WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

A Survey on UART controller design for SoC Implementation

[1]Manoj Patidar, [2]Deepak Sharma

[1]M. tech Scholar, [2]Assistant Professor

[1, 2]Dept. of ECE, LNCT (Bhopal) Indore Campus, Indore (M.P)

ABSTRACT :

UART (Universal Asynchronous Transceiver) is a serial communication protocol mainly used for short-distance, low-speed, and low-cost data exchange. We don't need all the functions of UART, we just integrate its core. The UART consists of three main modules of baud rate generator, receiver, and transmitter. UART implemented in VHDL language can be integrated into FPGA to realize compact, stable, and reliable data transmission. This is very important for the design of the SOC. Simulation results with Quartos II are fully compatible with the UART protocol.

Keywords— UART, asynchronous serial communication, VHDL, Quartus II, simulation.

Introduction

Serial communication is a way of transmitting data between two devices one bit at a time over a single communication line. It is commonly used in microcontrollers, embedded systems, and other electronic devices.

Serial communication can be either synchronous or asynchronous. In synchronous communication, the transmitter and receiver are synchronized using a clock signal, while in asynchronous communication, no clock signal is used. Instead, the data is transmitted as a series of bits, with each byte being preceded by a start bit and followed by one or more stop bits.

There are several types of serial communication protocols, including UART, SPI (Serial Peripheral Interface), and I2C (Inter-Integrated Circuit). Each protocol has its own set of advantages and disadvantages, and the choice of protocol depends on the specific requirements of the application.

Serial communication has several advantages over parallel communication, including the use of fewer communication lines, the ability to transmit data over longer distances, and the ability to transfer data at different speeds. However, it can be slower than parallel communication for large amounts of data.

Overall, serial communication is a widely used and versatile method of transmitting data between electronic devices.

There are several types of serial communication protocols, including:

1. UART (Universal Asynchronous Receiver-Transmitter): As described earlier, UART is a type of asynchronous communication that is commonly used in microcontrollers and other embedded systems.
2. SPI (Serial Peripheral Interface): SPI is a synchronous communication protocol that is used for short-distance communication between microcontrollers and other devices such as sensors, displays, and memory chips. It uses four lines for communication: a clock line, a data line, and two control lines.
3. I2C (Inter-Integrated Circuit): I2C is a synchronous communication protocol that is used for communication between microcontrollers and other devices such as sensors and memory chips. It uses two lines for communication: a clock line and a data line.
4. CAN (Controller Area Network): CAN is a protocol used for communication between microcontrollers and other devices over long distances, such as in automotive and industrial applications. It is a synchronous communication protocol that uses differential signaling for noise immunity.
5. USB (Universal Serial Bus): USB is a popular protocol used for communication between computers and other devices such as printers, cameras, and external hard drives. It is a complex protocol that supports both synchronous and asynchronous communication and uses multiple lines for communication.

These are some of the commonly used serial communication protocols, each with its own set of advantages and disadvantages, and the choice of protocol depends on the specific requirements of the application.

UART (Universal Asynchronous Receiver-Transmitter) is a type of serial communication interface that is commonly used in microcontrollers and other embedded systems. It allows for the transmission and reception of data between two devices, such as a microcontroller and a computer, over a single communication line.

UART is called "asynchronous" because it does not use a clock signal to synchronize the communication between the devices. Instead, it relies on the use of start and stop bits to mark the beginning and end of each data byte. The baud rate, or the rate at which data is transmitted, is specified in bits per second (bps).

UART typically uses two data lines, one for transmitting data (TX) and one for receiving data (RX). The data is transmitted as a series of bits, typically 8 bits per byte, and may also include parity and/or a stop bit. The receiver uses the start and stop bits to identify the boundaries of each byte and then converts the data into a format that can be processed by the receiving device.
UART is widely used in embedded systems due to its simplicity, low cost, and low power consumption. It is often used for communication between a microcontroller and other peripherals such as sensors, displays, and other microcontrollers.

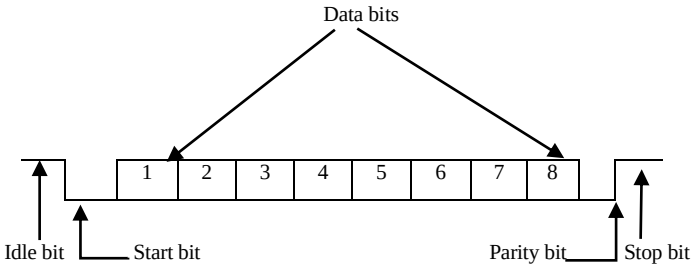

Fig. 1. UART Frame Format

LITERATURE REVIEW :

Related Works

This paper [9] tries to make a design demonstrated into VHDL that means to test UART communication protocols ability of being demonstrated and stable. The paper delivers an overview of the use of the protocol.
In the meantime, this paper [10] efforts to design and synthesis an UART block protocol communication. The research in this paper applies the Baud Rate Generator, Transmitter, and Receiver Modules.
In the paper [11], the investigation makes a design that can run SPI communication protocol in FPGA. The paper also defines how Clock Polarity and Phase which is a mode that can be used for synchronization between master and slave. The paper describes the port description used both on SPI Master and SPI Slave.
In the paper "SPI Execution on FPGA" [12], the study also tried to implement SPI communication protocol on FPGA. In the projected architectural design, the project uses registers on SPI Master and SPI Slave. Also in this architectural design is the use of Clock Generator.

Serial Peripheral Interface

In its implementation, the SPI protocol uses two modules, Master and Slave [12]. This communication protocol requires 3+N paths where the number of N depends on the number of slaves. The paths are MISO, MOSI, SCLK, and SS [13]. 1) Architecture of SPI: SPI communication protocol has three important parts that are Clock Generator, Master Module, and Slave Module. In Fig. 1, the master has three output paths that are MOSI, SS, and SCLK. MOSI is used by the master to send data to the slave by making the slave select value go to the low. The slave module has an output of only 1, MISO. The path is used for the slave to provide data according to the speed given by the Slave Clock of the master. 2) Timing Diagram of SPI: SPI communication protocol has several modes. In Fig. 2, there are four applicable models [12].

• Model 0
Model 0 it uses CPOL = 0 and CPHA = 0. In this mode, SCK starts with a low state. The data will be captured when the clock rises (low to high transition) and data will be received when the clock drops (high to low transition).
• Model 1
Model 1 it uses CPOL = 0 and CPHA = 1. In this mode, SCK starts with a low state. The data will be captured when the clock drops (high to low transition) and the data will be increased as the clock rises.
• Model 2 Model 2 is using CPOL = 1 and CPHA = 0. In this mode, SCK starts with a high state. The data will be captured when the clock rises (low to high transition) and data will be received when the clock drops (high to low transition).
• Model 3 Model 3 it uses CPOL = 1 and CPHA = 1. In this mode, SCK starts with a high state. The data will be captured when the clock rises (low to high transition) and data will be received when the clock drops.

Universal Asynchronous Receiver-Transmitter (UART)

It is the protocol for communicating that has asynchronous characters [9] [14]. It also could communicate in full-duplex, which will be used in data communications and control systems [9]. This protocol has two signal paths that is RXD and TXD for full-duplex communication [9].

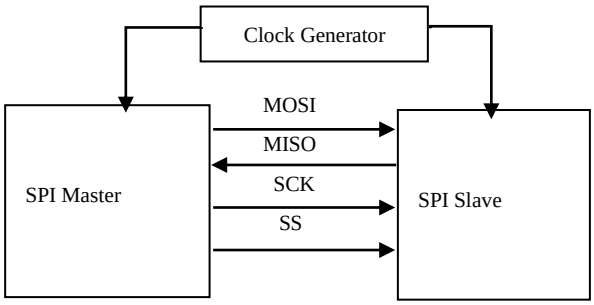

Fig. 2. Architecture SPI.

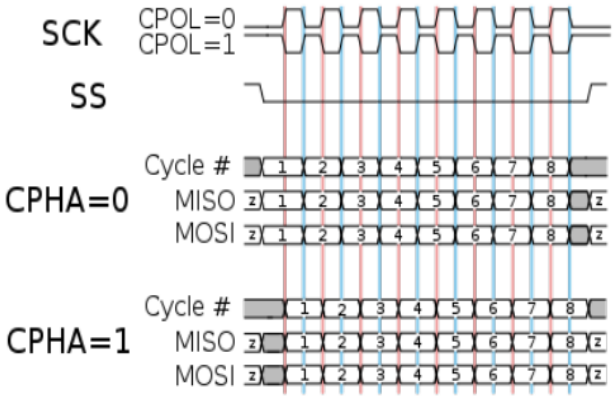

Fig. 3. Timing diagram SPI

Architecture of UART: UART communication protocol has three important parts, those are transmitter and receiver modules and Baud Rate generator as shown in Fig. 3.

- Baud Rate Generator

It has function as clock divider. There are two factors affecting the baud rate frequency such as clock of the clock oscillator and baud requests [9]. The purpose of using a baud rate generator is to speed up the asynchronous serial data [9] [10].

- Receiver Module

The receiver Module is responsible for receiving data from RXD input pins. In the process, the displacement from signal 1 to 0 means the beginning of the reception of data [10] [9]. • Transmitter Module

The function of the Transmitter Module is to change from parallel to serial data, adding some bits to data for controlling purpose [10] [9].

Timing Diagram of UART: UART communication protocol has a timing diagram as in Fig. 4.

Fig. 4 indicates that idle state is high or 1, when there is a displacement of signal 1 to 0, it can be said as the start bit, and after that the data is beginning to be delivered to up to 8 bits. After the 8 bits data has been sent, the last bit is the stop bit which indicates that the data has been read and then the signal will come back to idle or become a high signal.

III. METHODOLOGY AND SYSTEM DESIGN

Methodology

The methodology of the analysis and implementation of this communication protocol cover the design of architecture and communication protocols and how the communication protocol verification tests have been implemented towards the FPGA.

The design of the communication protocol is a process design block that will be required in accordance with the

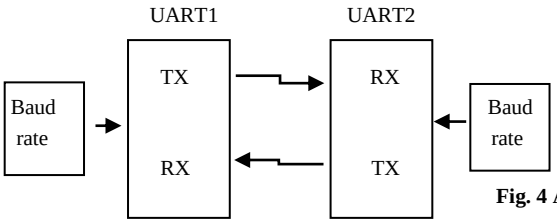

Fig. 4 Architecture UART.

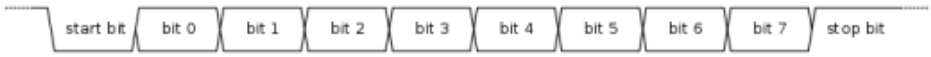

Fig. 5 Timing diagram UART

Standard provisions of the communication block and then the design with VHDL. After the design is finished, the next step is synthesizing the design. After completion of verification, it is then applied to the FPGA. Then the verification of the communication protocol, it will be done by communicating with Raspberry Pi.

1) Designing Architecture of Communication Protocol Design: In the Design of this Communication Protocol, communication protocols will be built, designed, and implemented towards the FPGA. The task of designing is to design a protocol to have standardized communication validation. The block design is built on the design and communication protocols.

2) Designing Architecture of Communication Protocol Verification: Verification of this communication protocol is a way used to know whether the design built on the FPGA can run on a standard communication protocol. This section uses Raspberry Pi hardware as the equipment that verifies it.

IV. CONCLUSION AND FUTURE SCOPE

Conclusion.

UART communication protocol plays a vital role in the field of serial communication. Because of its simple design and less wiring. It is also very famous because it is very easy to implement in any hardware either via software or hardware.

It's also very easy for programmers to implement because of its simple design protocol. In the available UART protocol, the suggested implementation will increase the communication speed and make UART for fast communication with available resources. The speed can be increased min of 1.22 times of available speed. That reduces the use of SPI which is a complex communication protocol as well as requires 4- wire to communicate.

Future Scope.

Serial communication is the backbone of a communication system because of its less no of required hardware and long-distance communication protocol.

Since serial communication sends the data in series it requires only a single line which reduces the cost of hardware. It also reduces the maintenance cost.

In serial communication, it is very easy to find errors in transmission lines because of less no wiring.

The Suggested Hybrid UART will increase the speed of UART communication protocol and increase the usability of UART.

In the Future speed of data, transmission is very important. So it is very essential to increase the speed of available serial communication protocol.

REFERENCES :

1. Zou,Jie Yang,Jianning。Design and Realization of UART Controller Bas ed on FPGA.
2. Liakot Ali， Roslina Sidek， Ishak Aris， Alauddin Mohd. Ali， Bambang Sunaryo Suparjo. Design of a micro - UART for SoC application [J].In: Computers and Electrical Engineering 30 (2004) 257– 268.
3. HU Hua, BAI Feng-e. Design and Simulation of UART Serial Communication Module Based on Verilog -HDL[J]. J ISUANJ I YU XIANDA IHUA 2008 Vol. 8.
4. Frank Durda Serial and UART Tutorial. uhclem@FreeBSD.org.
5. statista, "Internet of things (iot): number of connected devices worldwide from 2012 to 2020 (in billions)," https://www.statista.com/statistics/471264/iot-number-of-connected devices-worldwide/, 2017, accessed : 2017-02-11.
6. R. Buyya and A. V. Dastjerdi, Internet of Things: Principles and paradigms. Melbourne, Australia: Elsevier, 2016.
7. M. Abdurohman, A. Sasongko, and R. Rawung, "Mobile tracking system based on event driven method," in Applied Mechanics and Materials, vol. 321-324. Trans Tech Publ, 2013, pp. 536–540.
8. V. Suryani, A. Rizal, A. Herutomo, M. Abdurohman, T. Magedanz, and A. Elmangoush, "Electrocardiagram monitoring on openmtc platform," in 38th Annual IEEE Conference on Local Computer Networks - Workshops. Sydney, NSW: IEEE, 2013, pp. 843–847.
9. M. Abdurohman, A. Herutomo, V. Suryani, A. Elmangoush, and T. Magedanz, "Mobile tracking system using openmtc platform based on event driven method," in 38th Annual IEEE Conference on Local Computer Networks - Workshops. Sydney, NSW: IEEE, 2013, pp. 856–860.
10. M. Abdurohman, A. G. Putrada, S. Prabowo, C. W. Wijiutomo, and A. Elmangoush, "M2m device connectivity framework," International Journal on Electrical Engineering and Informatics, vol. 9, no. 3, pp. 441–454, 2017.
11. W. Dargie and C. Poellabauer, Fundamentals of wireless sensor net works: theory and practice. Chichester, United kingdom: John Wiley & Sons, 2010.
12. M. Abdurohman, Perancangan Embedded System Berbasis FPGA. Yo gyakarta: Graha Ilmu, 2014.
13. Y.-y. Fang and X.-j. Chen, "Design and simulation of uart serial commu nication module based on vhdl," in Intelligent Systems and Applications (ISA), 2011 3rd International Workshop on. IEEE, 2011, pp. 1–4.

14. G. B. Wakhle, I. Aggarwal, and S. Gaba, "Synthesis and implementation of uart using vhdl codes," in Computer, Consumer and Control (IS3C), 2012 International Symposium on. IEEE, 2012, pp. 1–3.
15. V. D. Veda Patil, "Implementation of spi protocol in fpga," International Journal of Computational Engineering Research (IJCER), vol. 3, no. 2, pp. 142–147, 2013.
16. T. D. Shingare and R. Patil, "Spi implementation on fpga," International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 2, no. 2, pp. 7–9, 2013.
17. K. Oudjida, M. L. Berrandjia, R. Tiar, A. Liacha, and K. Tahraoui, "Fpga implementation of i 2 c & spi protocols: A comparative study," in Electronics, Circuits, and Systems, 2009. ICECS 2009. 16th IEEE International Conference on. IEEE, 2009, pp. 507–510.
18. T. P. Blessington, B. B. Murthy, G. Ganesh, and T. Prasad, "Optimal implementation of uart-spi interface in soc," in Devices, Circuits and Systems (ICDCS), 2012 International Conference on. IEEE, 2012, pp. 673–677.