INTERNATIONAL SCIENTIFIC JOURNAL OF ENGINEERING AND MANAGEMENT (ISJEM) ISSN: 2583-6129
CONFERENCE PROCEEDINGS: VOLUME: 05 ACME – 2026 DOI: 10.55041/ISJEM.ACME127
AN INTERNATIONAL SCHOLARLY || MULTIDISCIPLINARY || OPEN ACCESS || INDEXING IN ALL MAJOR DATABASE & METADATA

Design of an FPGA-Based SPI Master Controller Supporting Multiple Slave Devices

D. Jithendra Reddy
Department of ECE, Annamacharya Institute of Technology and Sciences (Autonomous), Tirupati,AP,India
jithendrareddy.d@gmail.com

D. Mounika
Department of ECE, Annamacharya Institute of Technology and Sciences (Autonomous), Tirupati,AP,India
derangulamouni2@gmail.com

V. Lohith Vardhan
Department of ECE, Annamacharya Institute of Technology and Sciences (Autonomous), Tirupati,AP,India
vardhanlohith@gmail.com

T. Manasa
Department of ECE, Annamacharya Institute of Technology and Sciences (Autonomous), Tirupati,AP,India
manasat355@gmail.com

M. Manasa
Department of ECE, Annamacharya Institute of Technology and Sciences (Autonomous), Tirupati,AP,India
medapurammanasa9@gmail.com

Abstract— Serial Peripheral Interface (SPI) is a common synchronous data transmission protocol in high-speed data transfer among digital devices in embedded systems. The paper gives the design and implementation of a Single Master Multiple Slave (SMMS) SPI environment using FPGA to enhance scalability and flexibility of communication than the traditional Single Master Single Slave (SMSS) SPI frameworks. The given design allows a single master controller to interact with other slave devices through common communication lines like MOSI, MISO, and SCLK, whereas the peculiar chip-select signals are utilized to turn-on of certain slave modules. The SPI master and slave modules are developed in Verilog Hardware Description Language (HDL) and done in a Xilinx Kintex-7 FPGA (xc7k160tfbg676-2) with AMD Vivado 2023.1. Simulation validation shows that there was a functional communication between the master and four slave devices with proper data transmission and reception. The results of the implementation indicate that the proposed design requires about 1% Look-Up Tables (LUTs) and 1% Flip-Flops and I/O usage is approximately 43% as a result of multiple slave interfacing. The overall on-chip power consumption is 1.628 W, and it comprises 1.511 W dynamic power and 0.117 W static power with an operating junction temperature at about 28.1 o C. High-speed communication is also evidenced by the design with propagation delay of around 3.235ns. The results obtained affirm that the proposed SMMS SPI architecture offers an efficient use of hardware, low latency communication and scalable multi-device communication that is applicable in embedded systems based on FPGA, sensor networks and real-time digital communication.

Keywords — Serial Peripheral Interface (SPI), FPGA, Single Master Multiple Slave, Verilog HDL, AMD Vivado, Embedded Communication Systems etc.,

I. INTRODUCTION

Embedded systems today must operate with effective communication protocols that transmit information between the microcontrollers, sensors, and memory devices among other peripheral systems. Serial communications have gained popularity due to the fact that fewer interconnection wires are used and systems are becoming more reliable than parallel way of communication. The most common of these protocols is the Serial Peripheral Interface (SPI) that is so popular with its high rate of data transfer, simple hardware format and the ability to provide full-duplex communication. SPI allows real-time interaction of digital devices based on a master-slave design in which the master device is the

controller of the communication and slave devices are the followers of the commands of the master. The SPI protocol works with four major signals, i.e., Serial Clock (SCLK), Master out Slave In (MOSI), Master in Slave out (MISO), and Chip Select (CS). The clock signal is produced by the master device, which also identifies the slave device on the chip-select line and data is transmitted between the master and slave devices at the same time on MOSI and MISO lines. Spaniard SPI is very popular in embedded applications including sensor interfacing, memory communication, industrial automation and real-time monitoring systems due to its simplicity and efficiency [4], [8]. A number of authors have studied SPI communication systems and designs on FPGA platforms. Nguyen [1] developed a SPI-to-I2C protocol converter which was coded on an ASIC technology and on FPGA platforms to facilitate interoperability between various communication protocols. Rahimi et al. [2] came up with Single Master Multiple Slave SPI architecture on FPGA to enhance scalability and optimized communication. On the same note, Sahu et al. [3] dedicated their time to the optimization of SPI master design to enhance communication performance and hardware overhead reduction. These papers have proven that FPGA based applications offer flexible and efficient communication protocol design platforms. There is also another research on SPI verification, low-power design and improvement of reliability. In master-slave communication systems, Jose and Immanuel [5] proposed a Built-In Self-Test (BIST) architecture based on the SPI communication enhancements in fault detection. Priyadarshini et al. [6] came up with a 16-bit multi-slave SPI protocol in SystemVerilog with a focus on verification methods to provide a solid communication channel. According to Mohanapriya et al. [10], low-power SPI and I2C communication designs were studied in Verilog HDL, which underscores the significance of energy-efficient communication designs. Despite the number of works that have been undertaken in the implementation of SPI, most of the designs are a single Master Single Slave (SMSS) communication architecture, which only has a restriction in the scaling of system when more than two peripheral devices are required to be incorporated in the system. In real-world embedded systems, communication between one master device and several slave devices is often required to be effectively used. Thus, a scalable SPI architecture which can serve several slave devices and at the same time maintain high-speed communications as well as efficient hardware usage are required. This is done in a Single Master Multiple Slave (SMMS) SPI communication system that is implemented on FPGA with Verilog HDL. The suggested architecture allows one master to communicate to several slave modules via shared communication channels and chip-select signals. The design is however simulated and checked with the help of AMD Vivado and performance of the design is analyzed in terms of hardware usage, power consumption and communication efficiency. The findings indicate that the given system can offer scalable, practical and efficient SPI communication that can serve embedded applications powered by FPGA.

.

II. LITERATURE SURVEY

Nguyen [1] developed an ASIC-based SPI-to-I2C protocol converter and ran it on an FPGA. The suggested architecture allows communication between the devices that adopt various serial communication protocols and ensures the reliability of the data transfer as well as the simplification of hardware. Rahimi et al. [2] suggested a Single Master Multiple Slave (SMMS) SPI communication system on FPGA. Through their research, they showed that a master controller could be able to talk to many slave devices utilizing the common communication lines and also single chip-select lines, enhancing scalability and efficiency of the system. A design and optimization method of an SPI master module was introduced by Sahu et al. [3] where the aim was to enhance the performance of communication and minimize the hardware overhead. The optimized architecture enhanced the clock synchronization as well as minimizing the communication delay. The architecture used by Penurkar et al. [4] on FPGA in Verilog HDL was Single Master Single Slave (SMSS) SPI communication architecture. Hardware verification and simulation proved that there was reliable data transmission between the master and slave modules and low use of hardware resources. Jose and Immanuel [5] put forward Built-In Self-Test (BIST) embedded SPI communication system to enhance reliability and fault identification in master slave communication networks. The system combines testing functions to detect defects in the communication devices. Priyadarshini et al. [6] developed and tested a 16 bit multi-slave SPI protocol in System Verilog. Their contribution focused on techniques of functional verification to assure proper communication between the master and several slave devices. Ishak and Kumar [7] also used the I2C bus protocol in FPGA to communicate over master-slave data. The paper has shown that data transfer is efficient and has identified some similarities between SPI and I2C communication architectures. The design and verification of the SPI bus protocol HDL were

introduced by Maheshwari and Kharade [8]. The design proposed was able to accomplish the tasks of implementing the SPI communication and is able to test its functionality with the help of simulation tools. Islam [9] implemented a Ethernet based real time SPI protocol management system on hardware in the loop simulation with dSPACE and microcontroller boards. The system will permit remote monitoring and control of SPI communication systems. Mohanapriya et al. [10] presented a Verilog HDL implementation of a low-power design of the SPI and I2C protocols. The work was aimed at minimizing power usage in communication architectures and preserving the efficient data transfer. The master-slave communication networks were implemented using the I2C protocol and Osman [11] focused on the issue of the structured communication structures as the factors of the reliable communication. A master-slave control strategy of frequency regulation in hybrid power systems was presented by Iqbal and Gulzar [12]. Despite the emphasis on power systems, the idea of coordinated master-slave architecture has a similarity with digital communication devices like the SPI. Zhang et al. [13] have come up with the energy management system based on the Spring Boot framework to enhance the monitoring and control abilities of systems in distributed environments. Mohadeszadeh and Pariz [14] came up with an adaptive synchronization algorithm of the secure communication system based on the chaotic signals. The technique enhances security in communication and stability of the system. Aljabr and Kumar [15] described a framework of Internet of Medical Things (IoMT) infrastructure based on artificial intelligence to be used in mobile healthcare applications, which facilitates the effective communication between medical devices and monitoring systems. Hu et al. [16] developed a STM32 microcontroller-based wireless control system of a two-arm robot. The system permits effective communication and coordination of components through the robots. To remotely monitor and control electrical devices, Sanchez-Sutil and Cano-Ortega [17] created a smart plug system that is based on the LoRaWAN communication protocol and a remote monitoring system. Kanakararaja et al. [18] suggested a smart energy meter that implemented the LoRaWAN and IoT technology and allows the energy monitoring devices and energy control systems to communicate efficiently. Vaseghi et al. [19] proposed chaos synchronisation technique of secure communication in the case of office of digital frequency modulation systems and enhanced security of communications in digital transmission systems. Tutueva et al. [20] studied adaptive symmetry control methods in secure communication systems, which increases the reliability and the security of any data transmission. Rybin et al. [21] came up with a chaotic communication system with microcontrollers that is based on symmetry to transmit data securely. The Nguyen and Huang [22] have suggested the adaptive fuzzy disturbance observer based on the sliding-mode control of chaos-based communication systems to enhance the stability and robustness of the system. Rahman et al. [23] proposed a fractional-order chaotic communication system of a secure data transmission and showed better synchronization and circuit realization method. Lampinen et al. [24] developed a bilateral teleoperation system without the use of a force sensor grounded on masterslave control architecture which allows efficient control of a robotic system at a distance. Burunkaya and Duraklar [25] introduced an incubator of smart classroom based on IoT, which is an automated environmental control and monitoring system. Karimov et al. [26] came up with a chaotic communication system that utilized modulation based on symmetry to enhance excellent transmission of data in communication networks. Based on the literature review, it is noted that most of the studies are dedicated to the implementation of the SPI protocols, verification, optimization, and communication security methods. Nevertheless, not many studies have concentrated on lightweight FPGA based Single Master Multiple Slave SPI architectures and performance breakdown including power consumption, delay and hardware consumption.

III. PROPOSED METHOD

The suggested system emphasizes on the design and construction of a Single Master Multiple Slave (SMMS) Serial Peripheral Interface (SPI) communication system over an FPGA platform. In this architecture, a master device is in charge of managing communication with a number of slave devices via common lines of communication. The serial clock (SCLK) is produced by the SPI master and the transmission of the data takes place through the Master Out Slave In (MOSI) and Master In Slave Out (MISO) lines. The slave devices are enabled with a unique chip-select (CS) signal, which enables the master to select and communicate with a slave at a time with other slaves idle. The implementation flow chart shown in fig.4.The Implementation flow chart provides better flexibility and scalability of the communication system than the traditional Single Master Single Slave architecture of fig.1. the implementation flow chart shown in fig.3.

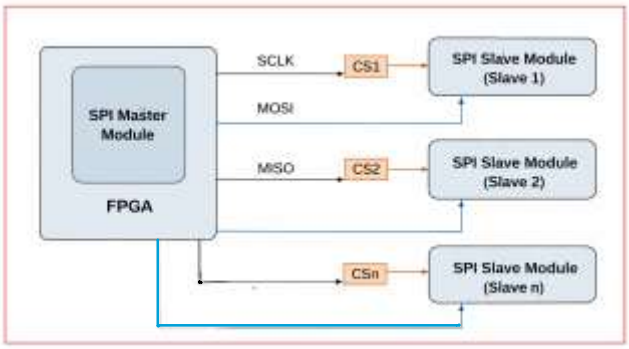

Fig. 1. Architecture of the proposed method

A. *Methodology*

The proposed system deploys a Single Master Multiple Slave (SMMS) Serial Peripheral Interface (SPI) communication architecture with the help of an FPGA platform. The methodology entails coming up with the SPI master module, a series of slave modules, and communication control logic to facilitate effective data transfer between the devices. The SPI protocol runs in time with a clock signal produced by the master device and hence data can be sent and received at the same time between the master and the chosen slave device.

1) Clock Generation

The master module forms the start of the SPI communication process through the generation of the Serial Clock (SCLK). The clock signal is used to regulate the timing of transmission of data between the slave and master devices. The input FPGA clock frequency is set in the proposed system to be as follows.

fclk=100MHz

The clock divider is used to get the SPI clock based on the system clock. The system clock/ SPI clock correlation is as follows:

fspi=Nfclk

where
fsp= SPI clock frequency
fclk= system clock frequency
N = clock division factor.
This guarantees that there is synchronization of data transfer between slave and master modules.

2) SPI Data Transmission

SPI communication facilitates two-way data transmission i.e. it is possible to transmit and receive data at the same time. The slave receives data sent by the master via MOSI (Master Out Slave In) line and the master receives the data sent by the slave via MISO (Master In Slave Out). Assuming that DM is the data being sent out by the master and that DS is the data being sent out by the slave, it can be said that the communication process is as follows:
DM→MOSI→Slave
DS→MISO→Master
The size of data frame in the proposed design is set as:

Data Width=16

Therefore, every communication cycle transfers 16-bit data frames between the master and slave devices.

3) Slave Selection Mechanism

In the Single Master Multiple Slave architecture, the master device then chooses a particular slave device by means of a Chip Select (CS) signal. Every slave device is provided with a unique chip-select line. The logic of selection of slaves can be expressed as:
CSi= {0, if slave i is selected1,
CSi= {0,1, if slave i is selected otherwise
where
CSi will be the signal of the ith slave device chip-select. The communication process only involves the slave that has an active signal of the chip-select and the other slaves do not take part in the process.

4) Data Shifting Operation

SPI protocol transmits data on a serial basis with the use of shift registers. Each clock cycle, one bit of data is sent out of the master and one bit is sent in of the slave. The changing operation can be presented in the following way:

Datanew=(Dataold<<1)

where the register shifts one bit to the left during each clock pulse. This process continues until all bits in the data frame are transmitted.

5) Verilog HDL Implementation

SPI master and slave modules are written by use of Verilog Hardware Description Language (HDL). The master module comprises of the following:

- Clock generator
- Shift register for data transmission
- Chip-select control logic
- Slave selection module

Each slave module contains:

- Data register
- Shift register
- MISO output control

This modular architecture allows easy expansion to support additional slave devices.

5) Simulation and Verification

The SPI architecture is simulated with the help of the AMD Vivado 2023.1 to ensure the ability of

communication signals to operate correctly. The signals verified in the simulation waveform include:

- Clock signal (SCLK)
- MOSI data transmission
- MISO data reception
- Chip-select signals for slave selection

Slave selection This is where a chip-select signal is used to select a slave. The data transfer between the master and the multiple slave devices is successful as the results of the simulation confirm.

6) FPGA Implementation

The design is then synthetized and run on a Xilinx Kintex-7 FPGA (xc7k160tfbg676-2) after being successfully simulated. The implementation outcomes reflect the use of efficient hardware with an estimate of. The approximate power consumption of the system is:

Ptotal=1.628 W

and the communication delay is approximately:

Delay≈3.235 ns

The findings prove that the suggested SPI design offers an effective communication, low hardware usage, and scalable multi-device interfacing. Fig.2 shows the process of transferring the data of the SPI between master and slave devices. The clock signal (SCLK) is created by the master, and data is sent to the slave via the MOSI line, and data is sent back to the master via the MISO line. The chip-select (CS) signal is used to active the slave device in communication so that the complete-duplex of data transmission is synchronized.

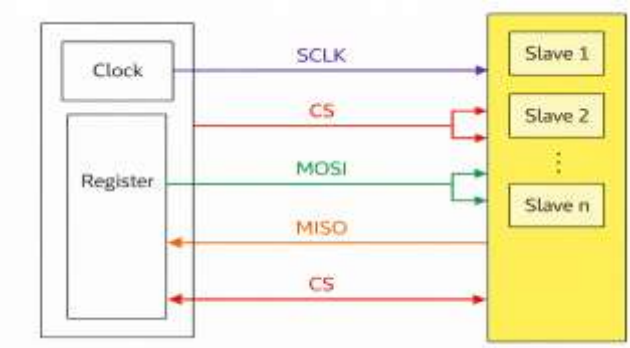

Fig. 2. Data transfer

B. Algorithm

Single Master Multiple Slave SPI Communication

Input: System clock (fclk), number of slave devices (n), and data frame size (D)

Output: Successful communication between the master and the selected slave device

Step 1: Set up the SPI system by defining the required parameters such as system clock frequency, data width, and the total number of slave devices.

Step 2: Create the SPI clock signal (SCLK) by dividing the main system clock using a clock divider circuit.

Step 3: Initialize the SPI master unit and reset all connected slave devices to their default state.

Step 4: Place the data that needs to be transmitted into the master's transmit register.

Step 5: Choose the slave device for communication by enabling its corresponding Chip Select (CS) line.

Step 6: Begin the SPI communication by activating the master control signal.

Step 7: For each clock cycle from 1 to D (where D is the data frame size):

a. Send one bit of data from the master transmit register through the MOSI line.

b. At the same time, receive one bit of data from the slave through the MISO line.

c. Store the received bit into the master receive register.

Step 8: Repeat the shifting process until all bits in the data frame are completely transmitted and received.

Step 9: Once the transmission is finished, disable the Chip Select (CS) signal of the slave device.

Step 10: Save the data received from the slave in the master receive register.

Step 11: If communication with another slave device is required, repeat Steps 5 to 10 for the next slave.

Step 12: After completing all communications, stop the SPI operation and return the system to the idle state.

C. *Implementation*

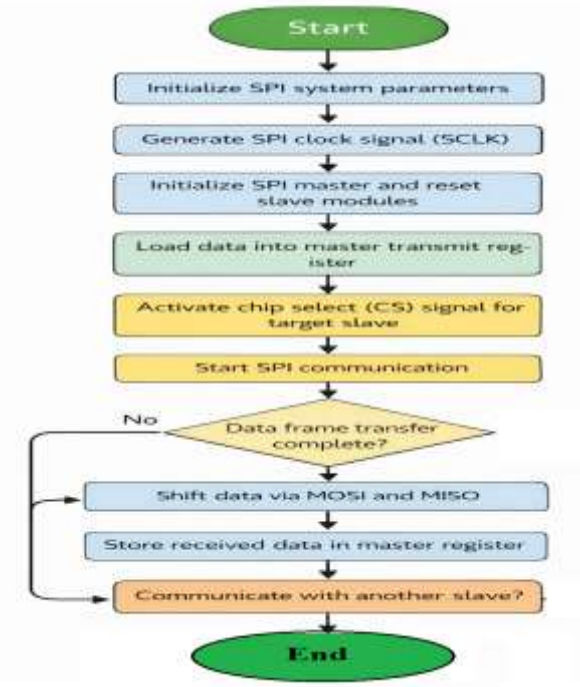

Fig. 3. Implementation of the flow chart

IV. EXPERIMENTAL RESULTS

The Register Transfer Level (RTL) schematic of the proposed Single Master Multiple Slave SPI communication architecture created within AMD Vivado is indicated in Fig. 4. The schematic will show the internal hardware design of the design comprising the SPI master unit, various slave units, and multiplexing logic applied to choose the slave. The master module produces the clock signal and data transfer by MOSI and MISO lines and sets the slave device by using chip-select signals. The RTL model proves that the SPI master is connected with several slave modules via shared communication lines and ensures the correct data synchronization.

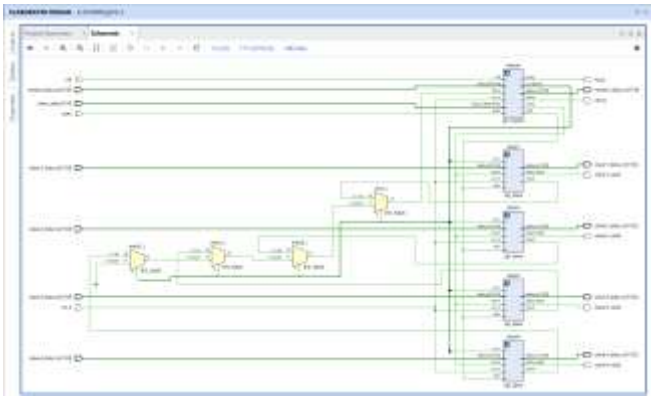

Fig. 4. RTL Schematic

The report of the FPGA resource usage can be seen in Fig. 5, and it has been obtained after the synthesis of the proposed SPI architecture. The findings point to the fact that the design occupies about 1 percent Look-Up Tables (LUTs) and 1 percent Flip-Flops, and I/O consumption is about 43 percent because of the communication lines needed to serve several slave devices. The clock distribution usage of the BUFG is about 6%. These findings show that the suggested design uses extremely low FPGA resources, and it can be used in scalable embedded system applications.

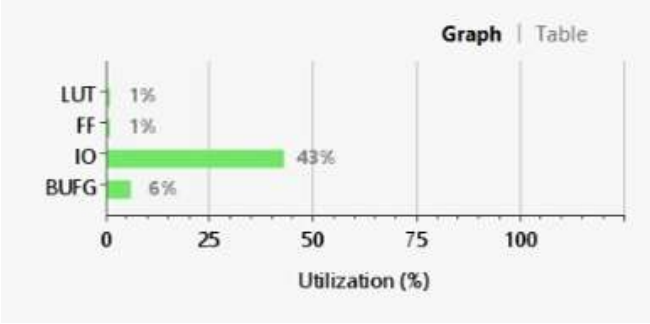

Fig. 5. Area Utilization

The power consumption analysis presented in fig.6 is the power consumption analysis of the implemented SPI communication system derived through the vivado power report. On chip power consumption is about 1.628 W of which dynamic power is 1.511 W and the power consumed in the state is about 0.117 W. I/O power is the most making percentage because of SPI data communication followed by signal and logic power. The findings verify that the proposed SPI architecture can be efficiently run with a moderate power consumption.

Fig. 6. Power Utilization

The waveform of functional simulation of the proposed Single Master Multiple Slave SPI system is presented in Fig.7. The waveform represents the functionality of the

following key signals together with clock (clk), reset (rst), slave-select, master transmit data, master receive data, busy, and done signals. It has been shown in the simulation that the master sends and receives 16-bit data frames sequentially to several slave devices. It is also confirmed by the waveform that slave valid signals go high when valid data is output and ensures that the SPI communication architecture is functioning correctly.

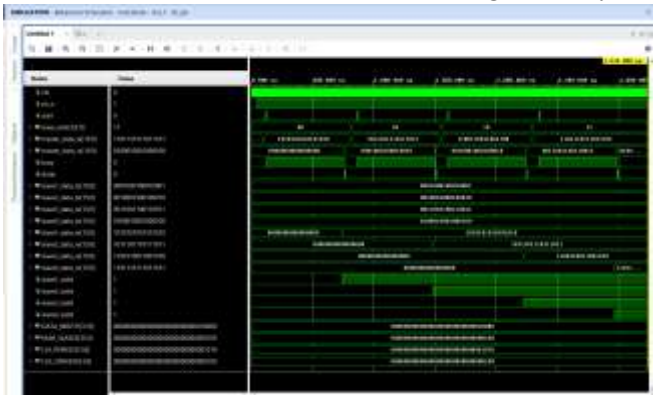

Fig. 7. Simulation Waveform

TABLE I. SIMULATION DATA TRANSMISSION RESULTS

Slave Device	Slave Select Value	Master Transmitted Data (Tx)	Slave Response Data (Tx)	Slave Received Data (Rx)	Status
Slave 1	00	1010101010101010	0001000100010001	1010101010101010	Successful
Slave 2	01	1011101110111011	0010001000100010	1011101110111011	Successful
Slave 3	10	1100110011001100	0011001100110011	1100110011001100	Successful
Slave 4	11	1101110111011101	0100010001000100	1101110111011101	Successful

Table I presents the results of the simulation of transmission of data between the SPI master and several slave devices. The master is able to communicate with four slaves with the help of various slave-select values (00, 01, 10, and 11). To every slave, the 16-bit data sent by the master is correctly received by the chosen slave and the slave reply data are transmitted successfully back to the master. The table II shows the utilization of the hardware resources of the proposed SPI design on the FPGA. The found results indicate that 1 Percent LUTs and 1 Percent Flip-Flops are utilized and I/O 43 Percent because of communication signals and BUFG 6 Percent clock distribution utilization. This shows that the proposed architecture has low demands in FPGA.

TABLE II. FPGA RESOURCE UTILIZATION

Resource Type	Utilization Percentage
LUT (Look-Up Table)	1 %
Flip-Flops (FF)	1 %
Input/Output (I/O)	43 %
BUFG (Clock Buffer)	6 %

Table III indicates the power consumption of the implemented SPI system. The overall power consumption on-chip is 1.628 W with a dynamic power consumption of 1.511 W and a static power consumption of 0.117 W. The highest dynamic element is the I/O power (56%), and the next one is signals and logic power.

TABLE III. **POWER CONSUMPTION ANALYSIS**

Power Component	Power (W)	Percentage
Signals	0.460 W	30 %
Logic	0.212 W	14 %
I/O	0.839 W	56 %
Device Static	0.117 W	7 %
Total On-Chip Power	**1.628 W**	**100 %**

Table IV shows the thermal performance of FPGA at the time of operation. Junction temperature is about 28.1 o C with an ambient temperature of 25 o C signifying safe operating temperatures. The thermal resistance is 1.9 O C/ W with a thermal margin of 56.9 O C, which proves the presence of stable operations of the device.

TABLE IV. THERMAL PARAMETERS OF THE FPGA DEVICE

Parameter	Value
Junction Temperature	28.1 °C
Ambient Temperature	25 °C
Effective Thermal Resistance (θJA)	1.9 °C/W
Thermal Margin	56.9 °C

TABLE V. COMPARISON BETWEEN EXISTING SMSS SPI [4] AND PROPOSED SMMS SPI

Parameter	**Existing System (SMSS SPI) [4]**	**Proposed System (SMMS SPI)**
Communication Type	Single Master – Single Slave	Single Master – Multiple Slaves
Number of Devices Supported	Only one slave device	Multiple slave devices
Scalability	Low	High
Hardware Flexibility	Limited	Flexible architecture
Chip Select Control	Single CS signal	Multiple CS signals
Data Communication	Point-to-point communication	Master communicates with multiple peripherals
Hardware Utilization	Simple but limited	Slightly higher but efficient
System Efficiency	Moderate	High
Application Suitability	Small embedded systems	Large embedded and multi-device systems

Table V makes a comparison between the current Single Master Single Slave (SMSS) SPI architecture [4] and the proposed Single Master Multiple Slave (SMMS) SPI system. The proposed architecture is better in terms of scalability, hardware flexibility as well as the efficiency of communication since one master is able to communicate with various slave devices. Table VI involves a comparison of hardware performance of the current [4] and proposed SPI systems. The suggested system will sustain four slave devices that have a little increased resource consumption and power consumption. Nevertheless, it has better communication ability and less propagation delay of 3.235 ns proving that it provides efficient system operation.

TABLE VI. AREA, POWER AND DELAY COMPARISON

Parameter	Rahimi et al. **[2]**	Sahu et al. **[3]**	Penurkar et al. **[4]**	Priyadarshini **et al. [6]**	Iqbal **et al. [12]**	**Proposed System**
LUT Utilization	3 %	2.6 %	2%	1.9 %	1.5%	**1 %**
Flip-Flop Utilization	2.8 %	2.6 %	2.5 %	2.7 %	2%	**1 %**
I/O Utilization	63 %	56%	65%	70 %	56%	**43 %**
Clock Buffers (BUFG)	10 %	9%	8%	19%	7%	**6 %**
Total On-Chip Power	4.58 W	3.58 W	2.4 W	2.2W	2.1W	**1.628 W**
Dynamic Power	1.9 W	1.8 W	1.8 W	1.9W	2.1W	**1.511 W**
Static Power	0.2 W	0.25 W	0.26 W	0.27 W	0.119W	**0.117 W**
Propagation Delay	7.5 ns	6.6ns	6.8ns	7.2ns	4.2ns	**3.235 ns**
Number of Slaves Supported	1	1	3	2	2	**4**

V. CONCLUSION AND FUTURE SCOPE

The design and implementation of a Single Master Multiple Slave (SMMS) Serial Peripheral Interface (SPI) communication system were provided in this paper on an FPGA platform. The proposed architecture allows one master device to interact with a number of slave devices through common communication lines like MOSI, MISO

as well as SCLK and separate chip-select (CS) lines are used to identify the needed slave device. The implementation of the SPI master and slave modules was done in the Verilog Hardware Description Language (HDL) and the Xilinx Kintex-7 FPGA (xc7k160tfbg676-2) was used with AMD Vivado 2023.1. The operation of functional simulation proved the effective communication between the master and the multiple slave devices and the correct data transmission and reception. The implementation achieved effective hardware use with about 1% LUT usage and 1% Flip-Flop usage which showed a lightweight and scalable design. Power analysis revealed that the cumulative on-chip power usage is about 1.628 W, whereas the system uses propagation delay of about 3.235 ns, which is sufficient to make sure that it is very fast in communication. The results confirm that the suggested SPI architecture has credible, scalable and efficient communication among multi device which can be used in embedded and FPGA applications. The future research can also be aimed at the implementation of dynamic slave selection and advancement in the method of arbitration in order to serve more peripheral devices. The possibility to integrate the SPI communication system with the IoT based embedded systems and sensor networks can also increase the scope of its practical application. Also, the design will be optimized to overcome power consumption and resource-saving architectures, which will be useful in battery-powered embedded systems and portable devices. These enhancements will ensure that SPI communication architecture becomes more appropriate in the high-performance and large-scale embedded system applications.

REFERENCES

[1] Nguyen, Hoang Dung. "Designing SPI to I2C protocol converter base on ASIC technology and implementing on the FPGA platform." Smart Systems and Devices 31.2 (2021): 19-26.

[2] Rahimi, A. H., 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). IEEE, 2024.

[3] Sahu[1], Rajat R., et al. "Design and Optimization in SPI Master." Computing Science, Communication and Security: 5th International Conference, COMS2 2024, Mehsana, Gujarat, India, February 6–7, 2024, Proceedings. Springer Nature, 2024.

[4] Penurkar, Shardul, et al. "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). IEEE, 2025.

[5] Jose, Bitty, and J. Samson 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). IEEE, 2021.

[6] Priyadarshini, S., et al. "Design and Verification of a 16-Bit Multi-Slave SPI Protocol in System Verilog." 2025 5th International Conference on Mobile Networks and Wireless Communications (ICMNWC). IEEE, 2025.

[7] Ishak, Mohamad Khairi, and Meenal Pradeep Kumar. "Design and implementation of I2C bus protocol on master and slave data transfer based on FPGA." Makara Journal of Technology 26.1 (2022): 5.

[8] Maheshwari, Nitya, and Falguni Kharade. "Design and Verification: SPI Bus Protocol Using Verilog." Transforming Tomorrow: Innovative Solutions and Global Trends in Electrical and Electronics Engineering 6 (2025): 152.

[9] Islam, Usama. Design and Implementation of Ethernet-Based Real-Time SPI Protocol Management using dSPACE HIL Simulator and STMicroelectronics Boards. Diss. Politecnico di Torino, 2025.

[10] Mohanapriya, K., M. Mohankumar, and L. Suganya. "LOW POWER DESIGN OF SPI AND I2C PROTOCOL IN VERILOG HDL." Suranaree Journal of Science & Technology 32.1 (2025).

[11] Osman, Mohammed El-Mogtaba Abbsher. Implementation of Simplified Master-Slaves Network Using inter integrated circuit Protocol. Diss. Al-Neelain University, 2021.

[12] Iqbal, Muhammad, and Muhammad Majid Gulzar. "Master-slave design for frequency regulation in hybrid power system under complex environment." IET Renewable Power Generation 16.14 (2022): 3041-3057.

[13] Zhang, Fang, et al. "Design and implementation of energy management system based on spring boot framework." Information 12.11 (2021): 457.

[14] Mohadeszadeh, Milad, and Naser Pariz. "An application of adaptive synchronization of uncertain chaotic system in secure communication systems."

International journal of modelling and simulation 42.1 (2022): 143-152.

[15] Aljabr, Ahmad Abdullah, and Kailash Kumar. "Design and implementation of Internet of Medical Things (IoMT) using artificial intelligent for mobile-healthcare." Measurement: Sensors 24 (2022): 100499.

[16] Hu, Huiqi, et al. "Design and Implementation of a Wireless Control System for Dual-Arm Robots Based on STM32." 2025 6th International Conference on Artificial Intelligence and Computer Engineering (ICAICE). IEEE, 2025.

[17] Sanchez-Sutil, F., and A. Cano-Ortega. "Smart plug for monitoring and controlling electrical devices with a wireless communication system integrated in a LoRaWAN." Expert Systems with Applications 213 (2023): 118976.

[18] Kanakaraja, P., et al. "Design and implementation of smart energy meter using LoRa-WAN and IoT applications." Journal of Physics: Conference Series. Vol. 1804. No. 1. IOP Publishing, 2021.

[19] Vaseghi, Behrouz, et al. "Finite time chaos synchronization in time-delay channel and its application to satellite image encryption in OFDM communication systems." Ieee Access 9 (2021): 21332-21344.

[20] Tutueva, Aleksandra, et al. "Adaptive symmetry control in secure communication systems." Chaos, Solitons & Fractals 159 (2022): 112181.

[21] Rybin, Vyacheslav, et al. "Prototyping the symmetry-based chaotic communication system using microcontroller unit." Applied Sciences 13.2 (2023): 936.

[22] Nguyen, Quang Dich, and Shyh-Chour Huang. "Synthetic adaptive fuzzy disturbance observer and sliding-mode control for chaos-based secure communication systems." IEEE Access 9 (2021): 23907-23928.

[23] Rahman, Zain-Aldeen SA, et al. "A new fractional-order chaotic system with its analysis, synchronization, and circuit realization for secure communication applications." Mathematics 9.20 (2021): 2593.

[24] Lampinen, Santeri, et al. "Force-sensor-less bilateral teleoperation control of dissimilar master–slave system with arbitrary scaling." IEEE Transactions on Control Systems Technology 30.3 (2021): 1037-1051.

[25] Burunkaya, Mustafa, and Kazım Duraklar. "Design and implementation of an IoT-based smart classroom incubator." Applied Sciences 12.4 (2022): 2233.

[26] Karimov, Timur, et al. "Chaotic communication system with symmetry-based modulation." Applied Sciences 11.8 (2021): 3698.

.