Combining Circuit Level and System Level Techniques for Defect-Tolerant Nanoscale Architectures

Teng Wang, Mahmoud Bennaser, Yao Guo, Csaba Andras Moritz

Electrical and Computer Engineering Department
University of Massachusetts Amherst, MA, USA
{twang, mbennase, yaoguo, andras}@ecs.umass.edu

Abstract

Recent research progress on nanoscale devices such as based on nanowire (NW) crossbars shows great promise towards building nanoscale computing systems. This paper is part of our ongoing effort to develop and evaluate high-density, defect-tolerant architectures on such fabrics. Our designs are based on Nanoscale Application Specific ICs (NASICs), and are primarily targeted towards microprocessor datapaths. In this paper we propose a new dynamic circuit scheme that enables efficient pipelining and temporary data storage with a 2× higher throughput than in previously published designs. In addition, we explore built-in defect-tolerance techniques in conjunction with system-level CMOS voting and evaluate their effectiveness to mask both defective transistors and broken NWs, as well as combination defects. Furthermore, we introduce a simple defect model for clustered defects. We evaluate the effectiveness of our defect-tolerant designs for both uniformly distributed as well as clustered defects.

1 Introduction

The general nanoscale fabric architecture we use in this paper is called Nanoscale Application-Specific IC (NASIC) [11, 7, 8]. NASICs are based on 2-D nanowire crossbars and use FETs as active switching elements. Our processor design called the Wire Streaming Processor (WISP-0) [9] is a simple but complete stream processor that exercises many different NASIC circuit styles and optimizations. WISP-0 is based on dynamic circuits for pipelining and temporary data storage. In this paper, we propose a new dynamic scheme that improves the throughput of a WISP-0 design by 2× with small area overhead.

As discussed by several researchers, defect tolerance is expected to be a key issue in nanoscale designs. Most nanoscale defect-tolerance techniques proposed by other researchers are based on reconfiguration [1, 3, 5] around defects. By contrast, our solution for NASICs is based on built-in redundant NWs and circuit-level redundancy in a cascaded *AND-OR* logic design [6, 10]. Our objective is to make NASIC designs self-healing (i.e., functional) even in the presence of defects.

In addition to built-in redundancy, fabric and circuit-level optimizations are applied to improve the effectiveness of fault masking. These optimizations include interleaving and weak pull-up/down wires [10]. We found interleaving of (redundant) NWs to be effective against clustered faults. The pull-up/down nano (or micro) wires are especially effective against faults caused by broken NWs. We also combine these low-level techniques with system-level CMOS voting to further improve the yield. Initial exploration of this approach was shown in an invited paper in [8].

This paper provides a more extensive evaluation: we extend the evaluation of our defect-tolerance techniques to cases with both defective transistors and broken NWs. Moreover, we assume both uniformly distributed and clustered defects. We found that the yield of WISP-0 remains above 30% even in the presence of 5% defective transistors in conjunction with 5% broken NWs. As expected, the yield of WISP-0 without defect-tolerance techniques drops quickly to 0 with either types of defects. This paper also introduces a simple defect model for clustered defects. Our preliminary results show that our defect-tolerance techniques can mask clustered defects very well.

Compared with reconfiguration-based approaches, our self-healing techniques eliminate the need for defect map extraction, do not require availability of reconfigurable devices, and dispense with the complex nano-micro interfacing required to address each crosspoint in a reconfigurable fabric. Despite the added redundancy, the WISP-0 design with a 10-nm NW pitch would still have a 3× density advantage compared to an equivalent 18-nm CMOS version.

The rest of this paper is organized as follows: Section 2 provides a brief overview of NASIC and shows an example design of a WISP-0 processor. The new dynamic circuit style is introduced in Section 3. Section 4 provides a discussion of our combined fabric and system-level defect-tolerance techniques. The simulation results in Section 5 show the effectiveness of our defect-tolerance techniques. Section 6 concludes the paper.

2 Overview of NASIC Designs and WISP-0 Processor

NASIC designs use FETs on 2-D semiconductor NWs to implement logic functions and various optimizations to work around layout, doping, and manufacturing constraints as well as defects [11, 9]. While still based on 2-level *AND-OR* logic style, our designs are optimized according to specific applications to achieve higher density and defect masking. Figure 1 demonstrates the design of a 1-bit full adder in a dynamic style. Each nanotile is surrounded by microwires (MWs), which carry V_{dd}, Gnd and some control signals. In multi-tile NASIC designs, NWs are often used to provide communication between adjacent nanotiles.

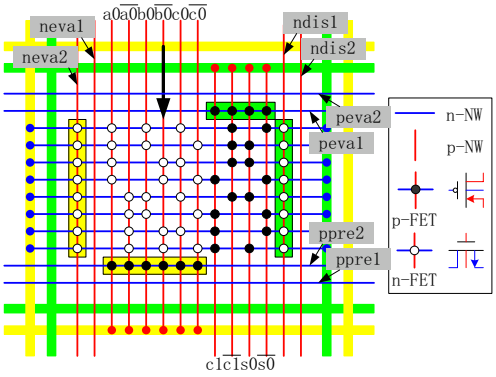

Figure 1. Dynamic implementations of a 1-bit full adder. The thicker wires represent microwires (MWs) and the thin ones are NWs. The doping type of the wires (p-type or n-type) along source-drain of a FET transistor determines the type of the transistor. The black and white dots, at the crosspoints of NWs, represent p-FETs and n-FETs respectively.

WISP-0 is a stream processor that implements a 5-stage microprocessor pipeline architecture including *fetch*, *decode*, *register file*, *execute* and *write back* stages. WISP-0 consists of five nanotiles. Figure 2 shows the layout of WISP-0. A nanotile is shown as a box surrounded by dashed lines in the figure. In WISP designs, in order to preserve the density advantages of nanodevices, data is streamed through the fabric with minimal control/feedback paths. By using dynamic circuits and pipelining on the wires, we eliminate the need for explicit flip-flops and therefore improve the density considerably. In this paper, WISP-0 is used for evaluating the efficiency of our defect-tolerance techniques.

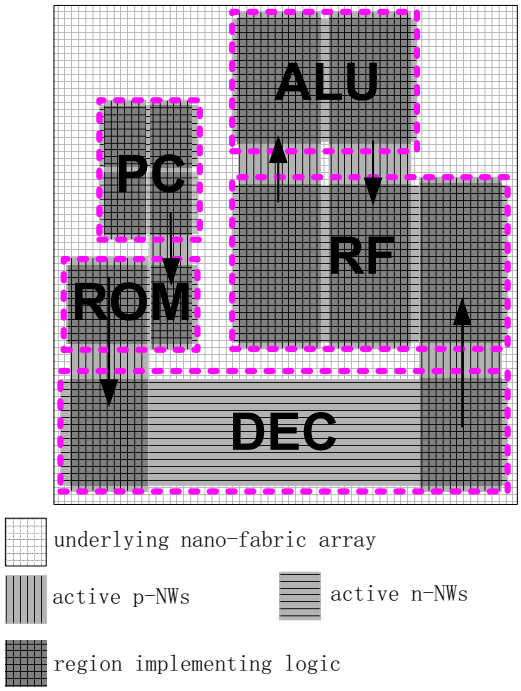

Figure 2. The floorplan of WISP-0.

3 Dynamic Circuits for NASICs

NASIC circuits are based on dynamic circuitry rather than static ratioed logic. In this paper, we propose a new dynamic circuit style that is an improvement compared to our previous work. Similarly, compared with [2], our new dynamic circuit requires two sets of *precharge/predischarge* and *evaluate* signals (i.e. *ndis1, neva1, ppre1, peva1* and *ndis2, neva2, ppre2, peva2* in Figure 1), in each dimension. To ease manufacturing, every nanotile is provided two sets of control NWs for selection. The area overhead is minimal requiring just two extra NWs in each dimension on the fabric. Two successive logic stages must select different control schemes: this allows cascaded circuits to function correctly.

Figure 3 compares our dynamic circuit scheme with the one in [2]. The circuit and waveform in [2] are shown in the left-side. In addition to the normal *precharge/predischarge*

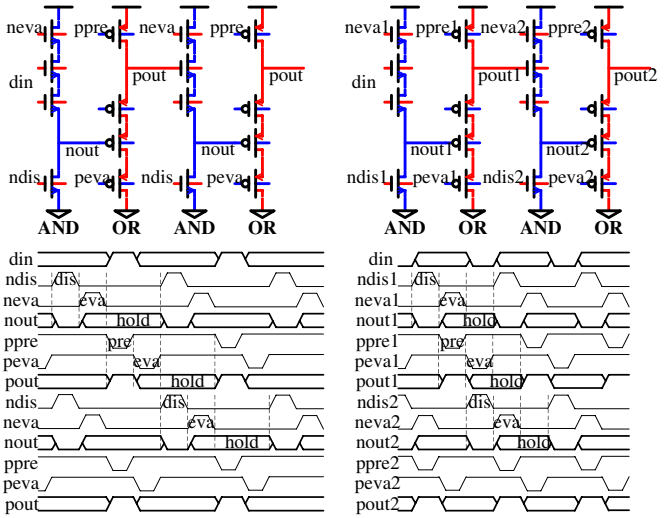

Figure 3. Two different dynamic circuit designs and their waveforms.

and *evaluate* phases, an extra *hold* phase is used for correct cascading. The right-side shows the new circuit and corresponding waveform. The key difference between these two designs is that in the new circuit, the *evaluate* phase is in parallel with the *precharge/predischarge* phase of the next stage. In our previous design and the design on the left in the figure, all operations are sequential. The waveforms indicate that the throughput of the new design is doubled compared with the circuit on the left-side.

4 Defect Tolerance Strategy

Although the manufacturing process is improving rapidly, the defect level of nanodevices is still close to a few percent range [4]. This fact makes defect tolerance a critical design aspect in any nanoscale system. The high rate also means that the problem will likely require a solution that involves all (or many) system layers to be effective.

Basically there are two main types of defects while building nanoscale systems: NWs may be broken and the transistors at the crosspoints may be stuck-short (channel is always open) or stuck-open (channel is always off). A stuck-open transistor can be treated as a broken NW; a stuck-short transistor means that there is no active transistor at that crosspoint.

If reconfigurable devices would be available, we could possibly devise techniques to work around defects. Reconfigurable proposals face many challenges however. One key challenge in such solutions is accessing crosspoints in the fabric: this is required for reconfiguration. That requires a special interface between the micro and the nanodevices capable of addressing every crosspoint. Such an interface involves a large number of MWs, which is not only a high area overhead but it is also very difficult to build due to the required alignment between the fabric-related nano and the interfacing-related microwires.

Alternatively, as proposed by this and our previous works, we can make the circuits and the architectures self-healing such that defects could be masked automatically without reconfiguration.

By replicating horizontal and vertical NWs and devices at crosspoints, we can devise schemes to tolerate a considerable fraction of transistor and NW defects. The faulty signals caused by such defects could be masked by correctly functioning replicas that are merged at each AND and OR stage in a design. Properties of AND-OR logic chains (like faulty 0s can be masked in OR logic and 1s in AND logic stages), can be used to mask faults. Note that either in the stage when they occur or in subsequent logic stages are these faults masked. Of course if both redundant signals are faulty, the correct signal could not be recovered. We found that by carefully interleaving NWs, we can minimize the regions on the fabric that are otherwise difficult to mask. Another technique is to insert a weak pull-up/down NW or microwire(s) between *AND* and *OR* planes or between tiles. Such weak pull wires could be implemented with MWs surrounding the tiles for better manufacturability. Weak pull resistances can be provided by a technique similar to micro-nano contact resistances. These wires are used to help mask certain defective floating NWs by pulling them to logic values that can be tolerated by AND or OR logic planes depending on their position. In addition to these low-level techniques, we also evaluate adding CMOS voting of interconnecting NWs across tiles or simply voting on the end results. Additional details on these defect-tolerance techniques could be found in [10, 8].

In our previous work we only considered defective transistors and broken NWs separately. However these two types of defects can likely happen simultaneously. In this paper, we include cases of both NW and transistor defects.

Moreover, previously we assumed all defects to be uniformly distributed. However, defect clusters will be likely common in grid-based architectures. Due to manufacturing limitations, however, a group of adjacent crosspoints or a group of adjacent wires could be defective at the same time. If clustered defects make two redundant signals faulty, these faults can not be masked. However, if any two redundant signals are set far-enough apart, clustered defects will unlikely make them both faulty simultaneously.

Figure 4 illustrates how NW interleaving helps to tolerate clustered defects. Assuming a defect cluster causes 4 transistors in the circle in Figure 4 to be stuck-open, in the top circuit, both o_1 and o_1' are set to "1" because they are disconnected from Gnd. These faulty signals can not be

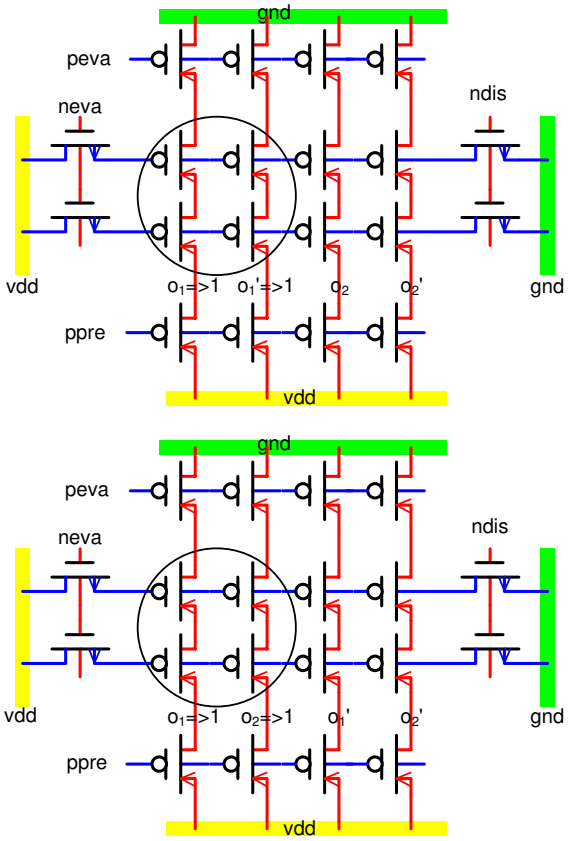

Figure 4. Interleaving helps to tolerate clustered defects. The transistors in the circle are stuck-open.

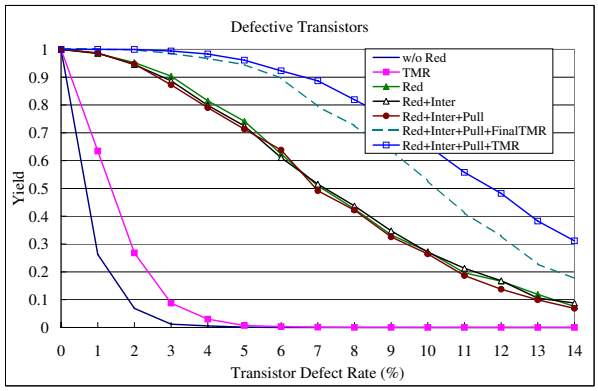

Figure 5. The yield achieved with different techniques when considering defective transistors. *w/o Red* means WISP-0 without built-in redundancy. *TMR* means WISP-0 with TMR voting at each stage. *Red* means WISP-0 with redundancy. *Inter* means interleaving of NWs. *Pull* means applying weak pull-up/down wires. *TMR* means Triple Modular Redundancy at system level and voting at each stage. *FinalTMR* means Triple Modular Redundancy at system level and voting on the final outputs. "+" means the combination of different techniques (e.g. *Red+Inter+Pull* means WISP-0 with built-in redundancy, interleaving and weak pull-up/down wires).

masked in such case. On the contrary, in the bottom circuit, although o_1 and o_2 are set to "1", they both have correct replicas (o_1', o_2'). In the *AND* plane of the next logic stage, they will be masked by their corresponding replicas. A similar analysis can be done for clustered broken NWs.

In the next section, we will show the impact of different types of defects and the effectiveness of our defect-tolerance techniques to mask these defects.

5 Evaluation

We apply the proposed self-healing techniques to WISP-0, including built-in signal redundancy (with signal merging at subsequent AND and OR planes), interleaving of NWs and weak pull-up/down wires. By simulating WISP-0 with randomly generated defects and comparing the outputs with a defect-free design, we evaluate the efficiency of our techniques on tolerating both defective transistors and broken NWs.

We first assume that defects are evenly distributed along NWs and among transistors.

Figure 5 shows the yield of WISP-0 assuming some defective transistors and Figure 6 shows the yield of WISP-0 with broken NWs. 7 curves are shown in every figure, each of them representing one configuration of WISP-0.

From Figure 5 and 6, we can see that our built-in redundancy works very well and the combined techniques improve yield considerably. With the exception of compensating faults, without defect tolerance, the presence of defects causes incorrect results. For WISP-0 with self-healing techniques applied, however, even if the defect rate of transistors reaches 14%, the yield still remains over 30%. If the defect rate of broken NWs is 10%, the yield is over 19%. Notice that interleaving and weak pull-up/down wires do not improve the yield of WISP-0 (Curves "Red", "Red+Inter" and "Red+Inter+Pull" are almost identical in Figure 5). However, they significantly improve the yield of WISP-0 with broken NWs.

We also find from our simulation that the CMOS TMR technique alone doesn't work when the defect rate of transistors or NWs are over 5%. This is because in these situations the possibility of correct signal on several NWs at the

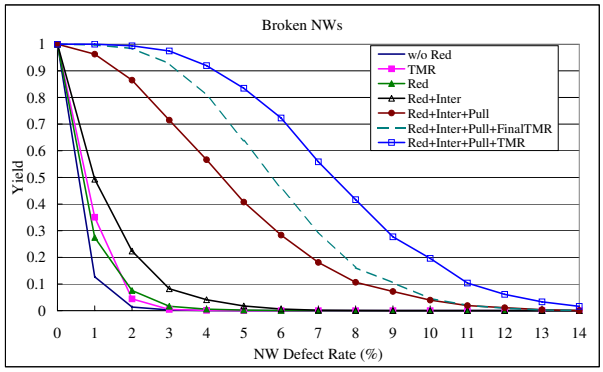

Figure 6. The yield achieved with different techniques when considering broken NWs.

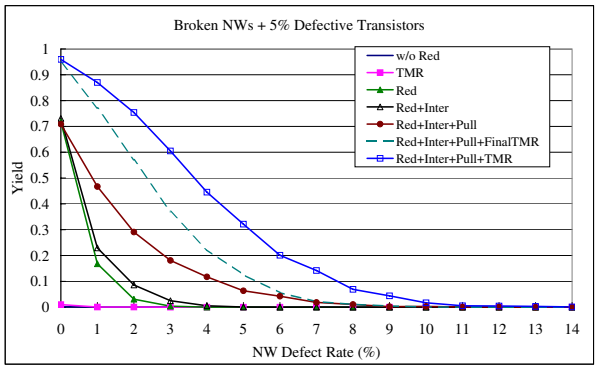

Figure 8. The yield achieved with different techniques when considering various defect rates of NWs and 5% defective transistors.

same time is very low. Assuming that the possibility of correct value on a single NW is α, a simple analysis indicates that the possibility of correct voting is $\alpha^2(3-2\alpha)$. This assumes the voting circuits are defect free because we implement the voting circuits in conventional CMOS. The defect rate of conventional CMOS devices is far less than nanodevices. When α is close to either 1 (defect free, best case) or 50% (worst case), the voting has no benefit. This explains why system-level TMR must be combined with other circuit and fabric-level techniques when the defect rate is very high. The simulation results shown in Figure 5 and 6 are consistent with this analysis (see the curves "TMR" and "Red+Inter+Pull+TMR").

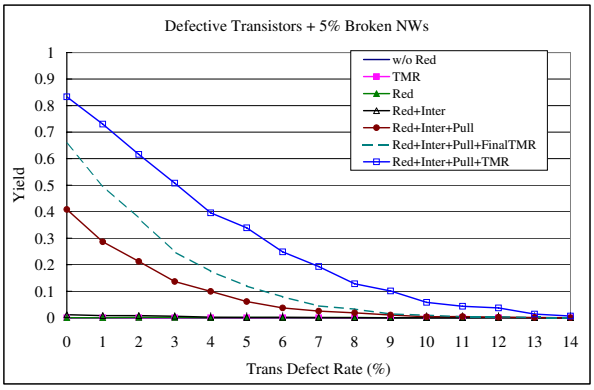

Figure 7. The yield achieved with different techniques when considering various defect rates of transistors and 5% broken NWs.

We extend our simulation to the cases with both defective transistors and broken NWs. We first explore the yield for various transistor defect rates in conjunction with a constant 5% defect rate for NWs. Similarly, we also show the yield for various NW defect rates when assuming a constant 5% defect rate for transistors. Our simulation results are shown in Figure 7 and Figure 8 respectively. We can see that our defect-tolerance techniques improve the yield of WISP-0 considerably even for fabrics with mixed defect types. There are some yield degradations compared to the cases with only one type of defect. However, the yield remains in reasonable range: e.g., even with 5% defective transistors and 5% broken NWs the yield of WISP-0 (with all self-healing techniques applied) remains over 30%.

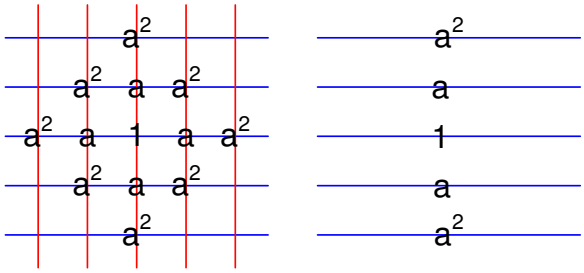

Figure 9. A simple clustered defect model. The value at a crosspoint or on a NW indicates the probability that this crosspoint or this NW is defective.

To evaluate the impact of clustered defects, we first introduce a model for clustered defects. In this paper, we assume a simple clustered defect model for NASICs. First, we set a probability for defect clusters (called *cluster rate*). Then, we determine randomly if a crosspoint belongs to a defect cluster, or not, based on the cluster rate. If yes, the crosspoints around this point would have larger probability to be defective than in uniformly distributed defect models. Intuitively, the probability of a crosspoint being defec-

tive decreases if the distance to the center of the cluster it belongs to increases. Figure 9 shows the defect distribution for clustered defective transistors and clustered broken NWs, where a is a value between 0 and 1. The other parameter in this model is n representing the maximum distance between outmost transistors or NWs in this cluster and the cluster center. n effectively determines the size of clusters.

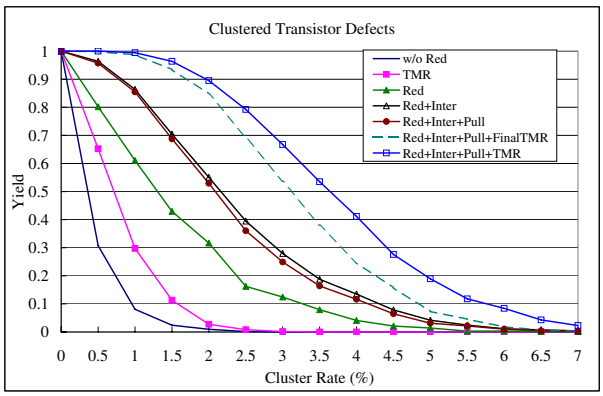

Figure 10. The yield achieved with different techniques when considering various rates for clustered transistor defects.

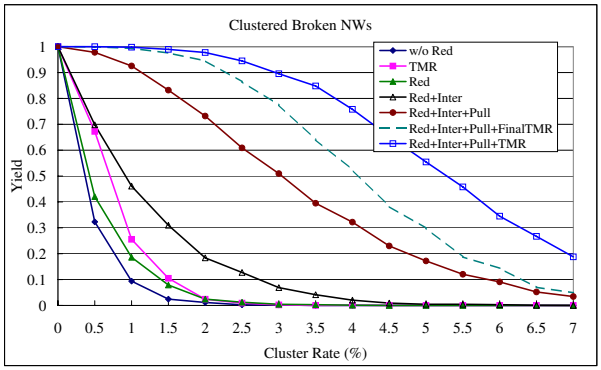

Figure 11. The yield achieved with different techniques when considering various rates for clustered NW defects.

We evaluate the impact of clustered defects based on the defect model discussed above. Figure 10 shows the yield of WISP-0 assuming clustered transistor defects and Figure 11 shows the yield of WISP-0 with clustered broken NWs. We assume the following parameters for clustered defect model: $a = 0.2$, $n = 2$. The results indicate that our defect-tolerance techniques can also improve the yield of WISP-0 with clustered defects. The yield of WISP-0 remains around 20% when the cluster rate of transistors is 5%. Note that each defect cluster may have multiple defects.

Comparing Figure 5 and Figure 10, we find that although interleaving does not help against evenly distributed transistor defects, it helps against clustered transistor defects (curve "Red" and "Red+Inter"). This is because interleaving set replicated signals far apart such that clustered transistor defects can hardly impact them simultaneously.

Another important aspect of nanoscale systems is their density. Built-in redundancy clearly impacts it. Compared with deep sub-micron CMOS technology, however, our self-healing WISP-0 has still great density advantage even with added defect-tolerance. Even at 18-nm CMOS, available in 12 years according to the ITRS 2005 roadmap, our self-healing WISP-0 design combined with system-level TMR would still be around 3× denser than the equivalent WISP-0 in 18-nm CMOS [10].

6 Conclusion

A novel defect-tolerant nanoscale NASIC design is discussed. First, a new dynamic circuit style is introduced to improve the performance of NASICs. After that, various defect-tolerance techniques are evaluated to improve the yield of a WISP-0 design built with NASICs. Both uniformly distributed and clustered defects are addressed. A fairly generic fault model is assumed including mixed defects with both broken NWs and defective transistors.

References

[1] A. DeHon. Nanowire-based programmable architectures. *ACM Journal on Emerging Technologies in Computing Systems*, 1(2), 2005.

[2] A. DeHon and M. J. Wilson. Sublithographic programmable logic arrays. In *Proceedings of the International Symposium on Field Programmable Gate Arrays, FPGA'04*, 2004.

[3] S. C. Goldstein and M. Budiu. Nanofabrics: Spatial computing using molecular electronics. In *The 28th Annual International Symposium on Computer Architecture, ISCA'01*, 2001.

[4] Y. Huang, X. Duan, Y. Cui, L. Lauhon, K.-Y. Kim, and C. Lieber. Logic gates and computation from assembled nanowire building blocks. *Science*, 1313(294), 2001.

[5] K. K. Likharev. CMOL: Devices, circuits, and architectures. *Introducing Molecular Electronics*, 2004.

[6] C. A. Moritz. Exploring nanoscale application specific ics (nasics) and a comparison with cmol: An ar-

chitect's perspective. In *Third ARDA NanoElectronics for High Performance Computing Workshop*, 2005.

[7] C. A. Moritz and T. Wang. Latching on the wire and pipelining in nanoscale designs. In *Non-Silicon Computing Workshop, NSC-3*, 2004.

[8] C. A. Moritz and T. Wang. Towards defect-tolerant nanoscale architectures. *Invited Paper - IEEE Nano2006*, 2006.

[9] T. Wang, M. Bennaser, Y. Guo, and C. A. Moritz. Wire-streaming processors on 2-D nanowire fabrics. In *Nanotech 2005*. Nano Science and Technology Institute, May 2005.

[10] T. Wang, M. Bennaser, Y. Guo, and C. A. Moritz. Self-healing wire-streaming processors on 2-d semiconductor nanowire fabrics. In *Nanotech 2006*. Nano Science and Technology Institute, May 2006.

[11] T. Wang, Z. Qi, and C. A. Moritz. Opportunities and challenges in application-tuned circuits and architectures based on nanodevices. In *CF '04: Proceedings of the 1st conference on Computing frontiers*, pages 503–511, New York, NY, USA, 2004. ACM Press.