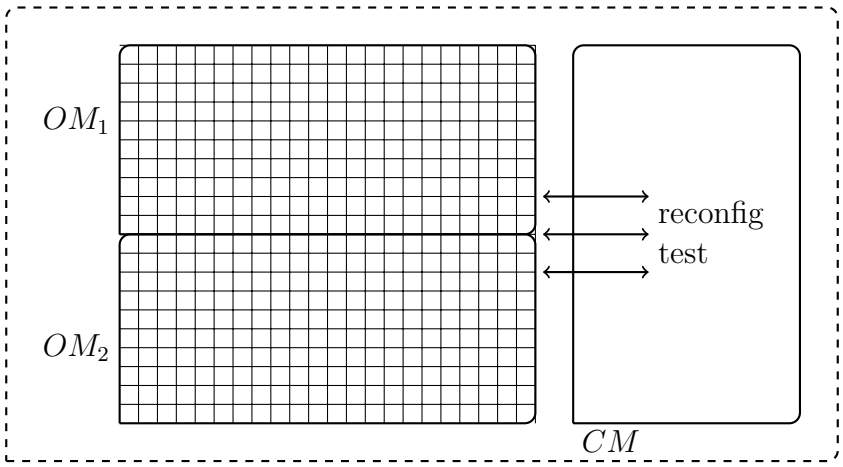
Fault Tolerant Circuits Using Evolutionary Repair Mechanisms

The issue of digital circuit degradation is certainly an important one; as devices become more complex and expensive, the cost of circuit failure is very high. A single "stuck-on" fault can render an entire complex system inoperable. To date, existing methods focus on *detecting* rather than *repairing* these faults. This document proposes a method and framework of fault identification and repair in consumer-oriented markets using a hardware evolution approach. This system results in highly robust and reliable digital circuitry. The targeted platform for initial proof of concept (PoC) demonstrations is Field Programmable Gate Array (FPGA) devices.

Previous research investigating evolutionary self healing circuitry has fallen short on several fronts. One approach demonstrated that an evolutionary approach was feasible, but the methods used are impractical in field use[1]. Their method was limited to the processing element (PE) level rather than the Configurable Logic Block (CLB) level and the self repair was limited to offline use. That is, during reconfiguration, the device was inoperable. Other work displayed promise, but was only demonstrated though simulation, and no furthering of their work was made[2].

With this in mind, the proposed method of circuit repair addresses many of these points, and even promises other advantages. First, the proposed method performs *online reconfiguration*, that is, the device is completely operable during partial reconfiguration. To the user, no interruption or degradation of service is detectable. Our method also utilizes intrinsic hardware evolution, where physical tests are run on the hardware block being reconfigured. This has the added benefit of significant speed and accuracy gains over software simulation (extrinsic evolution). The proposed method operates well for Single Event Upset (SEU) errors through partial FPGA scrubbing[3], as well as permanent/cumulative faults.



The basic design is presented in Figure 1. The system comprises of three components located on-chip: the two redundant Operation Modules OM_1, OM_2, and the Configuration Module CM. The OMs implement the actual function of the chip, for example, an image filter, with only one of two modules operating at any given time. The CM continually runs test vectors through the running module, ensuring correct operation. When an incorrect result is returned, the CM re-routes input/output access to the second OM, minimizing external exposure to faulty operation. The CM then attempts to diagnose the fault to a single location. An evolutionary algorithm run on the CM that designs an alternate design at the CLB level, avoiding use of the damaged area. Using the existing design as a starting point, the evolutionary algorithm tests possible configurations on the damaged operation module, until a potential solution is found. By using the existing configuration as a starting point, the evolution process arrives upon a new solution significantly fast than if it had to start from scratch and "reinvent the wheel". The evolutionary algorithm will use a Cartesian Genetic Programming (CGP) approach, which allows for efficient and intuitive implementation[4].

At this point, the workaround configuration is applied to the faulty OM, and is re-activated. The second

OM is turned back off to save resources. In-between test vector application, the CM will continue to evolve the existing configuration on the second OM, in the hopes that a significantly more efficient configuration can be found. In practice, both the OMs and the CM can be placed on the same FPGA board, with the CM implemented as a soft processor (e.g. Xilinx MicroBlaze). Alternately, the CM can be implemented as a separate hard processor.

Stakeholders

The three main stakeholders in this application are the user, client, and designer. Each is concerned with several aspects of the project.

User (Electronics consumers, military, aviation, etc):

- Do devices using this architecture perform at least as well as those that do not?
- How resilient are these devices to physical damage?
- Will they cost more than conventional architectures?
- How much longer are these devices expected to last?

Client (Manufacturers, board designers interested in using the proposed architecture):

- How expensive are these boards?
- Do they interact well with existing technologies?
- How difficult are they to make?
- How soon until a working model is available?
- Does it require specialized hardware?

Designer (Engineers designing the architecture):

- Who will want to use this architecture?
- How can we profit from this? Licensing IP or manufacturing our own boards?
- How much does a board cost?
- How can this be integrated into existing systems?
- How difficult is this design to make?
- How long will it take to produce a PoC?

Practicality

In terms of practicality, implemented the proposed method of circuit repair would be challenging, but well within current skill range. The most difficult aspect would simply be creating a system where FPGA blocks are able to interact with each other in read/write modes. The cost aspect is very reasonable; Spartan 6 family FPGA development kits range from $90 to $500, placing them well within price range.

A majority of the skills required to implement this project already exist. This includes experience in designing and implementing genetic algorithms, knowledge of FPGA platforms, basic knowledge of VHDL language, and a strong background in C/C++. More group members fluent in VHDL and digital system design would certainly benefit the design and implementation of this project.

SWOT Analysis

Overall, this project has great promise to produce a product that can be used in nearly every facet of digital electronic circuit. Simply put, a market exists for this project. It is well known that as digital circuitry becomes more and more complex, devices become more fragile. The method proposed here aims to make these devices significantly more robust, both in terms of physical damage resistance, and in terms of

circuit longevity. The proposed method also holds great promise in the area of manufacturing. When mass producing digital circuitry, a single error can render an entire device useless, and thus, a waste of money. Self repairing circuitry would adapt around any manufacturing flaws, driving down the rate of manufacturing error and cost.

While the proposed method has several strengths, there also exists several weaknesses that brings the use of this project into question. First and foremost is the issue with implementation. Producing self repairing circuitry is by no means simple, and scaling up to full devices may be impractical. Also, the entire concept of a self re-configuring FPGA is difficult to implement, and companies such as Xilinx do not provide tools to do this. A hard-processor based configuration tool make be required. Another issue is that of cost. FPGA devices are significantly more expensive that simple ASIC circuits, which may make devices using the proposed architecture economically unfeasible.

On the other hand, this project provides significant opportunity for success, specifically in the field of aerospace technology and military use. Naturally, the self-healing capabilities of this project would be of great interest to organizations such as NASA, who rely on hardened circuits to perform correctly for years, if not decades, in extreme environments. In fact, NASA dedicates an entire annual concept to the advancement of evolutionary hardware (Conference on Adaptive Hardware and Systems). On the other hand, military applications of this technology are also apparent. The need for reliable technology in environments where physical damage is likely is very present.

The threats this project faces are mostly internal, and are concerned with the ability to successfully implement the idea to the performance specifications presented above. That is, the implementation must perform significantly better than existing technology in terms of longevity and robustness in order to validate its higher cost. Finally, another issue exists in material sourcing. That is, this project relies on low cost CLBs being available to manufacturers for mass production. While a proof of concept can be run on a Xilinx development board, mass production would require access to smaller segments of CLBs without the "bells and whistles" (and price) of more user-friendly FPGA development board. Otherwise, the cost of using the proposed method would be high enough to negate any benefits this technology makes.

References

[1] R. Salvador, A. Otero, J. Mora, E. de la Torre, L. Sekanina, and T. Riesgo, "Fault tolerance analysis and self-healing strategy of autonomous, evolvable hardware systems," in *Reconfigurable Computing and FPGAs (ReConFig), 2011 International Conference on*, 2011, pp. 164–169.

[2] J. Lohn, G. Larchev, and R. Demara, "Evolutionary fault recovery in a virtex fpga using a representation that incorporates routing," in *Parallel and Distributed Processing Symposium, 2003. Proceedings. International*, 2003, pp. 8 pp.–.

[3] J. Heiner, B. Sellers, M. Wirthlin, and J. Kalb, "Fpga partial reconfiguration via configuration scrubbing," in *Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on*, 2009, pp. 99–104.

[4] S. Harding, J. Miller, and W. Banzhaf, "Self modifying cartesian genetic programming: Parity," in *Evolutionary Computation, 2009. CEC '09. IEEE Congress on*, 2009, pp. 285–292.