Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 12, December : 2024

DESIGN AND IMPLEMENTATION OF A BIST EMBEDDED INTER INTEGRATED CIRCUIT BUS PROTOCOL OVER FPGA

BHUMIREDDY VYSHALINI[1], G.LAKSHMI BHARATH[2]

[1]PG Student, Dept of ECE, SITS, Kadapa.

[2]Assistant Professor, Dept of ECE, SITS, Kadapa.

Abstract—

The I2C (Inter-Integrated Circuit) protocol is widely used for efficient and reliable communication between devices within complex integrated circuits (ICs), minimizing data loss. As IC designs grow in complexity, self-testability becomes essential to prevent product failures and ensure system reliability. A Built-in Self-Test (BIST) framework addresses this need by enabling automated system validation, improving cost-effectiveness and efficiency. Design and implementation of the I2C protocol with integrated self-testing functionality, eliminating the need for additional programming to configure device networks. The proposed design, developed in Verilog HDL, offers a compact and reliable solution for stable data transfer. Inputs to the design include the serial data (SDA) and serial clock (SCL) lines, which support communication between master and slave devices. The outputs encompass validation signals and fault detection indicators from the BIST.

Synthesis and simulation confirm the design's functional accuracy, demonstrating reliable data transmission, minimized power consumption, and enhanced speed with reduced delay. Test results show that the design improves communication reliability and simplifies testing by autonomously detecting and reporting faults. This BIST-enabled I2C implementation, with its efficient fault detection and stability, is a robust solution for modern IC applications, enhancing system resilience and reducing the risk of failures.

Keywords — Inter-integrated Circuit, Built-in Self-test Architecture, Verilog HDL.

I. INTRODUCTION

System-on-Chip (SoC) designs, in which complete systems, including processors, memory, communication protocols, and peripheral interfaces, are integrated into a single chip, are the result of the quick advances in integrated circuit (IC) technology. This integration is a crucial component modern electronics, especially in high-speed and power-efficient applications, as it significantly removes the need for external components, leading to lower power consumption, better efficiency, and decreased latency. However, in order to manage the speed and synchronization issues that arise, communication between various modules within a SoC frequently necessitates effective data transfer protocols. By utilizing two popular communication protocols—the Advanced High-performance Bus (AHB) and the Inter-

Integrated Circuit (I2C) protocol—this project aims to address these issues. Rapid data transfer in high-performance systems is made possible by AHB's well-known high bandwidth and full-duplex parallel communication capabilities. By offering a speed-matching mechanism between devices running at different clock speeds (such as GHz and MHz levels), I2C, on the other hand, shines in applications needing straightforward communication with fewer lines. The project's goal is to maximize the communication system's performance and power efficiency by integrating these protocols. By combining the AHB and I2C communication protocols onto a single chip, this initiative seeks to eliminate the need for additional peripherals and guarantee smooth, effective data flow. The project also intends to put in place a Built-In Self-Test (BIST) mechanism to guarantee the dependability of the master-slave communication on the Internet of Communications protocol. By automatically checking the system for errors and communication integrity, this BIST will increase the system's resilience and lower the chance of failure in intricate SoC designs.

The project's goal is to create a communication architecture that combines the I2C and AHB protocols into a single chip while utilizing each protocol's advantages to maximize power economy and performance. In order to guarantee the precision and dependability of master-slave communication over I2C, the design will integrate a Built-In Self-Test (BIST) mechanism into a smooth communication interface for effective data transfer. By identifying errors instantly, this BIST will increase the resilience of the system. The I2C protocol will be developed to meet AHB's high-speed needs, minimizing delays and maximizing data transfer rates in order to handle speed discrepancies between devices running at various clock speeds. Additionally, in order to reduce the need for external peripherals and make the design more compact, the project will prioritize maximizing area efficiency and lowering power usage.

The I2C protocol's SDA (Serial Data) and SCL (Serial Clock) lines serve as the design's main inputs. With SDA transporting the data and SCL supplying the clock signal, these inputs enable communication between the slave and master devices. Additionally, control signals are sent to the system to initiate the self-test mechanism and configure the communication process. Results of the Built-In Self-Test (BIST) mechanism's fault detection and validation signals verifying successful data transmission are among the design's outputs. To make sure that the master-slave communication works as intended, the BIST outputs will offer input on the condition of the communication system.

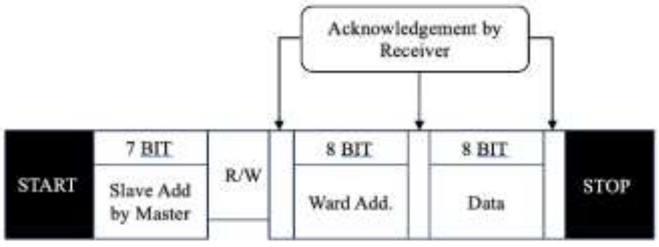

Fig.1. Format of I2C Protocol Bus

II. LITERATURE SURVEY

The AMBA AHB connection matrix and I2C Design Under Test (DUT) SoC level verification is described in the paper "SoC Level Verification Using System Verilog," which was published by P. D. Mulani at the 2009 Second International Conference on Emerging Trends in Engineering & Technology. The monitor and assertions are designed to verify that the multilayer AHB Lite's interconnect matrix is operating as intended. System Verilog hardware description and verification language are used for the entire verification process. The created verification environment can be used again. An interface idea is used to bridge the gap between AHB and inter-integrated circuits. The two designs are connected, and a few test cases are included in the verification to ensure that the design functions flawlessly for both scenarios. The bridging of both APB and AHB protocols with I2C is described in the work "Implementing communication bridge between I2C and AHB" by Busireddypally Jayaramudu et al. The interfacing notion is the methodology employed. The area, latency, and power checking of both APB and AHB tested independently with I2C are addressed in the final analysis. The Xilinx platform is used for the design, and the results are validated. The waveform provides a thorough analysis of the port mapping in the interacting block. Here, the AHB design serves as a slave, assisting IIC in disseminating the necessary data. Before being mapped in the interface block to be confirmed as a single module on the verification side, the blocks are formed as distinct modules.

Both the APB and AHB cases in the whole work often employ this methodology. Prabhakar et al.'s "ARM Advanced High-Performance Bus (AHB)Complaint Inter Integrated Circuit" combined AHB with I2C for data transfer, resulting in the correct waveform analysis results. The integration of two modules into one is explained in detail in this publication. In this case, the AHB is the master and the IIC is the slave, obeying orders and producing reliable results. Here, the wrapper logic approach is utilized to merge both modules. The signals from the AHB master are wrapped, decoded, and sent to the IIC slave appropriately using a separate logic for the wrapper block.

III. IIC WORKING SPEEDS AND MODES

A member of the ARM Advanced Microcontroller Bus Architecture (AMBA) protocol family, the Advanced High Performance Bus protocol addresses high frequency devices such as processors and Universal Asynchronous Receiver Transmitters (UART). It is a high performance protocol that uses a unique phasing idea in this protocol definition to improve device performance. It operates using a two-phase idea that will activate the data phase and the address phase in two cycles. The relevant data phase occurs in the subsequent clock cycle, not on its own, for each address phase. This idea is known as the AHB's pipelining architecture. As a result, the system performance that this protocol designs will be improved.

a. IIC

An inter-integrated circuit (IIC) is a block that facilitates communication between an integrated chip (IC) and a processor. Any CPU will operate in the high GHz frequency range. An IC, on the other hand, operates in the low frequency range of KHz or MHz. If a processor is just connected directly to an IC, this speed difference cannot be filled. In the end, issues like data loss, data mismatch, or perhaps even data trafficking will arise from the speed discrepancy. The gap must be filled in order to prevent this. This is carried out when a bus interface receives and debugs the processor's speed issue before sending it to the IC in accordance with its speed. Usually, this problem is resolved by utilizing a few external peripherals and at least two separate blocks between the processor and the IC. This will result in further power and area problems. This work aims to address this issue.

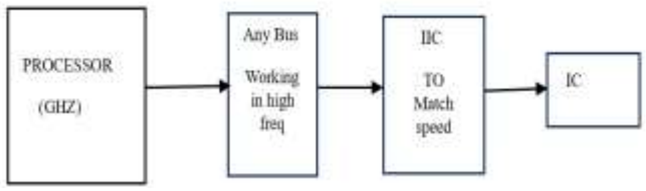

Fig.2. General processor to IC connection

b. IIC Working Speed

IIC operates at three distinct speeds depending on the mode of operation. IIC operates at 100 Kbps in regular mode, 400 Kbps in fast mode, and 3,5 Mbps in high mode. To construct the prescale register, which is the primary source for speed matching to bridge the gap between the processor and an IC, the mode on which the intended IIC will operate must be determined beforehand. The IIC in this architecture operates at 100 Kbps in standard mode. The following formula is used to construct the pre-scale register using the desired speed:

$$\text{Pre scale} = \frac{\text{AHB frequency}}{5*\text{SCL}} - 1$$

IV. METHODOLOGY

All of the signals are port mapped from AHB to IIC in this AHB and IIC interface architecture, ensuring a direct contact between the two modules. Here, the IIC is the slave and the AHB is the master. Since there is no interface or wrapper block between the slave and the master, any trigger in the master will have an immediate impact on the slave. By doing this, the extra space for all of these intermediary blocks is further avoided. There is just one intermediate block in the communication between the CPU and an IC because the AHB and IIC are made as a single module.

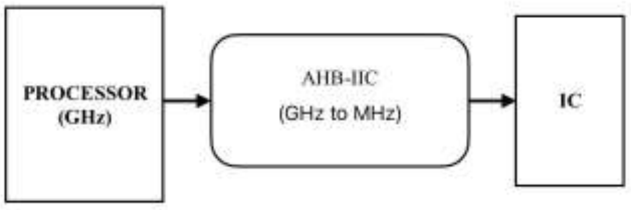

Fig.3. Processor to IC connection as per the proposed design

This will still more reduce the area and the external peripherals used is also avoided as both are been combined as one single chip.

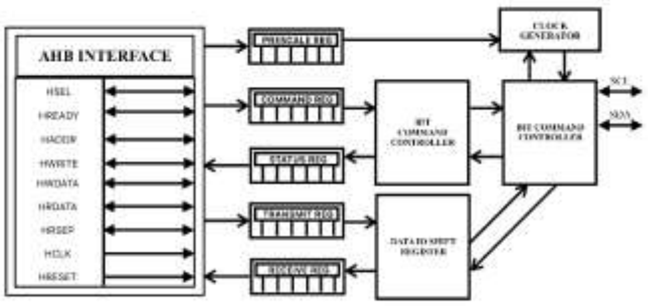

Fig.4. Framework of the design

Fig. 4 provides the structure of the overall design used here. Signals are produced in accordance with the specifications, and various blocks are designed in accordance with the AHB and IIC architectures. Each module performs the signal mapping, which highlights the relationship between the slave and the master.

a. I2C PROTOCOL EACH I2C

SCL and SDA are the two indicators that make up transport. The clock flag is SCL, while the information flag is SDA. Some slave devices may occasionally force the time low to delay the master delivering more information (or to demand more opportunity to plan information before the master strives to check it out). The clock flag is continuously produced by the current transport master. It's referred to as "clock extending" and on the convention page. According to I2C tradition, this would enable several slave fused circuits to interact with at least one circuit specialist. Every data trade has two requirements. They are in a state of starting and stopping bits. The condition starts when the start condition is acknowledged and ends when the stop condition is executed.Transport is thought to be engaged when the start condition arises and will be necessary in a similar state until all bus sales have been approved. As illustrated in figure 1, the slave's address is sent first for the read/make assignment, followed by the looking at data. Following each byte is the recognize. The recipient can flag the transmitter thanks to the recognize bit. that second byte might be transmitted after the byte was successfully obtained. All ticker beats, including the ninth clock beat, are produced by the master. The clock is defined as follows the transmitter discharge, followed by the SDA line during the detectable clock beat. Thus, during this clock beat's high period, the collector can pull the SDA line low and it will remain stable.

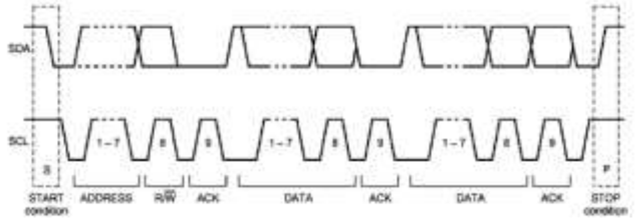

Fig.5. I2C Data Transfer Protocol

b. APBPROTOCOL

IDLE, the default state, is used to explain the edge cycle. SETUP: The bus enters the SETUP state, when the choose flag, PSELx, is avowed, when an exchange is necessary. After one clock cycle, the exchange switches to the ENABLE state and continues to run with the clock's rising edge. Attract: PENABLE, the enable hail, is given. While moving from the SETUP to ENABLE state, the address, make, and select signs must be steady. The car goes back to the IDLE condition if no more exchanges are needed. It goes without saying that the exchange will go to SETUP if another move needs to be made. In the middle of the advance, it would be possible to address, create, and choose signs.

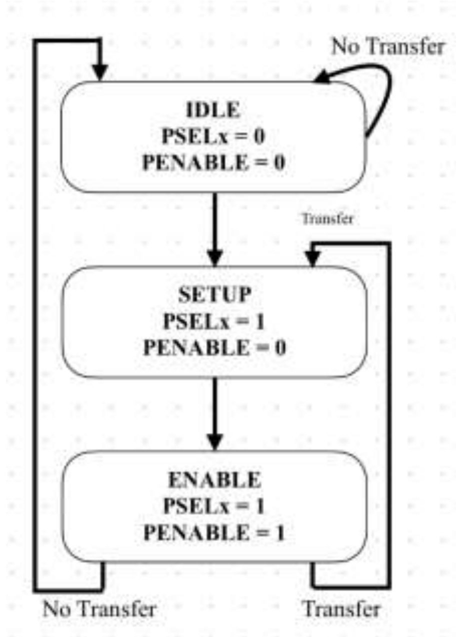

Fig.6. State diagram for state diagram

V. DESIGNED ARCHITECTURE

The finalized correspondence assistants between I2C and APB in an illustrative square arrangement. The APB Master and I2C Slave are the two fundamental components of the structure. I2C Slave provides the APB Master with the information that I2C Master has provided in an isolated course of action. In the APB Protocol, this APB Master also relays this information to the APB Slave. This concludes the correspondence between the APB Slave and the I2C Master.

a. Write Operation

1. The frameworks for the I2C Slave permit communication between the APB Slave and the I2C Master when necessary.
2. The I2C Slave will confirm the Data Valid and Address Valid signals.
3. The APB Master starts the APB state machine and verifies that the memory is ready after sensing these signals.

4. Four 8-bit data packets are serially transmitted via the I2C and stored in four distinct locations within the APB Memory.
5. The APB Master confirms that all four memory locations are updated accurately after each byte has been transferred. The APB Master sends the entire 32-bit data to the APB Slave after it has been updated.

b. Read Operation

1. Communication starts from the APB Master to the I2C Slave and then back to the I2C Master when the I2C Master wants to receive data from the APB Slave.
2. To let the APB Master know that the data is prepared for reading, the APB Slave sends a flag
3. The APB Slave sends the information to the APB Master after obtaining this flag.
4. This information is kept in the internal memory of the APB Master
5. When necessary, the I2C Slave can retrieve this data from the APB Master's memory, making it accessible to the I2C Master at the appropriate moment.

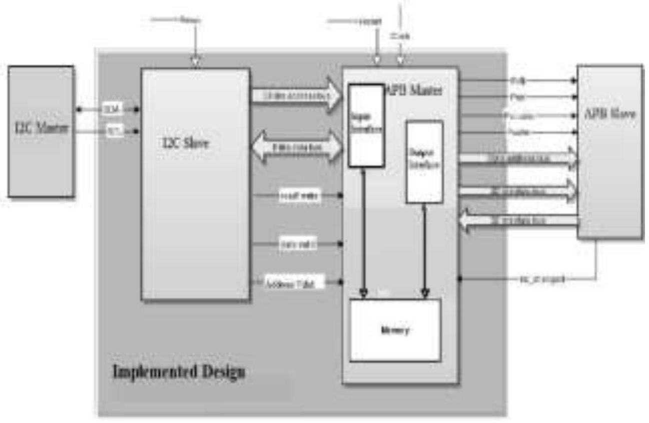

Fig.7. Communication Bridge

c. AHB APBInterface

The AMBA APB Master and AHB slave Focus AHB to APB provides an interface (associate) between the low-control APB area and the rapid AHB space. Core AHB to APB interacts with either Core APB via the APB interface or Core AHB/Core AHB Lite via the AHB interface.

d. Key Features

1. Connections between Advanced Peripheral Bus (APB), Advanced High-Performance Bus (AHB), and Advanced Microcontroller Bus Architecture (AMBA).
2. Smart Design's automatic connection to Core AHB/Core AHB Lite and Core APB.
3. AMBA and APB are compatible.

e. Maintained Interfaces

Center AHB to APB supports both an AMBA APB pro interface that interacts with an AMBA APB reflected pro interface (Core APB, for example) and an AHB or AHB-Lite slave interface associated with an AHB or AHB-Lite reflected slave interface (Core AHB or Core AHB Lite).

f. Block diagram

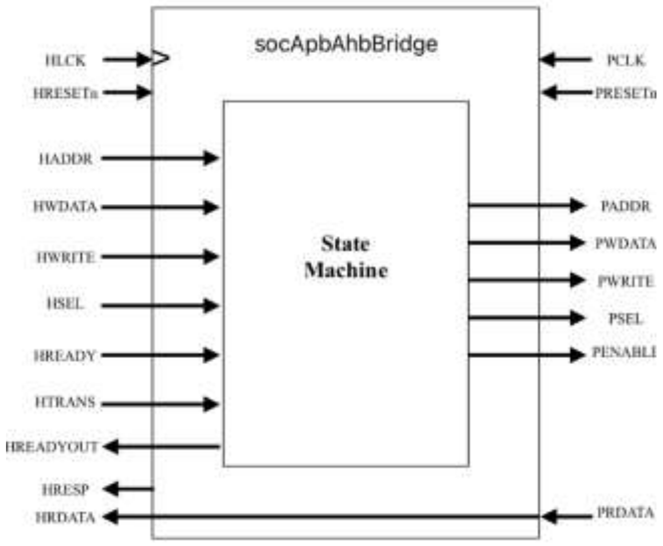

Fig.8. AHB and APB Bridge

g. Description

AHB indicators are converted to APB signals using the I2C-APB Bridge. Additionally, the tip top AHB (system transport) is separated from the slower APB (peripheral bus) by the I2C-APB Bridge. An AHB slave component called the I2C-APB Bridge can identify transactions that are focused on an APB periphery, decipher the address, and send an APB, periphery transport, or trade to the focused-on peripheral or memory. Up to sixteen APB peripherals can be decoded by the I2C-APB Bridge. The I2C-APB Bridge provides the focused-on periphery or memory with form control (PWRITE), select (PSELx), address (PADDR), and data (PWDATA) during creation transactions. The I2C-APB Bridge multiplexes the focused-on peripheral data (PRDATA_device) to the AHB HRDATA in the best possible arrangement on read trades. The Bridge between I2C and APB.Additionally, it restores the HREADYOUT movement to the AHB pro to demonstrate that the data is ready and the IPC-APB Bridge has finished the APB trade.

h. Interfacing to APB to AHB

At time T1, the exchange begins on the AHB, and at time T2, the APB interface looks into the address. This address is communicated and the optimal periphery choose flag is generated if the deal is for the periphery exchange. When the PENABLE banner is verified, the ENABLE cycle follows the SETUP cycle, which is the initial cycle on the perimeter exchange. The peripheral must provide the read data during the ENABLE cycle. This read data can be routinely routed, especially back to the AHB, where the transport ace exchange can test it at time T4, which is the rising edge of the time approaching the end of the ENABLE cycle.

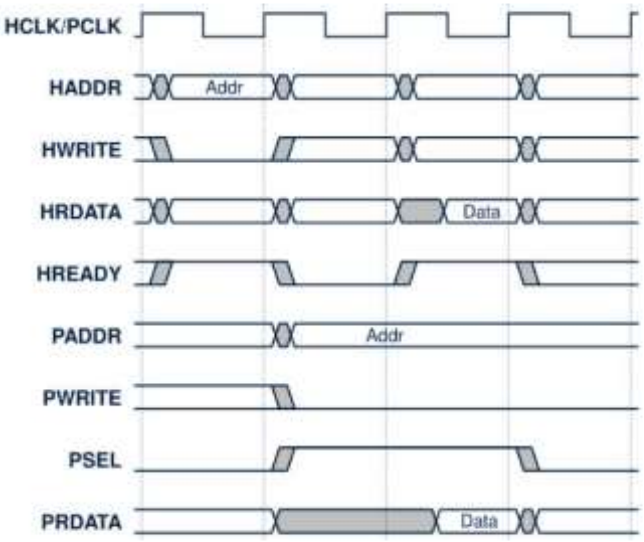

Fig.9. Interface APB to AHB

i. Write Transfer

At time T1, the transfers start on the AHB, and at time T2, the APB interface checks the address. This address is communicated and the appropriate periphery select flag is given if the deal is for the periphery exchange. When the PENABLE flag is asserted, the ENABLE cycle follows the SETUP cycle, which is the initial cycle on the peripheral exchange.

The peripheral must provide the read data during the ENABLE cycle. This read data can be consistently routed, especially back to the AHB, where the exchange master can test it at time T4, which is the rising edge of the time approaching the end of the ENABLE cycle.

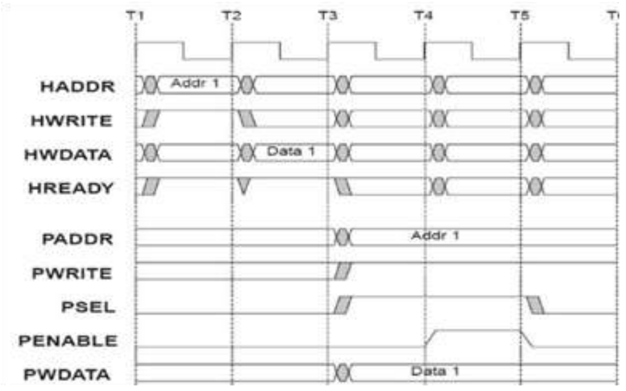

Fig.10. Write Transfer

VI. RESULTS

The primary inputs for the design are the SDA (Serial Data) and SCL (Serial Clock) lines of the I2C protocol. These inputs allow communication between the slave and master devices, with SDA carrying the data and SCL providing the clock signal. Furthermore, the system receives control signals to set up the communication process and start the self-test mechanism.

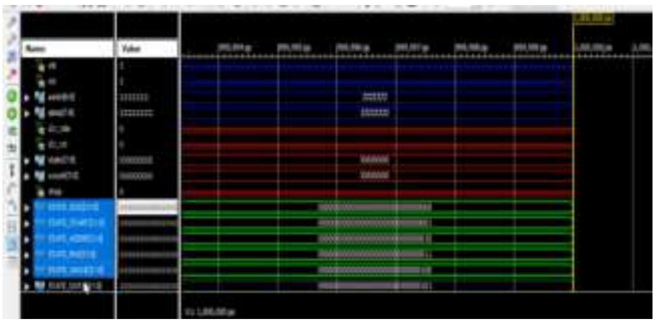
Fig.12. Input Simulation of I2C

The design's outputs include the Built-In Self-Test (BIST) mechanism's fault detection results and validation signals confirming effective data transmission. The BIST outputs will provide information on the state of the communication system to ensure that the master-slave communication functions as planned.

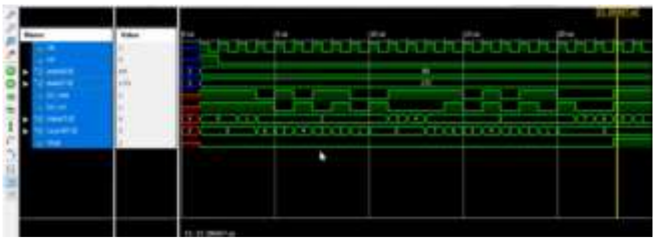
Fig.13. Output Simulation of I2C

CONCLUSION

As a result, the AHB-IIC module is designed and its outputs are verified. As a result, the developed module combines the benefits of both AHB and IIC and promises a high performance, speed matching module that will reduce the area and external peripherals while improving the efficiency of communication between the processor and an IC. The two cycle phase concept of the AHB also increases communication flexibility, which results in a very positive consequence. A 100% functional coverage is achieved. To improve consistency, the suggested personality has been used to arrange the data exchange speed of both vehicle transfers. In order to achieve a significantly faster rate of data trade, we plan to design a model with additional interface features. By extending the help duration at the built APB master interface, latency and the credibility of data loss can be reduced. Maintaining the impacted APB's working frequency allowed it to achieve amazing I2C mastery. Discuss strategies for managing and carrying out reading operations.

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 Testbench Architecture for Coverage Driven Functional Verification of SPI Protocol”, 2018 International

Conference on Advances in Computing, Communications and Informatics (ICACCI)I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[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, AzilahSaparon, " IP Core of Serial Peripheral Interface (SPI) with AMBA APB Interface", IEEE 9th Symposium on Computer Applications & Industrial Electronics (ISCAIE), 2019.

[5] ShumitSaha, 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.

[8] Bhagyashri,G.Allawagol,Vidyashree AC,Narendra Kumar,, "Design of SPI IP to communicate with I2C bus of a microcontroller" International Journal of Industrial Electronics and Electrical Engineering, ISSN: 2347-6982, Special Issue, Sep.-2016.

[9] Xiaole Cui, Miaomiao Zhang, Qiujun Lin, Xiaoxin Cui, Anqi Pang, "Design and Test of the In-Array Build-In Self-Test Scheme for the Embedded RRAM Array",IEEE ELECTRON DEVICE LETTERS, VOL. 38, NO. 5, MAY 2017.

[10] Abhas Singh, Gurram Mahanth Kumar, Abhijit Aasti, "Controller Architecture for Memory BIST Algorithms" IEEE International Students' Conference on Electrical, Electronics and Computer Science, MAY 2020.

[11] P. Venkateswaran, M. Mukherjee, A. Sanyal, S. Das, and R. Nandi, "Design and Implementation of FPGA Based Interface Model for Scale Free Network using I2C Bus Protocol on Quartus II 6.0," in Proc. 4th International Conference on Computers and Devices for Communication, Dec. 2009, pp. 1-4.

[12] R. Singh, and N. Sharma, "Prototyping of On-chip I2C Module for FPGA Spartan 3A series using Verilog," International Journal of Computer Applications, vol. 68, no. 16, April 2013.