The Design and Implementation of a Reconfigurable USART IP Core for Embedded Computing With Support for Networks

Ali H. El-Mousa, Nasser Anssari, Ashraf Al-Suyyagh and Hamzah Al-Zubi

***Abstract*— Embedded systems have drastically grown in importance and complexity in recent years. Many systems are now designed using Field Programmable Gate Array (FPGA) due to its size, flexibility, and resources. This paper presents the design and implementation of a flexible and user reconfigurable Universal Synchronous Asynchronous Receive Transmit (USART) IP core suitable for use in embedded systems and Systems on Chip (SoC). The design scheme employed allows the USART to be used in various modes of operation such as standalone and 9-bit addressable mode for multi-drop networks. It also supports high speed data rates of up to 3 Mb/s. The design utilizes Hardware Description Language (HDL) to describe the operation, ease implementation and allow cross platform utilization. The paper shows through comprehensive testing that the proposed design functions properly while consuming minimum resources from the target FPGA.**

***Index Terms*— Embedded systems, addressable USART, IP core, FPGA, HDL.**

I. INTRODUCTION

Interest in embedded systems has grown drastically in recent years; the world market for embedded software will grow from about $1.6 billion in 2004 to $3.5 billion by 2009, at an average annual growth rate (AAGR) of 16%. On the other hand, embedded hardware growth will be at the aggregate rate of 14.2% to reach $78.7 billion in 2009, while embedded board revenues will increase by an aggregate rate of 10% [1]. At the same time, embedded systems are increasing in complexity and more frequently they are also networked. As designs become more complex, embedded systems based on FPGA are required to interact with software running on stock commercial processors [2]. This interaction more often than not makes use of a serial communications transmission link. Design techniques based on hardware-software co-design are generally implemented on platforms that utilize FPGAs as accelerators together with embedded CPU cores for control and operational procedure definition [3]. Frequently, these platforms also require high speed serial data communication blocks.

USARTs have been around for years and they have become established for easy and simple serial transmission. However, most of these take the form of hardwired specialized ICs which make them unattractive and unsuitable for use in recent embedded systems; especially those utilizing FPGA technology, since they cannot be incorporated within the HDL design. Also, most are designed with limited features and capabilities; for example: limited speed and no capability for use in multi-drop networks. Attempts at the design of an HDL-based USART have been reported in the literature. Many are just HDL implementation of the well known industry standard 16550 UART without additional features [4-6].

This paper presents the design, implementation and verification of a high speed user configurable USART suitable to be used efficiently on platforms that utilize FPGAs. The architectural design allows serial communications in multi-drop networks using 9-bit operational mode using master-slave operation. It also features configurable high speed transmission rates and transmission error detection and recovery. User configuration is accomplished through specialized internal registers. The design is suitable to be used for inter-chip, inter-processor, and inter-system communications among others. The design implements both modes of operation synchronous and asynchronous.

The rest of the paper is organized as follows: Section 2 presents a general description of USART theory of operation and describes the modes of operation. Section 3 provides a detailed description of the specifications of the developed USART. Section 4 discusses the design methodology followed and provides a description and operational features of the various blocks used in the design. Section 5 is concerned with the testing and verification procedures followed. Section 6 is for discussions and conclusions.

II. USART THEORY OF OPERATION

USARTs operate as parallel to serial converters at the transmitter's side where data is transmitted as individual bits in a sequential fashion whereas at the receiver's side, USARTs assemble the bits into complete data and thus act as serial to

Manuscript received March 22, 2008.

Ali H. El-Mousa is with the Computer Engineering Department, Faculty of Engineering & Technology, University of Jordan, Amman, Jordan. (Phone: ++96265355000 ext. 23000, fax: ++96265355588, e-mail: elmousa@ju.edu.jo)

Nasser Anssari and Ashraf Al-Suyyagh are both with the Computer Engineering Department, Faculty of Engineering & Technology, University of Jordan, Amman, Jordan, e-mail: n.anssari@ju.edu.jo, a.suyyagh@ju.edu.jo .

Hamzah Al-Zubi works as a telecom engineer with UtilNet Company in Amman-Jordan, e-mail: hamzah_alzubi@yahoo.com .

parallel converters.

USARTs mandate preliminary configuration of data format, transfer rates and other options specific to the design model. The user can send individual or a block of data, whose size is determined by the design model, to the transmitter section of the USART. Data is stored in the internal buffer and framed according to the transfer mode and user defined options. Finally, the frame is sent to its destination one bit at a time. On the receiver side, the data frame bits are received and sampled. The extracted data from the received frame resides in the internal receiver buffer waiting to be read by the user. The receiver monitors the reception for possible errors and informs the recipient of their existence should the proper control bit(s) be set. Most USARTs offer a status monitoring mechanism via a dedicated status register(s) through which some of the internal operation aspects can be viewed. A simple block diagram of the internal structure of a USART is shown in Fig. 1 below.

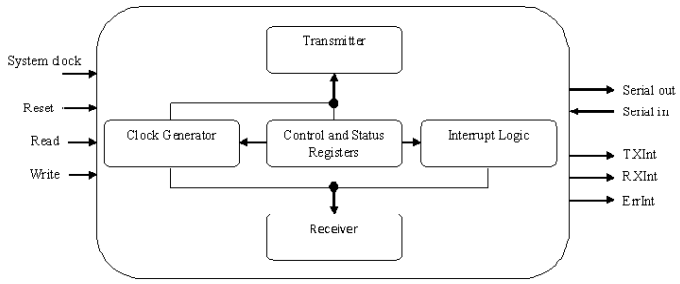

Fig. 1 The USART module

A. Modes of Operation

There are two major modes of operation in USARTs: synchronous and asynchronous modes. The latter prevails in most applications.

In synchronous mode, both the transmitter and receiver are synchronized to the same clock signal which is usually generated by the transmitter and sent along with the data on a separate link to the receiver. The receiver in turn utilizes the received clock in extracting the timing sequence and determining the beginning and end of the received bit interval.

In asynchronous mode of communication, the transmitter and receiver are preconfigured to the required timing parameters in advance and special bits are appended to the data frame for synchronization purposes. Thus timing is embedded in the frame and the need to send a timing signal to the receiver is eliminated.

B. Common Communication Errors Encountered in USARTs

The most common types of errors encountered in USARTs are parity, framing and overrun errors. A parity error indicates that noise affected the data during transmission; such errors occur frequently in hostile environments especially when cables are improperly shielded. A framing error indicates that the start and stop bits are not in their proper places which is mainly due to different baud rate configurations at the transmitter and receiver sides. Finally, the overrun error indicates that data have been lost in the receiver side because the internal reception buffer is full.

III. Specifications of the Developed USART

The specifications included in the design were chosen to meet the modern serial communication demands of high performance and reliability, taking compatibility with legacy devices into consideration.

Performance oriented features include an interrupt driven approach and universal eight bit addressability which make the USART ideal for industrial and control applications. Eight-level buffering allows for data block transfers that is beneficial considering the high speeds the USART can handle in data transfer which can reach 3 Mbs.

Reliability oriented specifications include the programmable odd/even parity bit with the ability to detect parity, framing and overrun errors.

To adapt to modern practices, the USART offers eight bit mode synchronous or asynchronous communication, variable stop bit options and full duplex mode. However to retain compatibility, it also offers five to seven bit transfer and half duplex mode of communication. Table 1 lists the detailed specifications of the proposed USART.

Table 1 Functional specifications of the USART

Specification	Justification
Support for the following transmission modes (Programmable): • Asynchronous (Full/Half duplex modes) • Synchronous (Full/Half duplex modes)	Full duplex mode is employed in modern systems while half duplex mode is retained for compatibility with legacy systems.
Supports a wide range of transmission/reception rates (from 50 Hz to 3MHz)	High frequencies are essential for high speed communication. Lower speeds are needed to communicate with older USARTs. Moreover, lower speeds can be used to minimize cross talk if similar links are adjacent.
Eight-level Transmitter/Receiver Buffer	To account for the high speeds of communication that the USART can reach, blocks of data can be received and buffered until read by the user. Also, this allows the user to issue the transmission of a block of eight-frame size in a single operation. This will also reduce the load on the

	USART operation in the system.
Parity Supported (Programmable –Enable/Disable parity and Odd/Even parity).	Single error detection techniques might prove beneficial in noisy operation environments.
Variable data lengths supported (Programmable - five to eight bits)	Byte communication is the modern norm. Five to seven bits data length is to retain compatibility with legacy systems.
Variable stop bits supported. (Asynchronous mode) (Programmable – One or two stop bits)	This is to comply with the RS232 standard where two stop bits mode is used to accommodate slightly different clocks in the transmitter and receiver sides when USARTs from different vendors are connected together.
Error Detection of the following errors: • Parity Error • Overrun Error • Framing Error (Asynchronous mode)	Parity error detection provides a measure of the reliability of communication. Framing error detection indicates the necessity of reconfiguring the internal clocking sources at both ends correctly. Finally, overrun error informs that data has been lost and the need to frequently read the received data.
Supports Interrupts (Programmable – with ability of Global Masking)	Most modern systems are interrupt-driven for the reason that interrupt techniques save processing clock cycles in comparison with polling techniques and are essential in real time applications.
Supports Addressability (8-bit Universal – Addresses up to 256 devices	Widely used in industrial and control applications in multi-drop networks where a master USART can communicate with a certain other slave USART.

IV. Design Methodology of the USART System

The methodology adopted in carrying out the USART system design was based on systems and software engineering approaches though no single definitive approach was considered in its own regard. Minor variations to systems engineering approaches were considered to account for the working environment under which the project was developed. It used a variation of both the waterfall and incremental approaches suited to the design environment and constraints. The design steps that the system went through are [7]:

1. System Requirements Definition. The requirements definition phase specified the functionality as well as the essential and desirable system properties. This involved the process of understanding and defining what services were required from the system and identifying the constraints on system operation and development.

2. System/Subsystem Design. This phase was concerned with how the system functionality was to be provided by the components of the system and where the system specification was converted into an executable system specification.

3. Subsystems Implementation and Module Testing. The subsystems identified during subsystem design were implemented and mapped into hardware code using the Verilog Hardware Descriptive Language HDL. In this critical stage, individual modules were extensively tested for correct functional operation. Each component, once implemented, was tested independently without the other system components and the assessment of the functional behavior was concluded from the simulation output.

4. System Integration. During system integration, the independently developed subsystems were merged together to build the overall USART system in an incremental approach.

5. System Testing. The overall integrated system was subjected to an extensive set of tests to assure correct functionality, reliability and performance. The tests were aimed to test the behavior of the system as a whole in addition to the interfacing between the subsystems.

Verilog HDL language was used to develop and simulate the USART system and subsystems under Xilinx ISE 8.2i environment [8]. USART modules have been designed and verified separately, before being integrated together.

A. The Baud Rate Generator

The USART system contains a programmable clock generator. Inputs are the system clock and the values of the two divisor registers. Fig. 2 shows a diagram of the clock generator.

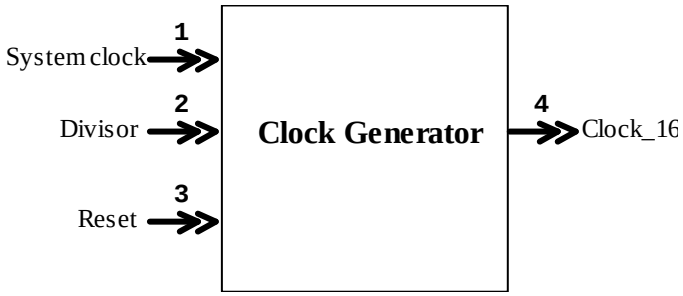

Fig. 2 Clock generator

This module is designed to generate a square clock irrespective of the divisor value (odd or even divisor). In

synchronous mode of communication, this clock is transmitted along with the data. Also, it is used to generate the baud-rate clock, through a division by 16.

B. Interrupt Logic

Interrupts are particularly useful when interfacing I/O devices that provide or require data at relatively low data transfer rates. Unlike polling techniques, interrupt processing allows the system to execute other operations while the USART device is in the process of sending or receiving data thus sparing processor cycles.

Interrupt signals are generated when one of the following actions happens:

1. Transmitter buffer becomes empty after being completely full (one empty slot can declare the buffer as empty).
2. New data frame is received and written on the receiving buffer. This means that if the buffer is full, overrun error will be declared and the received frame will get lost.
3. When one or more of the parity, framing or overrun errors occur(s).

All interrupts can be enabled or disabled individually or collectively by masking their corresponding bits in a control register.

C. The Transmitter Module

The transmitter module is responsible for transmitting the data serially utilizing the criteria set by the user through special control registers. These criteria include the data rate, number of data bits, number of stop bits, 9-bit address mode selection and parity. Fig. 3 shows the different subsystems used for the design of the transmitter section.

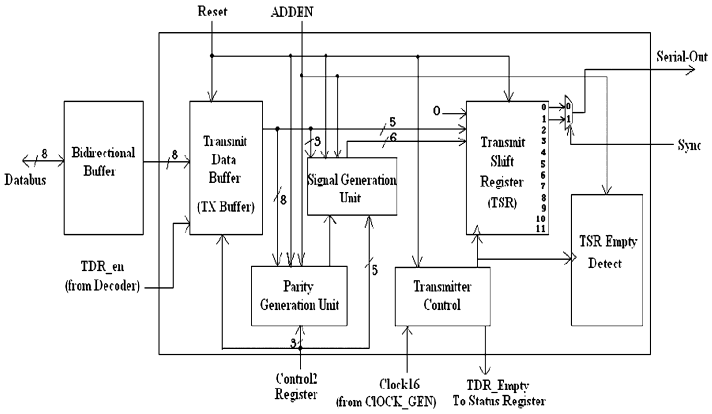

Fig.3 Transmitter subsystems block diagram

Due to space constraints, we show below a sample of the design flow followed in the developing of the different subsystems. As an example, we show the design of the transmit data buffer. Fig. 4 shows the input/output signals associated with this unit, while Fig. 5 shows its dataflow diagram.

D. The Receiver Module

Fig. 6 shows the different subsystems used for the design of the receiver section. Again, we chose a sample to show the internal design details of one of the subsystems. Fig. 7 shows Receiver Buffer Write Logic dataflow diagram.

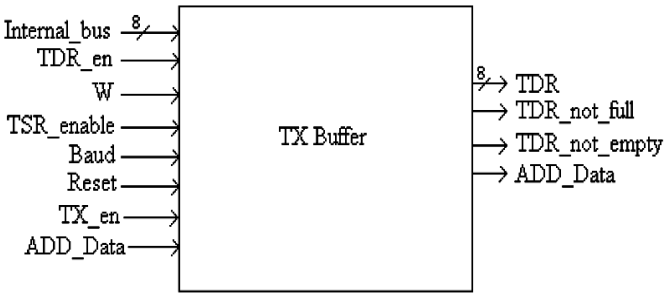

Fig. 4 Transmit data buffer I/O signals

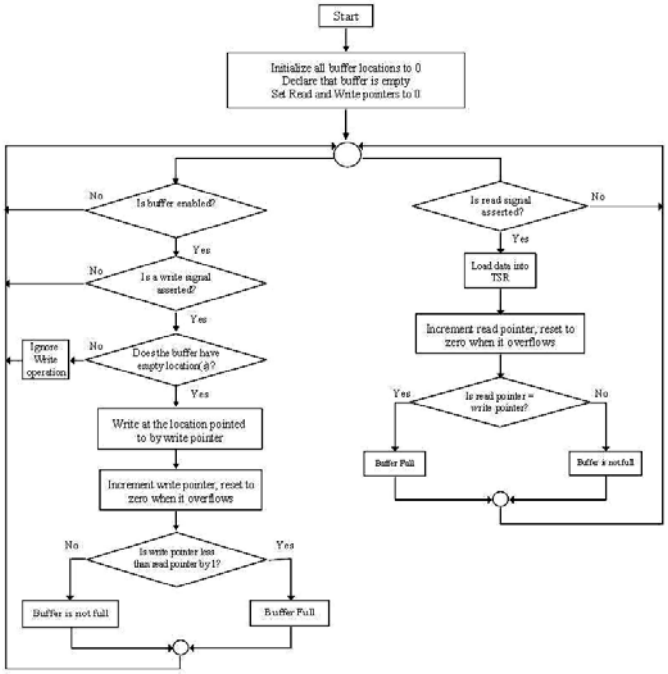

Fig. 5 Transmit buffer dataflow diagram.

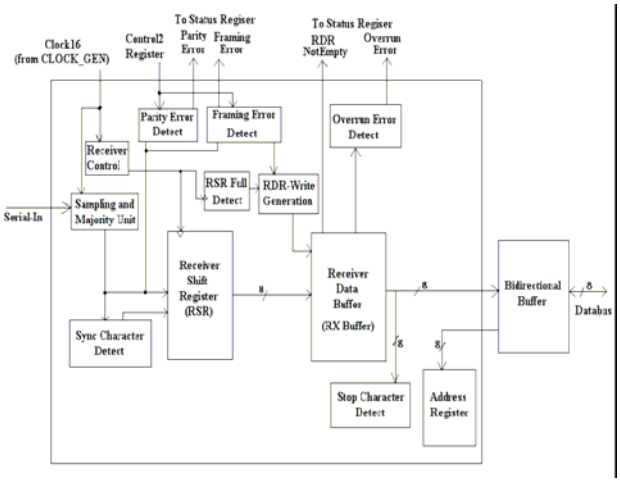

Fig. 6 Receiver subsystems block diagram.

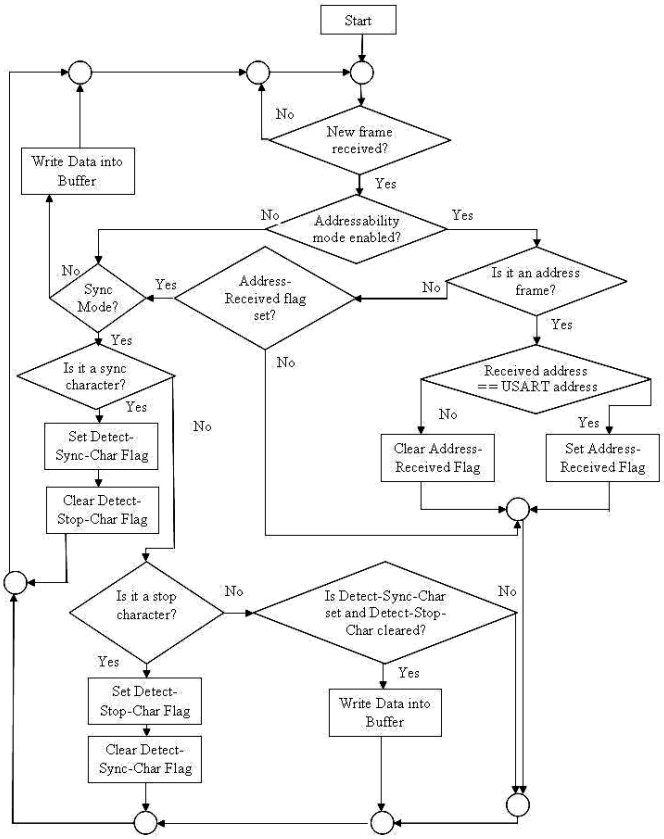

Fig. 7 Receiver buffer write logic dataflow diagram

V. Testing and Verification Procedures

In general, the system testing process has two distinct goals:

1. To demonstrate to the developer and the customer that the system meets its requirements.
2. To discover faults or defects in the system where the behavior of the system is incorrect, undesirable or does not conform to its specification [7].

The first goal leads to validation testing, where the system is expected to perform correctly using a given set of test cases that reflect the system's expected use. The second goal leads to defect testing, where the test cases are designed to expose defects. The test cases can be deliberately obscure and need not reflect how the system is normally used. For validation testing, a successful test is one where the system performs correctly. For defect testing, a successful test is one that exposes a defect that causes the system to perform incorrectly.

In general, the project went through several phases during the testing process as illustrated in Fig. 8:

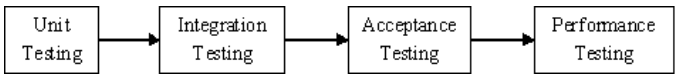

Fig. 8 System testing phases

A. Unit Testing

A unit test consists of a set of test cases used to establish that a subsystem performs a single unit of functionality to some specification. Test cases should be written with the express intent of uncovering undiscovered defects [9]. We employed two methods for testing: simulation and actual hardware-based testing. As an example of the simulation, Fig. 9 shows the results for the simulation of buffer operation in both the transmitter and receiver.

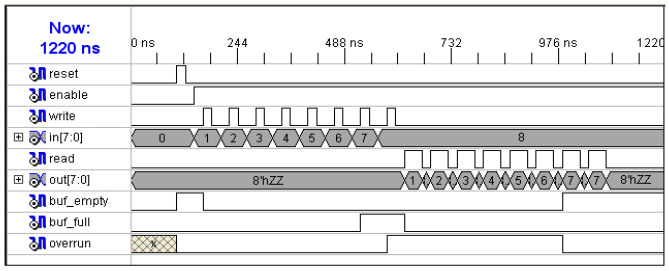

Fig. 9 Simulation of buffer operation.

B. Integration Testing

After the units of a system have been constructed and tested, they must then be integrated into larger subcomponents leading eventually to the construction of the entire system. The intermediate subsystems must be tested to make sure that components operate correctly together. The purpose of integration testing is to identify errors in the interactions of subsystems [9]. As an example, we show the results related to the testing of the transmitter section which illustrates the asynchronous mode of operation with word length of five bits, odd parity, one stop bit, and with the addressability feature disabled in Fig. 10.

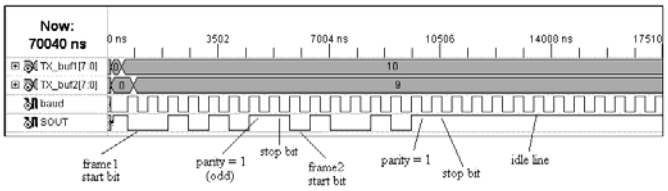

Fig. 10 Simulation of transmitter in asynchronous. mode

Actual testing was done using the available hardware kits and the results are shown in the snapshot taken by a digital storage scope in Fig. 11, which illustrates the serial output at a data rate of 110 bps in asynchronous mode of operation, with word length of eight bits and the addressability feature enabled.

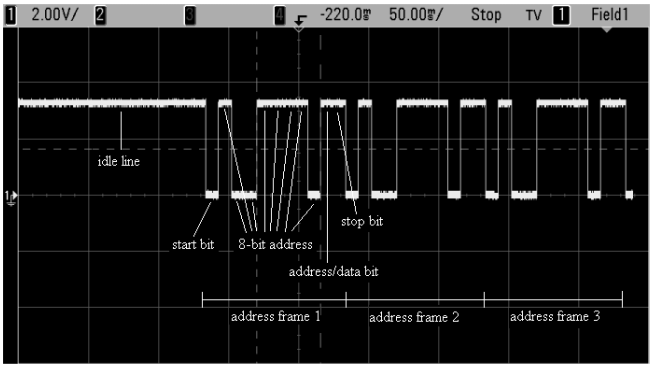

Fig. 11 Asynch. TX module output in 9-bit mode

C. Acceptance Testing

The primary goal of an acceptance test is to verify that a system meets its requirement specification. To demonstrate that the system meets its requirements, it must be shown that it

delivers the specified functionality, performance and dependability, and that it does not fail during normal use.

The entire system was implemented on two connected Xilinx development boards. Both were programmed with the same USART design. However, one was used to transmit data, while the other was used to receive the sent data. Special temporary modifications to the internal design were implemented to allow certain internal signals to be observed with the digital storage scope. From this test, more indications were obtained that the system complies with its requirements specification. Error conditions reflected the true state of the received data when the two USARTs were deliberately configured with conflicting communications options. Moreover, the USARTs functioned as expected in the 9-bit mode of operation.

Figs 12-14 illustrate some snapshots of the communication sessions that were established between the two USARTs. Each snapshot indicates the mode of communication options that were used in the session.

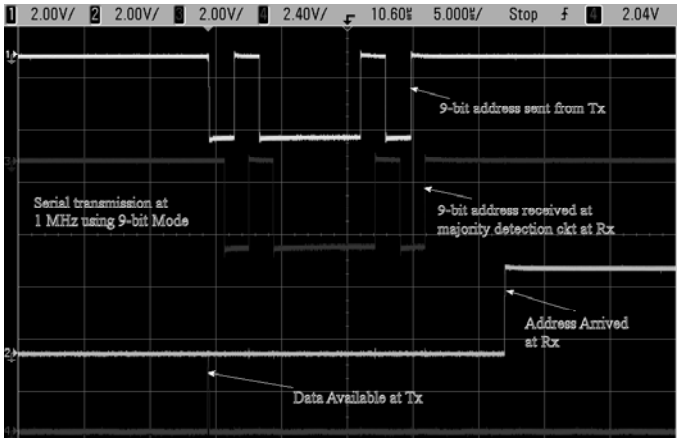

Fig. 12 Asynchronous 9-bit address transmission and detection between two USARTs

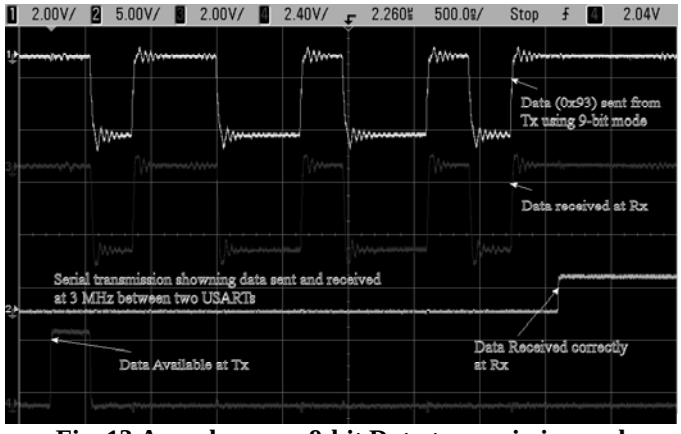

Fig. 13 Asynchronous 9-bit Data transmission and reception between two USARTs

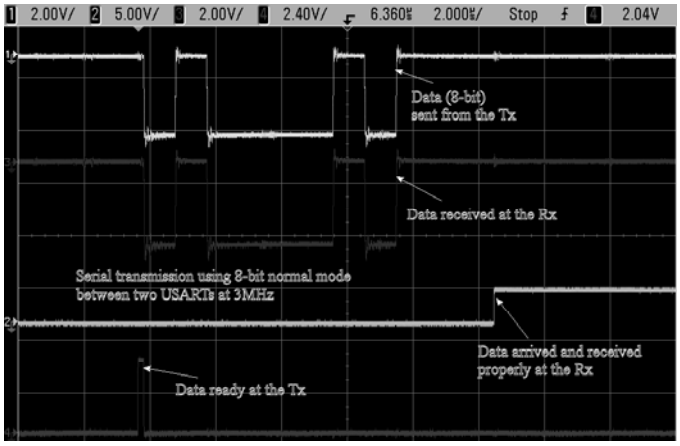

Fig. 14 Asynchronous 8-bit Normal mode data transmission between two USARTs at 3 Mbs

VI. Conclusion

The design and implementation of a reconfigurable USART suitable for use in FPGA based systems and systems on chip (SoC) was presented. It is shown through comprehensive testing that the design performs according to its specifications and can operate at high bit rates reaching 3 Mbps. The design features both synchronous and asynchronous operation and can uniquely operate using 9-bit address detection mode suitable for use in multi-drop networked serial communication devices. The USART is able to detect and recover from common serial communication errors such as overflow and framing errors. It can also detect false start bits. It features a universal 8-bit bus interface and interrupt driven operation. The design is implemented using HDL so it is platform neutral.

References

[1] Ravi Krishnan, "Future of Embedded Systems Technology", BCC Research Report, ID: IFT016B, June 2005.

[2] John A. Stankovic, Insup Lee, Aloysius Mok and Raj Rajkumar, "Opportunities and Obligations for Physical Computing Systems", *IEEE Computer Magazine*, Nov. 2005, pp. 23-31.

[3] Wayne Wolf, *High performance Embedded Computing*, Elsevier, 2007, pp.383-387

[4] Mohd Yamani Idna Idris, Mashkuri Yaacob and Zaidi Razak, "A Vhdl Implementation Of UART Design With BIST Capability", *Malaysian Journal of Computer Science*, Vol. 19 (1), 2006, pp. 73-86.

[5] Azita Mofidian, "DesignWare foundation DW_16550: A fine work of UART", *Designware Technical bulletin*, Technical Forum for Design Automation information, Volume 4 issue 3 Q4/99, http://www.synopsys.com/news/pubs/dwtb/q499/dwtb_art1.html.

[6] Shouqian Yu, Lili Yi, Weihai Chen and Zhaojin Wen, "Implementation of a Multi-channel UART Controller Based on FIFO Technique and FPGA", *Proceedings of 2nd IEEE Conference on Industrial Electronics and Applications ICIEA 2007*, May 2007, pp. 2633 – 2638.

[7] I. Sommerville, *Software Engineering*, 8th Edition. Addison Wesley, 2007.

[8] Xilinx web site, http://www.xilinx.com/ise/logic_design_prod/webpack.htm.

[9] Ford R.M, Coulston C., *Design for Electrical and Computer Engineers*, McGraw Hill, 2005.