOSI (Master Output-Slave Input), SCLK. SCLK refers to the clock, a unidirectional pathway, which act as input to the slave. MOSI which is also known as serial data out is the output from master. MISO which is also known as serial data in is the output from slave. The slave device is selected by SS which is an active low signal. In SPI clock cycle a full duplex data exchange occurs. SPI data transfer system is shown in Fig. 2.

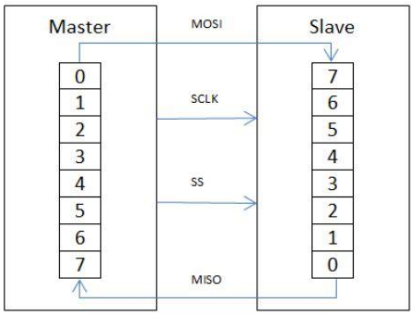

Fig -1: SPI Data Transfer System [2]

[7] shows that for transferring data to the slave from the master device to the slave device, mainly 3 components are required. SPI protocol data format is shown in Fig. 3.

Control	0	1	2	3	4	5	6	7
	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
Status	8	9	10	11	12	13	14	15
	SPIF	WCOL	-	-	-	-	-	SPI2X
Data	16	17	18	19	20	21	22	23
	MSB	-	-	-	-	-	-	LSB

Fig -2: SPI Protocol Bus Format [7]

1) *Control Address*: The control bus is represented by the first 0 to 7 bits in Fig. 3 and hence is 8 bit long. The SPI interrupt flag is enabled by SPIE which is interrupt enable signal. SPI is enabled by SPE. The data order is determined by the DORD. MSB will be sent first if DODR is 1. The selection of master or slave mode is done by MSTR. CPHA and CPOL are clock phase and clock polarity signals respectively. MOSI and MISO data shifted edges are determined by clock phase and clock polarity. The clock rate is determined by SPR0 and SPR1.

2) *Status Address*: 8 to 15 bits in Fig. 3 defines the status bus and hence is 8 bit long. The serial transfer is determined by SPIF bit. WCOL examines the transfer collision. Bit 10 to 14 are reserved bits. The clock speed can be doubled using SPI2X.

3) *Data Value*: 16 to 23 bits in Fig. 3 defines the data bus. Data values consist of 8 bits.

3. PROPOSED ARCHITECTURES

In [6] and [7] the architectures for I2C and SPI with BIST mode are proposed. Two modes of operations are involved. First is the BIST mode where the I2C or SPI tests itself. Second is normal mode. The device works like the usual I2C or SPI device in this mode.

3.1 BIST Module

The design technique that grants a circuit to examine itself is Built-In-Self-Test (BIST). Testing an embedded core in the System-on- chip (SoC) design poses a great challenge and hence BIST is useful. Both [6] and [7] use a similar BIST module.

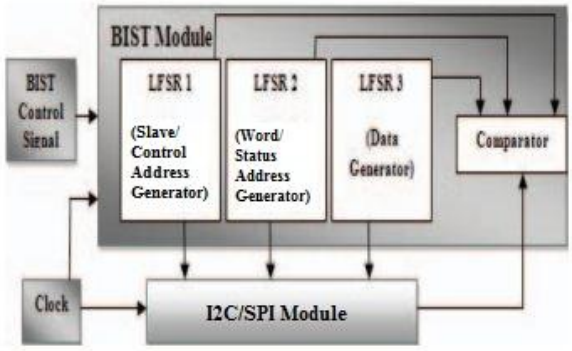

Fig -4: BIST Structure [6] [7]

The structure of I2C or SPI with BIST controlled by the BIST control signal is shown in Fig. 4. It has four functional blocks namely three LFSRs as random pattern generators and a comparator.

1) *Random Pattern Generator (RPG)*: Random Patterns need to be generated to verify the devices involving I2C or SPI. This is done by Random Pattern Generator. An efficient way to implement RPG is Linear Feedback Shift Register (LFSR). Three LFSRs are included in the proposed RPG. Slave address in case of I2C and control address in case of SPI is generated by LFSR 1. LFSR 2 produces the word address in case of I2C and status address in case of SPI. LFSR 3 gives the data for both I2C and SPI. Better fault coverage is obtained by using the generated bytes directly in I2C or SPI. Comparator evaluates the performance of I2C or SPI with these bytes.

2) *Comparator*: The transmitted and received bit patterns are compared by comparator. The value of the error is given if not matched,. If the output of comparator is 111 then the

device is working fine. If any other bit sequence is given, then it implies that there are some faults in the protocol.

3.2 I2C and SPI Structure

Table 1 depicts the operating modes of I2C and SPI. The operating modes are selected based on two control signals namely reset and reset_n signals. These operating modes are understood from [6] and [7].

Table -1: Operating Modes of I2C or SPI [6] [7]

reset	reset_n	BIST Mode	NORMAL Mode
1	0	OFF	ON
0	1	ON	OFF

The I2C bus format taken from [3] is shown in Fig. 5.

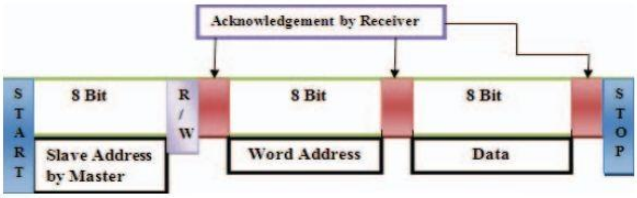

Fig -5: I2C Bus Format [3]

The step by step procedure of I2C protocol is described in [3]. Firstly, the master sends the start bit and initiates transmission. The data line going from high to low immediately after the clock line goes from high to low is the start condition. Then the slave's address is transmitted. An acknowledgement signal is sent to master by slave following which, the master transmits the register's address where the data needs to be saved which is again followed by an acknowledgement from the slave. Through the data bus, master then transmits the data to the slave. Once the data is received, the slave again sends an acknowledgement. In the end, master sends the stop bit which is indicated by the clock line going from low to high following the data line going from low to high.

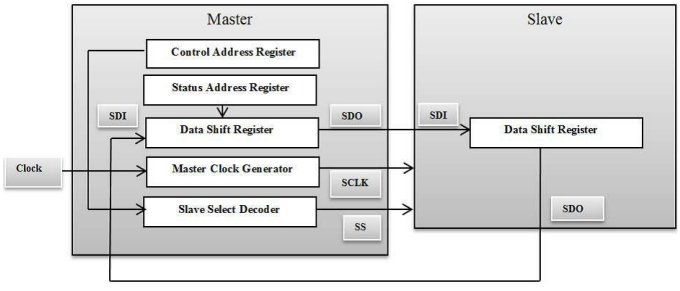

Fig -6: SPI Architecture [3]

In [3] the architecture of SPI is also described. The SPI architecture is shown in Fig. 6 as described in [3]. It has three registers namely Control, Status and Data registers. Data register is a type of shift register. When the data is transmitted to the slave from the master, it is Serial Data In (SDI) for slave and Serial Data Out (SDO) for master. The slave register starts to send data to master once it is completely filled. After this there is reversal of SDO and SDI. The clock is generated by Master Clock Generator and given to the slave. The slave device is selected by the Slave Select Decoder which is controlled by Control the register.

4.0 CIRCUIT SCHEMATIC

The pin diagrams, pin description and the BIST schematic are referred from [6] and [7]. The pin diagrams of I2C and SPI respectively are shown in Fig. 7 & Fig. 8. Table 2 and Table 3 refer to the pin descriptions of the implementation of the top level schematics on Verilog HDL depicted in Fig. 9 and Fig. 10. The pins are categorized into input and output pins in the table.

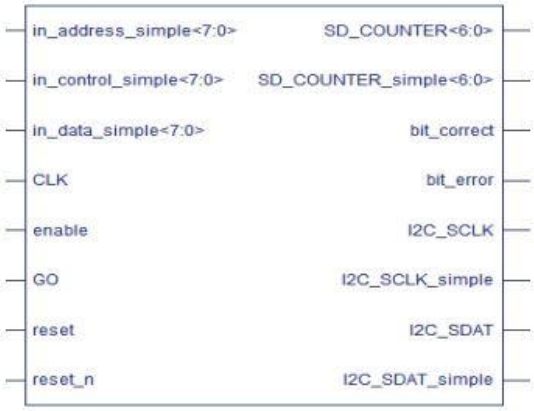

Fig -7: Pin Diagram of I2C [6]

Table -2: Pin Description of I2C[6]

Pin	IN/OUT	Description
CLK	IN	Clock generator
reset	IN	Bit for controlling BIST mode
reset_n	IN	Bit for controlling NORMAL mode
enable	IN	Enables the Random Pattern Generator
GO	IN	Control bit of the I2C
in_data_simple	IN	Input data byte of the I2C
in_control_simple	IN	Input control byte of the I2C
in_address_simple	IN	Input word address of the I2C
SD_COUNTER	OUT	CLK pulse counter for BIST Mode
SD_COUNTER_simple	OUT	CLK pulse counter for NORMAL Mode
I2C_SCLK	OUT	Output pin for I2C CLK for BIST Mode
I2C_SCLK_simple	OUT	Output pin for I2C CLK for BIST Mode
I2C_SDAT	OUT	Output data bus for BIST Mode
I2C_SDAT_simple	OUT	Output data bus for NORMAL Mode
bit_correct	OUT	Output pin of Comparator for correct bits
bit_error	OUT	Output pin of Comparator for wrong bits

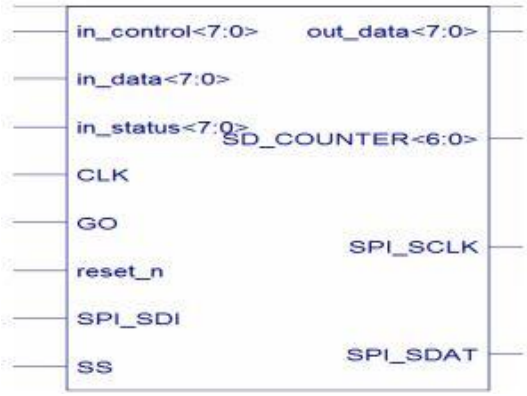

Fig -8: Pin Diagram of SPI [7]

Table -3: Pin Description of SPI[7]

Pin	IN/OUT	Description
CLK	IN	Clock generator
reset_n	IN	Control Bit for Normal & BIST Mode
GO	IN	Control bit of the SPI
in_data	IN	Input data byte of the SPI
in_control	IN	Input control byte of the SPI
in_address	IN	Input status address of the SPI
Out_data	OUT	Output Data byte of Master
SD_COUNTER	OUT	CLK pulse counter for BIST Mode
SPI_SCLK	OUT	Output pin for SPI CLK
SPI_SDAT	OUT	Output data bus
SPI_SDI	IN	Input data bus

BIST module top level schematic is shown in Fig. 9 and Fig. 10. One comparator and three LFSRs are included in this module.

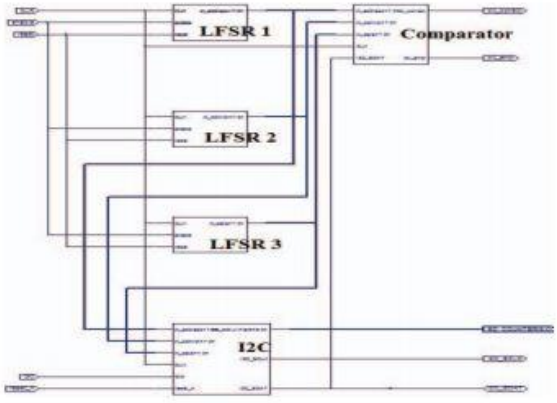

Fig -9: I2C BIST [6]

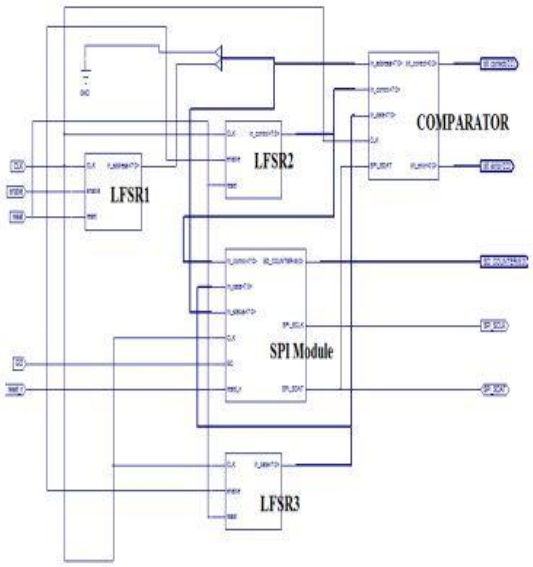

Fig -10: SPI BIST [7]

5.0 SIMULATION RESULTS

The waveforms obtained from Verilogger Pro 6.5 for both I2C and SPI are referred from [6] and [7]. The output is represented in 2 digit hexadecimal numbers.

5.1 BIST Mode Simulation Results

The simulation of BIST mode includes simulation of random pattern generator and the comparator.

a) Random Bit Pattern Generator (8 bits)

The random patterns generated by the 3 LFSRs are shown in Fig. 11 and Fig. 12. These patterns act as the input to I2C and SPI.

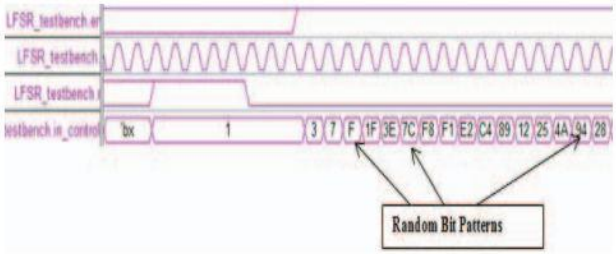

Fig -11: Random patterns for I2C [6]

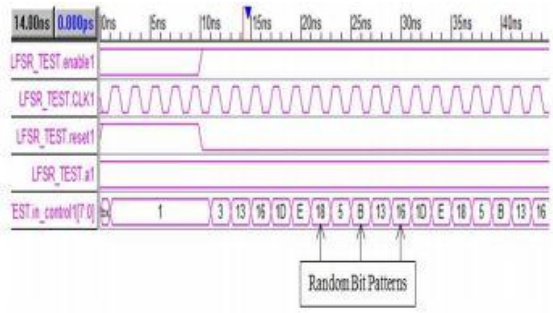

Fig -12: Random patterns for SPI [7]

b) BIST Mode Comparator Output

The waveforms of the I2C and SPI output in the BIST mode are shown in Fig. 13 and Fig. 14. Three LFSRs generate random patterns to act as control byte, word address and data. [6] shows that when "GO" is high and "reset_n" is low, then the random patterns are transmitted following the acknowledgement ("ACK") received from the slave device. Following the "START" bit, the bus carries the control byte. Consequently, the slave transmits an acknowledgement. The 8-bit word address or the memory address is transmitted following the acknowledgement. After this, again the slave sends an acknowledgement and 8 data bits are transmitted. Following this an acknowledgement is received from the slave and in the end, stop bit is transmitted to end the operation.

In [7] SPI BIST mode is described. The bit correct signal from the comparator is turned on if SPI module output matches with the control address, status address and data.

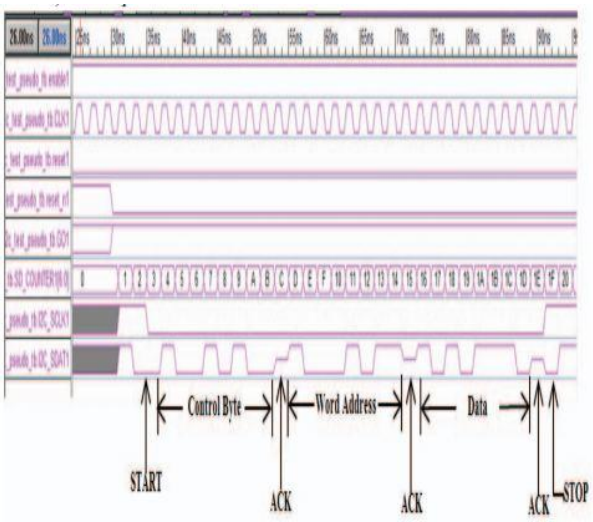

Fig -13: I2C BIST Mode Waveforms [6]

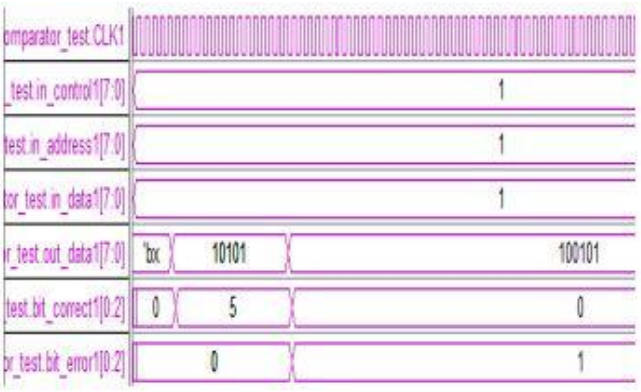

Fig -14: SPI BIST Mode Waveforms [7]

In Fig. 14, the signature value of bit correct is 101. The bit_error signal is turned on and the signature value is set as 001 if there is an error in the output stream. For example, if 1 is given to all the control, address and data bits and the data stream is 010101 and 101 must be the signature value if 10101 is the output which means that there is no error in the data. An error occurs and the bit_error line is 001 when the output is changed to 100101. Error is seamlessly detected with these two signature values.

5.2 Simulation Results for NORMAL Mode

In I2C, the NORMAL mode is shown in Fig. 15 which is referred from [6]. The hexadecimal values of Control Byte, Word Address and Data are CA, 00 and AA respectively which are converted to binary and loaded on I2C bus when "GO" becomes high and "reset_n" becomes low. Following the reception of each set of control byte, word address and data, the slave sends back an acknowledgement (ACK). START bit demonstrates beginning of activity and STOP bit shows the finish of activity.

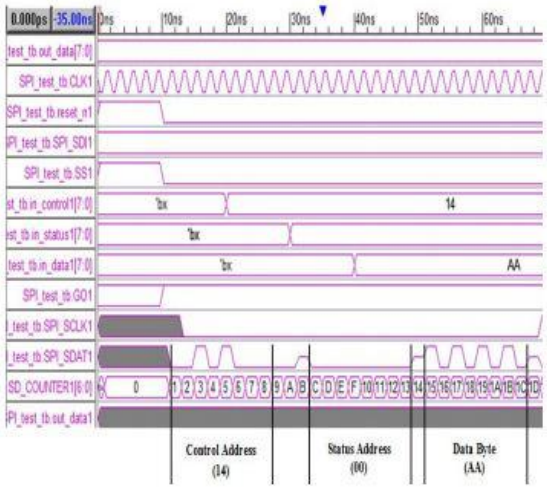

Fig -15: Normal Mode Operation of I2C [6]

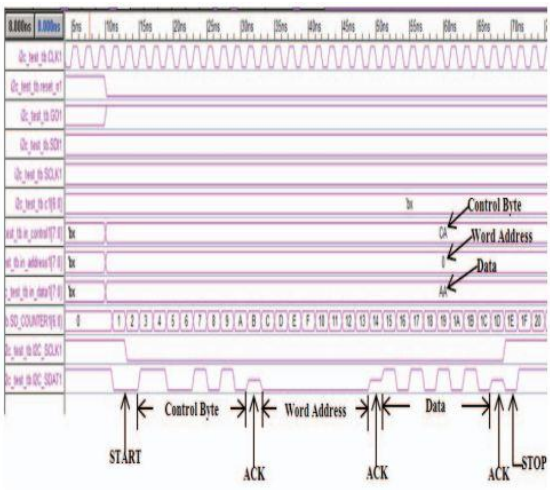

Fig -16: Normal Mode Operation of SPI [7]

In SPI, Fig. 16 shows the waveforms at NORMAL Mode referred from [7]. When "GO" signal becomes high, counting is begun by the "SD_Counter". The hexadecimal values of Control address, Status Address and Data are 14, 00 and AA respectively which are converted to binary and loaded on SPI bus as keeping the "SS" and "reset_n" low. Whether the given data is properly received or not is indicated by the SPI_SDAT line. 3 to A values of SD_Counter represent the control bits as first 8 bits. To the left, 1 bit is deliberately kept blank to prevent data collision. C to 13 values of SD_counter indicate status address and 15 to 1C indicating data byte.

6.0 CONCLUSION

In this review paper, design and simulation of I2C and SPI with BIST mode are reviewed. Verilog HDL is used to design and simulate all the modules. The normal and proposed architectures of both I2C and SPI are reviewed and concluded that the proposed architectures are efficient for testing the interfaces. As compared to the conventional interfaces, it is clear that both I2C and SPI are more speedy, flexible, stable and low cost as shown by the results of [6] and [7] it is clear that both I2C and SPI are much more speedy, flexible, low cost, and stable. The conclusion drawn is that BIST mode enables the industrial fabrication of chip with the capability of self-test with an addition of just one switch. A simple combination of three LFSRs and a comparator in BIST significantly saves time and cost of testing.

REFERENCES

[1] I2C Bus Specification, Version 2.1, Philips Semiconductors, 2000.

[2] Section 20. Serial Peripheral Interface ™, Microchip

[3] F. Leens, "An introduction to I2 C and SPI protocols," IEEE Instrumentation & Measurement Magazine, vol.12, no.1, pp.8-13, February 2009.

[4] A. K. Oudjida, M. L. Berrandjia, R. Tiar, A. Liacha, K. Tahraoui, "FPGA implementation of I2 C & SPI protocols: A comparative study," in Proc. 16th IEEE International Conference on Electronics, Circuits, and Systems, pp.507-510, Dec. 2009.

[5] S. Jamuna and Dr. V.K. Agrawal, "Implementation of BIST structure using VHDL for VLSI circuits," International Journal of Engineering Science and Technology, vol. 3, no. 6, pp. 5041-5048, June 2011.

[6] Shumit Saha, Md. Ashikur Rahman and Amit Thakur "Design and Implementation of a BIST Embedded Inter-Integrated Circuit Bus Protocol over FPGA", International Conference on Electrical Information and Communication Technology (EICT), 2013

[7] Shumit Saha, Md. Ashikur Rahman and 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