Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 45 No. 1 (2024)

Multi Protocol Transceiver :Bridging I2C and SPI With Verilog

B.Prathyaksha Sankaran[(1)],K.S.Dhanveer Santhosh[(2)],V.Shree Nagarajan[(3)],

G.B.Raj Gowtham[(4)],S.Karthika[(5)]

[(1) (2) (3)(4)]UG Student, National Engineering College, Kovilpatti.

[(5)]Assistant Professor in ECE Department, National Engineering College, Kovilpatti.

National Engineering College, Kovilpatti

Abstract:- A key difficulty in the dynamic world of embedded systems and the Internet of Things (IoT) is creating effective communication between various components. In order to solve this problem, this project uses the Verilog hardware description language to create a flexible multi-function serial interface that can simulate both the Serial Peripheral Interface (SPI) and Inter-Integrated Circuit (l2C) communication protocols, taking into account the unique benefits that each offers in various situations. The interface lowers design complexity, improves interoperability, and future-proofs devices by smoothly switching between SPI and I2C modes. Additionally, the project lays a strong emphasis on reducing space and power usage to make sure that this novel interface not only enhances communication but also complies with strict resource limitations in embedded systems and IoT applications.

Keywords: SPI, I2C, VERILOG

1. Introduction

Effective communication across many components continues to be a major difficulty in today's fast expanding technological landscape. The project, which aims to address this difficulty, heavily relies on Verilog, a specialized hardware description language. SPI is a full-duplex, master-slave communication protocol. It typically involves one master device that controls one or more slave devices. Communication occurs over multiple wires, including a clock signal (SCK), a data output from the master to the slave (MOSI - Master Out Slave In), a data input from the slave to the master (MISO - Master In Slave Out), and a chip select (CS or SS) line for each slave device. l2C stands for Inter- Integrated Circuit. It is a bus interface connection protocol incorporated into devices for serial communication. It was originally designed by Philips Semiconductor in 1982. Recently, it is a widely used protocol for short-distance communication. It is also known as Two Wired Interface (TWI) The main goal of the project is to create a flexible, multi-functional serial interface that can accurately simulate the Serial Peripheral Interface (SPI) and Inter-Integrated Circuit (I2C) communication protocols. The importance of Verilog rests in its capacity to precisely and effectively simulate complicated digital systems. Model Sim is used for thorough functional verification, thoroughly testing the performance of the interface under numerous circumstances in order to guarantee its dependability and functionality. Additionally, to meet the various requirements of reconfigurable devices, Xilinx Vivado improves the interface for effective deployment on field-programmable gate arrays (FPGAs). In the constantly changing field of advanced technology applications, this comprehensive approach, combining Verilog's high-level abstraction with meticulous testing and analysis, aids in the creation of reliable, scalable, and resource-efficient solutions.

2. Problem Statement

In contemporary vlsi systems design, interfacing with a multitude of peripheral devices is a common requirement. These peripheral devices often employ distinct communication protocols, such as SPI (Serial Peripheral Interface)

and l2C (Inter-Integrated Circuit), each characterized by unique electrical and timing specifications. The challenge is to develop an efficient and adaptable communication module that seamlessly integrates both SPI and l2C protocols into a single system, enabling the efficient exchange of data while adhering to the specific requirements of each protocol.

3. Literature Survey

In this Literature survey various works related to SPI protocol, l2C Protocol, their conversion and integration have been reviewed. Tatiana Leal-de! Rio, Gustavo Juarez-Gracia, and L. Noe Oliva-Moreno [1] have conducted research into communication protocols, specifically SPI (Serial Protocol Interface) and l2C (Inter-Interface Circuit). Their work delves into the intricacies of these protocols and their implementation using the versatile hardware programming language, Verilog HDL. SPI is highlighted as a dependable, full-duplex, and synchronous serial data link, essential for achieving high-speed data transfer in embedded electronic modules. In contrast, l2C offers versatility, supporting various configurations and operational modes, with data transfer rates ranging from 100,000 to an impressive 3.4 million bits per second. This comprehensive review underscores the significant progress made in this field through earlier research, emphasizing the critical role these protocols play in the realm of embedded systems.

Ricardo de Porras Bemacer [2] discusses the Synthetic Aperture Radar (SAR) system in this paper, a powerful remote sensing tool used for high-resolution terrain imaging at microwave frequencies. Typically mounted on aerial platforms such as airplanes or satellites, the SAR system periodically emits electromagnetic pulses to illuminate terrain and collects return echoes containing crucial reflectivity data. To improve imaging resolution, the system employs platform movement and synthetic aperture radar imaging techniques. A pivotal component of the system is the Formatting Block, which rapidly stores and formats radar pulses and echoes at a line rate of 1.5 Gbps, encoding them in 8b/10b at 75 MHz. Raw data is translated and processed by the Real-time Communications Block before being transmitted via serial FPDP protocol data units for further processing. The VME Access Block facilitates seamless system configuration communication. The TRX Communications Block utilizes the high-speed differential Aurora protocol for communication with the transmitting/receiving unit, while the Temperatures Block monitors up to 32 diodes for temperature changes in this critical environment. Clocking resources, including 300 MHz and 106.25 MHz clocks, are employed to streamline data handling, and hardware utilization information underscores the system's performance capabilities.

Konstantin Mikhaylov and Jouni Tervonen [3] have assessed the power efficiency of digital serial interfaces in microcontrollers. Their study, published in 2012, focuses on communication interfaces in low-power embedded systems commonly found in devices like computers, communication devices, and medical devices. The research evaluates energy consumption, resource requirements, data rates, and efficiency of popular digital interfaces, including UART, SPI, and l2C, across various scenarios. The findings reveal that hardware-based implementations generally outperform software-based ones, with SPI emerging as the most energy-efficient interface. In contrast, UART and l2C introduce extra bits and bytes, contributing to higher energy consumption. When transmitting nine data bytes, l2C consumes approximately 17.44 J of hardware energy and 83.6 J of software energy, while SPI requires approximately 22.52 J of hardware energy and 65.25 J of software energy. These results provide valuable insights for selecting energy-efficient communication interfaces, especially relevant for wireless sensor network applications. The study suggests potential for future research in predictive power consumption modeling in specific environments.

Konstantin Mikhaylov and Jouni Tervonen [4] conducted a study published in 2012, assessing the power efficiency of digital serial interfaces in microcontrollers. Their research focuses on communication interfaces within low-power embedded systems commonly found in devices like computers, communication devices, and medical devices. The study comprehensively evaluates energy consumption, resource requirements, data rates, and efficiency of popular digital interfaces, including UART, SPI, and l2C, across various scenarios. The findings indicate that hardware-based implementations generally outperform software- based ones, with SPI emerging as the most energy-efficient interface. Conversely, UART and 12C introduce additional bits and bytes, contributing to higher energy consumption. For instance, when transmitting nine data bytes, l2C consumes approximately

17.44 J of hardware energy and 83.6 J of software energy, while SPI requires approximately 22.52 J of hardware energy and 65.25 J of software energy. These results provide valuable insights for selecting energy-efficient communication interfaces, especially pertinent for wireless sensor network applications. The study also suggests potential for future research in predictive power consumption modeling tailored to specific environments.

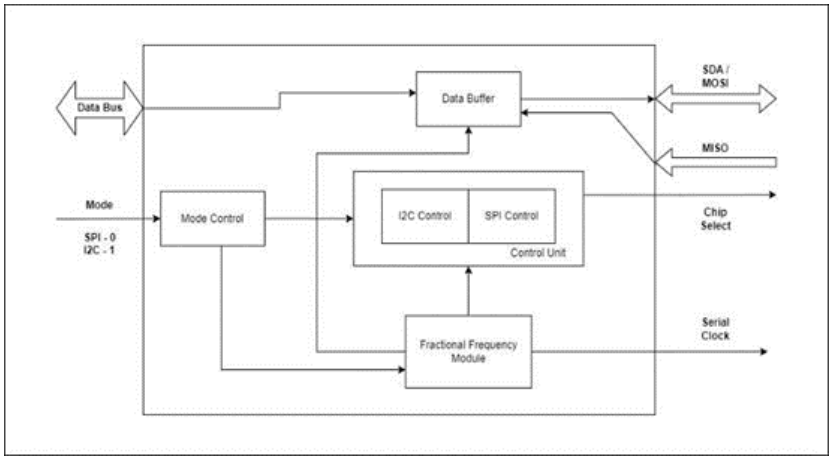
Xingchun Liu1, 2, Yandan Liu [5] introduced a versatile approach to serial communication interfaces in their 2017 study titled "Multi-functional Serial Communication Interface Design Based on FPGA." This design encompasses RS232, IIC (12C), and SPI interfaces, commonly used for short-distance data transmission. Recognizing the diverse features offered by these interfaces, including clock synchronization, communication distance, data rates, and protocol complexity, the study addresses hardware limitations by proposing a multi-functional serial communication interface design implemented in Field Programmable Gate Arrays

(FPGAs) using the Verilog hardware description language. Leveraging FPGA attributes like flexibility and rapid design cycles, the design accommodates IIC as a synchronous half- duplex protocol suitable for near-field and non-recurrent data communication, SPI as a synchronous full-duplex protocol reliant on clock signals, and asynchronous RS232 supporting full-duplex communication with specified baud rate requirements. The implementation employs fractional frequency modules for SPI and IIC, serial-parallel conversion modules for SPI, mode control for protocol selection, and baud rate settings for RS232. Notably, this multi-functional communication interface offers an adaptable and resource-efficient solution for diverse communication needs, resulting in approximately 27% fewer pins, 25% fewer registers, and 10% fewer logic elements when compared to the cumulative requirements of three separate interfaces.

4. Methodology

The overall design of the module is comprised into separate modules for SPI and l2C respectively, fractional frequency module for dividing the clock frequency of the processor/master device so as to match the data transfer frequencies of SPI and l2C protocols. Also, a top level module encapsules the individual SPI and 12C protocols and the fractional frequency module, and also works as mode select (SPI or 12C) module, and connects them as per the block diagram. The SPI and 12C modules are designed using the state machine approach and also common pins are used for similar functionalities for SPI and l2C protocols, in order to reduce the pin count.

Combining SPI and 12C modules into a single top module, rather than utilizing separate integrated circuits (ICs), presents several advantages. This approach offers space savings, critical in compact electronic devices with limited space. It simplifies communication between SPI and I2C devices, streamlining design and reducing potential errors. Resource sharing optimizes hardware utilization, enhancing power efficiency and simplifying debugging and testing. Scalability allows for future expansions or new protocols, accelerating development and leading to cost savings. Overall, this compact, resource-efficient, and scalable top time.



Overall Block Diagram

module design streamlines integration, reduces complexity, and enhances long-term adaptability and cost-effectiveness.

4.1 Software Tools used:

ModelSim is a versatile electronic design automation (EDA) tool widely employed for simulating, verifying, and debugging digital and mixed-signal designs. It supports hardware description languages like VHDL and Verilog, facilitating comprehensive analysis and testing of designs prior to physical implementation. Offering features such as waveform visualization, code coverage assessment, and advanced debugging tools, Model Sim is essential in industries like semiconductor and FPGA development, aiding engineers and designers in ensuring design correctness and performance. This project uses modelsim for writing the Verilog and and also for functional verification and timing diagram.

Xilinx Vivado, a comprehensive FPGA development environment, offers numerous advantages for digital design and development. It provides high-level abstraction tools like HLS, advanced synthesis, and optimization capabilities, streamlining the entire design process. Vivado's rich library of pre-designed IP cores simplifies complex functions' integration, while powerful debugging tools ensure thorough analysis and debugging. It supports heterogeneous computing with FPGA and ARM processors, caters to various FPGA families, and is optimized for Ultra Scale devices. Additionally, Vivado includes power analysis tools, benefits from a vibrant community and ecosystem, and receives regular updates with new features, making it an indispensable resource for efficient and high-performance FPGA design and development. Xilinx vivado is used in this project for the purpose of fpga analysis such as power analysis (static and dynamic) and also resource utilization.

4.2 SPI module overview:

The SPI module operates through a series of states outlined in the state diagram.

1. Idle State (idle): In this standby state, the SPI module awaits the start of a new data transaction. Key operations include initializing various signals like ss (Slave Select), mosi (Master Output Slave Input), and done to their idle states, typically logic low (0). The count variable is reset to zero, indicating the start of a new transaction, and the input_buffer is cleared to receive data.

2. Transmit State (state_t): This state manages data transmission onto the SPI bus. It activates the ss signal to select a specific slave device for communication. Data to be transmitted (mosi) is sourced from the data input, allowing data transfer to the selected slave device. Simultaneously, incoming data from the slave device is captured on the miso line and stored in the input_buffer. The count variable is incremented to facilitate the transmission of all 8 data bits, and this state remains active until count reaches 7, ensuring complete data transmission and reception.

3. Finish State (finish): The finish state marks the end of the data transmission phase in the SPI transaction. The ss signal remains active, enabling the selected slave device to finalize data reception. The done signal is set to logic high (1), indicating successful completion of the SPI transaction. Data transmission concludes, and the count variable remains unchanged.

4. Next State (next): After completing a transaction, the next state prepares the SPI module for a new transaction. The done signal is reset to logic low (0), indicating readiness for a new transaction. The SPI module transitions back to the idle state, awaiting the initiation of another SPI transaction. The count variable remains unchanged and is reset to zero only when a new transaction is initiated.

4.3 l2C Module Overview:

The 12C module operates through a series of states outlined in the state diagram.

1. Idle State (idle): This state indicates that the I2C module is in a waiting mode, prepared for the commencement of a new data transaction. Operations include setting the SDA (Serial Data Line) to its idle state, typically logic high (1), and keeping the SCL (Serial Clock Line) inactive, signifying no ongoing data transfer. Relevant variables and buffers are prepared for the new transaction.

2. Start State (start): The purpose of this state is to initiate the I2C data transfer by generating the start condition on the bus. It involves transitioning the SDA line from logic high (1) to logic low (0), indicating the start condition. The master device then transmits the slave address and the read/write bit.

3. Address State (address): In this state, the address state validates the acknowledgment of the slave's address and sets the direction of data transfer. The SDA line remains stable as it awaits the slave's acknowledgment. The direction of data transfer (read or write) is determined based on the acknowledgment received.

4. Data Transfer State (data): This state is dedicated to data transfer between the master and slave devices. Data is transmitted or received on the SDA line, depending on the direction determined in the previous state. Multiple bytes of data can be transferred in sequence.

5. Stop State (stop): The stop state concludes the I2C transaction and releases control of the bus. It involves transitioning the SDA line from logic low (0) to logic high (1), indicating the stop condition. The SCL line remains inactive, marking the end of the data transfer.

4.4 Top Level Module Overview:

The top-level module, referred to as the 'top module,' assumes a pivotal role as the central control unit overseeing communication between core system components and peripheral devices, with a focus on the utilization of both the Serial Peripheral Interface (SPI) and Inter-Integrated Circuit (I2C) protocols. Its primary objective is resource optimization and the provision of meticulous timing control to facilitate efficient communication.

One of the fundamental functions of the top module is Mode Selection (mode). This module incorporates an input wire called 'mode,' which serves as the mechanism for selecting between two distinct communication modes. When 'mode' is configured to '0,' it triggers the activation of the SPI communication mode. Conversely, setting 'mode' to 'l' initiates the I2C communication mode.

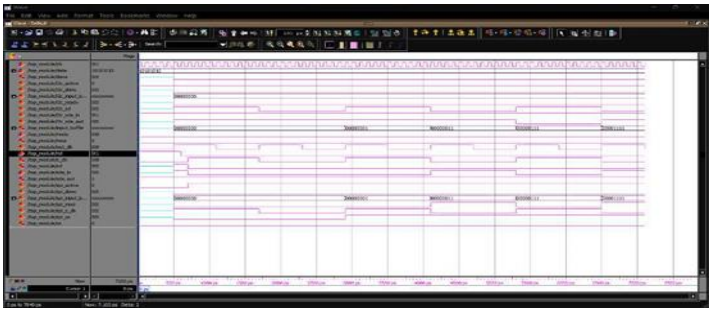
Within the top module, both SPI and I2C Interfaces are instantiated, each operating as master modules responsible for managing communication with peripheral slave devices. The SPI module is equipped to facilitate SPI-based communication through signals such as 'ss,' 'mosi,' 'miso,' and 's_clk.' Similarly, the I2C module is responsible for handling I2C-based communication, utilizing signals 'sda_in,' 'sda_out,' and 'scl.'.

The 'elk_div' submodule within the top module is pivotal for generating the 'out_elk' signal, which serves as the shared reference clock for both the SPI and 12C modules. It ensures synchronized data transfer, prevents errors, and provides precise timing control, enhancing communication reliability and accuracy.

5. Results and Discussions

This The simulation results encompass the functional verification of the SPI, 12C, and Top-Level Modules utilizing ModelSim software. In the SPI module, the waveform illustrates the transition from the reset state to operational states, depicting the initiation of data transmission and reception with meticulous timing control. Similarly, for the I2C module, the waveform portrays the initialization, address recognition, and error-free data exchange during read operations, with the 'done' signal indicating successful completion.

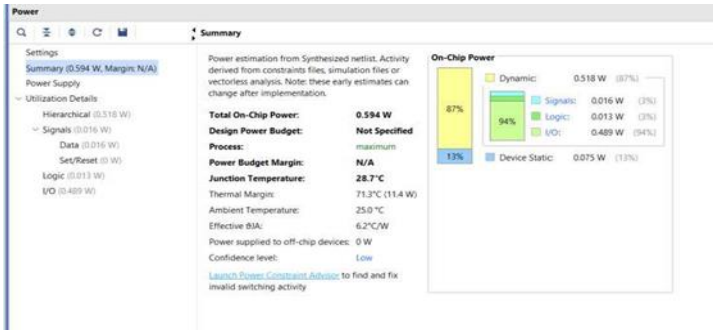
Within the top-level module, the 'mode' signal governs the selection of SPI or I2C communication. The clock divider, generating the 'out_clk' signal, ensures synchronization with peripheral devices operating at varying baud rates. This orchestration seamlessly activates the chosen communication protocol, facilitating startup, data transfer, and acknowledgment stages. The top-level module's adaptive design accommodates diverse communication requirements, utilizing peripheral device acknowledgment signals for data transmission confirmation and reattempts when necessary. Overall, these simulations confirm the precision and adaptability of the communication modules in interfacing with external devices



Waveform of top level module (SPI and I2C Combined) in Model sim.

The design of the system emphasizes efficiency, particularly within the SPT (Serial Peripheral Interface) and T2C (Inter- Integrated Circuit) modules. Both modules are optimized for low power consumption, with the SPI module excelling in typical conditions and the I2C module demonstrating impressive efficiency. Notably, the entire system maintains low power usage, highlighting the intelligent design that utilizes 12 I/0 ports for effective communication and data exchange. This energy- conscious approach positions the system as an ideal choice for applications where power conservation is a critical consideration, underscoring its practicality and reliability.

The power analysis was performed in Artix-7 FPGA board using Xilinx vivado.



Power analysis of the top level module in Xilinx vivado

Title	SPI	I2C	Overall Design
Static Power(Typical)	0.067W	0.06W	0.060W
Dynamic Power(Typical)	3.357W	0.284W	0.518W
Total On-Chip Power(Typical)	3.424W	0.345W	0.579W
I/O Ports	9	9	12
Static Power(Maximum)	0.087W	0.087W	0.075W
Dynamic power(Maximum)	3.357W	0.0284W	0.518W
Total On-Chip Power(Maximum)	3.444W	0.359W	0.594W

Comparison of SPI,I2C and Overall design

6. Conclusion

In conclusion, the Overall Design, which combines elements of both I2C and SPI, emerges as an efficient choice. While it offers a higher number of I/0 ports to accommodate diverse connectivity needs it is less than the number of i/o ports when the individual ports of SPI and l2C are added. The overall design maintains a power consumption profile that falls between the individual l2C and SPI designs. This balanced approach ensures efficient performance, making it a suitable option for applications that require versatility without sacrificing power efficiency.

In future iterations of this project, there is room for enhancements to the multi-protocol communication module. These enhancements could encompass improved performance in terms of bitrate, additional features such as error control, lower pin count and expanded compatibility with a broader range of devices. These advancements will further strengthen the capabilities of this project.

7. References

[1] F. Leens, "An Introduction to 12 C and SPI Protocols", IEEE Instrumentation & Measurement Magazine, pp. 8-13, February 2009.

[2] J.M. Irazabel & S. Blozis, Philips Semiconductors, "12 C- Manual", Application Note, ref. AN10216-0, March 24, 2003.

[3] A.K. Oudjida et al, "Universal Low/Medium Speed T2C Slave Transceiver: A Detailed FPGA Implementation," Journal of Circuits, Systems and Computers (JCSC), Vol. 17, No. 4, pp. 611-626, August 2008, ISSN: 02 I 8-1266, USA.

[4] Xiaomin Huang and Zhijie Zhang, "Design of ip core for IIC bus controller based on FPGA", Journal of Measurement Science and Instrumentation, vol. I, pp. 13-18, 2015.

[5] Yuwen Wang,Weixin Jin,Yibing Cai and Liping Yan, "Implementation ofspi bus interface based on FPGA", Modern Electronics Technique,vol.14, 2010.

[6] Xiaomin Zhong and Xiao Feng"Design of iic bus interface protocol and fpga realization", Modern Navigationvol.4pp.29 l-294, 2016.

[7] Abhinav Boddupalli, "Design and Implementation of l2C bus protocol on FPGA using verilog for EEPROM", Proceedings of IEEEFORUM International Conference, 01st October, 2017, Pune, India.

[8] Shaik.fazil ahmed, Y.Murali, "Implementation of l2C multi task and multi slave bus controller using verilog", International journal of engineering and computer science ISSN: 2319-7242 volume 4 issue 8, Aug 2015

[9] DVIJEN TRIVEDI, Aniruddha Khade, Kashish Jain, Kashish Jain, Ruchira Jadhav, "SPI to l2C Protocol Conversion using Verilog", Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), 2018.

[10] Zheng-wei HU, "l2C Protocol Design for Reusability", Third International Symposium on Information Processing, April-2010 IEEE

[11] A.K. Oudjida, M.L. Berrandjia, R. Tiar, A.Liacha, K. Tahraoui, "FPGA Implementation of 12c & SPI Protocols: A Comparative Study" Electronics, Circuits, and Systems, ICECS 2009.

[12] F. Leens, "An Introduction to l2C and SPI Protocols," IEEE Instrumentation & Measurement Magazine, , February 2009, pp. 8-13.

[13] Abhilash S.Warrier, Akshay S.Belvadi, Dhiraj R.Gawhane, Babu Ravi Teja K, FPGA Implementation Of SPI To l2C Bridge, International Journal of Engineering Research & Technology (IJERT), Vol. 2 Issue 11, November - 2013.

[14] P. Myers, "Interfacing Using Serial Protocoles: Using SPI and 12 C".

[15] Philips Semiconductors, "The IIC-Bus Specifications," version 2.1, January 2000.

[16] Motorola Inc., "SPI Block Guide V03.06," February 20