Partially reconfigured self test controller for fault detection and correction for FPGAs

Mrs.Jamuna.S[1], Dr. V.K. Agrawal[2]
1Associate Professor, Department of ECE, DSCE, Bangalore
2 Director,CORI,PESIT , Bangalore

Abstract

Field programmable gate arrays (FPGAs) are the reconfigurable logic devices which are widely used in many applications like system prototyping, complex computing systems, automotive electronics and mobile devices. Configurable Logic Blocks (CLBs) are the main logic resources for implementing sequential as well as combinatorial circuits in FPGA. However, increase in density and complexity also has resulted in more probability of faults. To effectively deal with the increased defect density, we need efficient methods for fault detection and correction. Many methods have been developed for fault detection, fault diagnosis and fault tolerance. Partial reconfiguration [PR] is a powerful solution which extends the capabilities of FPGAs. PR is a technique, which allows a portion of an FPGA to be reconfigured while other regions of the device continue to operate without any interruption. It is very useful for the devices operating in mission-critical applications which cannot be disrupted while some subsystems are being reconfigured. Here, we introduce an approach for FPGA testing that exploits the reprogram ability of a FPGA to create the self-test logic by configuring it using partial reconfiguration process. We have designed a reconfigurable built-in self test controller (RBIST) for coordinating the operations like detection and correction of faults. This technique is based on high level description without modifying the device architecture. All the reconfigurable modules are designed and implemented using *Xilinx ISE 12.4 and ML 507 FPGA board.*

Key words – Configurable logic block, Field programmable gate array, Partial Reconfiguration, RBuilt-in self test, SRAM,

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 [1]. 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 designs for testability (DFT) techniques. These techniques use the general concept of making all or some state variables controllable and observable [2].

A Field-Programmable Gate Array (FPGA) is a logic device that can be programmed to implement a variety of digital circuits. FPGAs are widely used both in product prototyping and development because of their ability for configuration and re-configuration. Some of the advantages are reduced design time and low non-recurring engineering cost. FPGA consists of an array of configurable logic blocks inter connected by programmable routing resources, and programmable I/O cells. The set of all programming bits establishes a configuration which determines the function of the device. With the increase in density, capability and speed, FPGAs have become more vulnerable to faults, as it is the case for all integrated circuits. A percentage of manufactured FPGA chips are determined to be faulty after initial application-independent tests. Faulty FPGAs can also be found after delivery to users, during system development or operation. They may be still usable for some particular application if only a portion of the circuitry is defective.

Reconfigurable Built-In Self Test is a technique of integrating the functionality of an automatic test system onto CLBs of FPGA. It is a Design for Test technique in which testing (test generation and test application) is accomplished through built in hardware features. The general RBIST architecture has a self test controller which controls entire tester circuit, test pattern 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). Testability is achieved without disturbing normal system functionality and with no area overhead, since the RBIST logic is configured on the CLBs which are not utilized by system function. We have utilized the flexibility of on-site programming and re-programming of FPGA for implementing the fault detection correction technique. Partial Reconfiguration (PR) is a process of modifying a subset of logic in an operating FPGA design by downloading a partial configuration file [3]. In this paper we explain how faults can be detected and corrected using RBIST. PR concept is been taken as a base for RBIST.

In this paper section II presents overview of FPGA CLB architecture, types of faults and fault models and previous techniques. Section III explains partial reconfiguration concept and self test controller. Section IV gives Implementation results and concludes with V.

II. General FPGA and CLB Architecture of VIRTEX5:

There are several families of FPGAs available from different semiconductor companies. These device families slightly differ in their architecture and feature set, however most of them follow a common approach: A regular, flexible, programmable architecture of Configurable Logic Blocks (CLBs), surrounded by a perimeter of programmable Input/output Blocks (IOBs). These functional elements are interconnected by a powerful hierarchy of versatile routing channels.

A. CLB and Slice Description:

Each CLB contains two Slices- SliceL(logic slice) and SliceM(memory slice).Each Slice has four basic logic elements. Logic element intern has a logic function generator (Look up table), a storage element, Multiplexers and Carry logic.The basic Virtex-5 logic element is illustrated in Fig1. It is composed of a 6-input look-up table (LUT), a configurable flip-flop/latch, and multiplexers to control the combinational logic output and the registered output (flip-flop/latch input).Fig.1 shows block diagram of a CLB.

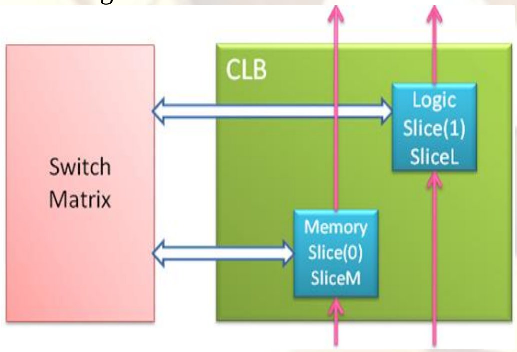


Fig.1 Structure of CLB

B. Fault & Fault Models

A fault is the representation of a defect reflecting a physical condition that causes a circuit to fail to perform in a required manner. A circuit error is a wrong output signal produced by a defective circuit. The reduction in feature size increases the probability that a manufacturing defect in the IC will result in a faulty chip. FPGA's are no exception from this. A very small defect can easily found when the feature size of the device is less than 100 nm. Furthermore, it takes only one faulty CLB or wire to make the entire FPGA fail to function properly. Yet, defects created during the manufacturing process are unavoidable and, as a result, some number of FPGA's is expected to be faulty; therefore, testing is required to guarantee fault free FPGA's.Faults can be divided into two categories:

1. Permanent Faults
2. Transient Faults

Fabrication faults and design faults are among the permanent faults. Transient faults, commonly called single event upsets (SEUs), are brief incorrect values resulting from external forces (terrestrial radiation, particles from solar flares, cosmic rays, and radiation from other space phenomena) altering the balance or locations of electrons, usually in a small area of the system. Fault models are required for emulation of faults or defects in a simulation environment. Because of the diversity of defects, it is difficult to generate tests for real defects. Fault models are necessary for generating and evaluating a set of test vectors.

Fault models can be divided into three types

1. Interconnect fault model
2. Logic Block fault model
3. Delay Fault

C. Previous Approaches:

In the related research, there are different approaches targeting fault detection in SRAM-based FPGAs. The first group aims at testing the FPGA by using the reconfigurability and the programming facilities of the device [4]–[12]. The second group aims at the modification of the original FPGA hardware to a new structure to make testing easy [13]–[16]. Conventionally, this class is named design for testability (DFT).

Testing of a single CLB: FPGA consists of NxN array of CLBs (2-D array). Testing can be done considering a single CLB at a time but in order to test all CLBs more time is required. Number of configurations applied depends on the number of CLB i/ps. Therefore testing a single CLB at a time becomes impractical. To reduce test time, [4], [5] [10] & [11] aimed to minimize the number of configurations. In [15], 21 CLB configurations were used for testing XC4000family.M.Renovell et al [4] Used 8 configurations. While T. Inoue et al [1] technique used only 4 configurations. C.Stroud et al [5] achieved maximal fault coverage only during offline testing. It used pseudo exhaustive testing excluding functionality of the FPGA.

III. PR concept and Self test controller

Partial reconfiguration is a powerful solution that can dramatically extend the capabilities of FPGAs. In addition to the potential for reducing size, weight, power, and cost, partial reconfiguration enables new types of FPGA designs that provide efficiencies unattainable with conventional design techniques[17]. PR takes this flexibility of FPGAs one step further, allowing the modification of an operating FPGA design by loading a partial configuration file, usually a partial bit file. After a full bit file configures the FPGA, partial bit files can be downloaded to modify reconfigurable regions in the FPGA without compromising the integrity of the applications running on those parts of the device that are not being reconfigured.[18].

In an SRAM-based FPGA, all user-programmable features are controlled by memory cells that are volatile and must be configured on power-up. These memory cells are known as the *configuration memory*, and they define the look-up table (LUT) equations, signal routing, input/output block (IOB) voltage standards, and all other aspects of the design[R]. In order to program the configuration memory, instructions for the configuration control logic and data for the configuration memory are provided in the form of a bit stream, which is downloaded to the device through a suitable configuration interface. An FPGA can be partially reconfigured using a partial bit stream. This partial bit stream in turn changes logic of a selected part of the design while rest of the device continues to operate.

We have designed an online self tester using PR for detection and correction of faults in the CLBs. HDL model of a CLB of VIRTEX5 is taken as circuit under test[CUT] . Two copies of the CLB are considered at a time. Faults are introduced at random locations in one and other is configured as fault free. CLB where faults are introduced is defined as Partial blocks using Plan ahead tool of ISE software.

Self test controller: RBIST 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 RBIST 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. 2 depicts the procedure.

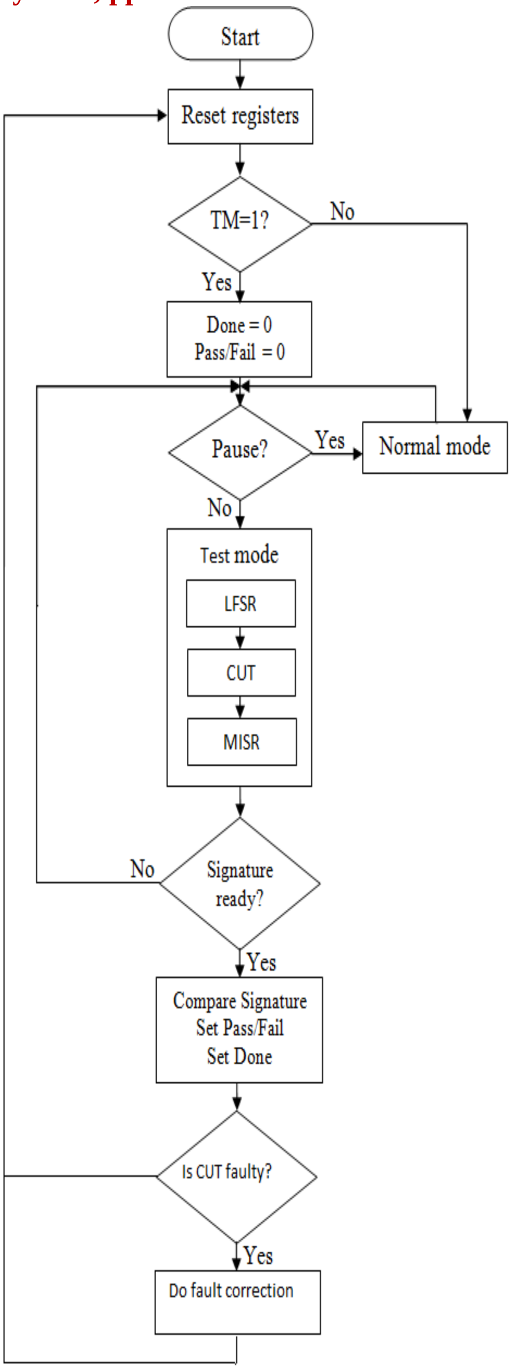


Fig. 2 flowchart for fault detection procedure.

Fault correction: once identification of stuck – at faults is over, fault correction logic is applied. Fault correction logic block also is implemented as a PR block as it has to take multiple configuration bit files depending on the position of fault. Correction of faults is done by modifying the content of CLB. Faulty cells of the logic slice are avoided with the help of partial reconfiguration. For fault correction, configuration memory is accessed through the internal configuration access port (ICAP).Detected faults are avoided by shifting the bits of configuration memory content as per the design requirement. Partial bit files are generated and downloaded on to the device depending on the position of fault. We have applied this technique for detection and correction of single as well as multiple faults. RTL schematic of the technique is as in fig.3

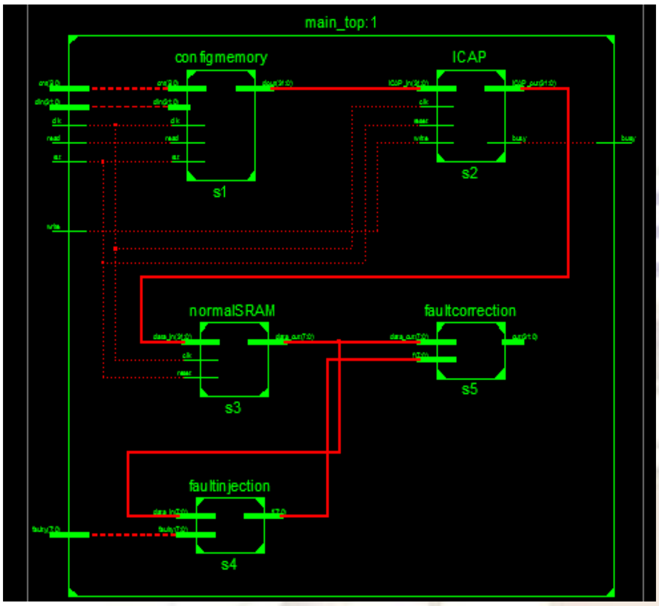
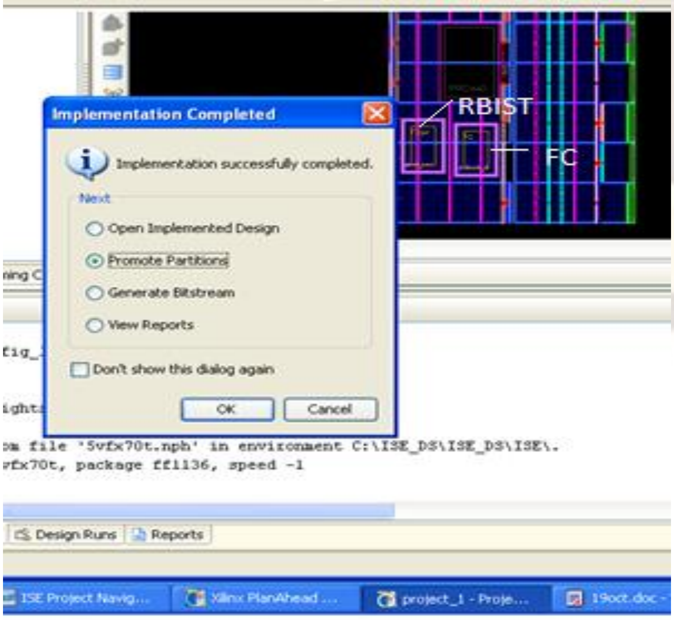


Fig. 3 RTL schematic.

IV. Implementation Results:

As we can see in fig .4, reconfigurable self tester and fault correction blocks have been defined as partially reconfigurable blocks and implemented successfully. Interconnected structure of the design is as shown in fig 5.Bit files are generated for the previously defined partial blocks which are then down loaded to VIRTEX5 on the ML507 board.



Implementation Completed

Implementation successfully completed.

Next

○ Open Implemented Design
● Promote Partitions
○ Generate Bitstream
○ View Reports

□ Don't show this dialog again

OK Cancel

```
om file '5vfx70t.nph' in environment C:\ISE_DS\ISE_DS\ISE\.
vfx70t, package ff1136, speed -1
```

Fig.4. Implemented Pblocks of RBIST and FC

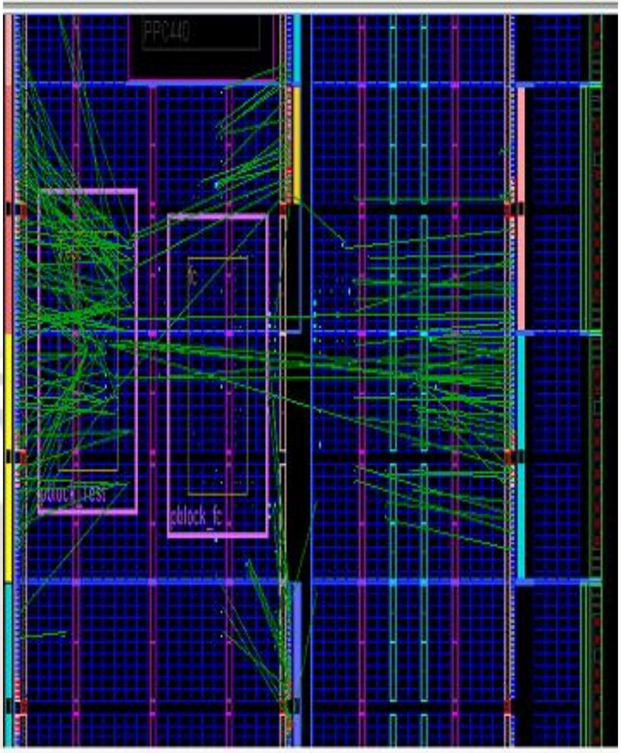
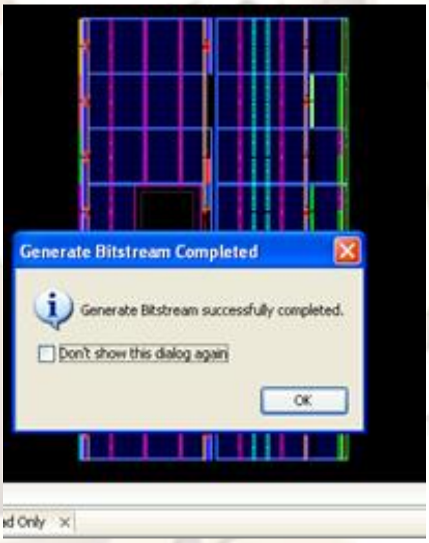


Fig 5. routed design on VIRTEX5 FPGA.


Generate Bitstream Completed

Generate Bitstream successfully completed.

□ Don't show this dialog again

OK

Fig. 6 result after bit file generation.

V. Conclusion

In this paper we have explained about detection and correction of stuck-at faults which occur in CLB of an FPGA. VHDL is used for modeling the device under test. Partial reconfiguration concept is used for correcting the faults through ICAP configuration interface. Using this single as well as multiple faults can be detected and corrected. As we can optimize placement and mapping of the logic blocks in plan ahead tool of XILINX ISE12.4, area overhead and implementation time is reduced.

REFERENCES

[1]. M.Bushnell and V.D. Agarwal, " Essentials of Electronic Testing for Digital, Memory and Mixed-signal VLSI Circuits" Kluwer Academic Publishers, 2000.

[2] C. Metra, G. Mojoli, S. Pastore, D. Salvi, and G.Sechi, "Novel technique for testing FPGAs," *Proc. IEEE Design Automation and Test in Europe*, pp. 89–94, 1998.

[3] Xilinx, "Two Flows for Partial Reconfiguration: Module Based or Difference Based," *XAPP290,*

[4] L.T. Wang, Cheng- Wen Wu and Xiaoqing Wen, "VLSI Test Principles & Architectures Design for testability".

[5] C. Stroud, S. Konala, P. Chen, and M.Abramovici, "Built-in self-test of logic in FPGAs," in *Proc. 14th Very Large Scale Integration (VLSI) Test Symp.*, 1996, pp. 387–392.

[6] W. K. Huang, M. Y. Zhang, F. J. Meyer, and F. Lombardi, "A XOR-tree based technique for constant testability of configurable FPGAs," in *Proc. Asian Test Symp.*, 1997, pp. 248–253.

[7] W. K. Huang, F. J. Meyer, and F. Lombardi, "Multiple fault detection in logic resources of FPGAs," in *Proc. Defect and Fault Tolerance in Very Large Scale Integration (VLSI) Systems*, 1997, pp. 186–194.

[8], "Testing configurable LUT-basedFPGAs," *IEEE Trans. VLSI Syst.*, vol. 6, pp. 276–283,June 1998.

[9] W. K. Huang and F. Lombardi, "An approach for testing programmable/ configurable field programmable gate arrays," in *Proc. IEEE Very Large Scale Integration (VLSI) Test Symp.*, Princeton, NJ, 1996, pp. 450–455.

[10] T. Inoue, H. Fujiwara, H. Michinishi, T.Yokohira, and T. Okamoto, "Universal test complexity of field-programmable gate arrays," in *Proc. 4th IEEE Asian Test Symp.*, Nov. 1995, pp. 259–265.

[11] M. Renovell, J. Figueras, and Y. Zorian, "Test of RAM-based FPGA: Methodology and application to the interconnect structure," in *Proc. 15th IEEE Very Large Scale Integraton (VLSI) Test Symp.*, 1997, pp. 204–209.

[12] F. Ferrandi, F. Fummi, L. Pozzi, and M. G. Sami, "Configuration-specific test pattern extraction for field programmable gate arrays," in *Proc. Defect and Fault Tolerance in Very Large Scale Integration (VLSI) Systems*, 1997, pp. 85–93.

[13] T. Liu, W. K. Huang, and F. Lombardi, "Testing of uncustomized segmented channel FPGAs," in *Proc. ACMInt. Symp. on FPGAs*, Feb. 1995, pp. 125–131.

[14] A. Doumar, T. Ohmameuda, and H. Ito, "Design of an automatic testing for FPGAs," in *Proc. IEEE Euro. Test Workshop*, May 1999.

[15] Fast testable design for SRAM-based FPGAs," *IEICE Trans. Inform. Sys.*, vol. E83-D, no. 5, pp. 116–1127, May 2000.

[16] "Virtex-5 FPGA User Guide," user manual: UG190 (v5.3)., May 17, 2010.

[17] WP374 (v1.0) July 23, 2010, Partial Reconfiguration ofVirtex FPGAs in ISE 12

[18] ug702 Partial Reconfiguration User Guide