Review on Performance Analysis of UART

Pratiksha Mathurkar[1], Prof. Rahul Nawkhare[2]
[1]*Student, Wainganga College of Engineering & Management*
[2]*Faculty, Wainganga College of Engineering & Management*

***Abstract*-A Universal Asynchronous Receiver/ Transmitter, abbreviated UART, is a type of "asynchronous receiver/transmitter", a piece of computer hardware that translates data between parallel and serial forms. UARTs are commonly used in conjunction with communication standards such as EIA RS-232, RS-422 or RS-485. A UART is usually an individual (or part of an) integrated circuit used for serial communications over a computer or peripheral device serial port. Modern ICs now come with a UART that can also communicate synchronously; these devices are called USARTs (universal synchronous/ asynchronous receiver/transmitter). The UART implemented with VHDL language can be integrated into the FPGA to achieve compact, stable and reliable data transmission. It's significant for the design of SOC. The simulation results with Quartus II are completely consistent with the UART protocol.**

Index Terms- **UART, TDRE, Ga DoSLD**

I. INTRODUCTION

The Universal Asynchronous Receiver Transmitter (UART) is the most widely used serial data communication circuit ever. UARTs allow full duplex communication over serial communication links as RS232. UARTs are available as inexpensive standard products from many semiconductor suppliers, making it unlikely that this specific design is useful by itself. The basic functions of a UART are a microprocessor interface, double buffering of transmitter data, frame generation, parity generation, parallel to serial conversion, double buffering of receiver data, parity checking, serial to parallel conversion. The data is transmitted asynchronously one bit at a time and there is no clock line. The frame format of used by UARTs is a low start bit, 5-8 data bits, optional parity bit, and 1 or 2 stop bits. Universal Asynchronous Receive/ Transmit consists of baud rate generator, transmitter and receiver. The number of bits transmitted per second is called baud rate and the baud rate generator generates the transmitter and receiver clocks separately. Transmitter interfaces to the data bus with the transmitter data register empty (TDRE) and write signals. When transmitting, UART takes eight bits of parallel data and converts it into serial bit stream and transmit them serially. Receiver interfaces to the data bus with the receiver ready and the read signals. When UART detects the start bit, it receives the data serially and converts it into parallel form and when stop bit (logic high) is detected, data is recognized as a valid data. Asynchronous serial communication has advantages of less transmission line, high reliability, and long transmission distance, therefore is widely used in data exchange between computer and peripherals. Asynchronous serial communication is usually implemented by Universal Asynchronous Receiver Transmitter (UART). UART allows full duplex communication in serial link, thus has been widely used in the data communications and control system. In actual applications, usually only a few key features of UART are needed. Specific interface chip will cause waste of resources and increased cost. Particularly in the field of electronic design, SOC technology is recently becoming increasingly mature. This situation results in the requirement of realizing the whole system function in a single or a very few chips. Designers must integrate the similar function module into FPGA.

SYSTEM MODEL

This project uses VHDL to implement the UART core functions and integrate them into a FPGA chip to achieve compact, stable and reliable data transmission, which effectively solves the above problem. Basic UART communication needs only two signal lines (RXD, TXD) to complete full-duplex data communication. TXD is the transmit side, the output of UART; RXD is the receiver, the input of UART. UART's basic features are: There are two states in the signal line, using logic 1 (high) and logic 0 (low) to distinguish respectively. For example, when the transmitter is idle, the data line is in the high logic state. Otherwise when a word is given to the UART for asynchronous transmissions, a bit called the "Start Bit" is added to the beginning of each word that is to be transmitted. The Start Bit is used to alert the receiver that a word of data is about to be sent, and to force the clock in the receiver into synchronization with the clock in the transmitter. These two clocks must be accurate enough to not have the frequency drift by more than 10% during the transmission of the remaining bits in the word. After the Start Bit, the individual data bits of the word are sent, with the Least Significant Bit (LSB) being sent first. Each bit in the transmission is transmitted for exactly the same amount of time as all of the other bits, and the receiver "looks" at the wire at approximately halfway through the period assigned to each bit to determine if the bit is a 1 or a 0. For example, if it takes two seconds to send each bit, the receiver will examine the signal to determine if it is a 1 or a 0 after one second has passed, then it will wait two seconds and then examine the value of the next bit, and so on. When the entire data word has been sent, the transmitter may add a Parity Bit that the transmitter generates. The Parity Bit may be used by the receiver to perform simple error checking. Then at least one Stop Bit is sent by the transmitter. When the receiver has received all of the bits in the data word, it may check for the Parity Bits (both sender and receiver must agree on whether a Parity Bit is to be used), and then the receiver looks for a Stop Bit. If the Stop Bit does not appear when it is supposed to, the UART considers the entire word to be garbled and will report a Framing Error to the host processor when the data word is read. The usual cause of a Framing Error is that the sender and receiver clocks were not running at the same speed, or that the signal was interrupted. Regardless of whether the data was received correctly or not, the UART automatically discards the Start, Parity and Stop bits. If the sender and receiver are configured identically, these bits are not passed to the host. If another word is ready for transmission, the Start Bit for the new word can be sent as soon as the Stop Bit for the previous word has been sent. Because asynchronous data are "self-synchronizing", if there are no data to transmit, the transmission line can be idle.

II. LITERATURE SURVEY

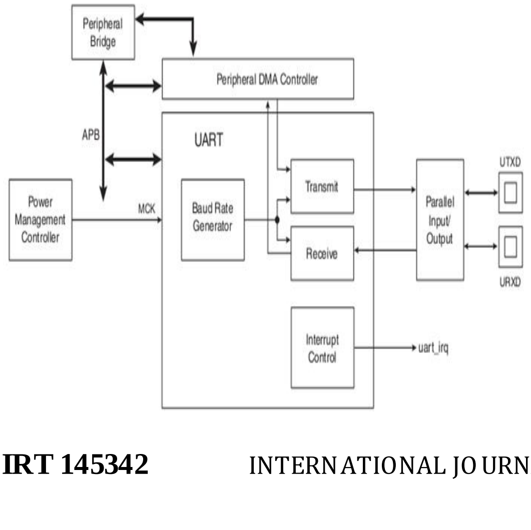
1. Neeraj Jain1 and Seema Sharma2 *LOW POWER UART DESIGN AND SIMULATION FOR SERIAL DATA COMMUNICATIONIJVD*: 3(1), 2012, Pp.5-10 UART (Universal Asynchronous Receiver Transmitter) is a kind of serial communication protocol; mostly used for short-distance, low speed, low-cost data exchange between computer and peripherals. During the actual industrial production, sometimes there is no need the full functionality of UART, but simply integrate its core part. UART includes three kernel modules which are the baud rate generator, receiver and transmitter. The UART implemented with VHDL language can be integrated into the FPGA to achieve compact, stable and reliable data transmission.

2.*Umakanta Nanda, Sushant Kumar Pattnaik "Universal Asynchronous Receiver and Transmitter (UART)" 2016 3rd International Conference on Advanced Computing and Communication Systems (ICACCS -2016), Jan. 22 – 23, 2016, Coimbatore, INDIA* All most all computers and microcontrollers have several serial data ports used to communicate with serial input/output devices such as keyboards and serial printers. By using a modem connected to a serial port serial data can be transmitted to and received from a remote location via telephone line. The serial communication interface, which receives and transmits the serial data is called a UART (Universal Asynchronous Receiver-Transmitter). RxD is the received serial data signal and TxD is transmitted data signal. In this project UART is implemented in virtex II pro FPGA chip due to low cost, high speed, reprogram ability and fast time to market.

III. RESEARCH METHODOLOGY

The "main" process has the primary task of being an interface to the CPU. It uses the internal device clock "clk". On getting a reset (internal/external), the main process initializes the global signals, local variables and ports. Then, at every rising clock edge, it checks for signals from the CPU. Since a mode command (from the CPU) must immediately follow a reset, it stores the next CPU control word in the "mode" global signal. On getting the mode word, it computes the following parameters : (a) number of bits per character (b) number of clock cycles per bit (c) number of stop-bit clock cycles (Async mode) d) number of clock cycles through which RxD has to remain low for Break-Detect in Asynchronous receive. These parameters are assigned to global signals and are used by the "transmitter" and "receiver" processes. Then, the process waits for SYNC character(s), if Internal Synchronization Mode has been programmed. The SYNC character(s) are also assigned to global signals for use by the "transmitter" and "receiver" processes. Then the process 37 waits for a command word from the CPU. (If Internal Synchronization Mode has not been programmed, the process waits for a command immediately after getting a mode word, since no SYNC character(s) are expected.) The command word is assigned to a global signal "command" and various operations (e.g. Internal reset, error flag reset, enter HUNT MODE) are performed, depending on which bits are set in the command word. After this initial phase, other control words sent by the CPU are interpreted as Command words. If the CPU sends data characters for transmission, that character is stored in the global signal "Tx_buffer" (if the transmitter is enabled) and the TxRDY status bit is reset. It also notes whether "CTS_BAR" (clear-to-send input) was low when the character was written, by conditionally setting the global signal "Tx_wr_while_cts". The CPU may want to read a data character that has been received by the USART. In that case, (if the receiver is enabled) the data bits in global signal"Rx_buffer" are placed on the data bus pins and "RxRDY" is reset. The CPU may want to read the status of the USART. In that case, the bits in global signal "status" are placed on the data bus pins and "SYNDET_BD" is reset pin..

. UART Functional Block Diagram



With the ever increasing amount and variety of data to be stored and transmitted in various mediums, the specification of security which has to be established at various levels of medium access and the accompanying issues of authentication and authorization has become a critical factor. Various steganographic, watermarking and data-embedding algorithms have usually manipulated the actual data in order to either hide any coveted information or to

provide some level of access control over the medium. The mediums are usually images, video, audio etc.,wherein specific portions or the overall space is usually ‘corrupted’ with ‘significant’ data. This paper is an attempt to bring out the significance of the steganographic techniques that are employed in information processing algorithms for data security. It deals with the problem of data security, focusing mainly on images, and tries to state the various properties and characteristics that the stenographic algorithms should possess. The paper also highlights the technique of masking used in the conventional steganographic LSB algorithms and in its variants.

With the ever increasing amount and variety of data to be stored and transmitted in various mediums, the specification of security which has to be established at various levels of medium access and the accompanying issues of authentication and authorization has become a critical factor. Various steganographic, watermarking and data-embedding algorithms have usually manipulated the actual data in order to either hide any coveted information or to

provide some level of access control over the medium. The mediums are usually images, video, audio etc., wherein specific portions or the overall space is usually ‘corrupted’ with ‘significant’ data. This paper is an attempt to bring out the significance of the steganographic techniques that are employed in Information processing algorithms for data security. It deals with the problem of data security, focusing mainly on images, and tries to state the various properties and characteristics that the steganographic algorithms should possess. The paper also highlights the technique of masking used in the conventional steganographic LSB algorithms and in its variants.In this project, an efficient GA-DoSLD algorithm is proposed for generating the DoSL attack profiles from multiple sensor nodes such that the attacker nodes can be prevented from the communication process. Initially, a WSN is simulated with 100 numbers of static sensor nodes; then the BS performs the operations such as key pair generation and behavior monitoring in parallel. The base station monitors the behavior of the sensor nodes and initializes every behavior as a chromosome. The MRSA algorithm is implemented in the base station for generating and distributing the key pair among the sensor nodes. Before initiating the communication between the sensor nodes, the AODV routing protocol estimates the optimal route. To validate the trustworthiness of the relay nodes in the route, the fitness value is estimated for every chromosome. If the chromosome is determined as unusual, it is validated against the existing attack profiles If there does not exist a match, the pair of chromosomes is subjected to the crossover and mutation operations. The resultant chromosomes are added to the existing chromosomes. Finally, the BS determines the attacker nodes broadcasting the blocked information to all the sensor nodes in the network. To prove the superiority of the suggested GA-DoSLD algorithm, it is compared against the existing X-MAC, ZKP, and TE$_2$S schemes for the metrics such as normalized energy consumption, effective packet number, end-to-end delay, average energy consumption, packet delivery ratio, and throughput ratio versus packet rate. The validation results prove that, when compared to the existing schemes, the proposed algorithm provides optimal results for all the metrics. The repeated execution of the GA-DoSLD algorithm in the sensor nodes consumes a considerable amount of energy. Thus, to achieve the energy optimization, a different soft computing algorithm other than GA can be used in future for detecting the denial-of-sleep attack in the WSN environment

IV. CONCLUSION

In this paper the design of UART is described along with the individual modules contained in it. Each module is described using Verilog and simulated using Xilinx ISE. This design provides a programmable UART suitable for variety of FPGA based systems reducing the routing problems, cost issues and improving flexibility and integrity.

REFERENCES

[1] Zou, Jie Yang, Jianning. Design and Realization of UART Controller Bas ed on FPGA.

[2] Liakot Ali, Roslina Sidek, Ishak Aris, Aladdin Mohd. Ali, Bambang Sunary, 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.

[5] Charles H. Roth, Jr, “Digital System Design by using VHDL”, PWS Publishing Company, 1998.

[6] Tomasi, Wayne, Advanced Electronic Communication Systems”, Third Edition, PrenticeHall, United States of America, 1994.

[7] Mohd Yamani Idna Idris, Mashkuri Yaacob, “A VHDL Implementation of BIST Technique in UART Design”, Faculty of Computer Science and Information Technology, University of Melaya, 2003C.

[8] Martin S. Michael, “A Comparison of the INS8250, NS16450 and NS16550AF Series of UARTs”, National Semiconductor Application Note 493 April 1989

[9] “PC I6550D Universal Asynchronous Receiver/ Transmitter with FIFOs”, National Semiconductor Application Note, June 1995 M. S. Harvey, “Generic UART Manual" Silicon Valley. December 1999.

[10] FANG Yi-Yuan CHEN Xue-Jun, “Design and Simulation of UART Serial Communication Module Based on VHDL’ Shanghai University of Engineering Science, 2011

[11] Amanpreet Kaur, Amandeep Kaur, “An approach for designing a universal asynchronous receiver transmitter (UART)”, International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com Vol. 2, Issue 3, May-Jun2012,pp.2305-2311

[12] Cowley, John (2007) communication and networking, An introduction, Springer, ISBN 9781846286452

[13] www.pccompi.com

[14] Shiva, S.G University of Alabama in Huntsville, AL, proceedings of the IEEE (volume 67, Issue: 12)

[15] Comparison of VHDL, Verilog and System verilog, by Stephen Bailey, www.model.com C. H. Roth, “Digital System Design Using VHDL", PWS Publishing Company, 2008cited.