Fault Tolerant System Design using Evolved Virtual Reconfigurable Circuit

*D.Dhanasekaran, **K.Boopathy Bagan and ***S.Ravi

*Assistant Professor, SVCE, Pennalur,Sriperumbudur-602105.
**Assistant Professor, Madras Institute of Technology, Chrompet, Chennai-44
***Professor, ECE Department, Dr.M.G.R.Deemed University, Chennai-95.

ABSTRACT

A majority of applications require cooperation of two or more independently designed, separately located, but mutually affecting subsystems. In addition to good behavior of each of the subsystems, an effective coordination is very important to achieve the desired overall performance. However, such a co-ordination is very difficult to attain mainly due to the lack of precise system models and/or dynamic parameters. In such situations, the evolvable hardware (EHW) techniques, which can achieve the sophisticated level of information processing the brain is capable of, can excel. In this paper, a new virtual reconfigurable circuit based sensor validation scheme using the techniques of evolved operators is presented. The idea of this work is to develop a system that is tolerant to sensor failures (fault tolerance) by utilizing multiple sensor inputs connected to a programmable VLSI chip. The approach chosen here is based on functional level evolution whose architecture contains many nonlinear functions and uses an evolutionary algorithm to evolve the best configuration. The system is tested for its effectiveness by choosing a real-time process control plant with three input sensors and introducing different sensor failures such as: sensor fails as open circuit, sensor fails as short circuit, noise added to individual sensors, multiple sensor failure etc.. In each case the mean square error is computed and used as the performance index.

Keywords: Virtual Reconfigurable circuit, Evolvable hardware, Sensor validation.

1. INTRODUCTION

On systems that perform real-time processing of data, performance is often limited by the processing capability of the system. Providing a high processing speed is therefore often a crucial factor to be considered when implementing real-time systems. Also there is a need to have flexible systems that can be changed according to new specifications. Systems based on software are flexible, but often suffer from insufficient processing capability. Alternately, dedicated hardware can provide the highest processing performance, but is less flexible for changes. Reconfigurable hardware [1] devices offer both the flexibility of computer software, and the ability to construct custom high performance computing circuits. Thus, in many cases they make a good compromise between software and hardware solutions. The structure of a reconfigurable hardware device can be changed any number of times by downloading into the device a software bit string called configuration bits. Field Programmable Gate Arrays (FPGA) and Programmable Logic Devices (PLD) are typical examples of reconfigurable hardware devices. FPGAs are hardware devices whose architecture can be determined by downloading a binary string, called architecture bits.

In this paper, a fault tolerant system using virtual reconfigurable circuit and evolved operators designed by coordinating the data from multiple sensors, (that are commonly present in a complex system for monitoring various states and environmental conditions) is used, to handle exceptions such as sensor faults or extreme situations incorrectly handled by the less sophisticated conventional systems. The proposed method finds application mainly in the area of sensor validation, control engineering and other related fields to estimate the true variations of the signal during the failure period of a sensor.

2. EVOLVING CIRCUITS

Evolvable Hardware (EHW) is a new concept in the development of online adaptive machines. In contrast to conventional hardware where the structure is irreversibly fixed in the design process, EHW is designed to adapt to changes in task requirements or changes in the environment through its ability to reconfigure its own hardware structure online and autonomously [2]. The capacity for adaptation is achieved through evolutionary algorithms such as Genetic Algorithm (GA).

Virtual reconfigurable hardware is the combination of Genetic Algorithms and the software reconfigurable devices. The structure of the reconfigurable device can be determined by downloading binary bit strings called the architecture bits [1]. The architecture bits are treated as chromosomes in the population by the GA, and can be downloaded to the reconfigurable device resulting in changes to the hardware structure. The modified functionality of the device can then be evaluated and the fitness of the chromosome is calculated. The performance of the device is improved as the population is evolved by GA according to fitness. This process is repeated till the desired performance is achieved or particular number of generations is reached. To design conventional hardware, it is necessary to prepare all the specifications of the hardware functions in advance. In contrast to this, VRC can reconfigure itself without such specifications and can be considered as an online adaptive hardware. The basic concept of evolving circuits is shown in figure 1. The GA implementation configures the evolving design by placing individuals in Random Access Memory (RAM). The fitness value is calculated by the GA from the feedback signals originating in the evolving design. Since the GA and the evolving design are implemented on the same chip, the evolution process may continuously observe the evolving design.

The proposed architecture has many advantages over traditional hardware and software systems. First, it can automatically improve its performance by changing its hardware configuration according to the GA. Second, the reconfigurable devices can change their functionality in an on-line fashion during execution.

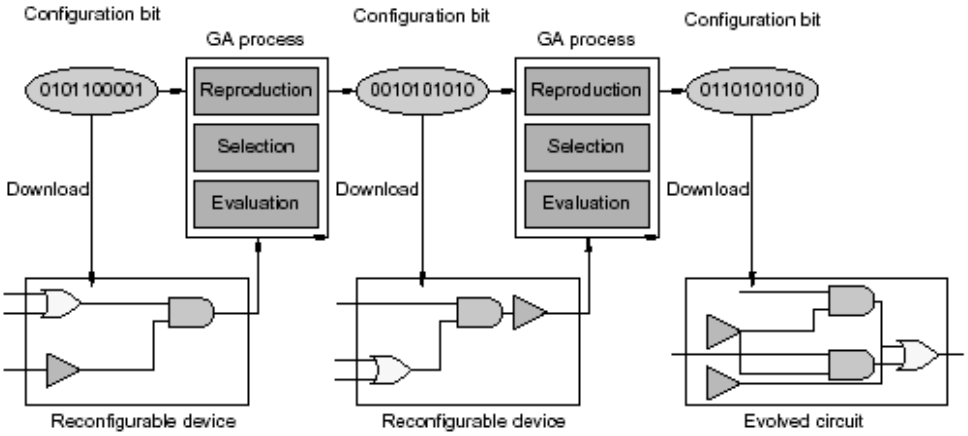

Figure 1 Basic concept of evolving hardware configurations

3. GENETIC ALGORITHM

Genetic Algorithm [6] determines how the hardware structure should be reconfigured whenever a new hardware structure is needed for a better performance. GA was proposed to model adaptation of natural and artificial systems through evolution, and is well known as one of the most powerful search procedures. The canonical GA has a population of chromosomes; each of them is obtained by encoding a point in the search space. Usually, they are represented by the strings of binary characters. The sequence of operations performed by the GA is shown in Figure 2. At an initial state, chromosomes in the population are generated at random, and processed by many operations, such as evaluation, selection, crossover and mutation. The latter three operations are called the genetic operations, and one cycle of the evaluation and the genetic operation is counted as a generation. The evaluation assigns the fitness values to the chromosomes, which indicates how well the chromosomes perform as solutions of the given problem. According to the fitness values, the selection determines which chromosomes can survive into the next generation. The crossover chooses some pairs of chromosomes, and exchanges their sub-strings at random. Finally, the mutation randomly picks some positions in the chromosome and flips their values. The major advantages of GA are its robustness and superior search performance in problems without a prior knowledge.

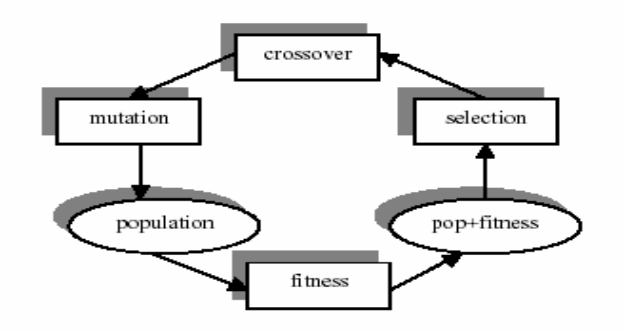

Figure 2 Operations in a Genetic unit

4. VIRTUAL RECONFIGURABLE CIRCUITS IN FPGA'S

Although various evolvable systems have been implemented as Application Specific Integrated Circuits (ASIC), this solution is relatively expensive [5]. Hence a great effort is invested to designing evolvable systems at the level of FPGAs. These solutions can be divided into two groups:
(i) FPGA is used for evaluation of circuits produced by evolutionary algorithm, which is executed in software.
(ii) The whole evolvable system is implemented in the FPGAs. This type of implementation integrates a hardware realization of evolutionary algorithm and a reconfigurable device. The typical feature of these approaches is that the chromosomes are transformed to configuration bit stream and the configuration bit stream is uploaded into the FPGA. Most families of FPGAs can be configured externally (i.e. from an external device connected to the configuration port).

The approach utilizing VRC offers many benefits, such as

1). It is relatively inexpensive, because the whole evolvable system is realizable using an ordinary FPGA.
2). The architecture of the reconfigurable device can be designed exactly according the needs of a given problem.
3). Since, the complete evolvable system is available at the level of Hardware Description Language (HDL) code it can easily be modified and synthesized for various target platforms (FPGA families).

The VRC consists of 8 programmable elements realized on top of an ordinary FPGA. Slices have to implement a new array of programmable elements, new routing circuits and new configuration memory. The virtual circuit can be configured internally or from FPGA's I/O pins if new configuration memory is connected to them. The VRC is shown in figure 3.

The main advantage of the VRC is that the array, the routing circuits and the configuration memory can be designed exactly according to the requirements of a given application. Furthermore, style of reconfiguration and granularity of new virtual reconfigurable circuit can exactly reflect the needs of a given application.

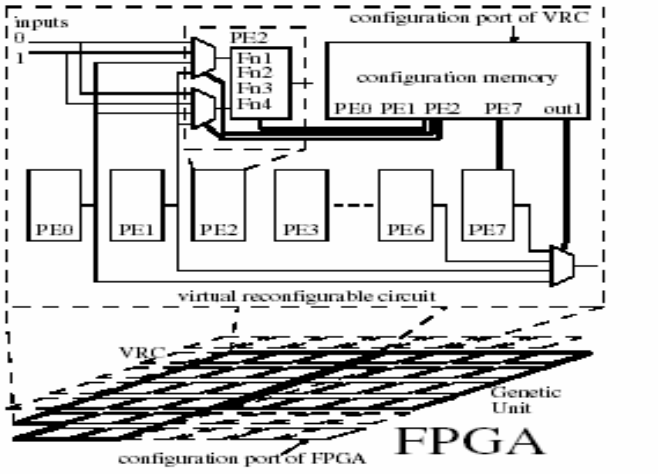

Figure 3 VRC module used in EHW

5. DETAILS OF EHW CHIP

The proposed EHW chip along with the VRC unit is shown in figure 4. It consists of the inputs routed through the communication component module. The configuration memory in the VRC is controlled by the genetic unit. The failure detection mechanism detects the failure of a sensor and accordingly the VRC is reconfigured. The evolvable system is composed of basic modules; input buffer, virtual reconfigurable circuit, pseudo random number generator, population memory, selection unit, mutation unit, fitness evaluator and output buffer and is shown in Figure 5. Both the Genetic unit and the Virtual reconfiguration unit reside in the chip and hence the configuration is complete. The configuration word contains details about the interconnection between the processing elements (PE) of the VRC and the functional operations performed within each PE. In this work, the interconnection between the PE's is not restricted to its nearest neighbors.

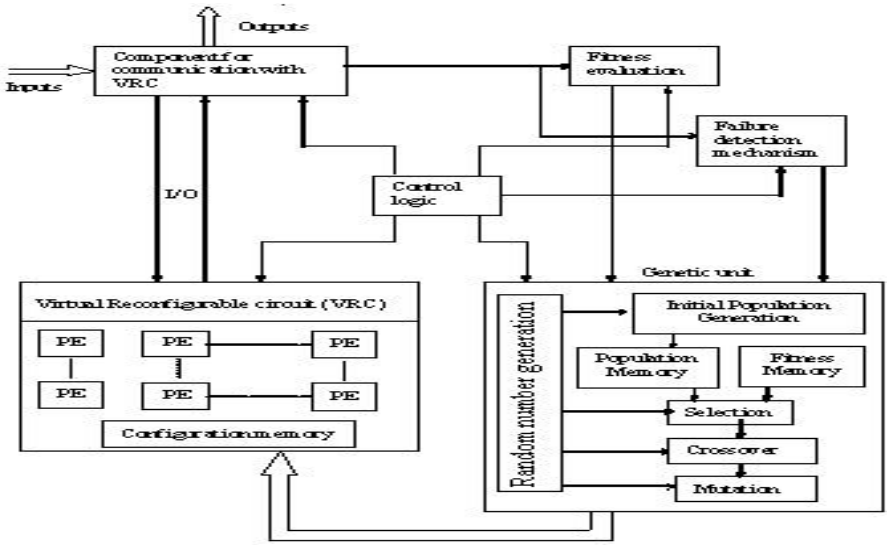

Figure 4 Proposed evolvable hardware chip

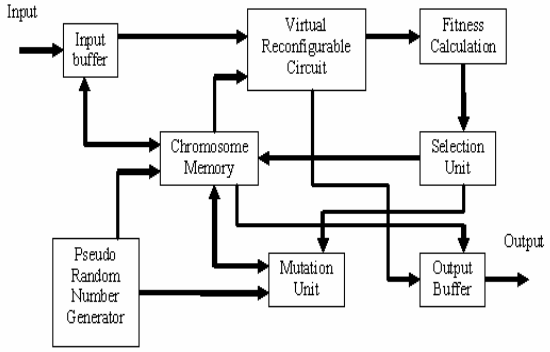

Figure 5 Block diagram of the VRC and the Genetic unit

5.1 Pseudo Random Number Generator

The Pseudo Random Number Generator (PRNG) [4] is used in two of the major steps in GA. Firstly, during initial population creation, and secondly to select individuals for crossover and mutation. One of the most common PRNG for FPGA implementation is a Linear Feedback Shift Register (LFSR). In this work, a word size of 12 is chosen. It is important to choose a good polynomial to ensure that the PRNG can generate a maximal sequence of 2^n-1 random numbers, while keeping the number of taps to a minimum for efficiency. For the chosen 12 bit word the polynomial

$$x^{12} \text{ (xnor) } x^6 \text{ (xnor) } x^4 \text{ (xnor) } x^1$$

is used. The block diagram of the LFSR is shown in Figure 6.

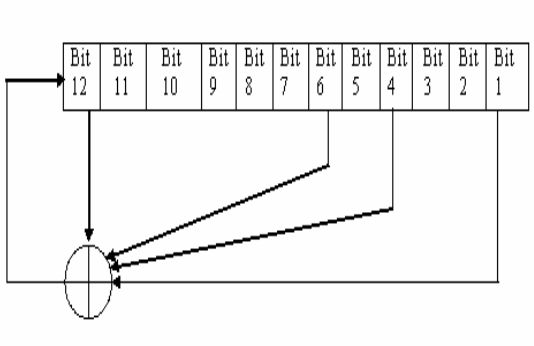

Figure 6 Pseudo random noise generator

The PRNG is designed so that a random number is generated in every clock. The 12^{th} bit is taken as the random bit. Initial seed value is loaded in to the PRNG. To create a 10 bit random number, 10 single bit PRNG are combined in parallel.

5.2 Input Buffer

Input buffer consists of RAM. Original and distorted images are read from the file and stored in the input buffer. During runtime pixels are given as input to the VRC from the input buffer.

5.3 Initial Population Generation

During initial population generation, a 250 bit chromosome is created using 10 bit random number generator in 25 clock cycles. Chromosomes are stored in the Block RAM of FPGA. The initial population size is taken as 16. Totally 16x25 clock cycles are needed for initial population generation.

5.4 Fitness Calculation

MDPP fitness function is used to evaluate the chromosome. The original and filtered images are taken from the memory and the absolute difference between the

corresponding pixel values is added and the fitness is evaluated.

5.5 Selection Unit

Selection unit selects the chromosome which has highest fitness as the best chromosome and it is retained for subsequent generations.

5.6 Mutation Unit

The chromosome which has highest fitness is selected for mutation. Using PRNG, bit by bit mutation is done for the creation of Childs. Fifteen new Childs are created in every generation and stored in the population memory.

5.7 Output Buffer

After the specified number of generations the evolution is completed and the best chromosome is stored in the memory. The fitness value and filtered image signal are calculated and stored in the output buffer.

6. PROGRAMMING THE PE'S IN VRC

The logical configuration of the circuit is defined by a set of 250 bits, 10 bits for each one of the 25 PEs in the reconfigurable architecture as shown

111 000 0000 010 001 1100 110 010 0011 111 011 0010 110 111 0100 …… 101 000 1000 110 010 1010

The first six bits of each ten bits represent the source of inputs to the PE (sel1& sel2). The other four bits of each ten bit (sel3) indexes the function from Table 1 to be applied by the PE. The configuration word contains details about the interconnection between the processing elements (PE) of the VRC and the functional operations performed within each PE. In this work, the interconnection between the PE's is restricted to its nearest two neighbors. The configuration word is updated the moment a sensor failure is detected. The genetic unit controls the configuration word in the VRC module. The genetic unit is programmed to give the best chromosome and using this, the initial configuration of the VRC is chosen. The reconfiguration of the circuit is required once a sensor failure is detected by the failure detection mechanism. In the present work, each PE except the first stage is assumed to receive inputs from any of the previous two stages. A total of 25 PE's is used in the VRC.

The logical configuration of the circuit is defined by a set of 25 integer triplets, one for each of the 25 PEs in the reconfigurable architecture. The first two integers of each triplet represent the source of inputs to the PE (sel1& sel2) and the third integer of the triplet (sel3) indexes the function to be applied by the PE. A total of 16 functions are used in each PE and these are listed in Table-I.

Table 1 List of Functions used in the Processing Element

Function Code	Function	Function Code	Function
0000	X >> 1	1000	X & 0x0F
0001	X	1001	X & 0xF0
0010	~ X	1010	X | 0x0F
0011	X & Y	1011	X | 0x F0
0100	X | Y	1100	Min(X,Y)
0101	X ^ Y	1101	Max(X,Y)
0110	(X+Y)>>2	1110	Y<<1
0111	(X+Y) >>1	1111	X+Y

The two inputs to the PE's are selected using two multiplexers as shown in figure 7. The function performed in each of the PE depends on the status of the three select lines. The routing circuits are implemented using multiplexers. The configuration memory is composed of flip-flops. All bits of the configuration memory are connected to multiplexers that control routing and selection of functions in PEs.

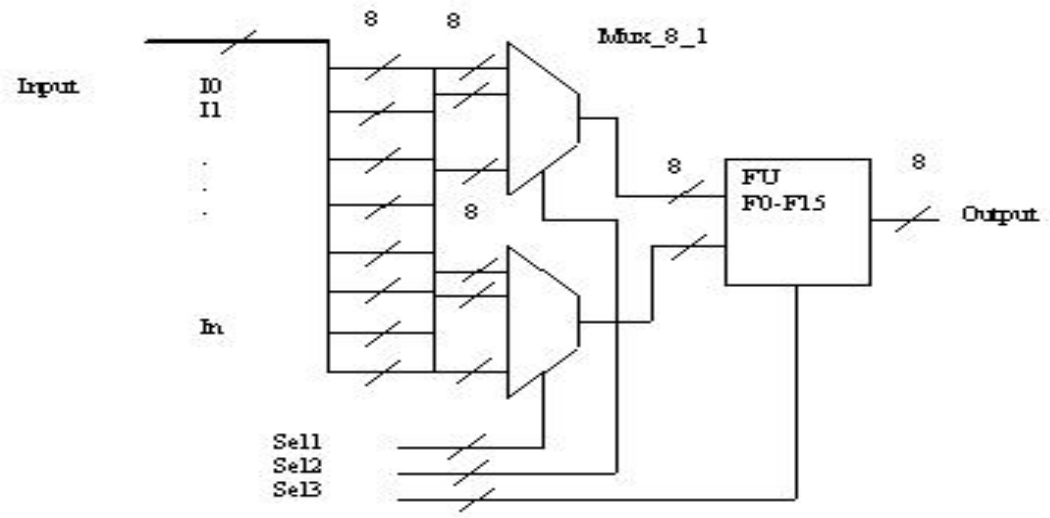

Figure 7 Selection of PE's using MUX logic

7. DISCUSSION OF RESULTS

Case study –I: All three sensors are faultless and noiseless

The results obtained by using the evolvable hardware chip on a real-time plant for this case are shown in figure 8. The first three plots show the variations of the three faultless sensors. The fourth plot is the average of the three sensor output and the last plot is the output of the reconfigured VRC.

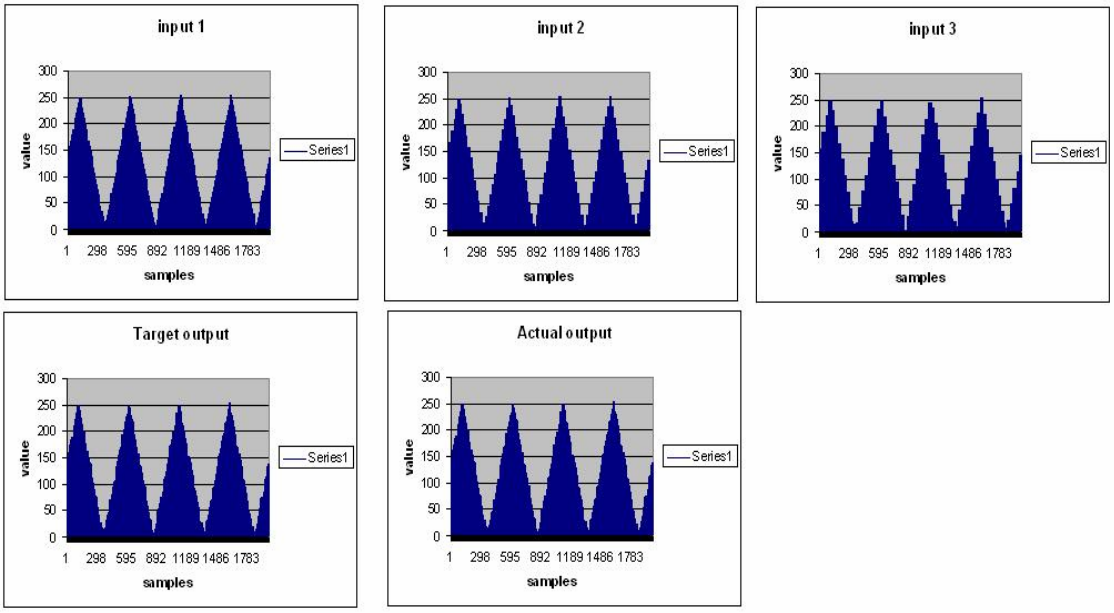

Figure 8 Experimental results obtained by using the EHW chip

Case study –II First Sensor input is noiseless and noise is added to sensor 2 and 3

The results obtained by using the evolvable hardware chip on a real-time plant for this case are shown in figure 9. The ability of the circuit to reconfigure itself and filter the noise is demonstrated in this study. Gaussian noise was added to the sensors 2 and 3 and the VRC output was found to match the required output.

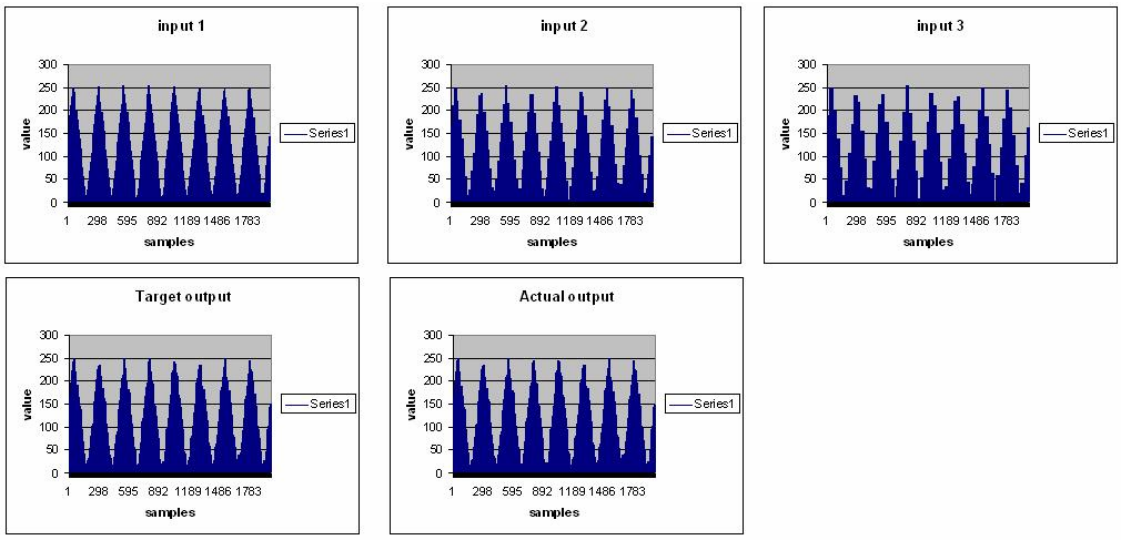

Figure 9 Experimental results obtained by using the EHW chip with noise added to sensors 2 and 3 alone

Case study –III Sensor 3 is open circuit and sensors 1 and 2 are faultless

The VRC output captured using the Model-Sim package corresponding to this condition is shown in figure 10. The VRC has reconfigured itself to reject the sensor 3 value and gives an output which closely matches with the average of the sensor 1 and 2 readings.

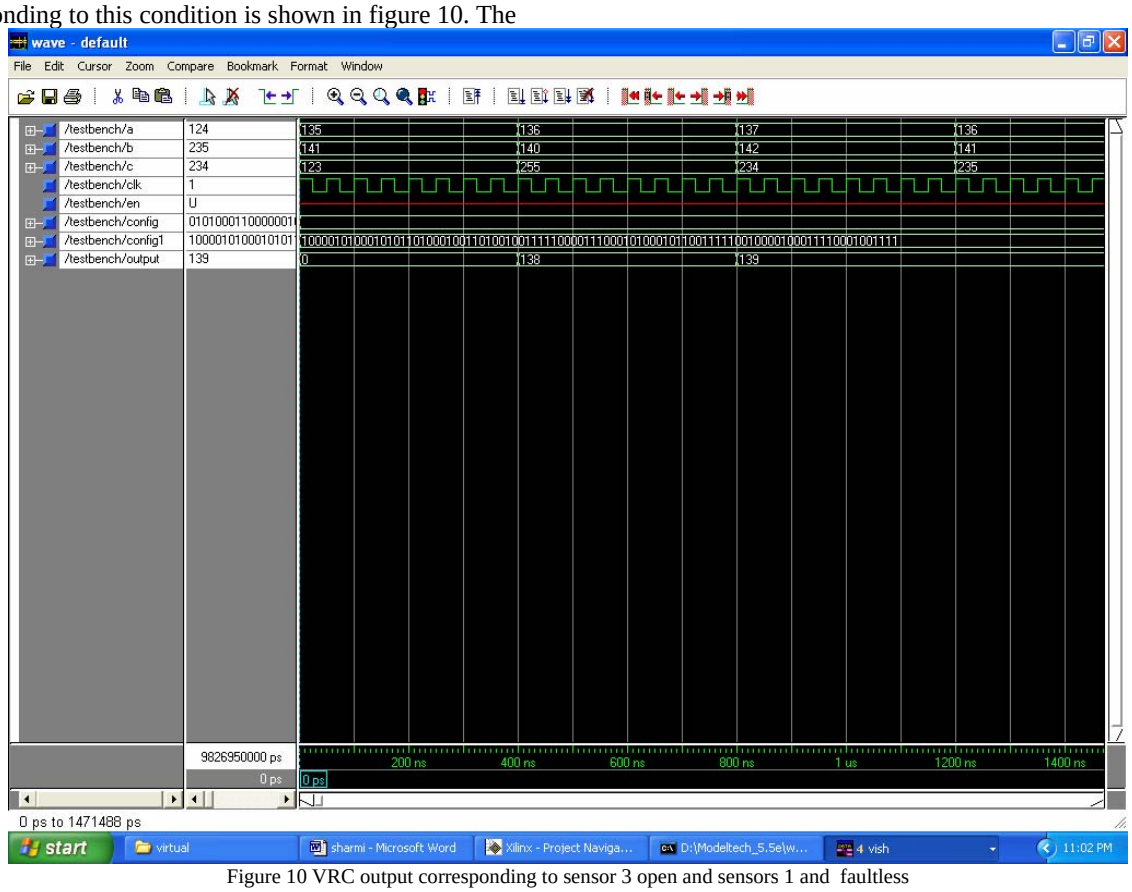

Figure 10 VRC output corresponding to sensor 3 open and sensors 1 and faultless

The performance of the evolved chip in each of the case is measured in terms of the MSE value and is given by

$$MSE = \sum_{k=1}^{N} [\text{actual output}(k) - \text{target output}(k)]^2$$

where actual output(k) and target output(k) are the VRC circuit output and ideal output respectively and N is the total number of samples.

9. CHIP IMPLEMENTATION RESULTS

The evolved filter is the result of the evolution of an array of 4x6 PEs with one PE at the output. Number of generations is 3000. The coding was done in VHDL and simulations were performed in ModelSim 6. The hardware evolution took 2 minutes on Xilinx Virtex FPGA xcv800 running at 49 MHz. This compares favorably with software simulations run previously which it took approximately 6 hours (Pentium III/800 MHz system) to achieve the best chromosome, the speed has been increased by 180 times and the evolution time has been greatly reduced. Number of generations was taken as 3000. Hardware evolution took 12 minutes in Xilinx VirtexE FPGA xcv2000e. This compares favorably with software simulations run previously which took approximately 6 hours to give the best chromosome.

The VRC takes 1754 slices of the Xilinx Virtex FPGA xcv800 (9408 slices) and the whole evolvable system including the GA takes 3204 slices. Since small amount of the resources are only used i.e only 34% of the resources, chromosomes can be operated in parallel and the processing time can be further reduced. The synthesis report obtained is given in Table-2 below.

Table 2 Synthesis Report - Device Utilization Summary
(Population Size = 16, Chromosome Length = 250)

Target information: Vendor: Xilinx, Family: Virtex, Device: v800fg680, Speed: -6, Optimization Goal: Speed		
Number of Slices	3204 out of 9408	34%
Number of Slice Flip Flops	1087 out of 18816	5%
Number of 4 input LUTs	6200 out of 18816	32%
Number of bonded IOBs	79 out of 516	15%
Number of BRAMs	8 out of 28	28%
Number of GCLKs	1 out of 4	25%
Minimum period	20.160ns	
Maximum Frequency	49.603MHz	
Minimum input arrival time before clock	27.706ns	
Maximum output required time after clock	6.887ns	

10. CONCLUSION

The work has presented a novel approach to fault tolerant system design based on the technique of evolvable hardware. FPGA model for the function level evolvable hardware is analyzed and associated with the evolutionary algorithms employed. The evolution time has been greatly reduced by implementing the evolutionary algorithm in hardware. The EHW architecture evolves circuits without a priori information and reconfigures itself and is tolerant to different fault conditions. The EHW system outperforms conventional ones in terms of computational effort (greatly reduced), robustness (improved) and implementation cost (reduced significantly).

11. REFERENCES

1.Gerald R. Clark (1999), "A Novel Function-Level EHW Architecture within Modern FPGAs", Proceedings of the Congress on Evolutionary Computation (CEC 99), IEEE.

2. Hollingworth G, Smith S and Tyrrell A (2000), "Safe Intrinsic Evolution of Virtex Devices", Proceedings of the Second NASA/DoD Workshop on Evolvable Hardware, IEEE, pp. 195-202.

3. Hollingworth G, Smith S and Tyrrell A (1999), "Design of Highly Parallel Edge Detection Nodes using Evolutionary Techniques", Proceedings of the 7th Euromicro Workshop on Parallel and Distributed Processing, IEEE, pp. 35 – 42.

4. Peter Alfke (1996), "Efficient Shift Registers, LFSR Counters, and Long Pseudo- Random Sequence Generators", Xilinx Application Note, XAPP052.

5. Paul Layzell (1999), "Reducing Hardware Evolution's Dependency on FPGAs", Proceedings of the Seventh International Conference on Microelectronics for Neural, Fuzzy and Bio-Inspired Systems (MicroNeuro '99), IEEE, pp. 171-178.

6. David E. Goldberg (1989), "Genetic Algorithms in Search, Optimization and Machine Learning", Pearson Education, Asia.

12. BIOGRAPHIES

1) Mr. D.Dhanasekaran is working as an Assistant Professor, ECE dept. In Sri Venkataswara College Engg. College, Pennalur,Sriperumbudur, affiliated to the Anna university. His areas of interest include Evolvable Computing, reconfigurable computing,VLSI signal processing and neural networks.

2) Dr. K.Boopathy Bagan completed his doctoral degree from Anna university . He is presently working as a professor, Department of Electronics in Madras Institute of Technology, Chrompet, Chennai. His areas of interest include DSP,VLSI signal processing, Genetic Algorithms and evolvable hardware.