FPGA Implementation of BIST (Built in Self Test) Enabled UART for Real Time Interface Applications

L. Supriya, J. Lingaiah, G. Kalyan

Abstract— **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. BIST (BUILT IN SELF TEST) is the technique of designing additional hardware and software features into integrated circuits to allow them to perform self testing, i.e., testing of their own operation using their circuits, there by reducing dependence external Automated Test Equipment (ATE).**

Manufacturing process are extremely complex, inducting manufactures to consider testability as a requirement to assure the reliability and the functionality of each of their designed circuits. One of the most popular test techniques is called Built in Self Test. A Universal Asynchronous Receiver /Transmit (UART) with BIST capability has the objectives of testing the UART on chip itself and no external devices are required to perform the test. Here VHDL Implementation of UART with embedded BIST capability using FPGA technology. The architecture of UART with BIST which tests the UART for its correctability.

In this project, the protocol of BIST Enabled UART is studied. The BIST Enabled UART architecture will be designed. Various blocks of BIST Enabled UART are modeled in VHDL. The design is functionally verified by simulating the code in ModelSim from Mentor Graphics. The FPGA synthesis is done using Xilinx ISE tool.

***Index Terms*—BIST, UART, VHDL, protocol, ATE.**

I. INTRODUCTION

As integrated circuits are produced with greater and greater levels of circuit density, efficient testing schemes that guarantee very high fault coverage while minimizing test costs and chip area overhead have become essential. As the complexity of circuits continues to increase, high fault coverage of several types of fault models becomes more difficult to achieve with traditional testing paradigms.

Integrated circuits are presently tested using a number of structured design for testability (DFT) techniques. These techniques rest on the general concept of making all or some state variables directly controllable and observable.

II. BIST

Built-In Self Test is a technique of integrating the functionality of an automatic test system onto a chip. It is a Design for Test technique in which testing (test generation and test application) is accomplished through built in hardware features. The general BIST architecture has a BIST test controller which controls the BIST circuit, test generator which generates the test address sequence, response verification as a comparator which compares the memory output response with the expected correct data and a circuit under test (CUT).

BIST Controller is a finite state machine, whose state transition is controlled by the Test Mode (TM) input. It provides the clock signal to the test pattern generator (LFSR), Circuit Under Test (CUT) and the signature generation circuit (MISR). The BIST controller also decides the input to the circuit under test based on whether the module is in normal mode or test mode on seeing the Test Mode (TM) input.

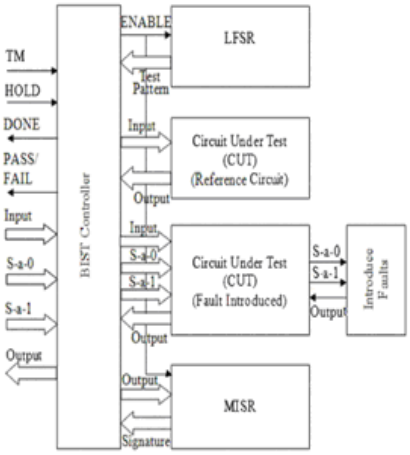

Fig.1 Block Diagram of Bist controller

A BIST circuit comprises a scan monitor with hold logic and a signature generation element. The hold logic is operable to suspend signature generation in the signature generation element at any desired point in the test sequence. In some embodiments, the hold logic comprises a scan-Loadable signature hold flip-flop which allows the logic BIST controller to be restarted from any selected pattern within a pattern range and to run to any subsequent pattern. The BIST session can be run incrementally, testing and reporting intermediate MISR signatures.

ISSN: 2278 – 7798

III. UART

The UART block diagram is shown in Figure 2. It consists of 5 blocks namely transmitter, receiver, register block, parity generator and clock divider.

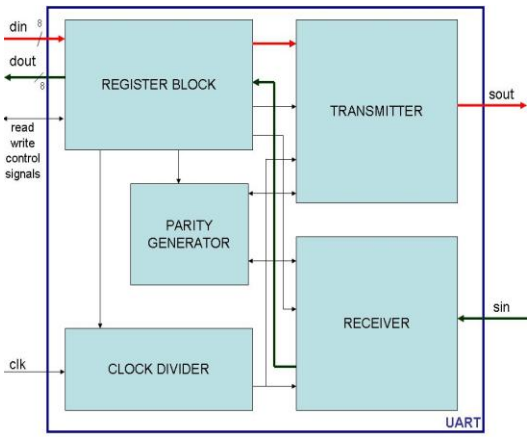

Fig.2 Block diagram of UART

The UART has serial interface to the RS232 driver. The operation of UART is controlled by an external host processor. There is an 8-bit data interface to host along with read and write control signals. Clock is fed from external crystal.

Transmitter:- The transmitter block is responsible for the transmission of serial data from UART. It takes 8-bit data from the transmit register and sends it to output in serial form. The data is accompanied by start and stop bits. An optional parity bit also may be used for error detection.

The transmitter stays in IDLE state unless transmit enable (tx_enable) is made as '1'. The data transmission starts with tx_enable = 1. As mandated by the protocol, a '0' is transmitted to indicate start of transmission or start bit. This is done in START state. Then data bits 0 to 7 are transmitted in states DATA0 to DATA7. If parity is enabled in configuration register, the data is attached with parity in PARITY state. Then transmitter enters STOP state and sends a '1'. This indicates the completion of transmission. Then the transmitter enters the IDLE state and waits for next data transmission.

Receiver: - UART receiver handles the reception of data from RS232 port. The main functions of receiver block are to convert the serial data to parallel data, check the correctness of data from parity and store the received data in receive register in register block.

The receiver is in IDLE state by default. When the serial data pin goes low, indicating the start bit, the state machine enters DATA0 state. The data is received; one bit at a time from LSB to MSB in states DATA0 to DATA7. If parity is enabled, the state machine checks the parity bit received against the parity obtained from received data. If the data received is fine, the data_rx (data_rx_done) bit is set to '1' and the receiver goes back to IDLE state again.

Register block: - the register block contains 4 registers namely transmit register, receive register, configuration register and clock division register. The bit assignments are shown in Fig. 3-4. These register are programmed by external host processor.

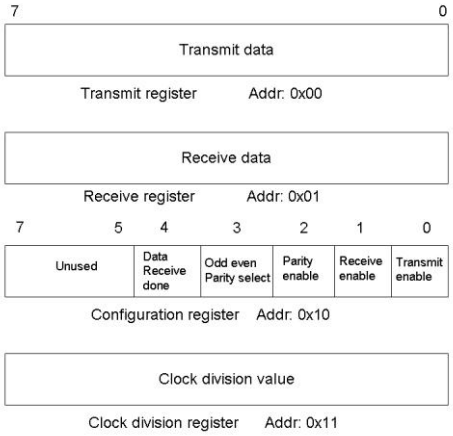

Fig.3 UART registers

Parity generator: - Parity generator generates even or odd parity for the input data bits. By default, even parity is generated. Odd parity needs to be enabled by setting bit (3) of configuration register to '1'.

Clock divider: - The clock divider block is useful in setting the required baud rate in UART. Based on value of input clock frequency and required baud rate, appropriate value of clock division is used.
Clock division value = (Input clock frequency) / (Required baud rate).

IV. IMPLEMENTATION

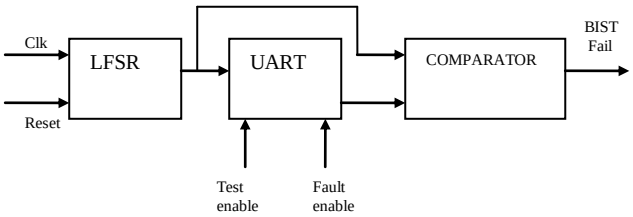

Fig.4 BIST enabled UART block diagram

The figure 4 shows the block diagram of UART with BIST capability. The above design results in effective working UART, because whenever any module of UART fails working, BIST will automatically generates a flag indicating that UART is misbehaving.

A linear feedback shift register (LFSR) is a shift register whose input bit is a linear function of its previous state. The most commonly used linear function of single bits is XOR. Thus, an LFSR is most often a shift register whose input bit is driven by the exclusive-or (XOR) of some bits of the overall shift register value.

The initial value of the LFSR is called the seed, and because the operation of the register is deterministic, the stream of values produced by the register is completely determined by its current (or previous) state. Likewise, because the register has a finite number of possible states, it must eventually enter a repeating cycle. However, an LFSR with a well-chosen feedback function can produce a sequence of bits which appears random and which has a very long cycle.

Applications of LFSRs include generating pseudo-random numbers, pseudo-noise sequences, fast digital counters, and whitening sequences. Both hardware and software implementations of LFSRs are common.

LFSR (Linear feedback shift register) generates 8 bit random data. This random data is fed to UART and it is used only in case of test enable is high which means circuit is in testing mode. The data received from LFSR is fed to receiver part of the UART. The receiver takes 8 bits of parallel data and gives serial data. This serial data is fed to transmitter which converts serial data to parallel again. This 8 bit parallel data coming from transmitter is given to comparator. Comparator one input is coming form transmitter output and the other input is fed from LFSR. If the data coming from transmitter is same as LFSR data, then the UART is working properly and it is indicated key BIST fail flag as '0'. In case the data is not same, then UART is misbehaving then, BIST fail becomes '1'.

The UART developed here has the capability of working with the testing mode. That means, at any time if circuit misbehaves then BIST fail flag indicates, the same in this design, test enable is meant for only proving all the test cases for the design. In real time use, this will not be there. Whenever test enable is low, the UART is going into normal working mode.

V. SIMULATION RESULTS

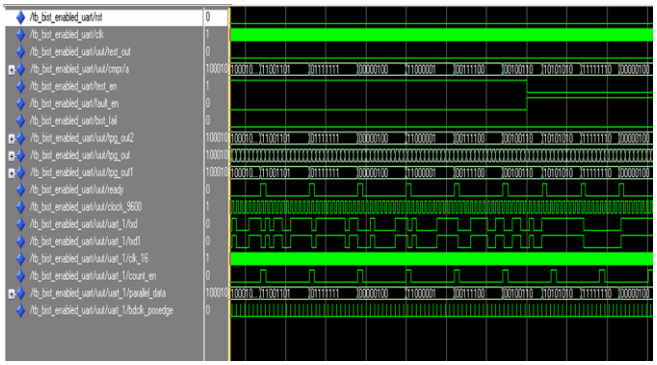

Fig. 5 Simulation results of BIST enabled UART

VI. CONCLUSION

In this paper, the architecture of BIST enabled UART was designed and various blocks of bist and uart are modeled in VHDL. The design is functionally verified by simulating the code in ModelSim from Mentor Graphics. The FPGA synthesis is done using Xilinx ISE tool and the design is implemented on SPARTAN 3E FPGA, which describes the characteristics and the architecture of the designed UART with embedded BIST.

REFERENCES

[1] Naresh, Vatsalkumar and Vikaskumar Patel, "VHDL Implementation of UART with Status Register", in the proceedings of International Conference on Communication Systems and Network Technologies, IEEE Computer Society, 11-13th May 2012, DOI: 10.1109/CSNT.2012.164, pp.750-754.

[2] Fang Yi-yuan and Chen Xue-jun, "Design and Simulation of UART Serial Communication Module Based on VHDL", in the proceedings of 3rd International Workshop on Intelligent Systems and Applications (ISA), IEEE, May 2011, DOI: 10.1109/ISA.2011.5873448, pp.1-4.

[3] Mohd Yamani Idna Idris, Mashkuri Yaacob and Zaidi Razak, "A VHDL Implementation of UART Design with BIST Capability", in the proceedings of Malaysian Journal of Computer Science, June 2006, Vol. 19(1), pp. 73-86.

[4] Dr. Garima Bandhawarkar Wakhle, Iti Aggarwal and Shweta Gaba, "Synthesis and Implementation of UART using VHDL Codes", in the Proceedings of International Symposium on Computer, Consumer and Control, IEEE June 2012, DOI: 10.1109/IS3C.2012.10.

[5] Norhuzaimin J and Maimun H.H, "The design of high speed UART", in the proceedings of Asia-Pacific Conference on Applied Electromagnetics, APACE 05, IEEE, 20-21st Dec. 2005, DOI: 10.1109/APACE.2005.1607831, pp.5-8.

[6] Dr. T.V.S.P.Gupta, Y. Kumari and M. Ashok Kumar, "UART Realization with BIST architecture using VHDL", in the proceedings of International Journal of Engineering Research and Applications, February 2013, Vol. 3, Issue 1, ISSN: 2248-9622, pp.636-640.

L. SUPRIYA is presently pursuing final semester M. Tech in VLSI at Arjun College of Technology and Sciences, Batasingaram village, Hayathnagar, R.R. Dist, Hyderabad, Telangana, India.

J. LINGAIAH is presently working as Head of Department & Associate Professor in the department of Electronics and Communication Engineering in Arjun College of Technology and Sciences, Batasingaram Village, Hayathnagar, R.R. Dist, Telangana, India.

G. KALYAN is presently working as Application Engineer in Unistring Tech Solutions Pvt Ltd, Hyderabad, Telangana, India.

ISSN: 2278 – 7798