DIGITAL COMMUNICATION SYSTEMS: SPI MASTER-SLAVE PROTOCOL DESIGN AND VERIFICATION IN VERILOG

[1]Vankdoth Bhavani, [2]Dr.Malothu Amru

[1]M.Tech Scholar, Department of Electronics and Communication Engineering, CMR Engineering College, Medchal, Kandlakoya, Seethariguda, Telangana, India

[2]Professor, Department of Electronics and Communication Engineering, CMR Engineering College, Medchal, Kandlakoya, Seethariguda, Telangana, India

ABSTRACT: SPI (Serial Peripheral Interface), which was introduced by the company Motorola, and a protocol for communication of serial synchronous about the communication between the master and slave devices, which is also used to provide communiqué between microcontroller and many devices which are additional and similar to external Analog to the Digital Converters, Digital to Analog Converters, and EEPROMs. Nowadays, communication protocols are at low end. This model presents the design and verification of a Serial Peripheral Interface (SPI) Master–Slave protocol using Verilog HDL and adopts a four-wire interface (MOSI, MISO, SCLK, CS). The multiple operating modes are supported by enabling full-duplex communication, with configurable parameters such as 8-bit data width and programmable CPOL/CPHA settings. The master core uses a finite state machine (FSM) with IDLE, START, TRANSMIT, and FINISH states for reliable data flow control. The Master-driven clock domain integrates Slave with synchronization and edge detection logic to ensure accurate operation. In a SystemVerilog/UVM-based environment, verification is done and includes directed and random test scenarios, scoreboards, and assertions to validate protocol timing, error handling and compliance. All protocol modes are further ensured when functional coverage metrics and boundary conditions are tested. Therefore, the design achieves robust communication, configurability, and protocol compliance. It is suitable for integration into digital communication systems requiring low-cost, reliable serial interfaces.

KEYWORDS: SPI (Serial Peripheral Interface), SCLK (Serial Clock), MOSI (Master Out Slave In), MISO (Master In Slave Out), CS (Chip Select), Digital Converters

I. INTRODUCTION

There are many communication protocols for both long and short distance communication purpose. Universal Serial Bus (USB), Serial Advanced Technology Attachment(SATA), Peripheral Component Interconnect- EXPRESS & ETHERNET are basically used for long distance, while Inter Integrated Circuits and SPI protocols are used for shorter distance communications[1]. SPI has high transmission speed compared to other protocols, and it is simple to use. Standard Serial Peripheral Interface protocol requires at least 4 interfaces. The devices that are based upon SPI protocol, it is divided into two, one is master device and another one is slave device[2]. The chip contains a select signal such as clock signal, which is provided from the master when exchange of data has been processed, it is often considered as the “little” protocol communication which is used for an On-Board communication[3]. This paper attempts to implement design of protocol and this design methodology can be reused. We first make a study of significant commercial SPI devices (datasheets) from dissimilar sellers to look at the necessities and then include the requirements features that satisfy ASIC/SoC applications[4].

SPI has high-speed with full-duplex and synchronous communication bus, which is used for information transmission between microcontrollers and peripherals. SPI is mainly used for saving the chip ports and gap on Printed Circuit Boards(PCB) layout[5].

It works with Master-Slave configuration, in full duplex mode there will be single master device and single slave device[6]. It requires only four lines, and the components used are Serial Data Input, Serial Data Output, Serial Clock (SCK), and Slave Select (SS)[7]. When the master of SPI wants to send data to slave, then SS line will get low for selecting slave, the clock signal will be activated, so that both master and slave can use it at the same time[8]. Master sends data to MOSI line (master's Serial Data Out and Slave's Serial Data Input) and itreceives the data from MISO line(master's Serial Data Input and slave's Serial Data Output)[9]. Clock pulse was provided by Serial Clock and Serial Data Input, Serial Data Output, these all are based on the pulse to make the information transmission., Data output is done through the SD Oline of master's either by rising or falling edge of the clock, and it is read by the slave in falling or rising edge followed. So, eight-bit data transfer needs at least eight times the clock signal changes[10].

Communication is always initiated by master, and will configure the clock first using that configure frequency that which must be less than or equal to frequency which is maximum then the slave device[11]. The master will select a desired slave for communication by pulling a chip through SS line of aslave-peripheral that is particular to low state. Shift register will organize for usage of data transfer with the given word sizes such as 8-bits in master device and slave device and they will connect in a ring[12]. MOSI line is used by the master to shift the register value, then slave shifts the data in to shift register and then the data is sent to master by slave through the line MISO[13].

Verifying the SPI protocol ensures reliability, robustness, and correctness in modern embedded systems, particularly for IoT and wearable devices. A structured verification approach must comprehensively evaluate the SPI protocol's behavior under various conditions to confirm its adherence to industry standards. One of the primary goals of SPI verification is ensuring protocol compliance[14]. This involves validating the correct operation of SPI modes, including proper clock polarity and phase settings, and confirming that the chip select signal behaves as expected to prevent unintended slave activation. Additionally, adherence to standard timing requirements, such as setup and hold times for data transmission, must be verified. Full-duplex communication must also be tested to maintain data integrity between the master and slave devices. Another crucial verification aspect is error handling and fault detection, which ensures that SPI communication remains reliable under various fault conditions. This includes detecting and addressing clock jitter, bus contention, and framing errors that may result from incorrect bit alignment or premature termination of communication. The system's response to unexpected events, such as simultaneous multiple slave selection or out-of-range clock frequency settings, must also be assessed[15]. Furthermore, verification must confirm that the protocol can handle transmission interruptions and implement retry or recovery mechanisms when necessary. Functional correctness is vital in SPI verification, as the interface must operate reliably across different scenarios.

This includes verifying data transmission integrity for various word lengths, ensuring that MSB-first and LSB-first transmission formats are correctly implemented, and validating the execution of register configurations that control SPI operations. Additionally, as the system's complexity increases, the verification process must confirm that SPI performs correctly in both austere and multi-slave environments.

II. LITERATURE SURVEY

V. U. Royal, A. Vijay Kumar, R. Sumathy, P. Durga Prasad, D. Gnaneshwar and U. T. Kumar Reddy, et al.[16] modern SoC designs, efficient data communication between high performance cores and low-speed peripherals is crucial in system functionality. The AMBA uses this kind of effective transaction and high SoC performance. Using AMBA as a reference, a hierarchy was created with protocols to facilitate efficient transactions between protocols from a (AXI) high-performance and high-bandwidth to low-performance and low-bandwidth protocol (SPI). This model requires a bridge between the protocols in order to regulate the transaction flow when managing high-speed transactions. By analysing the hierarchical transaction flow across these protocol bridges, this system can clearly observe how data moves across different protocol modules, ensuring efficient system performance.

S. Sasikumar, B. A. Balaji, N. J. Krishna and S. Reddy , et al.[17] InterIntegrated Circuit (I2C) offers a two-wire bus solution that enables many system devices to communicate with one another. Synchronous serial communication, or SPI, is widely used in embedded systems, where microprocessors and microcontrollers communicate with peripherals, enabling shortrange data transmission. In the meantime, I2C which has a reputation for having a slower communication speed works best in promoting communication between integrated circuits. In order to provide smooth communication between SPI and I2C, two commonly used serial communication protocols, this work aims to construct and simulate a Protocol Conversion Unit (PCU). I2C just needs two wires since it doesn't require a separate slave select line as SPI requires. The address and acknowledgment protocol is used in I2C instead of other protocols for communication with the slave.

S. Penurkar, A. Kumar, S. Srivastava and S. Nakhate, et al.[18] presents the design and analysis of a Single Master Single Slave Serial Peripheral Interface (SPI) system implemented on an FPGA. SPI is a communication protocol that enables data transfer between devices by designating master and slave roles. Despite its utility, existing peripheral communication protocols have limitations. While parallel communication requires extensive wiring and is prone to noise and crosstalk, serial communication can be slower, consume more power, and result in lower throughput. This study aims to develop an SPI module optimized for single-slave operation. Simulations conducted using AMD Vivado confirmed the module's functionality, and the design was successfully implemented on a Nexys 4 DDR FPGA board.

A. Maity, P. K. Samanta, B. P. De, S. K. Dash, W. Bhowmik and A. Bakshi, et sl.[19] design and simulation of a Serial Peripheral Interface (SPI) transceiver module using VHDL is being dealt with in this paper. SPI, a widely used communication protocol for communicating between micro-controllers and peripheral devices, is being utilized. VHDL (VHSIC Hardware Description Language), a hardware description language, is used for modelling digital circuits and systems. The work involves designing the SPI transceiver using VHDL and simulating it using a hardware simulator to ensure proper functionality. The design has been implemented on an FPGA board, and the performance of the SPI transceiver is being evaluated in terms of speed and power consumption.

M. Trehan, P. Kumar and N. Gaur, et al.[20] Multi-Protocol Conversion Unit (MPCU) is designed and simulated using Hardware Descriptive Language (HDL). This unit acts like a bridge and can perform data communication between three different protocol sets which are among the most prevalent methods used for serial communication of data. The following protocols are Serial Peripheral Interface (SPI), Inter Integrated Circuit (I2C), and Universal Asynchronous Receiver Transmitter (UART). The designed conversion module will take input from any of the three different protocols along with a value of Conversion Select (COSE) input. In accordance with the input value of COSE, the data will be transferred from the data bus that is within the MPCU module to any one of the three protocols slave. Thus, in terms of prototyping devices if there is a need for receiving data from any of the protocols and to send that data onto a slave based on different protocols there is no need for the data to be processed by the microprocessor or microcontroller.

S. Karthick, S. Venkatesh, P. S. Chaithanya and G. C. S. Royal, et al.[21] Software-Defined Vehicles (SDVs) have recently paved the way for a novel approach in the automotive industry where most of the vehicle's functionalities are controlled through software. This transition requires that the EE architecture be optimized so as to support efficient and reliable inter-subsystem communication. Specifically, this research addresses performance optimization of EE architecture specifically through evaluating and fundamental building blocks such as I2C, SPI, and CAN frame protocols. Inter Integrated (I2C) and Serial Peripheral (SPI) interfacing and protocols are widely used for short range intra system communication and both possess different merits and demerits depending on the requirements for speed, power consumption and complexity.

B. Jose and J. S. Immanuel, et al.[22] Serial-Peripheral Interface(SPI) Protocol also called as synchronous serial interface specification is used for communication between single master and single/multiple slaves. With the increase in number of slaves causing high complexity of circuit creates a demand in self testability feature for SPI module in order to test for fault free circuits. Built-In-Self-Test(BIST) is the answer for self-test in circuits as well as it helps in reduction of maintenance and testing cost. Design of BIST embedded SPI module with Single Master and Single Slave configuration has been introduced in this paper, here 8-bit data is transferred across the module ,where the circuit under test(CUT) is being self-tested with BIST feature for it's correctness. This SPI module is designed using Verilog Hardware Description Language(HDL) using EDA playground platform for applications like Application Specific Integrated Circuit(ASIC)or System on Chip(SOC).

Y. Si, N. Korada, R. Ayyanar and Q. Lei, et al.[23] presents a high performance 4-layer communication architecture for a smart micro-grid testbed which consists of a 2 kVA Distributed Energy Resource (DER) inverter with PV and battery channels capable of advanced grid supportive and grid forming functions in the Process layer, a Raspberry Pi computer in the Interface layer, a customized Edge Intelligent Device (EID) in the Sub-station layer, and an end-to-end solar energy optimization platform (e-SEOP) in the Supervisory layer. The Raspberry Pi serves as a communication interface which communicates with the DER inverter using Serial Peripheral Interface (SPI) communication and talks to the EID using Modbus TCP/IP protocol. The challenges of coordination between communication and system control and between different communication protocols have been addressed, which leads to a boost in the communication efficiency and makes the system highly scalable, flexible and adaptive.

J. P. Kadambarajan, P. Kadarkarai, B. Kalyani, M. R. Bathul, T. Sandhya and M. Greeshma, et al.[24] presents the design and implementation of an SPI verification monitor module using the Universal Verification Methodology (UVM). The monitor is tailored to observe, track, and validate SPI transactions within a testbench environment. The paper details the SPI protocol's intricacies and challenges during verification, emphasizing the need for robust monitoring solutions. Leveraging UVM's structure and methodology, the monitor module is architected to capture and analyse SPI interface behaviors, including data transfers, clocking, and control signals. Implementation details encompassing the monitor's interaction with the design under test (DUT) through a comprehensive testbench infrastructure are elucidated. The verification flow incorporates sequences, drivers, and a scoreboard within the UVM framework, ensuring thorough validation against expected behaviors. Simulation results demonstrate the efficacy of the monitor in detecting protocol violations, timing discrepancies, and data integrity issues.

O. Coşkun, E. Egeli, E. Tarcan, İ. Kurtuluş and G. Yılmaz, et al.[25] implications of using the daisy-chain configuration for SPI protocol-based communication between a PLC-CPU and multiple expansion modules have been thoroughly investigated. Equations for clock cycles and data length, for write/read functions have been derived and validated using oscilloscope outputs. Write and read functions have been confirmed by LEDs and the Modbus Poll software, respectively. A comparative analysis has been carried out to highlight the advantages and disadvantages of the daisy-chain configuration compared to the regular SPI configuration. When communicating with two slaves, the daisy-chain configuration required 1.25 times more bytes and time for writing than the regular configuration, while the read function remained the same as the regular configuration. When communicating with single slave, it required 2.5 times more bytes and time for writing and doubled for the reading.

III. FRAMEWORK OF DIGITAL COMMUNICATION SYSTEMS: SPI MASTER-SLAVE PROTOCOL DESIGN AND VERIFICATION IN VERILOG

In this section, framework of digital communication systems: SPI master-slave protocol design and verification in verilog is observed in figure 1. The sequence of data transfers is controlled by the master core. The master core contains a finite state machine (FSM) with four key states are IDLE, START, TRANSMIT, and FINISH and also includes configuration registers that allow adjustment of data width (fixed at 8-bit in this design) and the clocking parameters (CPOL and CPHA).

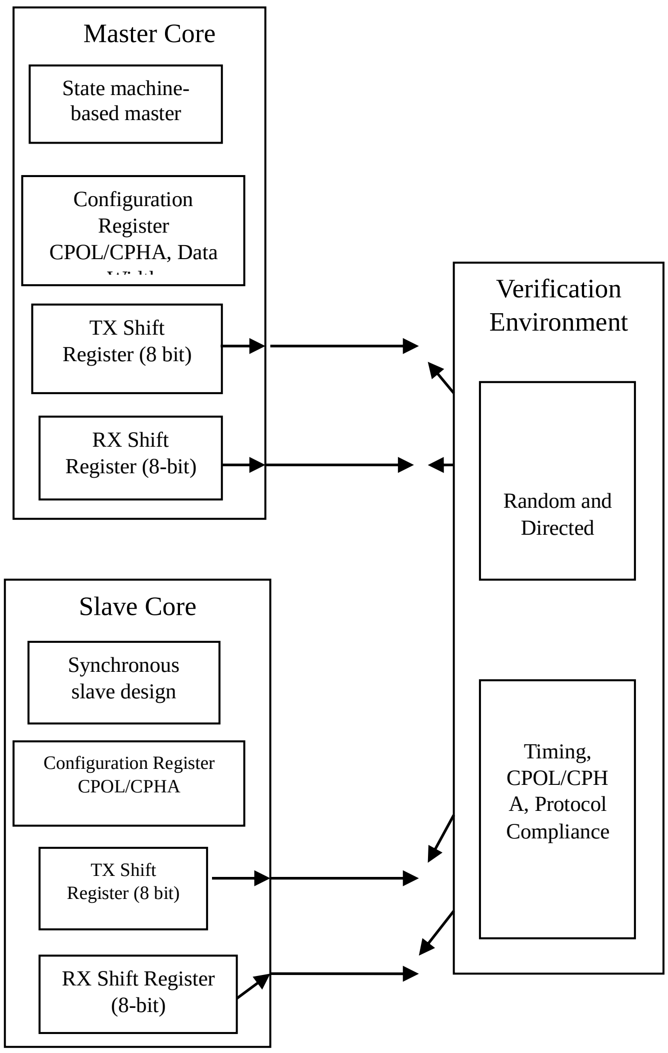
Master Core
State machine-based master
Configuration Register CPOL/CPHA, Data
TX Shift Register (8 bit)
RX Shift Register (8-bit)
Verification Environment
Random and Directed
Timing, CPOL/CPHA, Protocol Compliance
Slave Core
Synchronous slave design
Configuration Register CPOL/CPHA
TX Shift Register (8 bit)
RX Shift Register (8-bit)

Figure.1: Framework of Digital Communication Systems: SPI Master-Slave Protocol Design And Verification In Verilog

The Master can continuously send and receive data, enabling full-duplex communication to ensure dedicated transmit and receive shift registers. In Master-driven clock domain, slave core mirrors are integrated with additional synchronization and edge detection logic to ensure reliable sampling of signals in this structure. The CPOL/CPHA handles configurable registers and slaves, and it also contains 8-bit shift registers. SPI bus consists of a four-wire that is connected through both cores, and it contains MOSI, MISO, SCLK, and CS (Chip Select), which establishes the full-duplex communication channel. The verification environment is shown and randomized test scenarios are executed and testbench drives stimulus, while the scoreboard and reference model check functional correctness. The SystemVerilog assertions monitor protocol rules such as timing, CPOL/CPHA alignment, and chip-select behavior. Therefore, this design captures functional hardware design and also a comprehensive verification framework to ensure that the SPI protocol operates reliably under different configurations and test conditions.

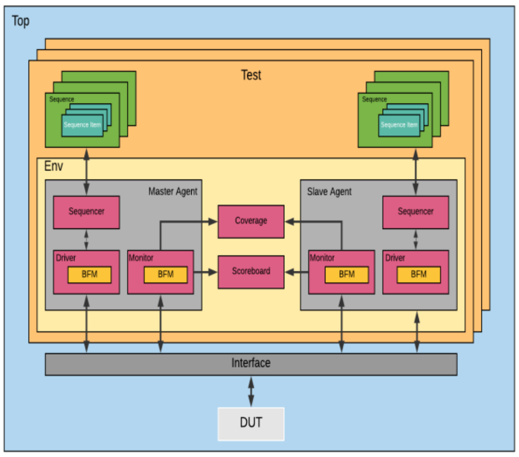

Figure.2: Test bench model

In SPI communication, a Finite State Machine (FSM) is used to control the serial data transfer between a master and slave device. The FSM manages the timing and control signals (like clock, chip select, MOSI, and MISO) based on the defined SPI protocol. It ensures proper data transmission and reception by transitioning through different states based on input signals and internal logic. The master FSM controls the overall communication sequence. It manages the chip select (CS) line to select the appropriate slave, generates the clock signal (SCLK), and sends data on the MOSI line while simultaneously receiving data from the slave on the MISO line.

In SPI (Serial Peripheral Interface), the master core is the component that initiates and controls communication with slave devices. It generates the clock signal and manages data transfer over the MOSI (Master Out Slave In) and MISO (Master In Slave Out) lines. The master core also handles the chip select (CS) signals to select which slave device is being communicated. The SPI master core typically includes registers for controlling its operation (e.g., clock polarity and phase, data order) and for monitoring its status (e.g., data being transmitted or received). In some designs, the SPI master core includes a Wishbone interface, allowing it to be integrated with other components or subsystems using the Wishbone bus protocol. MOSI, which stands for Master Out Slave In, is a crucial signal line in Serial Peripheral Interface (SPI) communication. It carries data from the master device to the slave device. In a typical SPI setup with one master and multiple slaves, the master controls the data flow on the MOSI line, sending commands and data to the selected slave device.

SCLK, or Serial Clock, is a crucial component of the Serial Peripheral Interface (SPI) protocol. It's a signal line used by the master device to synchronize data transfer between itself and slave devices. Essentially, the master generates clock pulses on the SCLK line, and these pulses dictate when data is transmitted and received on the MOSI (Master Out Slave In) and MISO (Master In Slave Out) lines, respectively. The master device outputs the SCLK signal, acting as the timing reference for all data transfers on the SPI bus. The SCLK signal ensures that both the master and slave devices are operating in sync, allowing for reliable data exchange. Data on the MOSI and MISO lines is transferred on the edges of the SCLK signal, either the rising or falling edge depending on the SPI mode configuration. The SCLK signal can be configured by the master using parameters like clock polarity (CPOL) and clock phase (CPHA), which affect how data is sampled and transmitted.

IV. RESULT ANALYSIS

In this section, result analysis of digital communication systems: SPI master-slave protocol design and verification in verilog is observed.

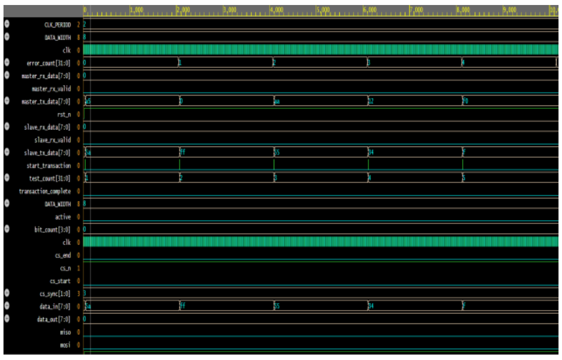

Figure.3: Output Waveform

Figure 3 shows simulation waveform of the SPI Master–Slave data exchange.

The functional verification of the SPI core controller was carried out successfully with the following result

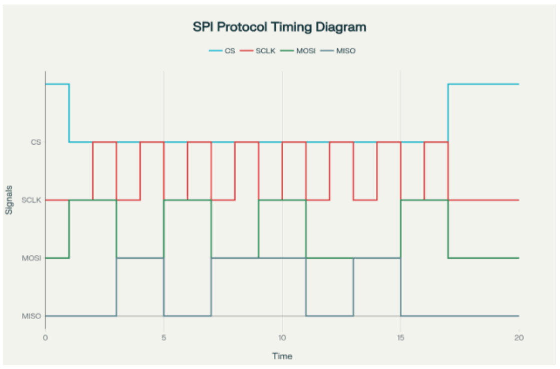

Figure.4: SPI Protocol Timing Diagram - 8-bit Data Transfer (0xA5 ↔ 0x5A)

V. CONCLUSION

In this section, result analysis of digital communication systems: SPI master-slave protocol design and verification in verilog is observed. In this design, a complete SPI Master–Slave communication system was designed and verified in Verilog with emphasis on configurability and protocol correctness, and also integration of a state-machine-based Master. The synchronous edge-detecting Slave, and parameterizable control registers ensures flexibility and reliability in data transfers. Verification through assertions, scoreboards, and functional coverage confirmed compliance with SPI protocol specifications under diverse test scenarios and the experimental outcomes validate that the proposed architecture supports stable full-duplex communication. The efficient synchronization and correct handling of various CPOL/CPHA modes in this work shows a comprehensive design and verification methodology. It extended to more complex digital communication protocols or adapted for low-power and high-performance embedded systems

VI. REFERENCES

[1] S. Navaneethan, U. Dinesh and V. D. Rajan, "Industrial Application of the Serial Peripheral Interface Protocol on Field Programmable Gate Arrays," *2023 International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS)*, Erode, India, 2023, pp. 1627-1631, doi: 10.1109/ICSSAS57918.2023.10331803.

[2] M. Hafeez and A. Saparon, "IP Core of Serial Peripheral Interface (SPI) with AMBA APB Interface," *2019 IEEE 9th Symposium on Computer Applications & Industrial Electronics (ISCAIE)*, Malaysia, 2019, pp. 55-59, doi: 10.1109/ISCAIE.2019.8743871.

[3] A. H. Rahimi *et al.*, "Design and Analysis of Single Master Multiple Slave Serial Peripheral Interface (SPI) on FPGA," *2024 IEEE International Conference on Applied Electronics and Engineering (ICAEE)*, Shah Alam, Malaysia, 2024, pp. 1-7, doi: 10.1109/ICAEE62924.2024.10667616.

[4] V.Muneeswaran, K. N. Kumar, P.Manikandan, M. N. Kumar, C. V. Chaithanya and M. G. Priyadarshini, "Performance Analysis of Communication Protocols (I2C, SPI) for Embedded Applications with IMU Sensor using Verilog," *2025 5th International Conference on Trends in Material Science and Inventive Materials (ICTMIM)*, Kanyakumari, India, 2025, pp. 685-689, doi: 10.1109/ICTMIM65579.2025.10988065.

[5] A. K, S. Sushma and V. S. Teja, "Implementing and Verifying the SPI Communication Protocol in ASICs with Cadence EDA Tools," *2024 Global Conference on Communications and Information Technologies (GCCIT)*, BANGALORE, India, 2024, pp. 1-5, doi: 10.1109/GCCIT63234.2024.10861947.

[6] T. -L. Liu, C. Yang, Y. -J. Chuang and H. Chiueh, "Radiation-hardened Configuration Registers with SPI Interface Protocol in a 65nm CMOS Technology," *2024 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, Taipei, Taiwan, 2024, pp. 212-216, doi: 10.1109/APCCAS62602.2024.10808372.

[7] C. K. Shaila, G. Manoj, P. S. Divya and M. Vijila., "Functional Verification of SPI Protocol using UVM based on AMBA Architecture for Flash Memory Applications," *2023 4th International Conference on Signal Processing and Communication (ICSPC)*, Coimbatore, India, 2023, pp. 311-315, doi: 10.1109/ICSPC57692.2023.10125890.

[8] A. Patra and L. M. Saini, "Analysis of Serial Peripheral Interface," *2023 Second IEEE International Conference on Measurement, Instrumentation, Control and Automation (ICMICA)*, Kurukshetra, India, 2024, pp. 1-6, doi: 10.1109/ICMICA61068.2024.10732859.

[9] M. Patel, S. Dhyani and S. Nath, "Generating High Switching Frequency by Integration of FPGA with DSP Using SPI," *2024 IEEE International Conference on Power Electronics, Drives and Energy Systems (PEDES)*, Mangalore, India, 2024, pp. 1-5, doi: 10.1109/PEDES61459.2024.10961234.

[10] M. Saravanan, R. Sandhiya, S. Sivaranjani, K. Tamilarasi, M. Tamilarasan and G. M. Varshini, "Implementation of 8-Bit SPI Protocol Using System Verilog and UVM," *2024 4th International Conference on Ubiquitous Computing and Intelligent Information Systems (ICUIS)*, Gobichettipalayam, India, 2024, pp. 1893-1898, doi: 10.1109/ICUIS64676.2024.10867233.

[11] P. K. Swain, S. Sarangi and S. Rout, "A compact SPI-based SRAM in 0.6um CMOS for Low-Power IoT Applications," *2023 1st International Conference on Circuits, Power and Intelligent Systems (CCPIS)*, Bhubaneswar, India, 2023, pp. 1-6, doi: 10.1109/CCPIS59145.2023.10291287.

[12] D. C. Costache, L. A. Perişoară and A. Florescu, "FPGA Implementation of a SD Card Controller using SPI communication," *2020 12th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, Bucharest, Romania, 2020, pp. 1-4, doi: 10.1109/ECAI50035.2020.9223238.

[13] M. B. Aykenar, G. Soysal and M. Efe, "Design and Implementation of a Lightweight SPI Master IP for Low Cost FPGAs," *2020 28th Signal Processing and Communications Applications Conference (SIU)*, Gaziantep, Turkey, 2020, pp. 1-4, doi: 10.1109/SIU49456.2020.9302434.

[14] A. Revani Sastaviyana and R. Hartono, "Performance of C-BUS Communication in CMX7032 IC with SPI Communication in ATxmega128A1 IC," *2019 IEEE International Conference on Aerospace Electronics and Remote Sensing Technology (ICARES)*, Yogyakarta, Indonesia, 2019, pp. 1-7, doi: 10.1109/ICARES.2019.8914355.

[15] K. C. Kumar, C. Vandana, G. Gowtham, E. Johnny and B. M. Babu, "Design of Low Power SPI Protocol using Clock Gating Techniques," *2024 International Conference on Emerging Technologies in Computer Science for Interdisciplinary Applications (ICETCS)*, Bengaluru, India, 2024, pp. 1-6, doi: 10.1109/ICETCS61022.2024.10543955.

[16] V. U. Royal, A. Vijay Kumar, R. Sumathy, P. Durga Prasad, D. Gnaneshwar and U. T. Kumar Reddy, "Integrated Bridge Design for Data Transmission from AXI Bus to SPI Peripheral Through APB," *2025 7th International Conference on Inventive Material Science and Applications (ICIMA)*, Namakkal, India, 2025, pp. 218-221, doi: 10.1109/ICIMA64861.2025.11074078.

[17] S. Sasikumar, B. A. Balaji, N. J. Krishna and S. Reddy, "Design and Implementation of a Protocol Conversion Unit for SPI to I2C Communication," *2024 8th International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, Coimbatore, India, 2024, pp. 399-406, doi: 10.1109/ICECA63461.2024.10801051.

[18] S. Penurkar, A. Kumar, S. Srivastava and S. Nakhate, "Design and Implementation of Single Master Single Slave Serial Peripheral Interface (SPI) on FPGA," *2025 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, Bhopal,

India, 2025, pp. 1-7, doi: 10.1109/SCEECS64059.2025.10940931.

[19] A. Maity, P. K. Samanta, B. P. De, S. K. Dash, W. Bhowmik and A. Bakshi, "FPGA Implementation of Serial Peripheral Interface Transceiver Module," *2024 International Conference on Computer, Electrical & Communication Engineering (ICCECE)*, Kolkata, India, 2024, pp. 1-5, doi: 10.1109/ICCECE58645.2024.10497241.

[20] M. Trehan, P. Kumar and N. Gaur, "Design and Analysis of Multi-Protocol Conversion Unit for SPI, I2C and UART," *2024 2nd International Conference on Device Intelligence, Computing and Communication Technologies (DICCT)*, Dehradun, India, 2024, pp. 324-329, doi: 10.1109/DICCT61038.2024.10533106.

[21] S. Karthick, S. Venkatesh, P. S. Chaithanya and G. C. S. Royal, "Implementation and Functional Verification of I2C, SPI and CAN frame protocols," *2025 International Conference on Emerging Smart Computing and Informatics (ESCI)*, Pune, India, 2025, pp. 1-5, doi: 10.1109/ESCI63694.2025.10987994.

[22] B. Jose and J. S. Immanuel, "Design of BIST(Built-In-Self-Test)Embedded Master-Slave communication using SPI Protocol," *2021 3rd International Conference on Signal Processing and Communication (ICPSC)*, Coimbatore, India, 2021, pp. 581-585, doi: 10.1109/ICSPC51351.2021.9451702.

[23] Y. Si, N. Korada, R. Ayyanar and Q. Lei, "A High Performance Communication Architecture for a Smart Micro-Grid Testbed Using Customized Edge Intelligent Devices (EIDs) With SPI and Modbus TCP/IP Communication Protocols," in *IEEE Open Journal of Power Electronics*, vol. 2, pp. 2-17, 2021, doi: 10.1109/OJPEL.2021.3051327.

[24] J. P. Kadambarajan, P. Kadarkarai, B. Kalyani, M. R. Bathul, T. Sandhya and M. Greeshma, "SPI Verification Monitor Module Using UVM," *2024 10th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Coimbatore, India, 2024, pp. 1838-1843, doi: 10.1109/ICACCS60874.2024.10717216.

[25] O. Coşkun, E. Egeli, E. Tarcan, İ. Kurtuluş and G. Yılmaz, "Analysis and Implementation of a Daisy-Chain Serial Peripheral Interface Bus for a Communication Network with Multiple PLC Modules," *2023 14th International Conference on Electrical and Electronics Engineering (ELECO)*, Bursa, Turkiye, 2023, pp. 1-5, doi: 10.1109/ELECO60389.2023.10416045.