**International Journal of Reconfigurable and Embedded Systems (IJRES)**
Vol. 3, No. 2, July 2014, pp. 39~48
ISSN: 2089-4864


# FPGA Evaluation of Reconfigurable Modules with Fault Detection and Repair Technique

**Pradeep C*, Radhakrishnan R****
*Department of ECE, SAINTGITS College of Engineering, Kottayam, Kerala, India
** Sri Shakthi Institute of Engineering &Technology, Coimbatore, Tamilnadu, India

**Article Info**

***Article history:***

Received Jan 22, 2014
Revised Apr 5, 2014
Accepted Apr 22, 2014

***Keyword:***

CLB
Fault Detection
FPGA
Partial Reconfiguration
Self Repair

**ABSTRACT**

This paper proposes a fault detection and repair algorithm which is suitable for fault free reconfigurable systems. In recent years Built in Self Repair digital systems have got very important role in the applications such as nuclear systems, space missions and communication systems etc where system reliability is very critical. Systems designed to operate in critical conditions will collapse due to even a single fault occurrence. To avoid these situations many methods have developed in recent years. This work proposes an area efficient and fast fault detection and repair algorithm. For the evaluation of the new approach and older methods a system with a standalone module and four add on modules were designed and evaluated for resource utilization using XUPV5 board. The entire FPGA is divided in to tiles and each module is implemented in different tiles using partial reconfiguration method using Xilinx Plan Ahead 14.2 with partial reconfiguration facility.



***Corresponding Author:***

Pradeep C,
Department of ECE, SAINTGITS College of Engineering, Kottayam, Kerala, India
09447661355
Email: pradeepcee@gmail.com

## 1. INTRODUCTION

In any systems, *system failure* occurs when the system deviates from its requirements, which may be catastrophic for specific critical applications [1]. Such systems have redundant hardware so that operational reliability is uncompromised by isolated failure [2]. Reconfigurable systems use runtime reconfiguration techniques to adapt the system operation to changes in its external parameters and also to increase the lifetime of application. The later can be achieved if the system is capable to detect, diagnose and repair the faults that occur during the run time.

In a Self Repairing System various techniques are used to find the deviation from its normal operation. For an optimal design of these techniques, tradeoffs in area and time are considered. Repairing algorithms are made to perform logical correctness and temporal correctness [3]. A system is called logically correct if it satisfies all the functionality and the system is temporally correct if all these functions are completed within a specific time. So the logical correctness of a fault tolerant system is the recovery of the original function if a fault occurs and temporal correctness is the recovery of the original function in a specific time bound. In real time fault tolerant systems fault detection and repair must be done in a time limit. Popular applications of reconfigurable systems are microwave filters for cellular phones [4], power management systems and space craft.

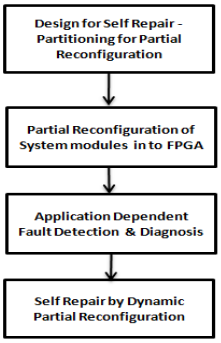


Figure 1. Design Cycle for Self Repair Reconfigurable Systems

In Dynamically reconfigurable Field Programmable Gate Arrays (FPGA) a partition can be configured for fast fault recovery [5]. The design of self repairable system consists of fault detection, fault diagnosis and self repair while system continues its operation with full or partial functionality. This paper presents an algorithm and its evaluation for fast fault recovery. The algorithm is useful in achieving high fault coverage if it is used with increased number of spare modules. The faulty module undergoes a self fault detection process and if the fault is transient the faulty module becomes the working module again.

## 2. STRUCTURE OF RECONFIGURABLE HARDWARE

Field Programmable Gate Arrays are used as the reconfigurable hardware. FPGA devices are widely used to implement complex systems also in applications that increasingly require capacity of autonomous self-checking and possibly of survival to faults, often within cost limitations that exclude massive redundancy.

The reconfigurable FPGA consists of an array of configurable logic blocks (CLBs) which can be programmed with configuration cell data to generate logical functions. The set of all configuration cell data makes up an FPGA configuration. Each CLB can be connected to the other CLBs via the interconnect networks. The input–output signals can be transmitted to/from outside the FPGA using the input–output blocks.

A switching block is programmable connecting element receiving lines on each side. The lines connect switching block pins identified as north, east, south, and west. Some pairs of pins in the block cannot be connected; these are called non connectable (NC) pins. Pairs that can be connected are called connectable pins. The connection of the set of pins is controlled by the configuration cells dataThe connectable pins are represented by dotted lines. A connection block has the same structure as a switching block, except for the fact that the connection blocks hold routing switches that serve to connect the logic pins from CLB to the wire segments, while the routing switches of the switching blocks allow wire segments to be connected to other wire segments. The connections between lines are also controlled by the configuration cells data.

If these types of reconfigurable hardware are used for systems which its responses are critical and those should operate for long time, a high level run time fault tolerance should be achieved .The built in self repair system capable of run time fault detection, runtime fault diagnosis and self repair will extend the life time of the reconfigurable systems.

A circuit under test fails when its observed behavior is different from its expected behavior. If the circuit is to be repaired, the cause of the observed fault must be diagnosed [8]. Repairing often consists of substituting one of its replaceable units (CLB) identified as containing faults and referred as a faulty CLB, by a good CLB. The diagnosis process is often hierarchical. But Xilinx released FPGA Like Virtex 5 has internal reconfiguration support [8]. In these type of FPGA's the reconfiguration of some logic blocks and wire segments are possible, while some other programmable hardware is in the functional mode ("Running").

## 3. RELATED WORK

To introduce fault tolerance in reconfigurable systems many approaches have been developed by researchers in this field and a few of them are considered in this paper. In modern FPGA's the previously unused parts can be reconfigured with the functional module to replace the faulty part of the FPGA. Also

there are many cost effective and efficient methods have been developed for tolerating faults without the complete shutdown of the system. Among these methods Triple modular redundancy (TMR) is the one of the first and most commonly used technique. In this method the same function is implemented using three equivalent modules which run simultaneously and the majority output is used. The fault is uncovered as soon as a mismatch occurs. The main drawback of this method is the hardware overhead for the functionality replication [6]. But this approach has immense timing performance advantage since there is no fault detection phase is required. The system will function correctly only up to one fault level and the implementation cost is very high.

Another method proposed in [7] the number of functional modules that can be expanded or reduced depends on the application. The functional module employs a double modular redundancy (DMR) for its own fault detection. Every functional unit has two modules one is the active (M) and its duplicate (M'). The fault detection of each working module is done by comparing the outputs of M and M' using an EXOR gate. If fault occurs XOR gate output is one and it is used to initiate fault recovery by means of partial reconfiguration. But this method also have a disadvantage of area overhead since two modules of each functional blocks are implemented and due to the area reserved for spare module.

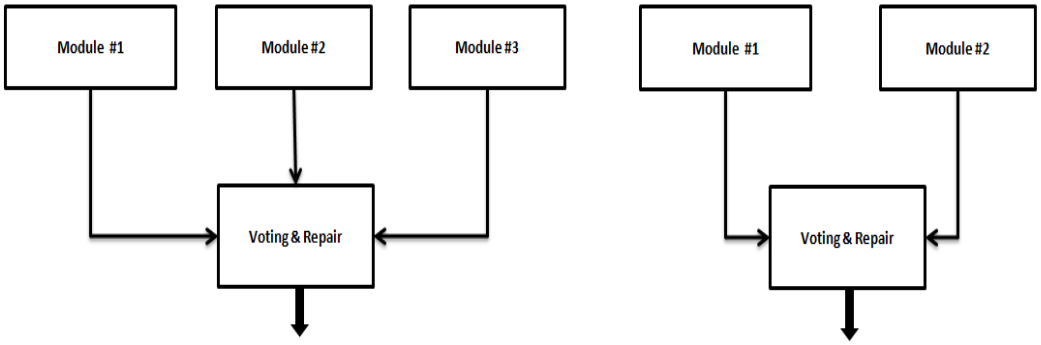


Figure 2. Fuctional diagram of TMR and DMR

The third method considers the FPGA at a low abstraction level, as a matrix of basic processing elements (CLBs – Configurable Logic Block) linked by switched interconnecting buses. The specific characteristics of the application system mapped onto the FPGA are not taken into account. Detection is performed by means of conventional techniques [11]. When an error is detected, the faulty FPGA block is identified, removed from the active computation and replaced by spare elements in order to preserve the nominal detection ability as long as possible. To this end, the interconnection structure is properly reconfigured by using one of the many reconfiguration techniques for regular array architectures. The interconnections and the configuration memory are usually considered fault free. The main drawbacks of this approach are due to the hardware redundancy needed to support detection (in particular, if checking is performed at low granularity) and to the fact that reconfiguration may be hampered by lack of sufficient redundant paths.

A fourth approach adopts a high-level view of the system. Fault detection techniques are introduced at application system level. Checking is executed by analyzing the intermediate or final application outputs, according to the adopted detection technique. There is no relationship between the location of the checking operation and the array structure of the underlying FPGA device. Whenever an error is detected, a diagnostic phase must be introduced in order to locate the faulty CLB [9]. Reconfiguration is then accomplished by excluding such element from the active computation through interconnection reconfiguration and spare activation. In this approach, circuit complexity and performance loss are reduced with respect to the previous case; however, the separate diagnostic phase leads to longer repair time, since appropriate testing need to be executed (obviously, reconfiguration is not "transparent" as far as time is concerned). Here also, reconfiguration may incur in lack of sufficient routing paths, although this case is less likely than in the previous approach since rerouting must be completely recomputed.

The fifth approach augments the physical structure of the FPGA device to directly support fault detection and confinement [10]. The basic CLBs and interconnections are modified so that detection is performed by internal check in every CLB. This solution is highly effective and efficient, but requires large

circuit complexity. The first two methods mentioned above are used here for the comparison with the new approach and the functional diagrams of those methods are given in figure 2 (a) and (b).

## 4. PROPOSED SELF REPAIR METHOD

Reconfigurable Systems by means of reconfiguration consists in partitioning the FPGA architecture into groups of FPGA resources (called *tiles*) on which portions of the computation are mapped. Most of the tiles are used to implement functional blocks of the desired computation. More functional units can be mapped in the same tile. Some tiles (the *spare tiles*) are left unused so that they can be exploited to replace the faulty tiles in reconfiguration to allow the system survival as shown in figure 3.

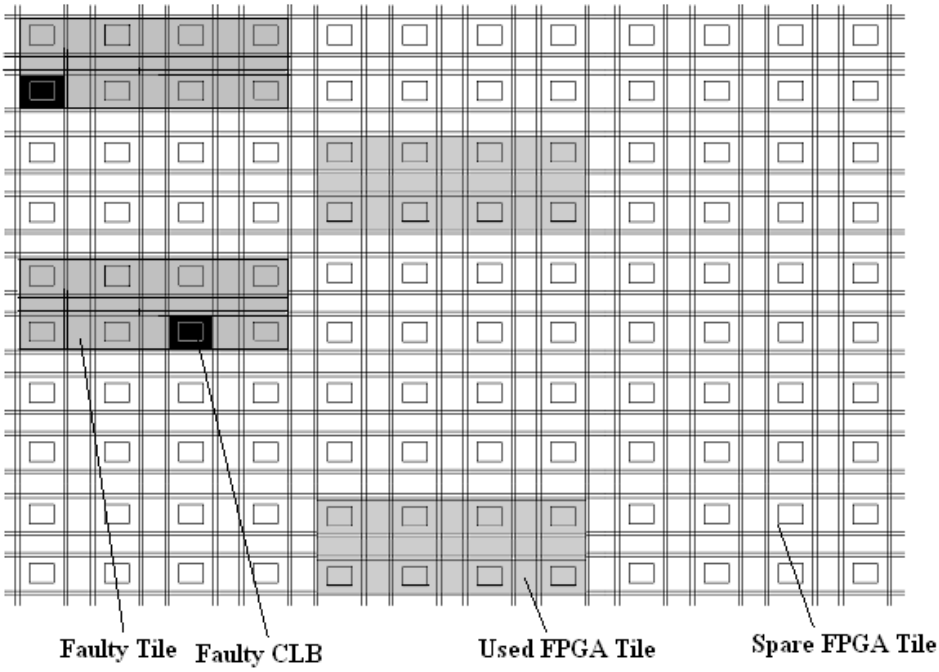


Figure 3. Tiles in the FPGA Architecture

In this approach the system is partitioned in to various modules for the ease of dynamic partial reconfiguration. The partition was done in a way that the system undergoes a mode change with basic functionality when a fault is detected. The functionality will be increased further by reconfiguration of more modules after ignoring the faulty tile. The overall scheme is presented using the flowchart in figure 4.

To detect the fault, an online BIST method is used here. The on-line testing is based on partitioning the digital system into tiles and using distributed checkers in the tiles to provide the fault location by identifying the checker by which the error is detected as shown in figure 5. Each tile is composed of a set of physical resources (CLBs and interconnect), an interface specification, which denotes the connectivity to neighboring tiles, and a net list. The tiles of small sizes help to reduce the amount of configuration memory by reducing the size of the component that is reconfigured. All possible configurations of the tiles are generated off-line. Upon detecting an error from a certain tile, the faulty tile only need to be diagnosed to locate the faulty CLB, which will be out of using in the original configuration or as a spare element. The faulty tile located by the diagnosis procedure is then circumvented by a new configuration downloaded from an external memory. This is possible because some of the CLB's are used as spares for the rest. Since the routing area represent the majority of the FPGA chip, the need for a detection and diagnosis approach of interconnections extremely high. This paper uses an online approach which reduces the detection and diagnosis time.

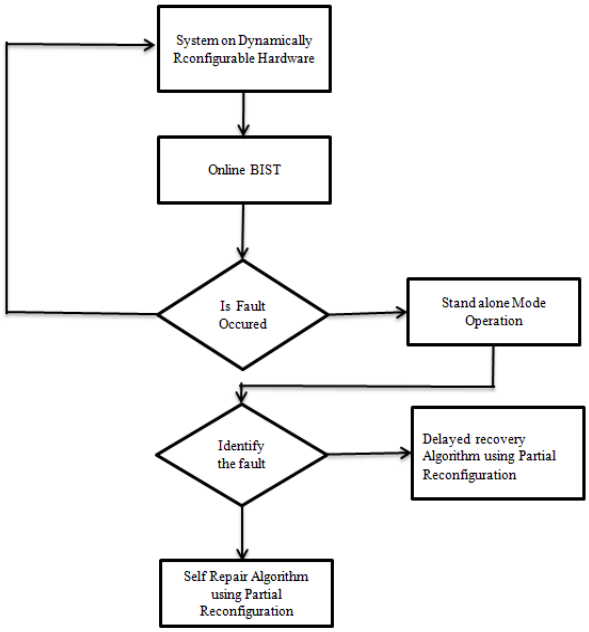


Figure 4. Flow Diagram for fault detection and repair mechanism

When a fault is detected, the system undergoes a mode change immediately. The system is designed in a way that it will function according to last completed operation and by providing necessary control signal to the peripherals. In this condition system is not able to do any other processing. So fault identification and repair was done with less configuration time. In the proposed approach the systems that are to be implemented in the FPGA is partitioned in to self checking tile shown in figure 5. The fault indicating signal from each tile is taken as the observable points and for a multiple output circuits many sub circuits can be partitioned.

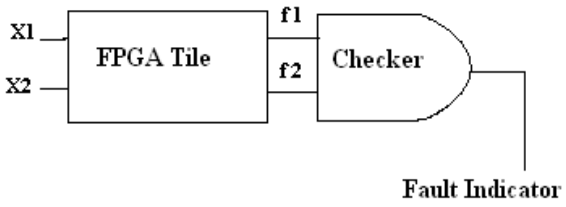


Figure 5. The proposed FPGA Tile

The new approach has two phases one is to identify the faulty tile using the new method and the second is to use any existing BIST method to identify faulty interconnect. The entire circuit level is dividing into sub circuit level until it becomes two levels. Then the sub circuits are divide into self checking tiles using the cone segment method [11-12] with certain error detection and correcting codes. Then the fault indicator output of self checking tiles are given to the complete checker. So when complete checker circuit receive any fault indication it can easily locate the faulty tile. In the next phase uses online BIST based diagnosis methods [13] to identify the faulty resources.

The fault recovery process is based on the type of fault occurred and separate algorithms for permanent fault and transient faults are used. Algorithm used in this approach for permanent fault repair is given in Algorithm 1. After fault identification availability of spare tile was checked and faulty tile is replaced by spare tile if it available. Also a module rearrangement procedure was introduced to reduce routing complexity (RC). System undergoes a module rearrangement otherwise by ignoring the least important (Re) functional module from the system if spare is not available.

Algorithm 1: Self Repair Algorithm.

```
1.  Do Self Testing of Tiles
2.  If (fault=1) go to stand alone mode
3.      Identify the faulty Tile.
4.      Check for the Spare Availability
5.      If Corresponding section spare available then
6.          Go To  Full Module Rearrangement.
7.          IF faulty Tile <=  High RC Tile
8.              Spare Tile= Faulty Tile
9.          Else
10.             Do rearrangement to have Less RC
11.         End
12.     Else
13.         Go to Partial Module Rearrangement
14.         If faulty Tile Module = Lowest Re
15.             Ignore it
16.         Else
17.             Ignore the lowest Re module
18.             do rearrangement
19.         End
20.     End
21.   Else
22.   Go to step1
23. End
```

## 5. EXPERIMENTAL SETUP

The above mentioned fault detection and repair approach is evaluated by designing a system core with stand alone module and four add on modules. The system is designed in way that it can work with any add on modules with basic functionality ie when a fault is detected system will be dynamically reconfigured to its basic mode. Add on modules designed for the evaluation are a double precision floating point unit, integer ALU capable of doing addition operation only, SDRAM controller and an asynchronous serial communication interface (UART). Each module is coded in Verilog HDL and simulated separately by using Xilinx ISE simulator.

The regular synthesis flow generates a single bit stream for programming the FPGA. This considers the device as a single atomic entity. In contrast, the Partial Reconfiguration flow physically divides the device in regions. One region is called the “static region”, which is the portion of the device that is programmed at startup and never changes. One region is the “dynamic region” also known as “the PR region”, which is the portion of the device that will be reconfigured dynamically, potentially multiple times and with different designs. It is possible to have multiple PR regions, but we will consider only the simplest case here [14] The PR flow generates at least two bit streams, one for the static and one for the PR region. Most likely, there will be multiple PR bit streams, one for each design that can be dynamically loaded. The PR region is a physical entity, with a given geometry. PlanAhead is the tool that allows you to define the exact location of the PR region on your target device [14].

The Virtex 5 LXT ML505 is a general purpose FPGA development board enables designers to investigate and experiment with features of the Virtex-5 LXT. Provides feature-rich general purpose evaluation and development platform. Includes on-board memory and industry standard connectivity interfaces. Delivers a versatile development platform for embedded applications [15].

The Virtex 5 FPGA is one of the latest and widely used FPGA for partial reconfiguration. The family of Virtex 5 FPGA has advanced features like ASMB column based architecture. The family has five different sub-families [16]. Each sub family has variety of features needed for modern logic designs. Few applications of Virtex 5 FPGA’s are Reconfigurable Processors, Soft processors and DSP processors etc. Virtex 5 FPGA is now used as an alternative to AIC design and it is built on 65 nm technology.

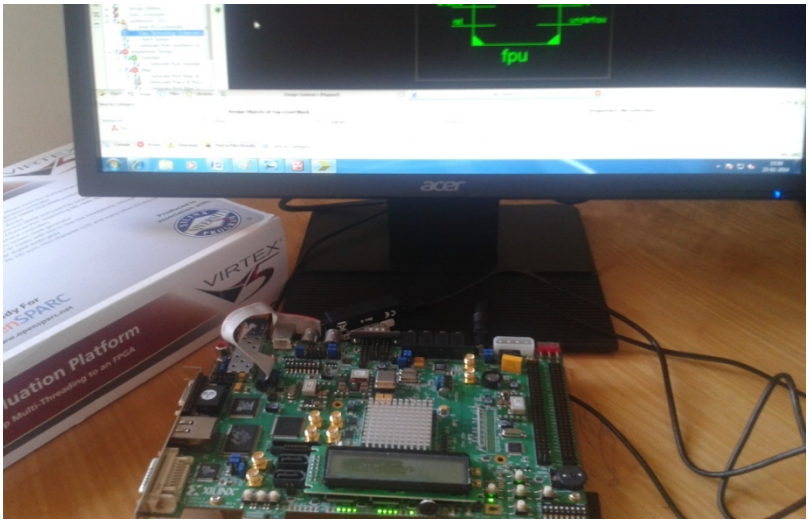


Figure 7. Experimental setup for on a Xilinx XUPV5 evaluation board

## 6. RESULTS AND DISCUSSIONS

For the evaluation of above mentioned algorithm the system was implemented on a Xilinx XUPV5 evaluation board with a Virtex-5 LX110. Xilinx Palnahead with Partial reconfiguration capability was the development tool used. The partitioned system is coded in Verilog HDL and simulated using appropriate test benches for verifying the functionality.

The system is designed in a way that it can function in various mode i.e. stand alone mode, stand alone mode with one add on module, stand alone mode with two add on modules, stand alone mode with three add on modules, and with full functionality modes. All the above modules are synthesized using Xilinx XST 14.2 under various methods to obtain the resource utilization. The result of this is given in following sections.

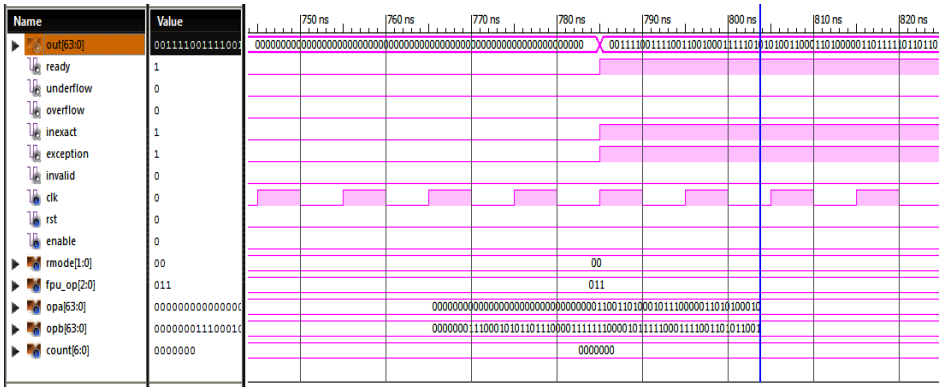


Figure 8. Simulation results for double precision floating point unit

Simulation result of floating point unit using Xilinx ISE simulator is shown in figure 8. For the design IEEE 754 double precision floating point standard used. The mantissa is 52 bit long and represented by bits 51-0. If the exponent field is greater than '0' a leading '1' is also included with the mantissa. 11 bits are used to represent exponent part and Bit field is 62-52.The actual value of exponent is calculated by 2^ (e-1024) the value represented by 11 bit field is offset by 1024.

Synthesis Results of various methods using Xilinx XST tool is given in Tables 1-4. To find the area overhead for the self repairing schemes Synthesis of the system without introducing any self repair concept is evaluated. The utilization shown in table 1depends on the functionality introduced in the design and the type of the coding style. Any way the same design is used for all the three evaluations so the design method and style of coding does not influence the comparison results.

Table1. Synthesis results for System without Fault Detection and Repair Method

| Without Fault Detection and Repair Method | | | | | |
|---|---|---|---|---|---|
| Tile No. | Module Name | Number of Slice Registers | Number of Slice LUT's | Number used as Logic | Number of fully used LUT-FF pairs |
| 1 | Floating Point Unit | 4408 | 6116 | 6115 | 3908 |
| 2 | Integer Unit | | 181 | 181 | |
| 3 | SDRAM Controller | 78 | 168 | 168 | 78 |
| 4 | UART | 95 | 129 | 128 | 84 |
| TOTAL | | 4581 | 6594 | 6592 | 4070 |

To perform the synthesis of TMR, all the four add on modules were synthesized separately by replicating it three times and introducing a function for majority voting. After the TMR voter determines the majority output value, it sends the correct information back to the three active modules. The TMR voter and repair circuit contains the necessary logic for the switching of which three modules are active in addition to reset and reinitialize any of the modules. The synthesis results shown in Tables clearly shows that the utilization is also three times higher than that of a system without self repair feature. There was no additional hardware designed for the fault detection.

Table 2. Synthesis results for System Triple Modular Redundancy Method

| TMR Method | | | | | |
|---|---|---|---|---|---|
| Tile No. | Module Name | Number of Slice Registers | Number of Slice LUT's | Number used as Logic | Number of fully used LUT-FF pairs |
| 1 | Floating Point Unit | 13289 | 18421 | 18418 | 11781 |
| 2 | Integer Unit | | 616 | 616 | |
| 3 | SDRAM Controller | 299 | 577 | 577 | 291 |
| 4 | UART | 350 | 460 | 457 | 309 |
| TOTAL | | 13938 | 20074 | 20068 | 12381 |

Table 3. Synthesis results for Double Modular Redundancy Method

| DMR Method | | | | | |
|---|---|---|---|---|---|
| Tile No. | Module Name | Number of Slice Registers | Number of Slice LUT's | Number used as Logic | Number of fully used LUT-FF pairs |
| 1 | Floating Point Unit | 8861 | 12285 | 12283 | 7873 |
| 2 | Integer Unit | 45 | 415 | 415 | 57 |
| 3 | SDRAM Controller | 201 | 389 | 389 | 213 |
| 4 | UART | 235 | 311 | 309 | 225 |
| TOTAL | | 9342 | 13400 | 13396 | 8368 |

Table 4. Synthesis results for New Approach

| New Approach | | | | | |
|---|---|---|---|---|---|
| Tile No. | Module Name | Number of Slice Registers | Number of Slice LUT's | Number used as Logic | Number of fully used LUT-FF pairs |
| 1 | Floating Point Unit | 4408 | 6116 | 6115 | 3908 |
| 2 | Integer Unit | | 181 | 181 | 0 |
| 3 | SDRAM Controller | 78 | 168 | 168 | 78 |
| 4 | UART | 95 | 129 | 128 | 84 |
| 5 | Spare Tile | 4408 | 6116 | 6115 | 3908 |
| TOTAL | | 8989 | 12710 | 12707 | 7978 |

Finally all recourse utilization of the new approach was compared with the resource utilization of TMR and DMR method. Figure 9 shows the comparison and it has been clearly observed from the diagram that the new approach is having less area overhead than the TMR and DMR methods.

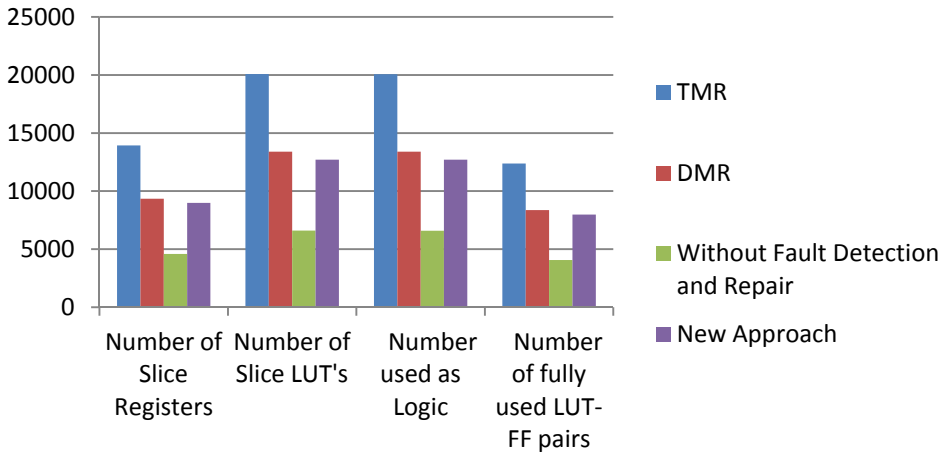


Figure 9. Comparison of Recourse Utilization of Various Methods

## 7. CONCLUSION AND FUTURE WORK

This paper presented the evaluation of a fault detection and repair algorithm for Reconfigurable Systems. In this approach effective utilization of FPGA resource by means of Partial Reconfiguration is used. The FPGA is partitioned into various tiles where the system modules are implemented. The comparison of FPGA recourses utilization shows that this approach has less area over head than the other methods. So this method is very useful in real-time embedded systems and long life mission critical applications. The efficiency of this method is limited by the structure of dynamically reconfigurable FPGA. These method also has the design complexity of partitioning the system as a standalone module and add on modules. In this approach the occurrence of only one fault is evaluated. The correctness of this repair algorithm for more fault can be evaluated by considering another system with more add on modules and by dividing the FPGA in to more number of Tiles.

## REFERENCES

[1] L. Moser and M. Melliar-Smith. Demonstration of fault tolerance for corba applications. In *ARPA Information Survivability Conference and Exposition*, volume 2, page 87, Washington, DC, USA, 2003. IEEE Computer Society.

[2] R. Ellisona, R. Linger, H. Lipson, N. Mead, and A. Moore. Foundations for survivable systems engineering. Technical report, CERT Coordination Center, Software Engineering Institute Carnegie Mellon University, 2001.

[3] G.W.Greenwood, "On the practicality of using intrinsic reconfiguration for fault recovery", IEEE Trans. Evol. Comput. vol. 9, no. 4, pp. 398–405, Aug. 2005.

[4] Y. Kasai, H. Sakanashi, M. Murakawa, S. Kiryu, N. Marston, and T. Higuchi. Initial evaluation of an evolvable microwave circuit. In J. Miller, A. Thompson, P. Thomson, and T. C. Fogarty, editors, *Evolvable Systems: From Biology to Hardware, Third International Conference on Evolvable Systems (ICES-2000)*, pages 103–112, Edinburgh, 2000. LNCS volume 1801, Springer Verlag.

[5] J. D. Hadley and B. L. Hutchings, "Designing a partially reconfig.d system in FPGAs for fast board development and reconfigurable computing," in Proc. SPIE 2607, 1995, pp. 210–220.

[6] R. E. Lyons and W.Vanderkulk, "The use of triple modular redundancy to improve computer reliability," IBM J., vol. 6, no. 2, pp. 200–209, Apr. 1962.

[7] Reshma Mary John, Pradeep "Self--Repairing Algorithm with Shared Spare Allocation for Reconfigurable Systems". International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 3, Issue 8, August 2013

[8] Abramovici, M.; Strond, C.; Hamilton, C.; Wijesuriya, S.; Verma, V., "Using roving STARs for on-line testing and diagnosis of FPGAs in fault-tolerant applications", *Proc. International Test Conference*, 1999

[9] El-Ayat, K.; Cahn, R.; Chung Lau Chan; Speers, T., "Array architecture for ATG with 100% fault coverage", *Proc. Workshop Defect and Fault Tolerance in VLSI Systems*, 1991
[10] Chapman, G.H.; Dufort, B., "Making defect avoidance nearly invisible to the user in wafer scale field programmable gate arrays", *Proc. Int. Symp. Defect and Fault Tolerance in VLSI Systems*, 1996
[11] M. Abramovici, M. A. Breuer, and A. D. Friedman, Digital Systems Testing and Testable Design. Jaico Publishinf House, Mumbai 2004..
[12] K. Elshafey, J. Hlavicka, "On-Line Detection and Location of Faulty C I A in FPGA-Based Systems", IEEE DUECS Workshop, Bmo, Czech Republic, April 17-19, 2002, pp. 183-190.
[13] Miron Abramovici, Charles E. Stroud, John M. Emmert, "Online BIST and BIST-Based Diagnosis of FPGA Logic Blocks", *IEEE Transactions On Very Large Scale Integration (Vlsi) Systems*, Vol. 12, No. 12, December 2004, Pp 1284-1294.
[14] Xilinx, Inc. "Partial Reconfiguration User Guide", Xilinx UG702. July 6, 2011. *http://www.xilinx.com / support/documentation/sw_manuals/xilinx13_2/ug702.pdf*
[15] Xilinx, Inc. "ML505 Hardware User Guide." Xilinx UG347 May 16, 2011. *http://www.xilinx.com/ support/documentation/boards_and_kits/ug347.pdf*
[16] Xilinx, Inc. "Virtex-5 Family Overview."Xilinx DS100 (v5.0) February 6, 2009. *http://www.xilinx.com/ support/documentation/data_sheets/ds100.pdf*