International Journal of Research and Development in Engineering Science
ISSN: 2582 - 4201(Online) , http://www.ijrdes.com/
Regular Issue, Vol. 6, Issue. 4 (July - August), 2024 , Pages: 1 - 7
Received: 20 April 2024 ; *Accepted:* 14 June 2024 ; *Published:* 15 September 2024
DOI: https://doi.org/10.63328/IJRDES-V6RI4P1
Original Paper : Applied Research - Experimental Research Paper

64 - BIT Implementation of SPI AMD I2C Protocols Using VERILOG HDL

P. Gangadhara Reddy [1*], G.M.Anasuya[2], N. Umamaheswar Reddy [3], K. Sai Charan [4] , B. Subramani Pavan [5], N. Ramesh [6]

[1*, 3, 4] Associate Professor, Department of ECE, Aditya College of Engineering, Madanapalle, Andhra Pradesh, India;
gangadhararedy.p@gmail.com
[3] maheswarreddyuma123@gmail.com, [4] saicharannaidu04@gmail.com , [5] pavanyadav0916@gmail.com , [6] nramesh9392@gmail.com

[2] Assistant Professor, Department of EEE, Viswam Engineering College, Madanapalle, Andhra Pradesh , India;
anasuyagm92@gmail.com

** Corresponding Author:* gangadhararedy.p@gmail.com

Abstract: The Inter-Integrated Circuit (I2C) protocol is a versatile communication standard used in embedded systems and integrated circuits. Two bidirectional lines are present in it: a serial clock line (SCL) and a serial data line (SDA). Its design is master-slave. In order to offer flexibility in data transfer, the number of bytes to be sent or received is chosen using the parameter N. The Serial Peripheral Interface (SPI) protocol, which makes use of a clock gating mechanism, is another efficient communication standard. It enables full-duplex, high-speed data exchange between several slave devices and a master device. Clock gating is a mechanism that selectively enables and disables clock signals to individual blocks or components, hence optimising power consumption. The parameter N is utilised in both SPI and I2C to choose how many bytes are delivered or received. As a result, the number of bytes transported can be varied, and the operation can adapt since data transfer control is handled by a state machine.

Keywords: SPI, I2C, Master & Slave, SDL,SCL, Parameter N, Flexibility, Clock Gating Techniques, State Machine.

1. Introduction

SPI (Serial Peripheral Interface) and I2C (Inter-Integrated Circuit) are two widely used serial communication protocols in the field of embedded systems and microcontroller-based applications.

SPI is renowned for its quick and effective data transport. It works in a master-slave architecture, in which a single master device connects to several slave devices. Four essential lines facilitate this communication: SCLK, MOSI, MISO, and SS/CS. The SCLK line synchronizes the data transmission, while MOSI and MISO facilitate data exchange between the master and slaves. The SS/CS line enables the master to select specific slaves for communication. SPI is widely used in applications where quick data transfer is essential, like flash memory, digital to analogue converters (DACs), and display drivers.The SPI module is powered by an eight-bit serial shift register present in both the slave and the master. The master enters the data into the shift register, and the slave enters the data into the shift register. Eight clock pulses are created, and the bits in the master shift register are transferred to the slave shift register via the MOSI line. Conversely, the slave uses the MISO line to transmit the contents of its shift register back to the master.The two shift registers' contents are essentially switched during this procedure. Figure 1 illustrates it. SPI transmits data via its interface using a variety of signals.

SS (Slave Select): The matching slave device is chosen for communication with the master when this pin goes high. SCK (Serial Clock): The data transmissions over the bus are synchronized by this signal

IJRDES, Vol. 6, Issue 4 , 2024 , https://doi.org/10.63328/IJRDES-V6RI4P1

MOSI (Master Out Slave In): Based on the internally shifted value of the master data register, the SPI master generates this serial data line.

MISO (Master In Slave Out): The SPI slave and master exchange data across this serial data channel.The bits from the slave data register that have been serially moved are sent out.

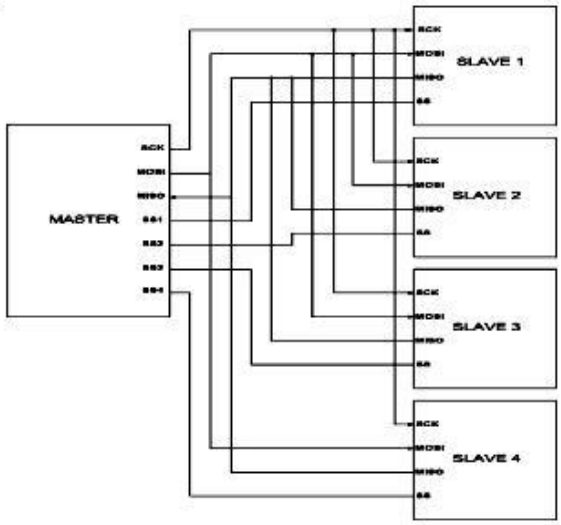

Fig.1 SPI Master Slave's Block diagram.

I2C Protocol is a two-wire, bidirectional serial bus called the I2C protocol enables communication between numerous devices. It is a well-liked option due to its accessibility and simplicity of usage. The two wires that the protocol uses are SDA (Serial Data Line) and SCL (Serial Clock Line). All data transfers are synchronized via the SCL line, and data is transmitted on the SDA line. At the logic 1 level, both SCL and SDA indicate that the I2C bus is idle. The start sequence, which is a high-to-low transition on the SDA line when the SCL line is high, is issued by the master device to begin a data transfer. The slave address is sent by the master following the start sequence. Pulling the SDA line low, the matching slave acknowledges the response with a bit. Eight-bit sequences, beginning with the most significant bit (MSB), are used to convey data. The slave transmits an acknowledgement bit per eight bits.

The slave is prepared to accept another byte if it transmits a low ACK bit, which signifies that it has received the data. When it transmits a high acknowledgment bit, the device is unable to receive any more data, and the master needs to send a stop sequence to break the connection. Standard mode (100 kbps), fast mode (400 kbps), fast mode plus (1 Mbps), high-speed mode (3.4 Mbps), and ultra-fast mode (5 Mbps) are among the several communication modes that the I2C protocol offers. There are various timing constraints for each mode.

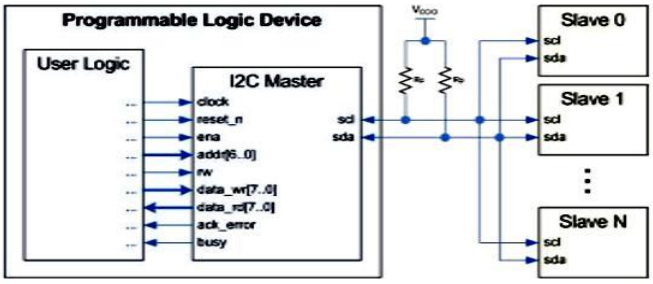

Fig.2 Block Diagram of I2C

Objective

The objective of this work is to propose a parameterized method for I2C and SPI communication protocols. I2C uses a master-slave architecture and employs only two bidirectional lines (SDA and SCL), making it ideal for applications with many devices. The parameter N is used to select the number of bytes to be sent or received, providing flexibility. In data transfer. SPI, on the other hand, facilitates high-speed, full-duplex data exchange between a master device and multiple slave devices. It uses a clock gating technique to optimize power consumption. Both protocols use a parameter N to select the number of bytes to be sent or received, allowing for flexibility in the number of bytes transferred. The proposed method aims to enhance the efficiency of data transfer in both I2C and SPI protocols, providing a flexible and adaptable solution for various applications in embedded systems.

2. Literature Survey

Easy communication with no data loss is provided with I2C protocol. When compared to older protocols like UART, USB, and CAN, it offers exceptional speed. It is affordable, widely available, and lightweight. In comparison to other protocols, it also speeds up data transport. The paper's primary objective [1] is to create a protocol that will allow for high-speed communication and control of data that can be stored on registers and inside devices, allowing for the manipulation of various parameters.

I2C is utilised in data surveillance for efficiency and accuracy. The ideal surveillance architecture employing I2C protocol is characterised by high flexibility, performance, low development costs, easy upgradability, and a migration path to lower costs as the application expands and volume improves. Developed in 1980 to facilitate data flow, particularly between slow and fast devices, Philips Semiconductor (now known as NXP Semiconductor) created the well-known serial communication standard known as I2C [2]. Compared to other protocols, it is more straightforward

and less expensive because it only requires the two external wires, SDA and SCL, and can transfer data without experiencing any loss. This work [3] aims to present the important research that various researchers have done over the years on the I2C protocol.

A communication mechanism called SPI bus enables serial data transfer between a slave and a master device. The objective of this research [4] was to provide an in-depth review of the design and implementation of a Master/Slave serial peripheral interface. The material in this document is intended to facilitate communication between serial peripheral interfaces (SPI) Master and SPI Slave. The entire design was translated onto a Xilinx FPGA device using Verilog HDL [5].

A comparison of the physical implementation characteristics of SPI and I2C protocols using many of the most recent FPGA generations from Xilinx is provided in [6], along with an explanation of which protocol components are responsible for the majority of the area overhead. Design engineers can create careful, precisely suited architecture resolutions with the use of this invaluable data. According to a recent market analysis of a significant number of lucrative I2C and SPI devices, both protocols are made as general-purpose IP solutions that integrate all the capabilities needed by contemporary SoC/ASIC applications [7]. This information is vital for a thorough comparison study. The technology-neutral Register Transfer Level (RTL) coding adds approximately 25% area overhead for I2C compared to SPI and almost identical propagation durations for both systems. I2C and SPI are sometimes considered as "little" communication protocols in the field of communication protocols when compared to other protocols like Ethernet, USB, SATA, PCI-Express, and others that have throughput in the region of multiples of 100 Mb/s, if not Gb/s. It's critical to remember the objectives of any protocol [8][9].

J. Zhang, J.Wang, C.Wu, W.Zhang, “The design and realization of a comprehensive SPI interface controller,” Second International Conference on Mechanic Automation and Control Engineering (MACE), 2011 IEEE,the design and realisation of a comprehensive SPI interface controller are the topics of this study. In the field of digital communication and embedded systems, the creation of a thorough SPI (Serial Peripheral Interface) interface controller is an important task. An overview of the main elements and importance of the design and execution of such a controller is given in this summary [10].

In the field of digital electronics and embedded systems, Leens claims that the SPI (Serial Peripheral Interface) protocol is a commonly used serial communication standard [11]. Regarding the I2C Bus Controller's design and modelling, O. Romain, T. Cuenin, and P. Gardner State [12].Professors Vishu Sharma, Mukesh Tiwari, and Jai Karan Singh states regarding the I2C's design and implementation [13].

Y.N. Alhoumays, A.K. Oudjida, M.L. Berrandjia, A. Liacha, R. Tiar, and K. Tahraoui demonstrates how SPI master and slave testing is done on the OPB bus[14].Samir Palnitkar gives a presentation on the Verilog HDL digital design and synthesis guide [15].

The I2C protocol, its characteristics, and its implementation are covered in detail in the Philips Semiconductors I2C Bus Specification and User Manual[16]. The Serial Peripheral Interface (SPI) Basics page from Texas Instruments provides an overview of the fundamentals of the SPI protocol, together with timing diagrams and examples [17]. The I2C protocol, characteristics, and applications are thoroughly explained in NXP Semiconductors Understanding the I2C Bus [18].

3. Implementation of Proposed Method

Both I2C (Inter-Integrated Circuit) and SPI (Serial Peripheral Interface) are widely used serial communication protocols in the digital world.

The SPI protocol uses a master-slave architecture, in which the master device supplies the clock signal to the slave device to synchronise data exchange. The master controls the clock line (SCK), and data transfer only happens when the clock is manipulated. Before attempting another transfer, the data that is being transmitted must be read. A device cannot participate in the SPI protocol if it is only a transmitter or a receiver; it must be able to send and receive data.

An eight-bit serial shift register present in both the slave and the master powers the SPI module. The master loads the byte of data to be sent into the shift register, and the slave loads its data into the shift register. The bits in the master shift register are moved to the slave shift register through the MOSI line as eight clock pulses are generated. On the other hand, the slave sends the content of the shift register back to the master across the MISO line. The two shift registers' contents are essentially swapped throughout this procedure. The two main elements of the SPI system are the data register and the eight-bit shift register. A little amount of data is moved from the SPI master's data register and then serial data from the slave's data register is serially moved into the master's data register during an SPI transfer.

IJRDES, Vol. 6, Issue 4 , 2024 , https://doi.org/10.63328/IJRDES-V6RI4P1

Additionally, the SPI system has a control register that gives the user some initial control over how the SPI operates. This register can be used to change the clock polarity, enable the SPI, configure it in slave or master mode, set the data sampling, and perform other operations related to SPI data transfer control. There are several divider steps that make up the baud rate production. Eight bits in the SPI baud rate register control the value by which the bus clock is divided, giving the end user a wide range of options for baud rate generation with divisors ranging from 2 to 128.

The number of bytes to be sent and received is chosen by the proposed method using a parameter N. The size of the counter and data array are specified by this option. The counter is used to track the current byte being transmitted or received, and the data array is used to hold the data that is delivered or received.

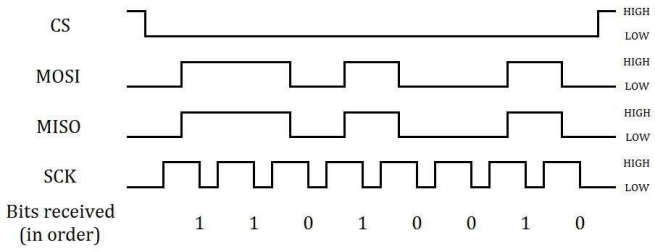

Fig.3 Waveform of SPI. Protocol

When the master wants to send data, it fills the data array with the data to be sent, and the SPI controller sends the data byte by byte. When the master wants to receive data, it sets the counter to the number of bytes to be received, and the SPI controller receives the data byte by byte. The SPI controller uses a state machine to control the data transfer. The state machine has different states for each operation, such as data transmission. The state machine transitions between these states based on the current operation and the responses from the slave. This method allows the SPI controller to send and receive a variable number of bytes, making it flexible and adaptable to different applications.

A two-wire, bi-directional serial bus called the I2C protocol enables communication between numerous devices. Its affordability and ease of use make it a popular choice. SCL(Serial Clock Line) and SDA (Serial Data Line) are the two wires used by the protocol. All data transfers are synchronised via the SCL line, and data is transmitted on the SDA line. At the logic 1 level, both SCL and SDA indicate that the I2C bus is idle. The start sequence, which is a high-to-low transition on the SDA line when the SCL line is high, is issued by the master device to begin a data transfer. The slave address is sent by the master following the start sequence. Pulling the SDA line low, the matching slave acknowledges the response with a bit.

Eight-bit sequences, beginning with the most significant bit (MSB), are used to convey data. The slave transmits an acknowledgement bit per eight bits. The slave is prepared to accept another byte if it transmits a low ACK bit, which signifies that it has received the data. When it transmits a high acknowledgment bit, the device is unable to receive any more data, and the master needs to send a stop sequence to break the connection.

The number of bytes that we receive and send in the proposed method is chosen using a parameter called N. The size of the counter, byte_counter, and the data array, data[N-1:0], are both defined by this option. The data that is sent or received is stored in the data array, and the current byte being sent or received is tracked by the byte_counter.

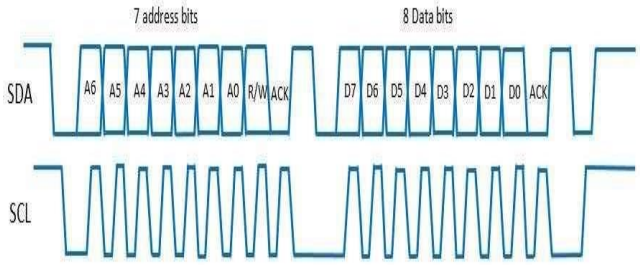

Fig.4 Waveform of I2C Protocol

When the master wants to send data, it fills the data array with the data to be sent, and the I2C controller sends the data byte by byte. When the master wants to receive data, it sets the byte_counter to the number of bytes to be received, and the I2C controller receives the data byte by byte.

The I2C controller uses a state machine to control the data transfer. The state machine has different states for each operation, such as start condition, address transmission, data transmission, and stop condition. The state machine transitions between these states based on the current operation and the responses from the slave. This method allows the I2C controller to send and receive a variable number of bytes, making it flexible and adaptable to different applications.

4. Results and Discussion

4.1 Results Of SPI Protocol

MOSI:

The data transferred across the SPI bus from the master device to the slave device is referred to here. Usually, eight bits are transferred in each byte of data, with the

IJRDES, Vol. 6, Issue 4 , 2024 , https://doi.org/10.63328/IJRDES-V6RI4P1

most significant bit (MSB) sent first. In this case, the slave receives and the master sends the data "abcdef69."

Fig.5 Waveform for SPI Protocol

MISO:

This is the information that the master device obtained over the SPI bus from the slave device. It is transmitted in 8-bit sequences, with the MSB coming first, just like transmitted data. In this case, the slave sends and the master receives the data "12345678."

This is shown in the above figure 5.

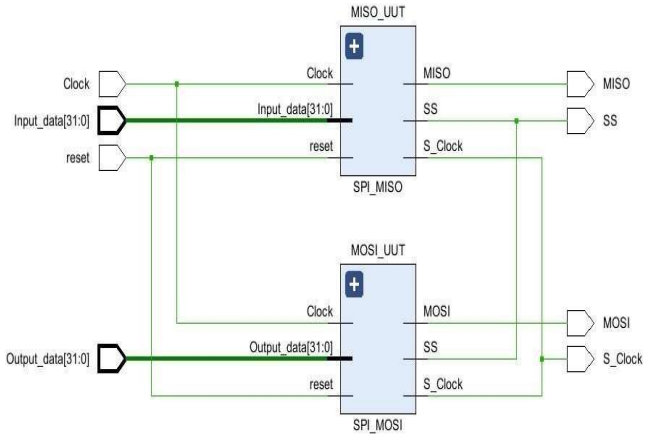

Fig.6 Elaboration Design of SPI Protocol

The above shown fig 6 is the elaborated design of single master and single slave SPI protocol

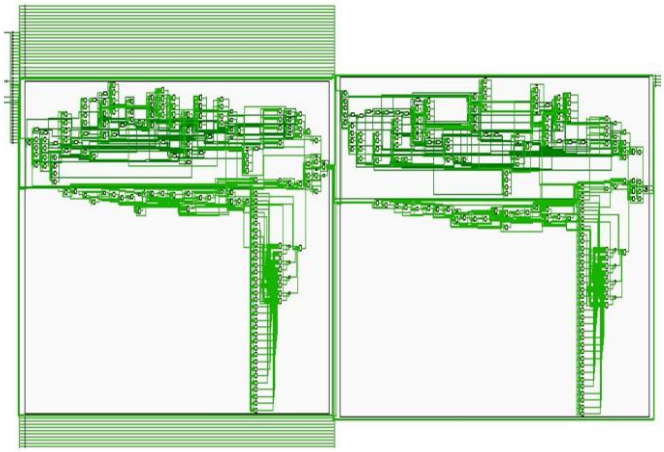

Fig.7 RTL Schematic of SPI Protocol

The above shown fig 7 is the RTL schematic diagram of single master and single slave SPI protocol.

Table 1 Area of SPI Protocol

Utilised Resource	Resources Available	Resources Utilization	Resources Utilization %
LUT	20800	154	0.74
FF	41600	161	0.39
IO	106	70	66.04

The above given table gives the value of utilization of LUTs, FF, IO of SPI Protocol.

Table 2 Delay of SPI Protocol

TYPE	TOTAL ENDPOINTS
MAX DELAY	385
MIN DELAY	385

The above given table gives the value of maximum and minimum delay of SPI Protocol.

4.2 Results of I2C Protocol:

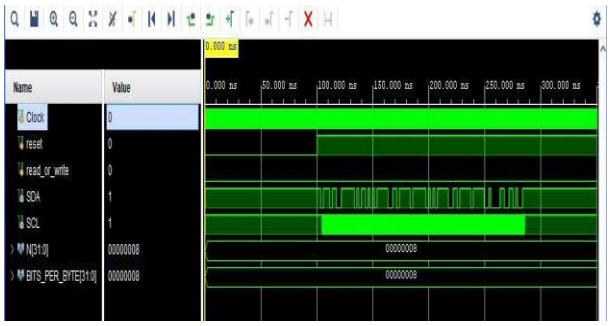

Fig.8 Waveform for I2C Master write Protocol

The above given figure8 shows the waveforms of the I2C single master and single slave protocol.

In the above fig shows the write operation of I2C protocol.

The data transferred is “566688abcdef2134”.

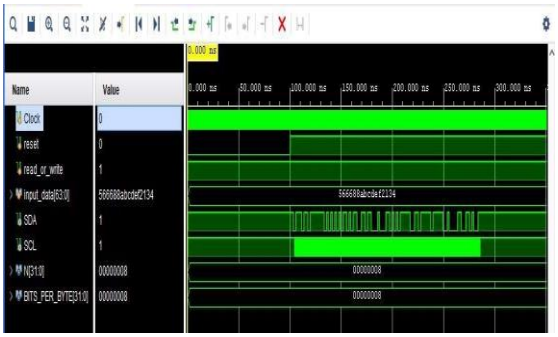

Fig.9. Waveform for I2C Slave Read Protocol

The above given figure 9 shows the waveforms of the I2C

IJRDES, Vol. 6, Issue 4 , 2024 , https://doi.org/10.63328/IJRDES-V6RI4P1

single master and single slave protocol. In the above fig shows the write operation of I2C protocol. The data transferred is "566688abcdef2134".

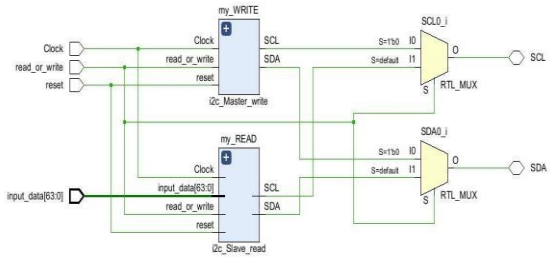

Fig.10 Elaboration Design of I2C Protocol

The above given fig shows the elaborated design of the I2C single master and single slave protocol.

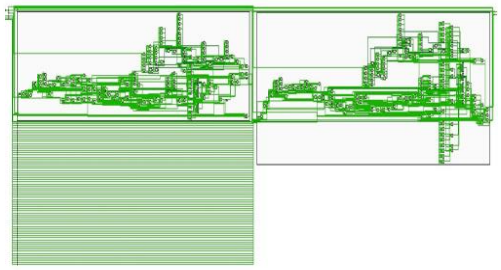

Fig.11 RTL Schematic of I2C Protocol

The above given figure 11 shows the RTL schematic daigram of the I2C single master and single slave protocol.

Table 3 Area of I2C Protocol

Resource	Utilization	Available	Utilization %
LUT	150	20800	0.72
FF	120	41600	0.29
IO	69	106	65.09

The above given table gives the value of utilization of LUTs, FF, IO of the I2C single master and single slave protocol.

Table 4 Delay of I2C Protocol

NAME	SLACK	LEVELS	ROUTES	HIGH FAN OUT	FROM	TO	TOTAL DELAY
Path 1	∞	3	4	4	READ OR WRITE	SCL	5.231
Path 2	∞	3	4	4	READ OR WRITE	SDL	5.231

The above given table gives the value of time delay of the I2C single master and single slave protocol.

5. Conclusion and Future Scope

In the proposed method for I2C, a parameter is used to select the number of bytes to be sent or received. This allows for flexibility in the number of bytes transferred, and the use of a state machine for data transfer control makes the process adaptable to different applications .A parameter is also used in the proposed SPI technique to choose how many bytes are delivered or received. As a result, the amount of bytes transported may be varied, and the process can be tailored to various applications because it uses a state machine for data transfer control. To sum up, SPI and I2C are both strong communication protocols, each having advantages and disadvantages of its own.

The particular needs of the application, including the quantity of devices, data rate, power consumption, and system complexity, will determine which option is best. Integration into advanced chips like System-on-Chips (SoCs) and Microcontroller Units (MCUs) ensures seamless connectivity, with potential enhancements like DMA support for faster data transfers. Their compatibility with existing hardware and software ecosystems ensures they'll remain relevant, vital for backward compatibility and interoperability. SPI and I2C excel in energy efficiency and low pin count, making them ideal for battery powered and space-constrained devices. Supported by industry standards, they guarantee ongoing development, interoperability, and widespread adoption across various sectors. Their applications extend beyond traditional embedded systems to automotive electronics, industrial automation, consumer electronics, and healthcare devices, adapting to new requirements as they arise. Despite the emergence of newer protocols like I3C, SPI and I2C's widespread adoption and versatility ensure their continued relevance in diverse embedded systems and IoT applications.

Declaration

Conflicts of Interest: The authors declare no conflict of interest.
Author contribution: All authors wrote the main manuscript text and also consent to the submission
Ethical approval: Not applicable.
Consent to Participate: All authors consent to participate.
Funding: Not applicable, and No funding was received
Institutional Review Board Statement: Not applicable.
Informed Consent Statement: Not applicable.
Personal Statement: We Declare with our best of Knowledge that this research work is purely Original Work and No third party material Not

used in this article drafting. If any such kind material found in further online publication, we are responsible only for any judicial and copyright issues.

Acknowledgements
We thank everyone who inspired our work.

References

[1]. J.Mankar, C. Darode, K. Trivedi, M. Kanoje, and P. Shahare, "Review of I2C protocol," International Journal of Research in Advent Technology, Vol. **2**, no. **1**, **2014**.

[2]. V. K. Pandey, S. Kumar, V. Kumar, and P. Goel, "A Review Paper on I2C Communication Protocol," International Journal of Advance Research, Ideas and Innovations in Technology, Vol. **4**, no. **2**, pp. **340-343, 2018.**

[3]. T.Leal-del Río, G. Juarez-Gracia, and L. N. Oliva-Moreno, "Implementation of the communication protocols SPI and I2C using a FPGA by the HDL-Verilog language," Research in Computing Science, **2014,** pp. **31-41.**

[4]. A.K.Oudjida, A. Liacha, D. Benamrouche, M. Goudjil, R. Tiar, and A. Ouchabane, "Universal Low/Medium Speed I2C-Slave Transceiver: A Detailed FPGA Implementation," Journal of Circuits, Systems, and Computers, Vol. **17**, no. **04**, pp. **611-626, 2008.**

[5]. Ghanekar, B. Kishor, and S. Bandewar, "Design and Implementation of SPI Bus Protocol," International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, Vol. **5**, no. **5**, pp. **4155-4157, 2016.**

[6]. J. K.Singh, M. Tiwari, and V. Sharma, "Design and Implementation of I2c master controller on FPGA using VHDL," IJET, Vol. **4**, no. **4**, **2012**.

[7]. A.K.Oudjida, M. L. Berrandjia, R. Tiar, A. Liacha, and K. Tahraoui, "FPGA implementation of i2c & spi protocols: A comparative study," in 2009 16th IEEE International Conference on Electronics, Circuits and Systems- (ICECS 2009), pp. **507-510**, IEEE, **2009.**

[8]. F. Leens, "An introduction to I2C and SPI protocols," in 2009 IEEE Instrumentation & Measurement Magazine.

[9]. A.K. Oudjida, A. Ouchabane, and M. Liem, "Master-slave wrapper communication protocol: A case study," in Proceedings of the 1st IEEE international computer systems and information technology conference ICSIT, Vol. **5**, pp. **461-467, 2006.**

[10]. J.Zhang, J. Wang, C. Wu, and W. Zhang, "The design and realization of a comprehensive SPI interface controller," Second International Conference on Mechanic Automation and Control Engineering (MACE), **2011** IEEE.

[11]. F.Leens, "An introduction to SPI and I2C protocol," IEEE Instrumentation and Measurement magazine, February 2009.

[12]. O. Romain, T. Cuenin, and P. Garda, "Design &modeling of an I2c Bus Controller," FDL 0'3, Frankfurt, Deutschland, Sept **23-26, 2003.**

[13]. P. K. Mehto, P. Mishra, and S. Lal, "Design and Implementation for Interfacing Two Integrated Device Using I2C Bus," IJRICCE, ISSN (Online): **2320- 9801**, Vol. **2**, Issue **3**, March **2013.**

[14]. A.K. Oudjida, M. L. Berrandjia, A. Liacha, R. Tiar, K. Tahraoui, and Y. N. Alhoumays, "Design and Test of General-Purpose SPI Master/Slave IPs on OPB Bus," in 2010 IEEE International Conference on Computer Design (ICCD), **2010**, pp. **655-660.**

[15]. S.Palnitkar, "Verilog HDL: A Guide to Digital Design and Synthesis," SunSoft Press, **1996**.

[16]. Philips Semiconductors, "I2C Bus Specification and User Manual," **1995**.

[17]. Texas Instruments, "Serial Peripheral Interface (SPI) Basics," **2013**.

[18]. NXP Semiconductors, "Understanding the I2C Bus," **2018**.

Author's Profiles

Dr.P.Gangadhara Reddy is currently working as Associate Professor in Aditya College of Engineering,Madanapalle. He completed Ph.D in the area of Digital Image Processing from S.V.University College of Engineering, S.V.University, Tirupati, Andhra Pradesh, India. He completed his M.Tech degree in Electronic Instrumentation and Communication Systems from Sri Venkateswara University College of Engineering, Tirupati and B.Tech in ECE from Rajiv Gandhi College of Engineering &Technology, Nandyal. He has published 18 technical papers in different reputed International journals and International conferences. His areas of research interest include Machine learning,Deep Learning,Digital Image Processing and Embedded System design. He is a Life Member of ISTE.

N. Umamaheswar Reddy received his engineering diploma in Electronics and Communication Engineering from Government Polytechnic-Ananthapur in 2021. He is currently pursuing B.Tech., in Electronics and Communication Engineering in Aditya college of engineering, madanapalle.

K. Sai Charan is currently pursuing B.Tech., in Electronics and Communication Engineering in Aditya college of engineering, madanapalle.

B. Subramani Pavan is currently pursuing B.Tech., in Electronics and Communication Engineering in Aditya college of engineering, madanapalle.

N. Ramesh is currently pursuing B.Tech., in Electronics and Communication Engineering in Aditya college of engineering, madanapalle.