IJDACR
ISSN: 2319-4863

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary Research
Website: www.ijdacr.com (Volume 4, Issue 10, May 2016)

An Improved Approach of UART Implementation in VHDL using Status Register

Abhay Malviya
M. Tech. Scholar
Swami Vivekanand College of Engineering, Indore (India)
er.abhaymalviya@gmail.com

Vijay Kumar Sharma
Asst. Professor
Swami Vivekanand College of Engineering, Indore (India)
vijaysharma@svceindore.ac.in

Abstract –**This paper introduces a novel approach of status register along with 8 bit UART, to overcome testability and data integrity. The complete design is implemented in VHDL and simulated throughout in Modelsim, synthesis is carried out using Xilinx ISE 14.5. The IP core is implemented on FPGA device xc4vfx20-10ff672.**

Keywords –**FPGA, Modelsim, UART, VHDL, Xilinx.**

I. Introduction

The FPGA Prototyping System consists of the I/O Processor (IOP) core and the Front-End Subsystem. IOP is a soft core that can be used to handle data communication between the PC and Design-Under-Test (DUT) in the FPGA-based Prototyping Board. The UART module in IOP is a soft core that used to conduct serial I/O communication. A universal asynchronous receiver transmitter (UART) is a type of asynchronous receiver transmitter computer hardware which is used to translate data between parallel and serial interfaces. It is commonly used for serial data telecommunication. A UART converts bytes of data to and from asynchronous start-stop bit streams represented as binary electrical impulses. It is mainly used at broadband modem, base station, cell phone, and PDA designs.

It is crucial that the IOP is functioning correctly and is fault free in real silicon or FPGA board to ensure that the data sends to the DUT inputs are correct and outputs from the DUT send back to the Front End Subsystem is reliable.

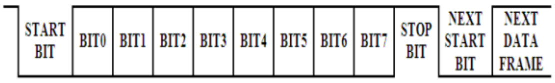
The IOP is migrated to Verilog HDL which contributes several advantages. Verilog HDL allows different levels of abstraction to be mixed in the same models and thus can define a hardware model in terms of switches, gates, RTL, or behavioural code. Besides, most popular logic synthesis tools support Verilog HDL. This makes it the language of choice for many ASIC companies. More important, all fabrication vendors provide Verilog HDL libraries for post logic synthesis simulation. Thus, designing a chip in Verilog HDL allows the widest choice of vendors. On top of that, compared to VHDL, Verilog HDL provides better code efficiency and easier to learn. Nowadays, most IC design companies have migrated HDL design from VHDL to Verilog.

To overcome the testability and data integrity, this research work implemented a novel approach of UART with status register. This paper describes the problems of Very-Large-Scale-Integrated (VLSI) testing followed by the behaviour of UART circuit using VHISC Hardware Description Language (VHDL). In the implementation phase, the UART design is synthesized by means of reconfiguring the existing design to match testability requirements. The UART is targeted at broadband modem, base station, cell phone and PDA designs.

II. Proposed Method

Proposed UART Architecture

UART supports asynchronous communication in which clock information is not shared between transmitter and receiver; several overhead bits are sent along with data bits for synchronization purpose. This indicates that data bits are transmitted in the form of frame. This frame is received at the receiver input where de-framing is done and only the data bits are available in parallel form at the receiver output. The frame format is shown in Figure 1:



START BIT	BIT0	BIT1	BIT2	BIT3	BIT4	BIT5	BIT6	BIT7	STOP BIT	NEXT START BIT	NEXT DATA FRAME

Figure 1: Frame Format for UART [12]

The proposed design of UART, shown in Figure 2, has LCR, Baud Rate Generator (BRG), Transmitter and Receiver as its functional units. All these blocks are explained in brief as course of rest of this section.

Serial data is transmitted via its serial port. A serial port is one of the most universal parts of a computer. It is a connector where serial line is attached and connected to peripheral devices such as mouse, modem, printer and even to another computer. In contrast to parallel communication, these peripheral devices communicate using a serial bit stream. A VHDL Implementation of UART Design with BIST Capability protocol (where data is sent one bit at a time). The serial port is usually connected to UART, an integrated circuit which handles the conversion between serial and parallel data [13] [14].

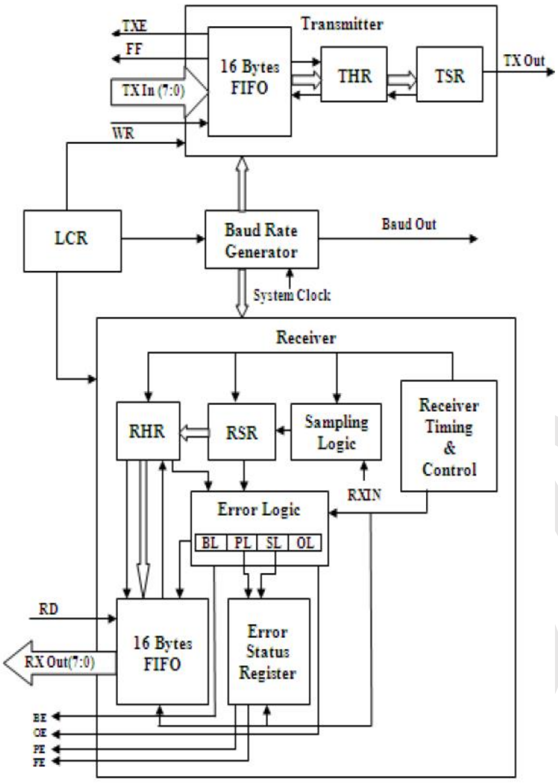


Figure 2: Proposed UART architecture

UART receives a byte of parallel data and converts it to a sequence of voltage to represent 0s and 1s on a single wire (serial). To transfer data on a telephone line, the data must be converted from 0s and 1s to audio tones or sounds (the audio tones are sinusoidal shaped signals). This conversion is performed by a peripheral device called a modem (modulator/demodulator). The modem takes the signal on the single wire and converts it to sounds. At the other end, the modem converts the sound back to voltages, and another UART converts the stream of 0s and 1s back to bytes of parallel data.

UART Transmitter

The transmitter accepts parallel data from peripheral/processor, makes the frame of the data and transmits the data in serial form on the Transmitter Output (TXOUT) terminal (Figure 3). The baud rate generator output will be the clock for UART transmitter.

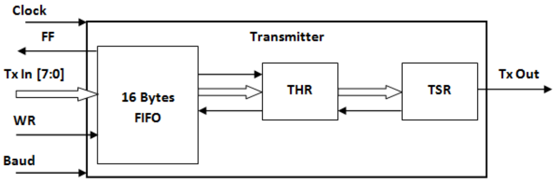


Figure 3: UART transmitter

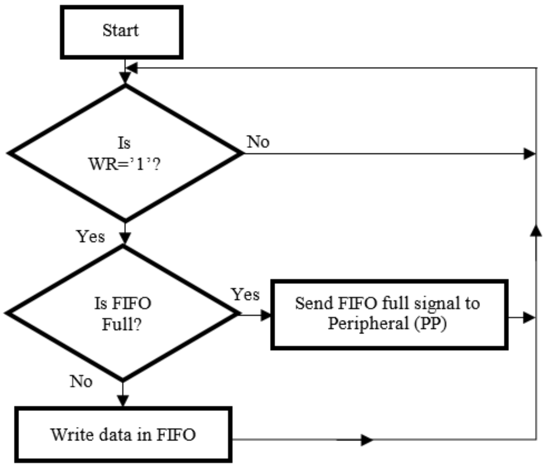


Figure 4: Transmitter flowchart – Input to FIFO

Data is loaded from the parallel inputs TXIN0-TXIN7 into the Transmitter FIFO by applying logic high on the WR (Write) input. FIFO is 16-byte register. If FIFO is full, it sends FIFO Full (FF) signal to peripheral as shown in Figure.

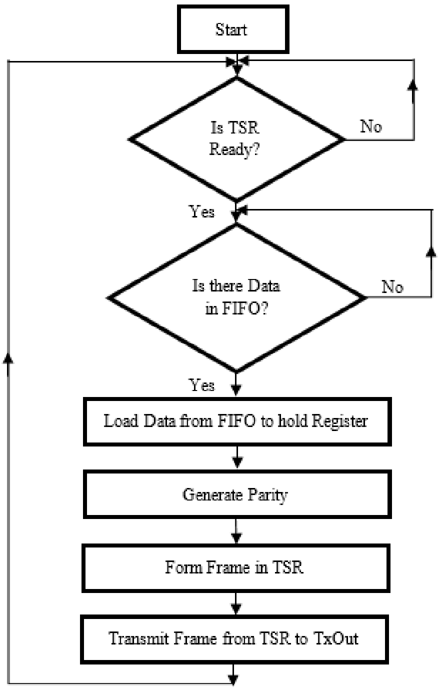


Figure 5: Transmitter flowchart – FIFO to TXOUT

When FIFO contains some data, it will send the signal to Transmitter Hold Register (THR), which is an 8-bitregister. At the same time, if THR is empty it will send the signal to FIFO, which indicates that THR is ready to receive data from FIFO. If Transmitter Shift Register (TSR) is empty, it will send the signal to THR and it indicates that TSR is ready to receive data from THR. TSR is an 11-bit register in which framing process occurs. In frame, start bit, parity bit and one stop bit will be added as shown in Figure 6. Now data is transmitted from TSR to TXOUT serially. Figure 5 is the flowchart explaining transmission of serial data from FIFO to transmitter output.

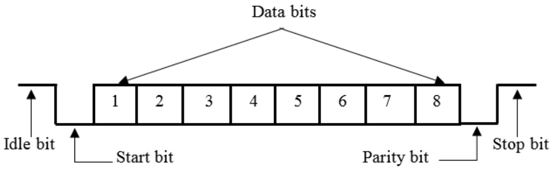


Figure 6: Transmitter flowchart – FIFO to TXOUT

UART Receiver

The received serial data is available on the RXIN pin. The received data is applied to the sampling logic block. The receiver timing and control is used for synchronization of clock signal between transmitter and receiver. The receiver sampling is 16 times to that of the transmitter baud rate. In the architecture of UART receiver (Figure 7), initially the logic line (RxIn) is high.

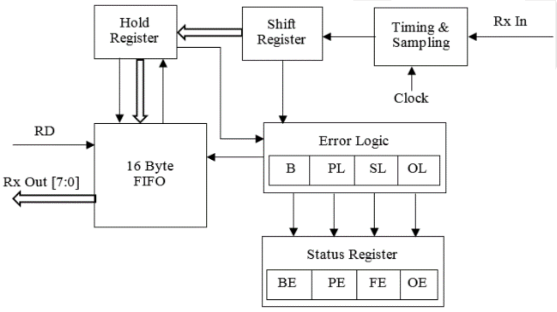


Figure 7: UART Receiver

Whenever it goes low sampling and logic block will take 4 samples of that bit and if all four are same it indicates the start of a frame. After that remaining bits are sampled in the same way and all the bits are send to Receiver Shift Register (RSR) one by one where the entire frame is stored. RSR is a 12 bit shift register. Figure 8 shows the receiver logic.

Now if the Receiver Hold Register (RHR) is empty it sends signal to RSR so that only the data bits from RSR goes to RHR which is an 8 bit register. The remaining bits in the RSR are used by the error logic block. Then, if receiver FIFO is empty it send the signal to RHR so that the data bits goes to FIFO. When RD signal is asserted the data is available in parallel form on the RXOUT0-RXOUT7 pins.

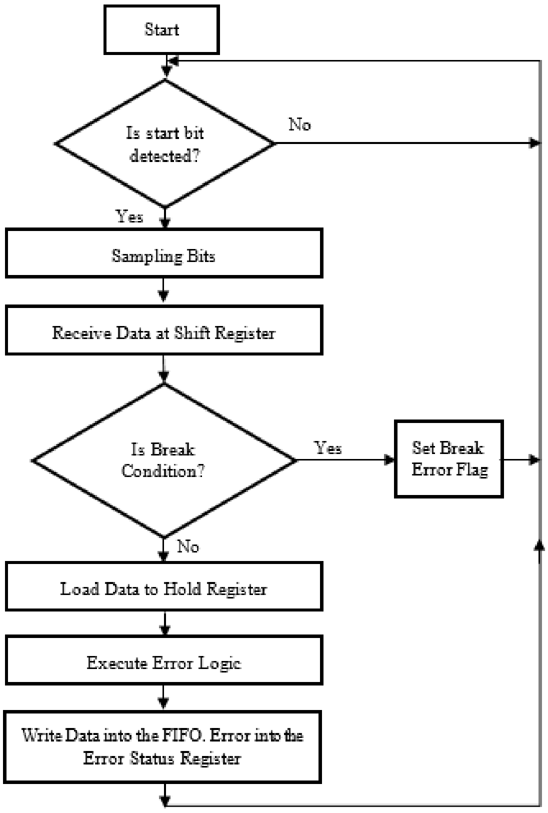

Start
Is start bit detected?
No
Yes
Sampling Bits
Receive Data at Shift Register
Is Break Condition?
Yes
Set Break Error Flag
No
Load Data to Hold Register
Execute Error Logic
Write Data into the FIFO. Error into the Error Status Register

Figure 8: Receiver flowchart (Input to FIFO)

The status register is implemented with flags for error logic operations performed on the received data. The error logic block handles 4 types of errors: Parity error (PE), Frame error (FE), Over run error (OE), Break error (BE). If the received parity does not match with the parity generated from data bits, PE bit will be set which indicates that parity error occurred. If receiver fails to detect correct stop bit or when 4 samples do not match frame error occurs and FE bit is set. If the receiver FIFO is full and other data arrives at, RHR overrun error occurs and OE bit is set. If the RXIN pin is held low for long time than the frame time then there is a break in received data and break error occurs and BE bit is set. Reading of data from receiver is explained by means of flowchart in Figure 9.

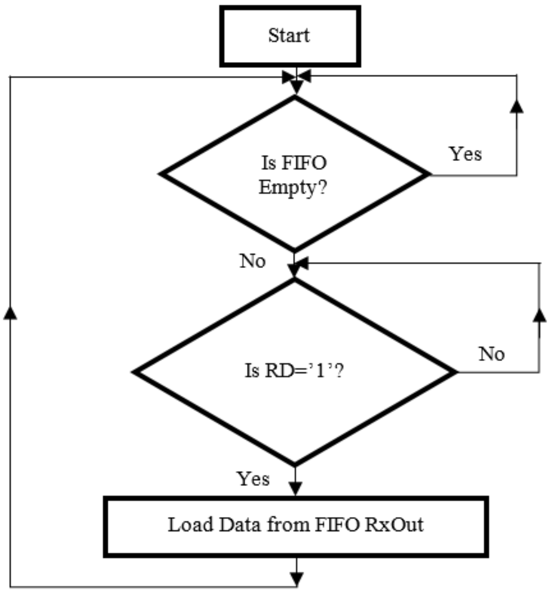


Figure 9: Receiver flowchart (FIFO to Output)

III. Simulation And Results

Proposed architecture is synthesized using Xilinx 13.1 ISE and simulated using Modelsim 6.3.

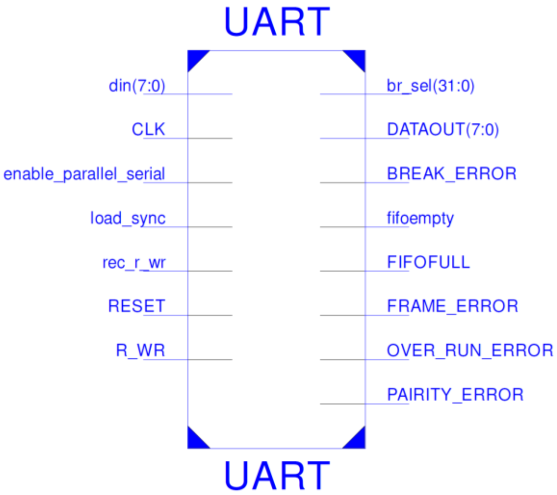


Figure 10: Pin diagram of UART top entity

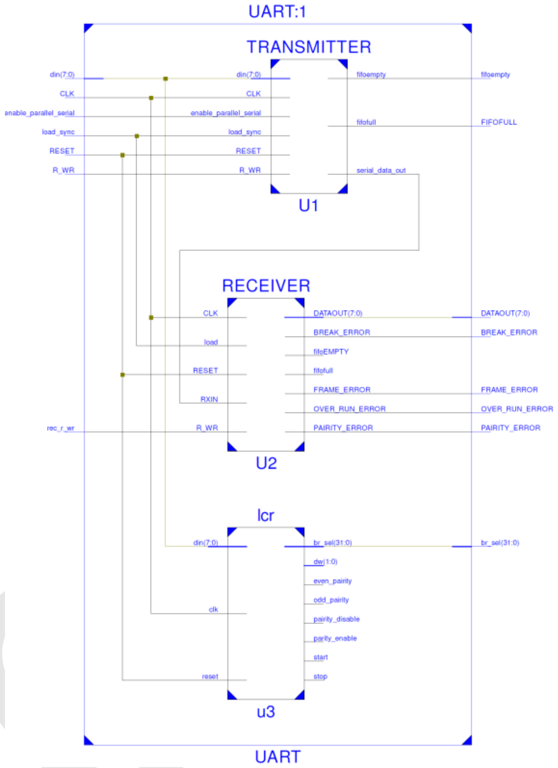


Figure 11: RTL view of UART top entity

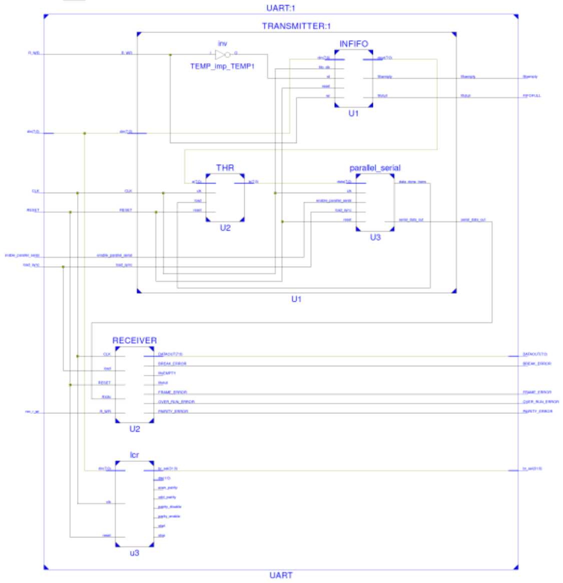


Figure 12: RTL view of transmitter block

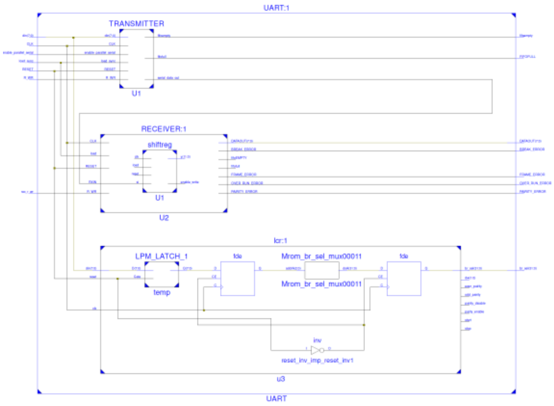


Figure 13: RTL view of receiver block

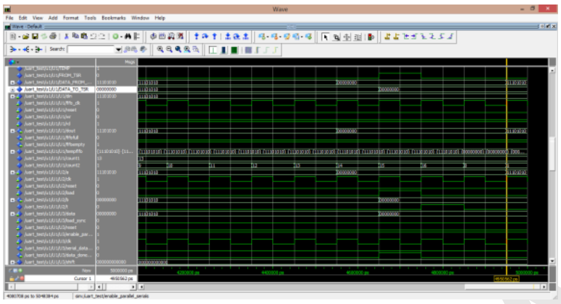


Figure 14: Simulation waveform of transmitter FIFO top

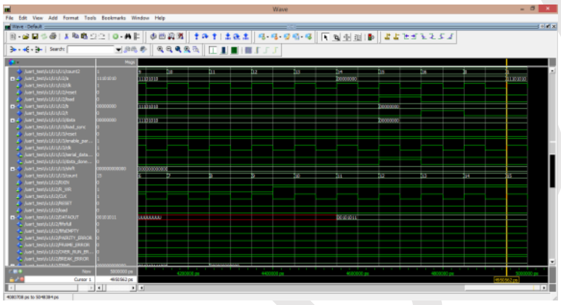


Figure 15: Simulation waveform of Hold and TSR

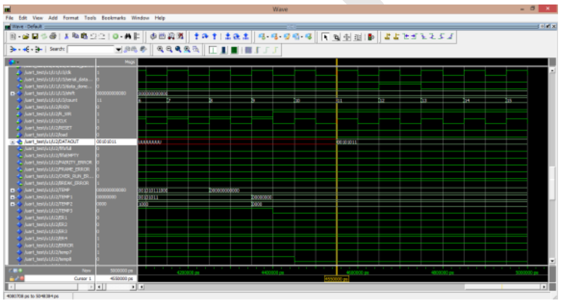


Figure 16: Simulation waveform for receiver shift register (RSR)

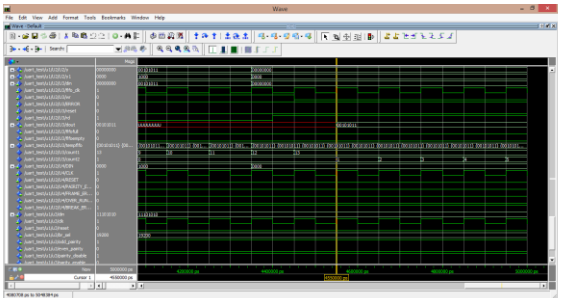


Figure 17: Simulation waveform for Receiver FIFO and Error logic

Table 1: Device utilization summary

Logic Utilization	Previous Method [12]			Proposed Method		
	Used	Available	Utilization	Used	Available	Utilization
Number of Slices	453	3584	12%	314	8544	3%
Number of Slice Flip Flops	467	7168	6%	393	17088	2%
Number of 4 input LUTs	711	7168	9%	516	17088	3%
Number of bonded IOBs	68	141	48%	60	320	18%
Number of GCLKs	6	8	75%	2	32	6%

IV. CONCLUSION

UART supports various data word length, parity selection and different baud rates for serial transmission of data. Working of UART has been tested using Xilinx ISE simulator, which is implemented on FPGA.

Additionally we can detect the different types of errors occurred during communication and hence correct them. Performance comparison of proposed research work with previous work [12] has been shown in Table 1. It was found that the proposed architecture outperforms the previous work [12] on the basis of different parameter values.

REFERENCE

[1] Bibin M C, Premananda B S, “Implementation of UART with BIST Technique in FPGA”, International Journal of Inventive Engineering and Sciences (IJIES), ISSN: 2319–9598, Volume-1, Issue-8, July 2013.

[2] Shumit Saha, Md. Ashikur Rahman, Amit Thakur, “Design and implementation of a BIST embedded high speed RS-422 utilized UART over FPGA”, Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), pp. 1-5, July 2013.

[3] M. Schrader, R. McConnell, "SoC Design and Test Considerations," Proc. Design, Automation and Test in Europe, pp. 202-207, 2003.

[4] M. Abramovici and C. Stroud, "BIST-Based Test and Diagnosis of FPGA Logic Blocks," IEEE Trans. on VLSI Systems, Vol. 9, No. 1, pp. 159-172, 2001.

[5] C. E. Stroud, K. N. Leach, T. A. Slaughter, "BIST for Xilinx 4000 and Spartan Series FPGAs: A Case Study," Proc. IEEE International Test Conf., pp. 1258-1267, 2003.

[6] J. Sunwoo, S. Garimella, C. Stroud, "On Embedded Processor Reconfiguration of Logic BIST for FPGA Cores in SoCs," Proc. IEEE North Atlantic Test Workship, pp. 15-22, 2005.

[7] K. Iijima, A. Akar, C. McDonald, and D. Burek, "Embedded Test Solution as a Breakthrough in Reducing Cost of Test for System on Chips," Proc. IEEE Asian Test Symp., pp 311 – 316, 2002.

[8] C. E. Stroud, A Designer's Guide to Built-In Self-Test, Springer-Verlag, New York, 2002.

[9] Mohd Yamani Idna Idris, Mashkuri Yaacob, Zaidi Razak, "A VHDL Implementation of UART Design with BIST Capability", Malaysian Journal of Computer Science, Vol. 19 (1), 2006.

[10] Shikha Kakar, Balwinder Singh, Arun Khosla, "Implementation of BIST Capability using LFSR Techniques in UART", International Journal of Recent Trends in Engineering ,Vol 1, No. 3, May 2009.

[11] Manjit Kaur, Gurmohan Singh, "Area Overhead and Delay Analysis for Built-In-Self-Test (BIST) Technique", International Journal of Recent Trends in Engineering, Vol 2, No. 7, November 2009.

[12] Patel N, Patel V, Patel V., "VHDL Implementation of UART with Status Register", IEEE, International Conference on Communication Systems and Network Technologies (CSNT), pp. 750-754, May 2011.

[13] S. Wang, "Generation of low power dissipation and high fault coverage patterns for scan-based BIST", in Proceedings of International Test Conference, pp. 834 –843, 2002.

[14] SBS Implementers Forum. System Management Bus (SMBus) Specification, Version 2.0, August 2000.