Performance Enhancement Defect Tolerance in the Cell Matrix Architecture

C. R. Saha, S. J. Bellis, A. Mathewson and E. M. Popovici

Abstract- This research concentrates on the area of fault tolerant circuit implementation in a field programmable type architecture. In particular, an architecture called the Cell Matrix, presented as a fault tolerant alternative to field programmable gate arrays using their Supercell approach, is studied. Architectural constraints to implement fault tolerant circuit design in this architecture are discussed. Some modifications of its basic structure, such as the integration of circuitry for error correction and scan path, to enhance fault tolerant circuits design are introduced and are compared to the Supercell approach.
Keywords: Cell Matrix, Supercell, fault tolerance, FPGAs, Hamming code, Scan path.

I. Introduction

Fault tolerance plays an important role in modern digital systems. In recent years research into fault tolerant systems has also been driven by the rapidly increasing use of computer based systems in railway and traffic control, aircraft flight, hospital patient monitors, telecommunication and others where failure can cost lives, money or both. State-of-the-art programmable devices such as Field Programmable gate Arrays (FPGAs) offer many new opportunities for the implementation of digital circuits in the computing and digital signal processing areas. The reconfigurability of these computing architectures is an exciting and promising area for the future research into fault tolerance. Over the past decades, a great deal of research has been done in this area but little commercial support for this technology has been available. With this deficit and future predictions for the requirement of a nanotechnological, fault tolerant and reconfigurable architecture in mind, the Cell Matrix Corporation have developed the Cell Matrix Architecture [1]. The architecture consists of a simple regular two-dimensional collection of cells. Fig. 1 represents a simple four-sided cel1. It has four data and four control inputs/outputs designated north, south, east and west. Each cell consists of a small amount of logic and a local memory that is used as a look up table (LUT).

C. R. Saha, S. Bellis and A. Mathewson are with National Microelectronic Research Center, Lee Malting, ProspectRow, and Cork, Ireland. E-mail: chitta@nmrc.ie

E. Popovici is with the Department of Microelectronic Engineering, University College Cork, Cork, Ireland. E-mail: e.popovici@ucc.ie

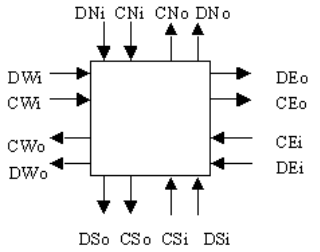

Fig. 1. Cell Block.

Each cell operates independently in one of two modes at any time: either data/normal operational mode (D-mode) or configuration/control mode (C-mode). The memory of each cell specifies the output behaviour in response to every possible combination of input values. If all control(C) inputs are zero, the cell is said to be in D (Data-) mode otherwise it operates in C-mode. When a cell is in D mode, it is basically a simple combinatorial device whose input to output mapping is specified by the content of its LUT. In C-mode a cell can be read and written synchronously upon a global clock signal. Thus in C-mode, a cell's memory can be examined and modified.

In the architecture there are lots of repeated unit cell blocks and each cell is connected to its neighbour and formed a large matrix. If any cell is faulty, then it is possible to bypass the faulty cell using another routing path through the architecture or that faulty cell can be replaced by a spare or adjacent cell. However, it is necessary first to identify the faulty cell from the outside of the matrix. If any fault occurs in the circuit then observability is required to check the faulty state and a test sequence is required to observe the final state of the circuit. To investigate the architecture in software, VHDL [2], Mentor Graphic [3] and QHSIM simulators were used. This paper presents an analyses of fault tolerance in the architecture and suggests some modifications of its basic structure to enhance fault tolerant circuit design such as using a Scan Path approach and Hamming Code error correction.

II. Redundancy based Fault Tolerance

One way to start consideration of a fault tolerant approach with the architecture is redundancy. Using a redundancy approach, each cell would have a number of associated extra cells for rerouting and reassignment of that cell's function in case of failure. The number of extra cells needed for designing redundancy based fault tolerant

circuits such as adders, multipliers in the architecture were investigated [4]. This concluded that fault tolerant circuit design in the architecture is difficult to achieve using redundancy because the architecture has limited and very rigid communication. If data is sent to any inner cell of a large matrix through a particular row/column, then it is not possible to send data in any other cell in that particular row/column through the same direction in that particular cell. Therefore extra cells would be required to reroute wires. Complex software is needed to program a fault tolerant circuit without modification of the basic structure of cell because there is no direct way to access internal cells from the outside of the matrix and no direct communication between non adjacent cells. The architecture could be modified with additional hardware for fault tolerance. Vertical and horizontal scan path techniques plus Hamming code error correction and error detection circuit designs could be incorporated into a modified architecture. This integration is explained in the following sections.

III Error-correcting Codes

In order to provide automatic fault detection, location and masking, error-correcting codes are used. One of the most widely used is Hamming code [5]. Hamming codes are used in many applications where errors are common including DRAM memory chips and satellite communication hardware. Hamming code can be used as single error detection and correcting codes (SED-SEC) and single error correction and double error detecting codes (SEC-DED). Russo and Meyer [6] also proposed the single fault tolerant sequential circuit implementation, using error-correcting codes. The basic idea of coding is to add check bits to information bits/data bits so that errors can be detected or the original information bits can be reconstructed if errors occur. The process of adding check bits to the data bits is called encoding. The error detecting and correcting capability of a code can be defined in terms of the Hamming distance of a code. The distance between two words is just the number of places in which they differ. The relationship between the Hamming distance of a code and its error detecting and correcting capabilities must follow this mathematical formula [7]:

$$W = C + D + 1 \text{ with } D = C,$$

Where W = Hamming distance of a code, D = Number of bit errors which can be detected, C = Number of bit errors which can be corrected to obtain a correct code. So, to detect and correct single bit error, Hamming code-3 is needed and for double error detection and single error correction distance 4 Hamming code is needed.

A. Single Bit Error Correction

It is possible to correct a single bit error in cell memory by using Hamming code. Fig. 2 shows how the Hamming code circuitry could be incorporated into the basic cell of the architecture. To correct a single bit error in cell memory using Hamming code, the XOR gates and a decoder circuit will have to be added in to the cell structure. Since each cell has four data outputs, three parity check bits are needed to apply the Hamming code method. During the read operation a typical code word will be $DNoDSoDWoP_3DEoP_2P_1$. From this code word, syndrome bits could be generated and any value of syndrome bits apart from zero indicates the location of the particular faulty output. A decoder could be used to correct the faulty output through an XOR gate. The parity check bits would be generated from a workstation which is under software control. The downloaded data (PC) and control signals (CC) are stored in an extended Cell Matrix memory and these cells include the syndrome bit generator circuitry and a decoder circuit as shown in Fig. 2.

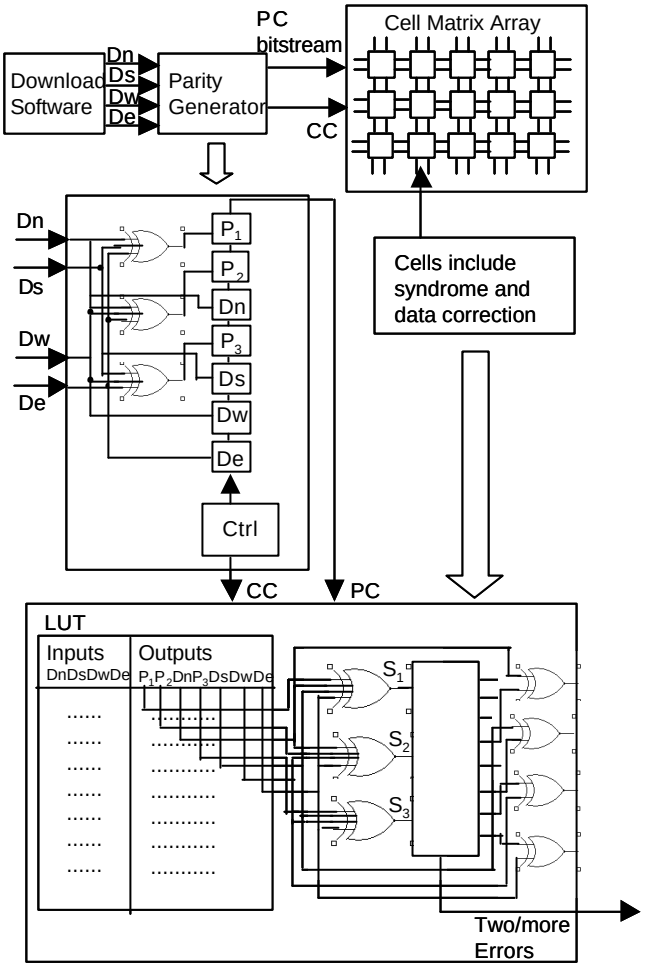

Fig.2. Integration of Hamming error correction circuitry in the basic cell.

IV Scan Path Design Approach

In scan path design techniques the circuit operates in two modes, normal mode and test mode [8]. During normal operation, each cell's response depends on the output of its adjacent cells. In the test mode, all flip-flops in the circuit are connected in a chain to behave as a shift register. This shift register allows input test vectors to be loaded into any cell of the circuit. The test results from the output of the circuit can then be captured and shifted out for inspection. To apply scan path techniques in the architecture, additional hardware could be added to the structure. In the architecture each cell has four data outputs, so four flip-flops and eight two input multiplexers will be needed for each cell in the scan path chain.

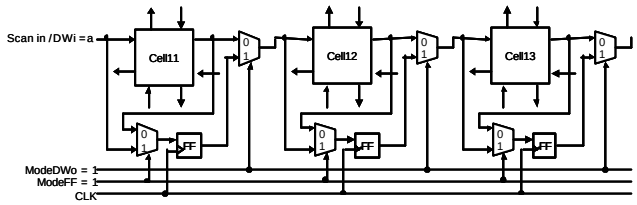

Fig.3(a). Set up to load test data into scanpath

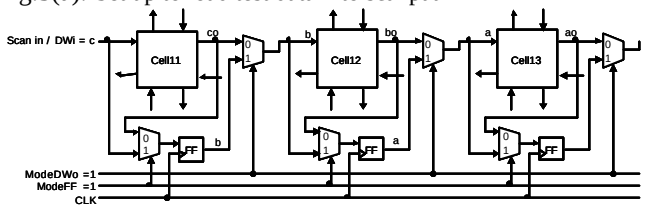

Fig.3(b). Test Mode of scan path after two clocks

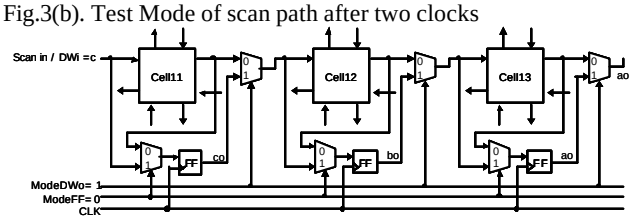

Fig.3(c). Capture the output test results

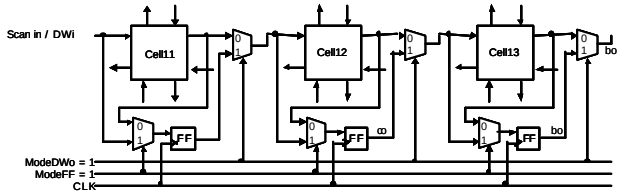

Fig.3(d). Pipe out the test results

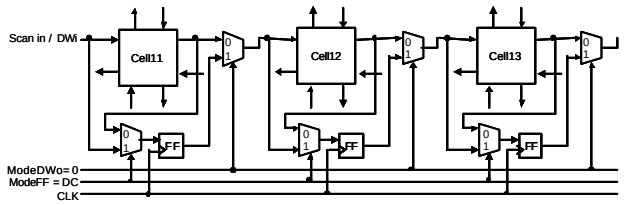

Fig.3(e). Normal operational mode after testing

Fig. 3 shows the scan path operation for a DWi to DWo row in the architecture. Similarly three other scan paths will be needed to complete full scan path in the architecture. Fig. 3 (a) shows the initial condition of the circuit where the control inputs ModeDWo=1 and ModeFF=1 are set such that input test data from the Scan in/DWi input can be loaded into the flip-flops. Fig. 3 (b) shows the state of the circuit after two clock cycles such that test vector *c b a* is loaded into the flip-flops and onto the DWi inputs of each cell. Fig. 3 (c) shows the state of the circuit after the resulting cell under test outputs *co bo ao* have been captured bysetting ModeFF=0 and clocking the flip-flops. The test results are retrieved by setting ModeFF=1 and shifting out the data as shown in Fig. 3(d). Switching ModeDWo=0 restores normal operational mode where the DW inputs and ouputs are directly connected via the multiplexers (Fig. 3(e)).

Fig. 4 shows the vertical and horizontal scan path-testing approaches that could be implemented in the architecture. If any cell is faulty, then it is possible to detect that particular cell by observing scan out data of the rows and columns. Different test vectors will be needed to detect particular faults. Scan flip-flops and multiplexer circuitry has to be assumed to be fault free for this circuit to work correctly.

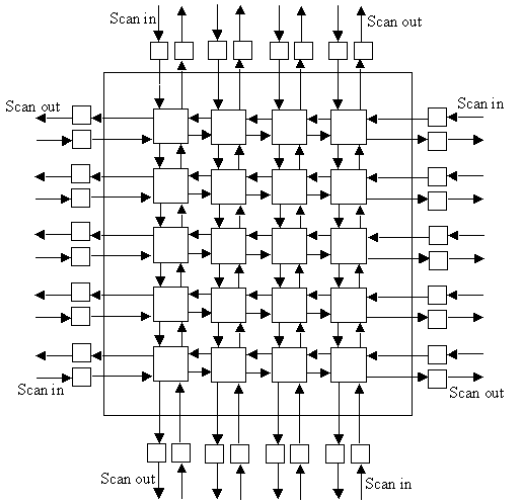

Fig.4. Vertical and horizontal scan test of Cell Matrix

Hamming single bit error correction could be applied together with this scan vertical/horizontal path technique to identify whether a particular cell should be used or not. Seven EXOR gates, a syndrome decoder circuit, five flip-flops and six multiplexers are needed for each cell of the architecture to implement Hamming code together with the vertical/horizontal scan path technique. then it will automatically correct single bit error in each cell's LUT and detect two error bits.

V. Supercells Approach

The Cell matrix Corporation talked about defect tolerance and autonomous fault handling capacity in the architecture using a Supercell approach [9], [10]. In this approach a small region of the array is first used to implement an initial building block called a Supercell. A short configuration bit stream is needed to configure a small Supercell and this bitstream is supplied externally but it is fixed for a given target circuit. The initial Supercell must be fault free then it performs a series of tests on nearby regions of the matrix looking for defective or faulty areas. In the regions that are found to be defect free then the initial Supercell configures additional Supercells and these seceond generation Supercells. This process continues for a fixed number of generations.

Durbeck and Macias present Supercells that require 40 x 40 basic Cell Matrix cells which takes 37000 steps to configure [9] and another version composed of 270 x 270 cells (configuration steps not given) [10]. The advantage of the Supercell network lies in the ability of a Supercell to configure and test a neighbouring supercell and avoid that supercell if necessary. However, on implementing the target circuit after testing the resultant functionality resembles a two-input, one –output functional block which is somewhat similar to that of a single basic cell, thus there is large overhead.

VI. Results

In the basic Cell Matrix architecture, each cell consists of approximately 27 logic gates (LG), a 7 bit address RAM (128 bits), a D-Flip-flop (DFF) and a 7 bit counter (~7 more DFFs). In our scan path design, each cell needs approximately another 32 LGs for the eight multiplexers and four D-FFs. For Hamming Code error correction another 23 logic gates for a syndrome decoder circuit and five DFFs will be needed plus 8 logic gates and 1 DFF to scan out the error indicator bit. Table 1 shows a comparison of the resources and test time needed Supercell approach, scan path and Hamming code error correction.

TABLE I
Comparison of Different Test methodologies.

Approach	**Resources**			**Test Time** (n = no. of cells across)
	LG	**RAM (bits)**	**DFF**	
Cell Matrix	27	128	8	NA
Supercell (40x40)	43200	204800	12800	37000n
Scan Path	32	0	4	2n+1
Hamming	31	48	6	2n+1

From this table, it can be deduced that inclusion of the scan path technique and Hamming code error correction increases the cell size by about 3 times compared to one without fault-tolerance. The test time is significantly much lower than the Supercell approach, although extra stages are required for the original configuration of the cell RAMs before test.

VII. Conclusion

Fault tolerance can be achieved by applying a set of analysis and design techniques to create systems with dramatically improved dependability. As new technologies are developed and new applications arise, new fault tolerance approaches are also needed. Cell Matrix Corporation have published the advanced Supercell approach where a number of cells can be used for self-fault detection. However, this approach has large overhead as 1600 basic cells are required for one Supercell which has the target functionality similar to that which could be achieved by a single basic cell. Also a large number of clock cycles are required for configuring a Supercell (37000) and the initial Supercell configuration must come from outside the matrix. It has been demonstrated in this paper how scan path design techniques and Hamming code could be applied to the architecture with some additional hardware. These techniques could be used to identify a faulty cell and correct single bit error in the LUT's automatically with a significantly smaller overhead in both area and time compared to the Supercell approach. In conjunction with the redundancy techniques discussed it has been shown how a good degree of fault tolerance could be incorporated into the Cell Matrix architecture.

References

[1] www.cellmatrix.com
[2] Stefan Sjoholm and Lennart Lindh, " VHDL for Designers", Prentice Hall Europe 1997.
[3] http:/www.mentor.com
[4] C. R. Saha, A. Mathewson and S. Bellis "On Fault Tolerance in a Reconfigurable Architecture",Irish Signal and System Conference, pp. 272-277, July 2003.
[5] Paterson, W.W. and E.J. Weldon, " Error Correcting Codes", MIT, Press 1972.
[6] Lin. S, " An Introduction to Error Correcting Codes", Prentice Hall, 1970.
[7] Davies, D. and J.F. Wakerly " Synchronization and matching in redundant systems" IEEE Transaction computer, 531-539, June 1978.
[8] H. Fujiwara, "Logic Testing and Design for Testability" MIT press 1985.
[9] L. J.K. Durbeck " Defect-tolerant, fine grained parallel testing of a Cell Matrix", Cell Matrix Corporation, 1004 Palmer Drive, Blacksburg. www.cellmatrix.com
[10] N. J. Macias and L. J. K. Durbeck " Self-Assembling Circuits with Autonomous Fault Handling", Cell Matrix Corporation, 1004 Palmer Drive, Blacksburg. www.cellmatrix.com