

# ASIC Design and Implementation of UART with DFT logic for Built-in Self-test using Verilog HDL

Author: Kistipati Karthik Reddy<sup>1</sup>; Jeeru Dinesh Reddy<sup>2</sup>

Affiliation: M.Tech. student<sup>1</sup>; Asst. professor<sup>2</sup>

E-mail: karthikreddykisti@yahoo.in<sup>1</sup>

## ABSTRACT

Asynchronous serial communication is usually implemented by Universal Asynchronous Receiver/Transmitter abbreviated UART, mostly used for short distance, low speed, low cost data exchange between processor and peripherals. UART allows full duplex serial communication link, and is used in data communication and control system. There is a need for realizing the functionality of UART in a single or a very few chips. Further, design systems without full testability are open to the increased possibility of product failures and missed market opportunities. Also, there is a need to ensure the data transfer is error proof. In this paper, introduction of Built-in self-test (BIST) and status register for UART to overcome the above two constraints of testability and data integrity. The 8-bit UART with status register and BIST module is coded in Verilog HDL, synthesized and simulated using Xilinx ISE design suite 14.2. The results indicate that this model eliminates the need for higher end, expensive testers and thereby it can reduce the development cost and valuable time.

**Keywords:** Built in self-test(BIST), DFT, System on Chip(SOC).

## 1. INTRODUCTION

A Universal Asynchronous Receiver/Transmitter, UART is a piece of computer hardware that translates data between parallel and serial forms. The Universal term states that the data format and transmission speeds are changeable. It performs temporal-to-spatial conversion on data character received from host processor into serial data stream and spatial-to-temporal conversion on serial bits

received from serial device to the host processor. UART's are commonly used in conjunction with communication standards such as RS-232, RS-422 or RS-485. A complete data frame format for UART consists of a start bit '0', 5-8 bits of data, optional parity bit and stop bit '1'. The stop bit can be in length of 1, 1.5 or 2 bits[1].

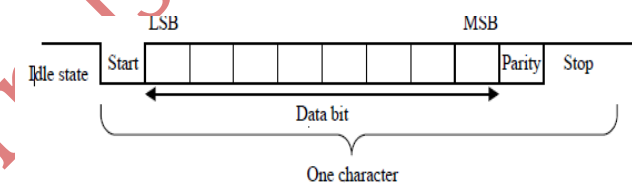


Fig.1 showing 8-bit data word

A start bit logic low(0) at the beginning of the data frame will cause a falling edge on the serial data line. This marks the detection of a data character. The idea of start bit and stop bit in UART is to achieve data synchronization. An optional parity bit can be in odd parity or even parity. Odd parity means that sum of all bits gives an odd number, while even parity means sum of all bits gives an even number. The Start Bit is used to alert the peripheral 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. After the Start Bit, the individual data bits of the word are sent, with the Least Significant Bit (LSB) being sent first. Each bit is transmitted for exactly the same amount of time as all of the other bits, and the receiver samples at the wire at approximately halfway through the period assigned to each bit to determine if the bit is a 1 or a 0. When the entire data word has been sent, the transmitter adds 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. Designers of digital ASICs use a hardware description language (HDL), such as Verilog or VHDL, to describe the functionality of ASIC.

## 2. LITERATURE

Manufacturing processes are extremely complex, inducing manufacturers to consider testability as a requirement to assure the reliability and the functionality of each of their designed circuits. Testing of integrated circuits (ICs) is important to ensure a high level of quality in product functionality in both commercially and privately produced products. In the modern System-on-a-Chip (SOC) design, many cores are integrated into a single chip. Some of them are embedded, and cannot be accessed directly from the outside of the chip. Such SOC[3] designs make the test of these embedded cores a great challenge. As ICs grow in gate counts, it is no longer true that most gate nodes are directly accessible by one of the pins on the package. This makes testing of internal nodes more difficult as they could neither no longer be easily controlled by signal from an input pin (controllability) nor easily observed at an output pin (observability). Hence internal diagnostic capabilities are to be introduced to test the embedded cores. Due to the high demand for speed and performance, most of the functions that were implemented in the software are shifting towards the hardware. This means functions like the communication protocols, DSP algorithms, audio and video compression algorithms and many other functions find it most advantageous to be implemented in the hardware for meeting the performance. Another factor that leads to this situation is miniaturization. Due to the fast changes happening in the semiconductor technology the ability to pack more circuits in a small area has increased many fold [4]. VLSI testing problems like Test generation problems, input combinatorial problems, gate to I/O pin ratio problems are discussed and this motivated designers to identify reliable test methods in solving these difficulties. An insertion of special test circuitry on the VLSI circuit that allows efficient test coverage is the answer to the matter. This has been addressed by the need for design for testability (DFT) and hence the need for BIST. This is to specify test as one of the system functions and thus becomes self-test. BIST is an on-chip test logic that is utilized to test the functional logic of a chip, by itself. With the

rapid increase in the design complexity, BIST has become a major design consideration in DFT methods and is becoming increasingly important in today's state of the art SOCs[5].

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 (functionally, parametrically, or both) using their own circuits, thereby reducing dependence on an external automated test equipment (ATE). A properly designed BIST is able to offset the cost of added test hardware while at the same time ensuring the reliability, testability and reduces maintenance cost. BIST is a Design-for-Testability (DFT) technique, because it makes the electrical testing of a chip easier, faster, more efficient, and less costly. The concept of BIST is applicable to just about any kind of circuit, so its implementation can vary as widely as the product diversity that it caters to. As an example, a common BIST approach for DRAM's includes the incorporation onto the chip of additional circuits for pattern generation, timing, mode selection, and go-/no-go diagnostic tests.

The main drivers for the widespread development of BIST techniques are the fast-rising costs of ATE testing and the growing complexity of integrated circuits. It is now common to see complex devices that have functionally diverse blocks built on different technologies inside them. Such complex devices require high-end mixed-signal testers that possess special digital and analog testing capabilities. BIST can be used to perform these special tests with additional on-chip test circuits, eliminating the need to acquire such high-end testers.

By detecting a malfunctioning chip at an earlier level, the manufacturing cost may be kept low. For instance, the approximate cost to a company of detecting a fault at different levels[6] is as:

WAFER	\$0.01-\$0.1
PACKAGED CHIP	\$0.1-\$1
BOARD	\$1-\$10
SYSTEM	\$10-\$100
FIELD	\$100-\$1000

BIST is also the solution to the testing of critical circuits that have no direct connections to external

pins, such as embedded memories used internally by the devices. In the near future, even the most advanced tester may no longer be adequate for the fastest chip, a situation wherein self-testing may be the best solution for. BIST is fast becoming an alternative solution to the rising costs of external electrical testing and increasing complexity of devices. This approach will find greater deployment in a wider variety of circumstances as more and better BIST techniques are developed. This does not mean, however, that BIST will eventually replace external electrical testing altogether. Still, BIST proponents are optimistic that BIST will someday be the preferred mode of testing, instead of being merely an alternative to external ATE[7] testing as it is today.

Generic BIST architecture components are:

**Circuit under Test (CUT):** This is the portion of the circuit tested in BIST mode. It is delimited by their Primary Input (PI) and Primary Output (PO).

**Test Pattern Generator (TPG):** It generates the test patterns for the CUT. The patterns may be generated in pseudorandom or deterministically. Normally, the pattern generator generates exhaustive input test patterns to the CUT to ensure the high fault coverage.

**Test Response Analysis (TRA):** It analyses the value sequence on PO and compares it with the expected output, based on the value compacted signature is also stored for future reference.

**BIST Controller Unit (BCU):** It controls the test execution; it manages the TPG, TRA and reconfigures the CUT and the multiplexer. During BIST mode, it selects input from the pattern generator to CUT while during functional mode, selects primary inputs.

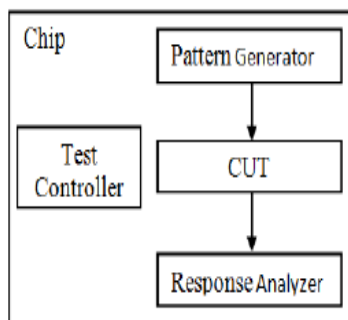


Fig.2 showing BIST generic module

### 3. PROPOSED SYSTEM

#### 3.1 BIST block with UART system

The proposed model has two major modules viz. UART and BIST. Further in the UART, we have transmitter, receiver, and baud rate generator. Baud rate generator works at 50 MHz and further reduced as required for the operations in transmitter and receiver to achieve baud rate of 9600 bps. BIST has a control register, pattern generator and a comparator

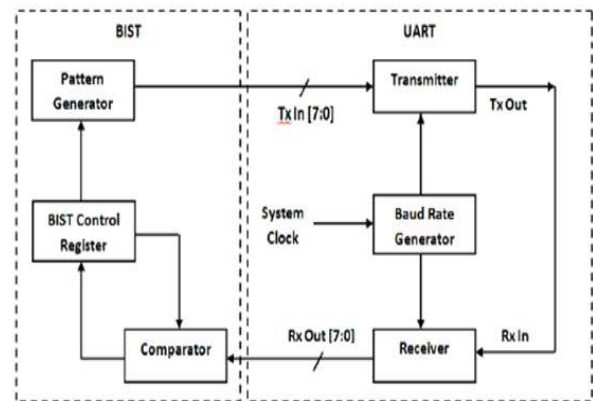


Fig.3 showing proposed system block

#### 3.2 Operation of BIST

LFSR is used to generate pseudo-random test pattern for the BIST. A LFSR is a shift register where the input is a linear function of two or more bits (taps). It consists of D flip-flops and linear exclusive-OR (XOR) gates.

The bits contained in selected positions in the shift register are combined in some sort of function and the result is fed back into the register's input bit. The selected bit values are collected before the register is clocked and the result of the feedback function is inserted into the shift register during the shift, filling the position that is emptied as a result of the shift. The bit positions selected for use in the feedback function are called "taps". The largest state space possible for such an LFSR will be  $2^n - 1$ , all possible values except the zero state. All zero is not allowed in LFSR, as it will always produce 0 in spite of how many clock iteration. Because each state can have only once succeeding state, an LFSR with a maximal length tap sequence will pass through every non-zero state once and only once before repeating a state.

For BIST, UART is set in an internal loop back mode to test both the transmitter and receiver of the UART. This will loop-back the serial data and transmit the data back to the receiver. For the BIST, the test pattern is generated by LFSR as mentioned in the last section and the pattern is loaded to the FIFO of the UART transmitter. Each test byte is then padded with start, parity and stop bits and sent from transmitter and is looped back to receiver. The receiver will extract the data from frames received and loads to receiver FIFO.

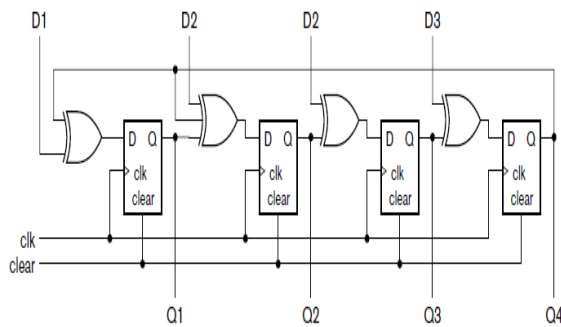


Fig.4 showing LFSR logic for pattern generation

### 3.3 HDL Synthesis Report

# Counters	: 4
4-bit up counter	: 4
# Registers	: 32
1-bit register	: 30
8-bit register	: 2
# Comparators	: 6
4-bit comparator great equal	: 2
4-bit comparator greater	: 1
4-bit comparator less	: 1
4-bit comparator less equal	: 2
# Tristates	: 1
8-bit tristate buffer	: 1
# Xors	: 2
1-bit xor2	: 2
# IOs	: 27

### Cell Usage :

# BELS : 61	
# INV	: 4
# LUT2	: 12
# LUT3	: 16
# LUT4	: 19
# LUT4_D	: 2
# LUT4_L	: 7
# VCC	: 1
# Latches	: 62
# FDC	: 9
# FDC_1	: 16
# FDCE	: 29
# FDP	: 4
# FDP_1	: 2
# FDPE	: 1
# FDPE_1	: 1
# Clock Buffers : 3	
# BUFG	: 2
# BUFGP	: 1
# IO Buffers : 26	
# IBUF	: 12
# OBUF	: 6
# OBUFT	: 8

### Timing Summary:

Speed Grade: -4

Minimum period: 10.572ns (Maximum Frequency: 94.589MHz)

Minimum input arrival time before clock: 1.973ns

Maximum output required time after clock: 4.394ns

Maximum combinational path delay: 5.310ns

Total REAL time to Xst completion: 4.00 secs

Total CPU time to Xst completion: 4.47 secs

### 3.4 Schematic

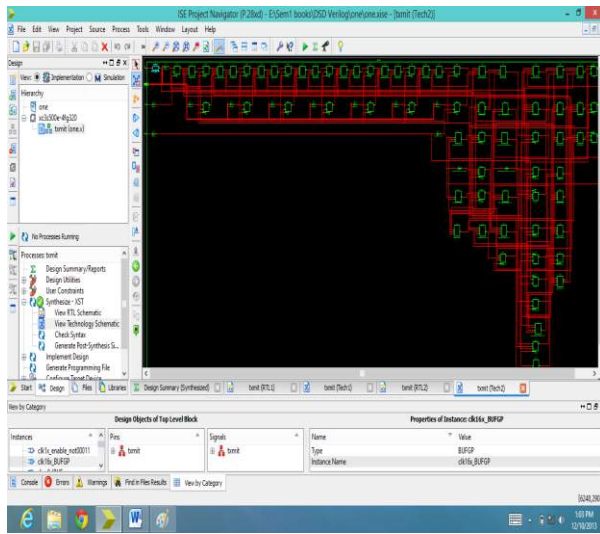


Fig.5 showing Technology schematic of transmitter

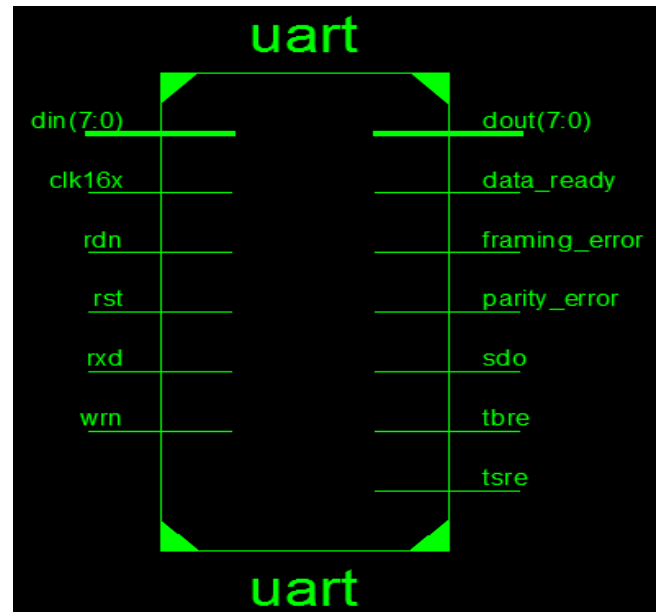


Fig.7 showing LFSR schematic

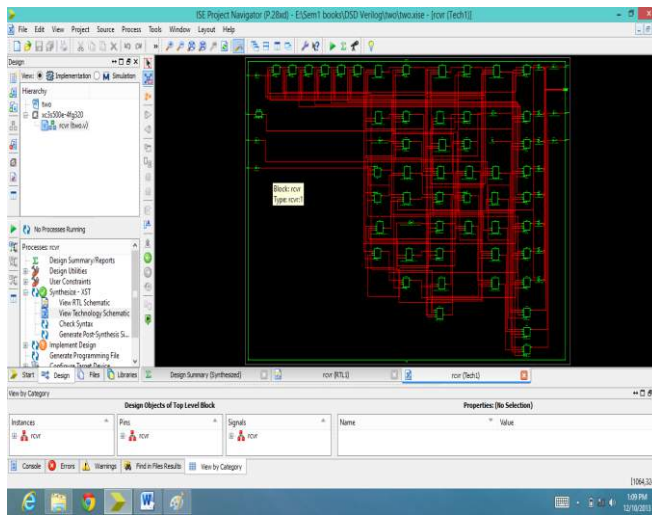


Fig.6 showing schematic of receiver

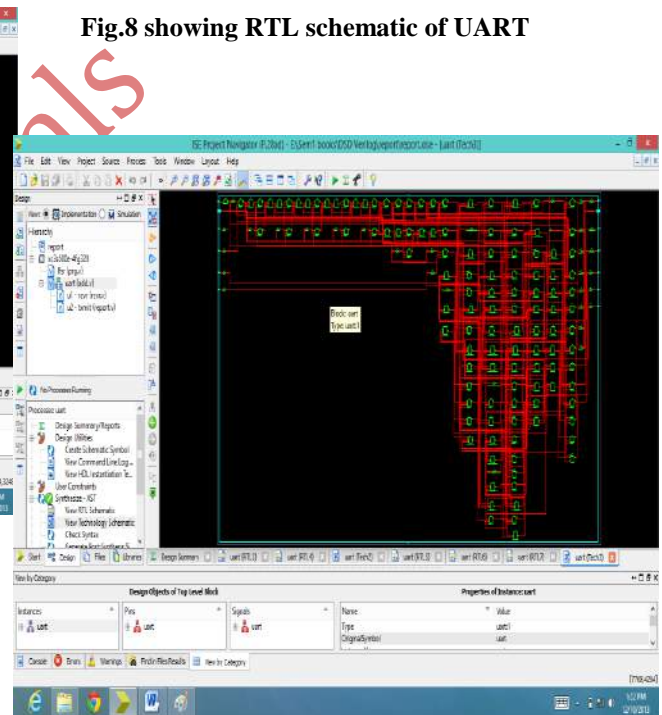


Fig.9 showing proposed structure schematic of UART with BIST



of overall production cost..

## 6. REFERENCES

- [1]. Nennie Farina Mahat, member ieee "Design of 9-bit UART module" IEEE-ICSE2012 Proc., 2012, Kuala Lumpur, Malaysia
- [2]. J. Norhuzaimin, and H.H. Maimun, "The design of high speed UART," Asia Pac. Conf. on Appl. Electromagnetics (APACE 2005), Johor, Malaysia, Dec. 2005
- [3]. Douglas A. Pucknell and K. Eshraghian, Basic VLSI design, PHI
- [4]. Nazeih M. Botors, HDL Programming VHDL and Verilog, Dreamtech Press, 2009
- [5]. Design for testability and BIST, Jin-fu LI, ARES lab
- [6]. Principles of CMOS VLSI design, by Neil H.E. Weste and Kamran Eshraghian.
- [7]. 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.
- [8]. [www.asic.co.in/BIST2- Design for testability](http://www.asic.co.in/BIST2-Design-for-testability)
- [9]. Bibin M C, Premananda BS, "Implementation of UART with BIST technique in fpga" International Journal of Inventive Engineering and Sciences (IJIES), Volume-1, Issue-8, July 2013
- [10]. Peter J Ashenden, Digital Design – An Embedded Systems Approach using Verilog.

IJournals