ON SUITABILITY OF FPGA BASED EVOLVABLE HARDWARE SYSTEMS TO INTEGRATE RECONFIGURABLE CIRCUITS WITH HOST PROCESSING UNIT

*P.Nirmalkumar, **J.Raja Paul Perinbam, @S.Ravi and #B.Rajan
*Research Scholar, Department of ECE, Anna University, Chennai-25
**Professor,Dept. of ECE, Anna University, Chennai-25
@Professor, Dept. of ECE, Dr.M.G.R.Deemed University, Chennai-95.
#Dept. of ECE, Dr. M.G.R.Deemed University, Chennai-95.

ABSTRACT

Integrating the reconfigurable logic and the host processor can eliminate the communication bottleneck that is present in current custom computing units. This integration is of great advantage if the reconfigurable block consumes less power. Once power optimization of the VRC is possible, the combined system can provide high speed computing with low power consumption. This paper describes experiments conducted to analyze the power consumed by the individual processing elements of a virtual reconfigurable circuit (VRC) according to the functionality performed. The experiment is performed on a model VRC designed to perform sensor validation and automatic functional reconfiguration in case of occurrence of single or multiple sensor faults. The power analysis done in this work will assist to estimate how the use of VRC's influence the integration of FPGA based evolvable systems with host processor and can facilitate reconfigurable computing to enter the mainstream and provide high performance benefits.

Keywords: Virtual Reconfigurable circuit, Power Analysis, Evolvable hardware.

1. INTRODUCTION

Reconfigurable hardware [1] devices offer both the flexibility of computer software, and the ability to construct custom high performance computing circuits and make a good compromise between software and hardware solutions. Virtual reconfigurable circuits were introduced for digital evolvable hardware as a new kind of reconfigurable platform utilizing conventional FPGAs. Virtual reconfigurable circuit (VRC) is, in fact, an implementation of a domain-specific reconfigurable circuit on top of an ordinary FPGA. 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. A VRC can be designed to exactly fit the needs of a given evolvable hardware-based application. Also, reconfigurable systems have provided significant performance improvements by adapting to computations not well served with current processor architectures. The inherent redundancy present in the VRC can protect the circuits from faults.

On the pessimistic side, the implementation of VRC's are relatively expensive in terms of gates used since interconnection circuits of VRC's are selected using multiplexers which are area expensive. Rather than only optimize the speed, optimizing the power consumed by the PE's in the VRC is also a crucial factor to be considered when integrating reconfigurable circuits with host processor. This paper is organized as follows: Section 2 provides an overview of the evolutionary design of digital circuits. Section 3 describes the details of the model VRC designed using evolved operators to handle exceptions such as sensor faults. Section 4 discusses the implementation of the PE's using the tanner software tool. The experimental results and discussions showing the power consumed by the different PE's as well as the overall VRC corresponding to the different sensor failure conditions are presented in Section 5.

2. EVOLVING DIGITAL 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). The main requirement in an evolved digital circuit is that the evolved circuit should have the ability to control the granularity of configurable elements and provide a transparent structure of the configuration data.

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. The typical feature of these approaches is that the most families of FPGAs can be configured externally (i.e. from an external device connected to the configuration port).

The evolved circuit shall be evaluated either using software simulation models or entirely in hardware. Accordingly the evolution process is classified as extrinsic or intrinsic evolution respectively. In both cases, the evolution process itself is carried out in software. Alternately, it is also possible to have the evolution process itself done on hardware and this is called as complete evolution. The advantage of the last one is that it provides a speed up in the process. An evolved circuit can either be of gate level evolution or functional evolution. For gate level evolution, the gene is considered as an AND, OR, XOR, NAND, NOR or XNOR gate. For a functional level evolution, the gene is considered as a 'm' input LUT. A functional level evolved VRC using GA based techniques for handling sensor faults in a process using multiple sensors is proposed and examined in the next section.

3. VIRTUAL RECONFIGURABLE CIRCUITS

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]. When a VRC is uploaded into the FPGA, its configuration bit stream will create the following units at specified positions: (i) an array of programmable elements called the PE (ii) a programmable interconnection network (iii) a configuration memory (implemented as a register array) and (iv) a configuration port. The basic idea of the VRC with the evolved circuit is shown in figure 1. The VRC shown can be described in HDL and can be synthesized using common synthesis tools and for various target platforms. In this work, the following evolvable systems are implemented using the idea of VRC in an FPGA.

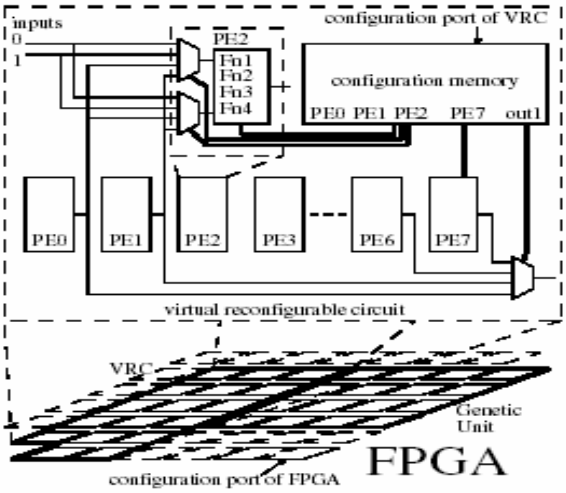

Figure 1 Basic VRC model

(i) Evolvable circuit to detect single sensor failure
(ii) Evolvable circuit to detect multiple sensor failures
(iii) Evolvable circuit to filter the noise present in the input sensors

The chromosomes are transformed into configuration bit stream and the configuration bit stream is uploaded into an SRAM-based FPGA. The evolved circuit along with the VRC, Genetic unit and the host processing units to perform the estimation and failure detection mechanism is shown in figure 2. The approach utilizing VRC offers many benefits, such as 1) It is relatively inexpensive, because the whole evolvable system is realizable using an FPGA. 2) The architecture of the reconfigurable device can be designed exactly according to the needs of a given problem. Slices have to implement a new array of programmable elements, new routing circuits and new configuration memory. The hypothetical VRC chosen for experimental study is shown in figure 3 and consists of 25 PEs, having 3 inputs and 1 output and the behavior

fully defined using 217 configuration bits of SRAM.

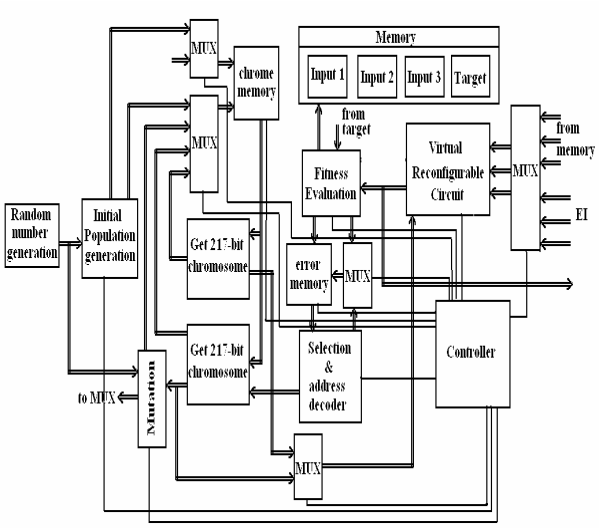

Figure 2 Block Diagram of EHW with VRC

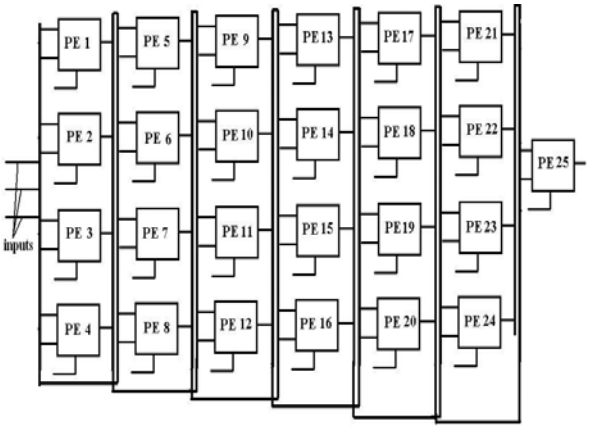

Figure 3 Model VRC with 25 PEs

Each of the PEs can be configured to operate with any of the 13 functions provided in table-1.

Table 1

F0: 0000	X<<1
F1: 0001	~X
F2: 0010	X | Y
F3: 0011	X^Y
F4: 0100	(X+Y) >> 2
F5: 0101	(X+Y) >> 1
F6: 0110	X & “F0”
F7: 0111	X | “F0”
F8:1000	X | “0F”
F9:1001	Min(X,Y)
F10:1010	Max(X,Y)
F11:1011	X >> 1
F12:1100	X+Y

4. IMPLEMENTING THE PE’S IN VRC

The configuration word contains details about the interconnection between the processing elements (PE) of the VRC and the functional operations performed within each PE. The inputs of a PE can be connected to circuit inputs or to the outputs of preceding PEs. However, only up to 8 combinations are permitted in order to reduce the number of configuration bits. While 8 configuration bits define the complete behavior of the PEs of the 1st column (PE0-PE3), 10 configuration bits define the behavior of the remaining PEs (PE4-PE25). It is evident that both combinational and sequential functions can be created in the model VRC. The reconfiguration of the circuit is required once a sensor failure is detected by the failure detection mechanism. 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. 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. This is shown in figure 4 for PE2.

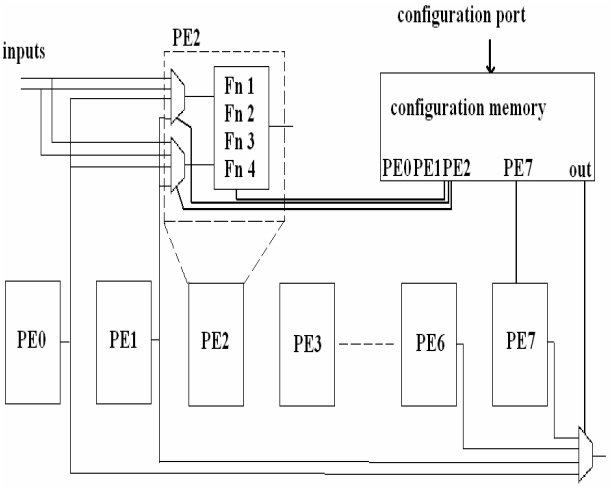

Figure 4 PEs with the MUX and function selector

4.1 *Tanner* Simulator

In this work the implementation of the PEs of the hypothetical VRC in an FPGA is simulated using the Tanner simulator tool. The advantage of this approach is that the reconfigurable circuit can be made available as a soft IP core i.e. the model VRC can easily be removed or modified from or on FPGA. Once simulated it is possible to (i) know the power consumed by both the individual PEs and the complete VRC model and (ii) develop effective methods to improve fault tolerance in FPGAs and recover the functionality by means of a smart reconfiguration strategy.

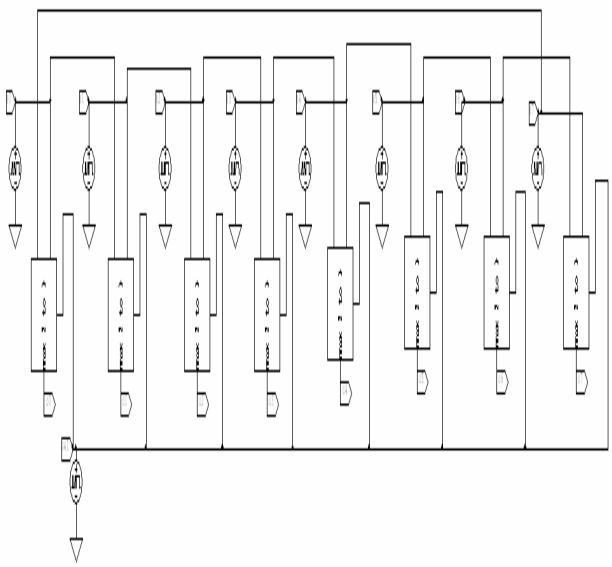

Figure 5.1 First input left shift by 1 function

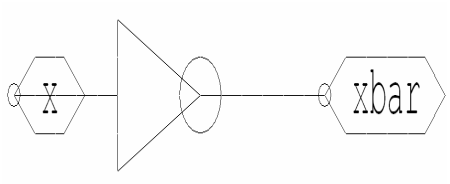

Figure 5.2 Inverter Function

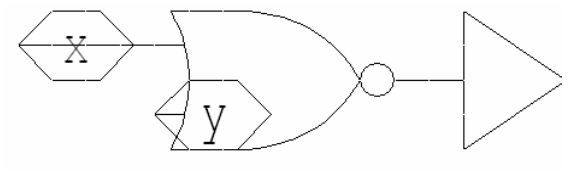

Figure 5.3 OR Function

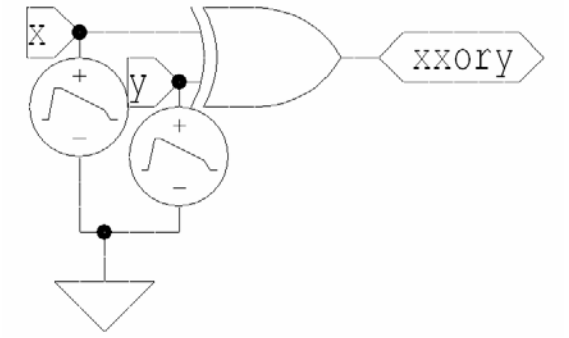

Figure 5.4 Ex-OR Function

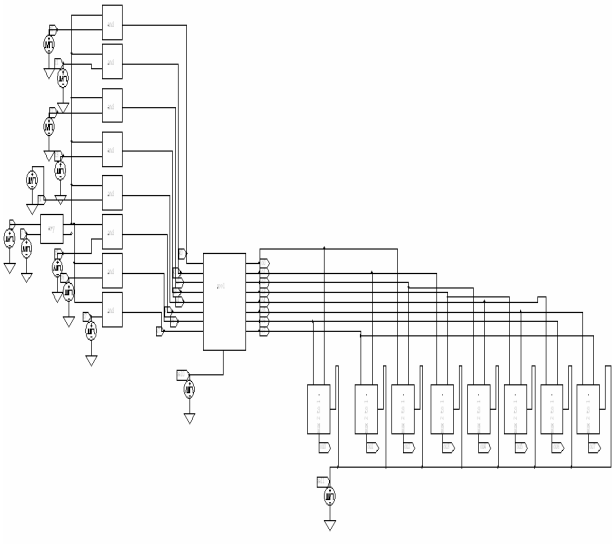

Figure 5.5 Adder + Right shift by 2 function

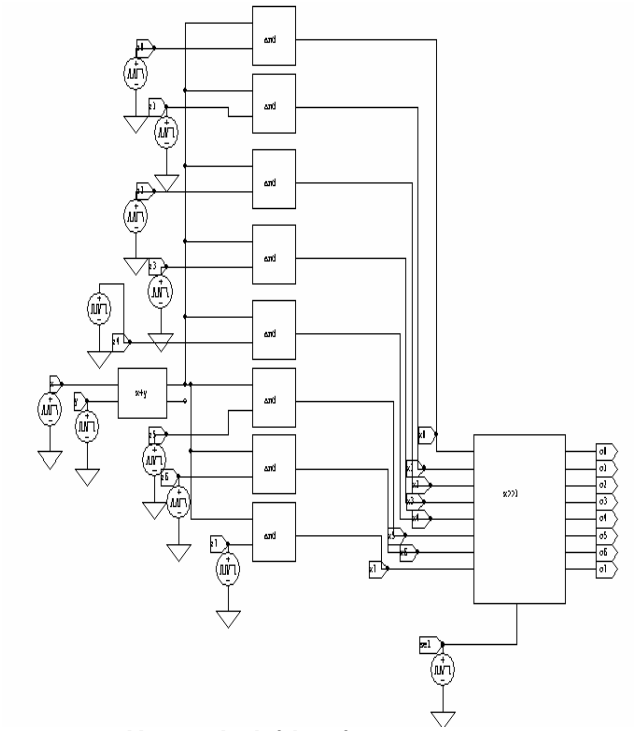

Figure 5.6 Adder + Right shift by 1 function

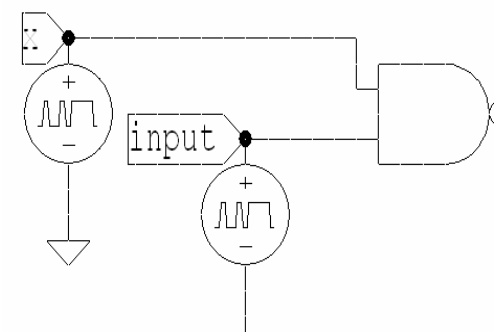

Figure 5.7 First input ‘AND’ with 0xF0 function

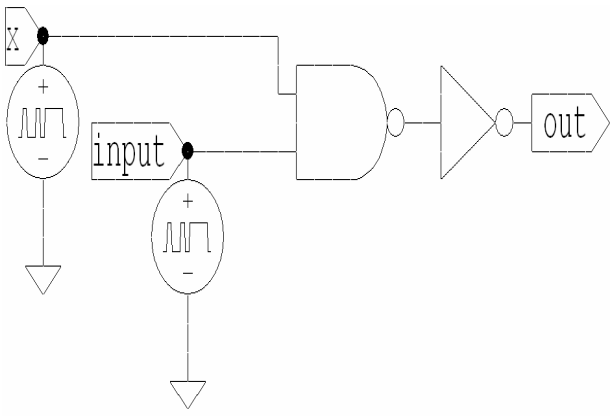

Figure 5.8 First input ‘OR’ with 0xF0 function

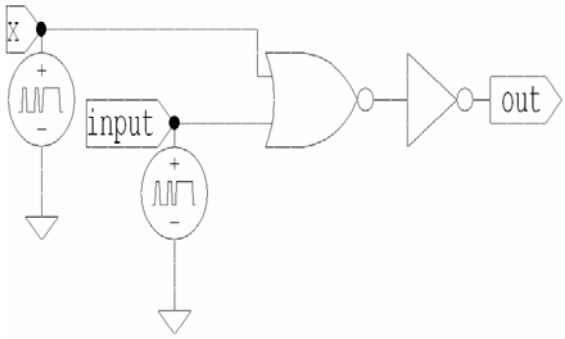

Figure 5.9 First input ‘OR’ with 0x0F function

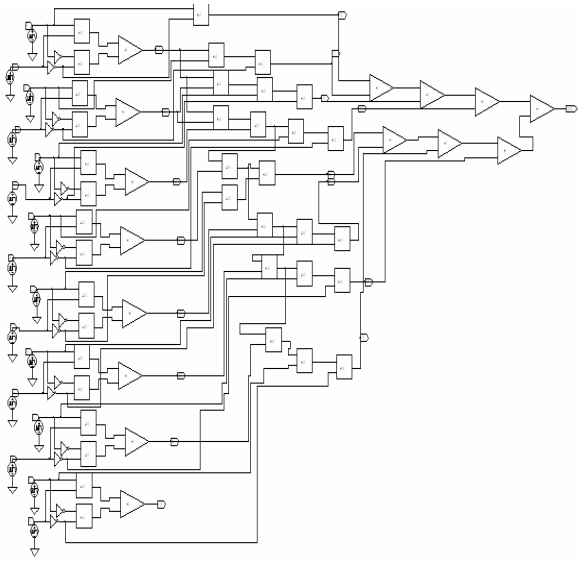

Figure 5.10 Minimum Of 2 input function

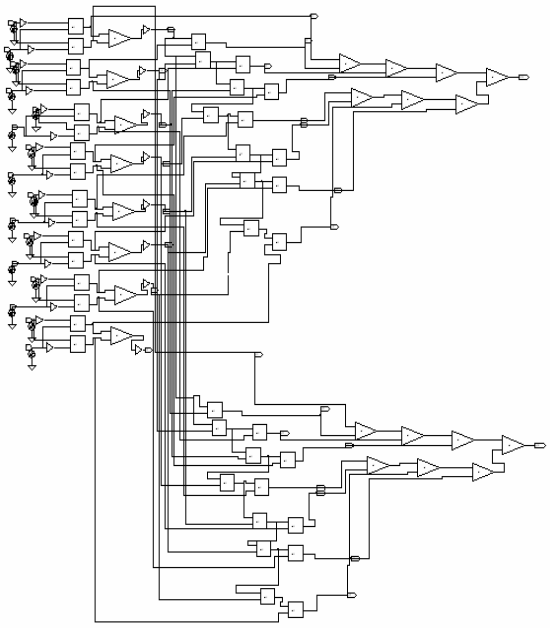

Figure 5.11 Maximum/minimum of two inputs function

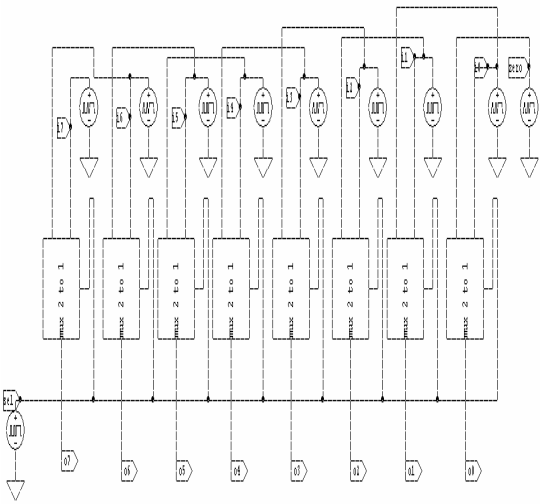

Figure 5.12 First input right shift by1 function

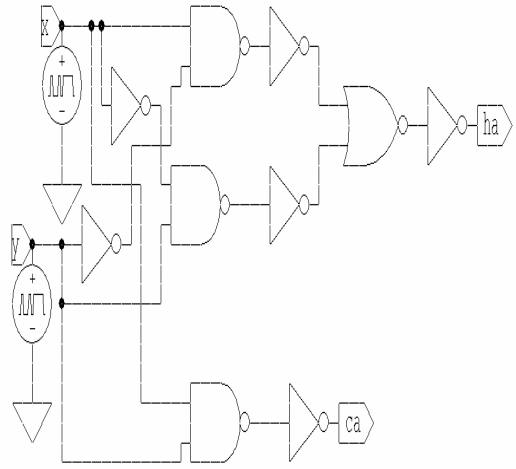

Figure 5.13 Adder Function

The different functions that each PE can perform are simulated in the *Tanner* package tool and are shown in figures 5.1 to 5.13.

5. EXPERIMENTAL RESULTS

Case–I: Circuit evolved to handle single sensor failure (Sensor 3 is assumed to be at fault)

The power consumed by the VRC chip configured to handle single sensor failure is shown in figure 6. The VRC has reconfigured itself to reject the sensor 3 reading and give an output, which closely matches with the average of the sensor 1 and 2 readings.

Case–II: Circuit evolved to take care of multiple sensor failure (Sensors 1 and 2 are assumed to be faulty)

The result obtained by using the evolvable hardware chip on a real-time plant for this case is shown in figure 7. The VRC has reconfigured itself to reject the fault reading of sensors 1 and 2 and give an output, which closely matches with the sensor 3 reading.

Case–III: Circuit evolved to filter the noise present in the input sensors

The VRC output captured using the Tanner tool corresponding to this condition is shown in figure 8. In this case, a Gaussian noise of mean zero and variance 0.1 was added to all the three sensors. The VRC configures itself to act as a filter and filters the noise present in the three input sensors.

6. CONCLUSION

This work has presented the use of *Tanner* tool to perform an analysis of the power consumed by a model VRC for three different reconfigured architectures. The other functions of the complete EHW chip namely the estimator and the fault decision unit is performed in the host processor (PC). The results presented here can be used to assist the integration of VRC with host processor.

7. References

1. Canham, R. O. and Tyrell. A (2002). Evolved fault tolerance in evolvable hardware. IEEE Congress on Evolutionary Computation. IEEE Computer Society.
2. Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization & Machine Learning. Pearson Education, Inc.
3. Hereford, J., Pruitt, C. (2004). Robust sensor systems using evolvable hardware. Proc. of the 2004 NASA/DoD Conference on Evolution Hardware. IEEE Computer Society.
4. Hollingworth, G., Smith, S. and Tyrell, A. (2000). Safe intrinsic evolution of Virtex devices. In Proc. of 2nd NASA/DoD Workshop on Evolvable Hardware. IEEE
5. Iwata, M., Kajitani, I., Liu, Y., Kajihara, N., Higuchi, T.. Implementation of a Gate-Level Evolvable Hardware Chip. IEEE.
6. Keymeulen, D., Stoica, A., Zebulum, R. and Dong, V. (2000). Results on the Fitness and Populational based Fault Tolerant Approaches using a Reconfigurable Electronic Device. IEEE.
7. Layzell, P. (1999). Inherent Qualities of Circuits Designed by Artificial Evolution: A preliminary study

of populational fault tolerance. In Proc. of the 1st NASA/DoD Workshop on Evolvable Hardware. IEEE Computer Society.
8. Murakawa, M., Higuchi, T., Iwata, M., Kajitani, I., Liu, W., and Salami, M. (1997). Evolvable Hardware at Functional Level. IEEE
9. Ramsden, E., Greenwood, G. W., and Hunter, D. (2004). Fault Recovery in Linear Systems via Intrinsic Evolution. Proc. of the 2004 NASA/DoD Conference on Evolution Hardware. IEEE 1Computer Society.
10. Sekanina, L. (2003). Towards Evolvable IP cores for FPGAs. In Proc. of the 2003 NASA/DoD Conference on Evolvable Hardware. IEEE Computer Society.
11. Sekanina, L., and Friedl, S. (2004). On Routine Implementation of Virtual Evolvable Devices Using COMBO6. Proc. Of the 2004 NASA/DoD Conference on Evolution Hardware. IEEE Computer Society.

8. Biographies

1)Mr.P.Nirmalkumar is a research scholar in the faculty of ECE, Anna Uty. Chennai. His areas of interest include evolvable hardware, fault tolerant systems, genetic algorithms etc.
2)Dr.Raja Paul Perinbam is a Professor in the faculty of ECE, Anna Uty. Chennai. His areas of interest include EHW, embedded systems, low power VLSI etc.
3)Dr.S.Ravi is a Professor in the faculty of ECE, Dr.M.G.R. Deemed Uty. Chennai. His areas of interest include reconfigurable architectures, signal processing, low power VLSI etc.
4)Mr.B.Rajan is in the department of ECE, Dr.M.G.R. Deemed Uty. Chennai. His areas of interest include image processing, ASIC, Evolvable hardware etc.

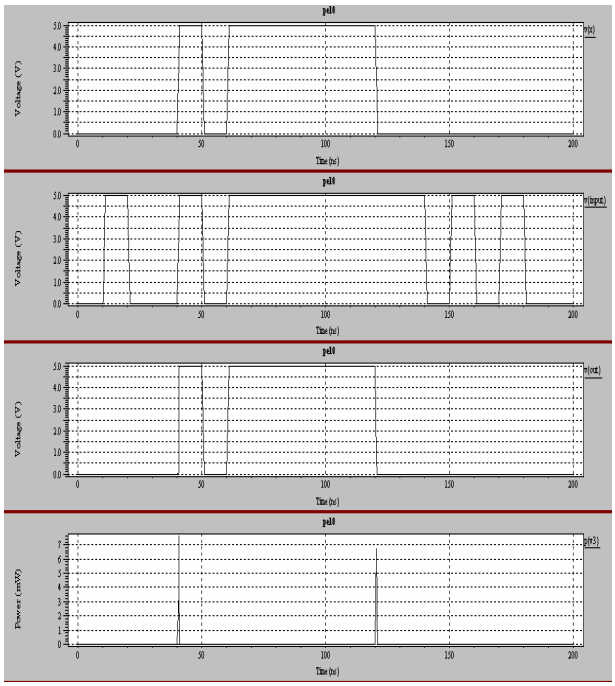

Figure 6 Power consumed by evolved VRC for case (i)

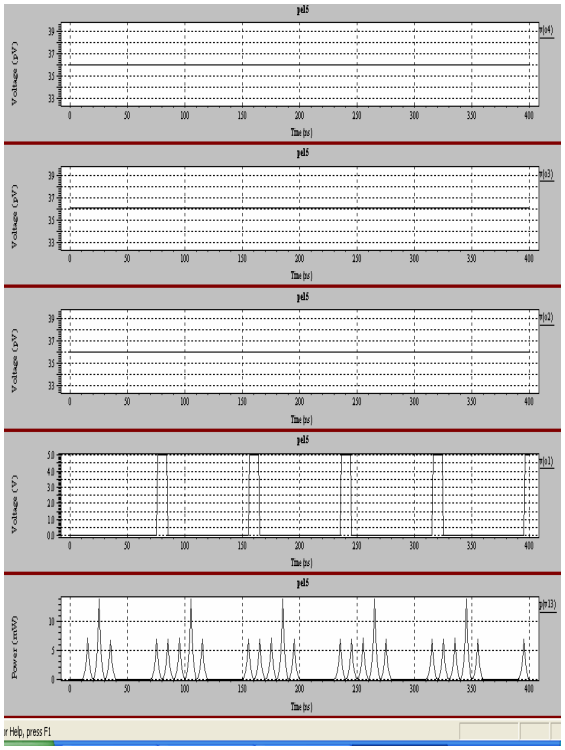

Figure 7 Power consumed by evolved VRC for case (ii)

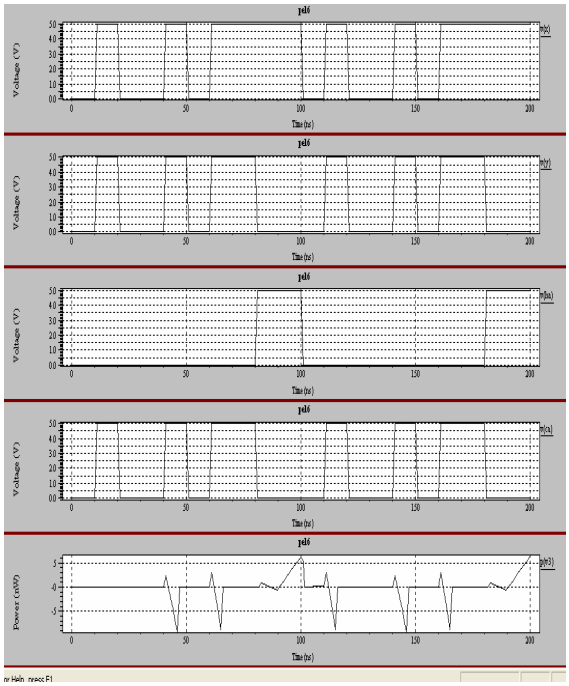

Figure 8 Power consumed by evolved VRC for case (iii)