Fault-Tolerant Nanoscale Processors on Semiconductor Nanowire Grids

Csaba Andras Moritz, Teng Wang, Pritish Narayanan, Michael Leuchtenburg,
Yao Guo, Catherine Dezan, and Mahmoud Bennaser

***Abstract*—Nanoscale processor designs pose new challenges not encountered in the world of conventional CMOS designs and manufacturing. Nanoscale devices based on crossed semiconductor nanowires (NWs) have promising characteristics in addition to providing great density advantage over conventional CMOS devices. This density advantage could, however, be easily lost when assembled into nanoscale systems and especially after techniques dealing with high defect rates and manufacturing related layout/doping constraints are incorporated. Most conventional defect/fault-tolerance techniques are not suitable in nanoscale designs because they are designed for very small defect rates and assume arbitrary layouts for required circuits. Reconfigurable approaches face fundamental challenges including a complex interface between the micro and nano components required for programming. In this paper, we present our work on adding fault-tolerance to all components of a processor implemented on a 2-D semiconductor nanowire (NW) fabric called NASICs. We combine and explore structural redundancy, built-in nanoscale error correcting circuitry, and system-level redundancy techniques and adapt the techniques to the NASIC fabric. Faulty signals caused by defects and other error sources are masked on-the-fly at various levels of granularity. Faults can be masked at up to 15% rates, while maintaining a 7X density advantage compared to an equivalent CMOS processor at projected 18nm technology. Detailed analysis of yield, density, and area tradeoffs is provided for different error sources and fault distributions.**

***Index Terms*—Defect tolerance, fault tolerance, semiconductor nanowires, nanoscale fabrics, NASIC, nanoscale processors.**

I. Introduction

THE recent progress on manufacturing and assembling of semiconductor nanowires (NWs) is driving researchers to explore possible circuits and architectures. Examples of proposed architectures include [7][8][9][10].

A fabric architecture proposed based on NWs and targeting datapaths is the Nanoscale Application Specific IC (NASIC) [13]. NASIC is a tile-based fabric built on 2-D NW grids. Based on NASICs, nanoscale processors are being explored. For example, the Wire Streaming Processor (WISP-0) [14] is a processor design that exercises several NASIC design principles and optimizations. In this paper we use NASIC as the underlying fabric and evaluate the impact of built-in fault-tolerance techniques on WISP-0's yield and area. Additionally, WISP-0's density is compared with an equivalent CMOS version developed with state-of-the-art conventional CAD tools and scaled to projected technologies at the end of the ITRS-defined semiconductor roadmap.

Two main directions have been proposed to handle defects/faults at nanoscale: reconfiguration and built-in fault tolerance. Most conventional built-in defect/fault-tolerance techniques, however, are not suitable in nanoscale designs because they were designed for very small defect rates and assume arbitrary layouts for required circuits. Moreover, the circuits used for fault correction are often assumed to be defect free, which cannot be guaranteed in nanoscale fabrics.

Secondly, if reconfigurable devices are available, defective devices might be replaceable after manufacturing. Reconfiguration based approaches, however, include significant technical challenges: (i) highly complex interfaces are required between micro and nano circuits for accessing defect maps and reprogramming around defects - this is considered by many researchers a serious manufacturing challenge due to the alignment requirement of a large number of NWs with programming microwires (MWs), (ii) special reconfigurable nanodevices are needed requiring unique materials with programmable and reversible characteristics, and (iii) an accurate defect map has to be extracted through a limited number of pins from a fabric with perhaps orders of magnitude more devices than in conventional designs.

In addition to the potentially intractable complexity, it might not always be possible to correctly extract such a map from a fabric with very high defect rates. Reconfiguration has been proposed at higher levels (e.g., node level in [15]) where it may not require a fully accurate defect map, assuming that self-checking at node-level is supported. However, the complexity of a node might make the nanoscale implemnetation almost always defective.

Furthermore, reconfiguration-based approaches would primarily address permanent defects; it might be difficult, if not unfeasible, to work around faults caused by device parameter

Manuscript received January 15, 2007. This work was supported in part by awards from the Center for Hierarchical Manufacturing (CHM), and NSF awards CCR:0105516, NER:0508382, and CCR:0541066.

Csaba Andras Moritz is with the University of Massachusetts in Amherst and BlueRISC Inc, Amherst, 01002 MA, USA. Phone: 413-320-7669; fax: 413-825-0217; e-mail: andras@ecs.umass.edu. All other authors are with the University of Massachusetts in Amherst. Catherine Dezan is visiting at the University of Massachusetts in Amherst from the Universite de Bretagne Occidentale in Brest, France.

variations visible only for certain input combinations, or internal/external noise related faults that are transient.

Alternatively, as shown in this paper, we can introduce fault tolerance at various granularities, such as fabric, circuit, and architecture levels, to make nanoscale designs functional even in the presence of errors, while carefully managing area tradeoffs. Such built-in fault tolerance could possibly address more than just permanent defects. Faults caused by speed irregularities due to device parameter variations, noise, and other transient errors could be potentially masked. Compared with reconfiguration based approaches, this strategy also simplifies the micro-nano interfacing: no access to every crosspoint in the nanoarray is necessary. Furthermore, a defect map is not needed and the devices used do not have to be reconfigurable.

In this paper, we introduce several fault-tolerance techniques into all parts of WISP-0 while simultaneously managing their area efficiency. The fault tolerance approach used is based on both structural/fabric redundancy, built-in error-correcting circuitry (EC) at nanoscale, and CMOS-based voting at the architectural level. Error correction in general has been proposed by other researchers for nanoscale designs [17][18], however, error correction was used either in memory or at the interface between micro and nanoscale circuits. When used in arithmetic circuits, e.g., with *residue codes* [27][47], components of the correcting circuitry are often assumed to be defect free, and/or, such as in the case of arithmetic with *stochastic computing* and *serialized data* [36] operand lengths are increased prohibitively.

We are the first to apply an EC technique directly on a logic and fabric with significant layout constraints and the first group to evaluate a nanoscale processor design with a combination of EC, structural, and system-level techniques. The combined techniques make redundant circuits more tuned for specific designs and better tradeoff between area overhead and fault tolerance can be achieved. For example, our simulation results show that a hybrid fault tolerance approach is up to 11X better than 2-way structural redundancy alone in terms of its achieved yield on WISP-0. It gives a 12% improvement at 2% defect rate, a 103% improvement at 5% defect rate, and 11X at 10% defect rate. The improvement in the density-yield product compared to 2-way redundancy alone is 52% at 5% defect rate and 4.2X at 10% defect rate.

We found that the yield of WISP-0 is as high as 20% at 10% defective devices while the density of this design is still 7X denser[1] than of the 18-nm equivalent CMOS processor. Much additional experimental data for various fault rates and error sources is provided. The paper is organized as follows.

In Section II, we provide a brief overview of NASICs and WISP-0. The fault model is described in Section III. Section IV describes the built-in fault tolerance techniques. The yield and density simulation results for WISP-0 with uniformly distributed and clustered faults are provided in Section V. A detailed comparison with a CMOS WISP-0 designed with conventional CAD tools is shown in SectionV.B. Section VI shows a sensitivity analysis including the impact of a larger NW pitch on the density of WISP-0. Section VII estimates delay and power consumption. Section VIII discusses related work. Section IX concludes the paper.

II. NASICS AND WISP-0 PROCESSOR

A. *Overview of NASICs*

NASIC designs use FETs on 2-D semiconductor NWs to implement logic functions; various optimizations are applied to work around layout and manufacturing constraints as well as defects [10][13]. While still based on cascaded 2-level logic style, e.g., AND-OR, NASIC designs are optimized according to specific applications to achieve higher density and defect/fault-masking. The selection of this logic family is due to its simplicity and applicability on a 2-D style fabric where arbitrary placement and routing is not possible. Furthermore, due to manufacturing constraints (such as layout and uniform doping in each NW dimension) it may be impossible to use, for example, complementary devices close to each other, such as in CMOS or orient devices in arbitrary ways. By using dynamic circuits and pipelining on the wires, NASICs eliminate the need for explicit flip-flops in many areas of the design and therefore can improve the density considerably [12].

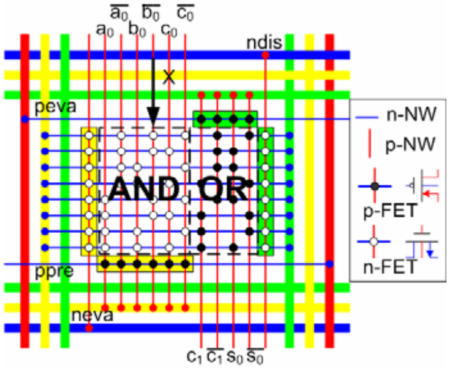

Fig. 1. 1-bit NASIC full adder in dynamic style.

Fig. 1 demonstrates the design of a simple 1-bit NASIC full adder in dynamic style [14]. The signals *ndis*, *neva*, *ppre*, and *peva*, correspond to discharge, evaluation, precharge and evaluation phases on the different NWs. Each nanotile is surrounded by microwires (MWs) (thicker wires in the figure), which carry V_{dd}, *Gnd (or Vss)* and control signals for the dynamic style evaluation of outputs. In multi-tile designs, local communication between adjacent nanotiles is provided by NWs. For more details, please refer to [10][11][12][13][14].

B. *Single-Type vs. Complementary Type NASICs*

In order to produce complementary FETs, two different types of doped NWs must be used. Complementary FETs have been demonstrated in zinc oxide [35], silicon [33], and germanium [34], but in all cases differences in transport properties were found between the two types, sometimes much greater than those seen in today's traditional CMOS FETs. By suitably modifying the NASIC dynamic control scheme and circuit style, we can implement arbitrary logic functions with one type of FETs in NASICs. A design using only n-type FETs

[1] 3X when structural redundancy is combined with CMOS TMR. The NW pitch assumed is 10nm.

will implement a NAND-NAND cascaded scheme whereas a design using only p-type FETs will implement a NOR-NOR scheme. Fundamentally, these are equivalent with the original AND-OR. These schemes may thus be used with manufacturing processes where complementary devices are difficult to achieve. The 1-bit adder example with n-FETs is shown in Fig. 2. A detailed analysis of the control scheme for this circuit is beyond the scope of this paper; we refer the interested reader to [16] for more details.

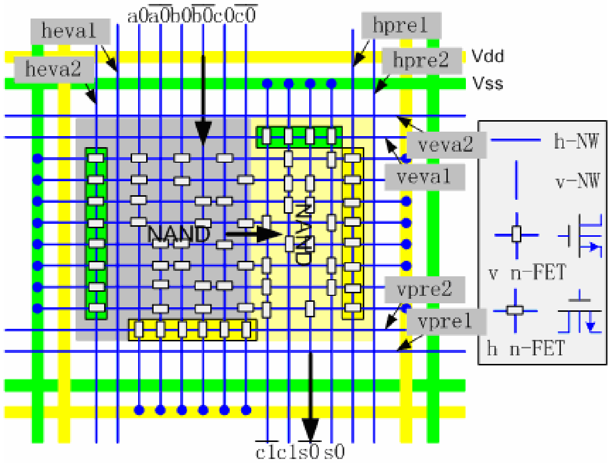
heva1
heva2
$a0\overline{a0}b0\overline{b0}c0\overline{c0}$
hpre1
hpre2
Vdd
Vss
veva2
veva1
vpre2
vpre1
NAND
NAND
h-NW
v-NW
v n-FET
h n-FET
$\overline{c1}c1\overline{s0}s0$

Fig. 2. n-FET only version of a 1-bit adder using the NAND-NAND cascaded scheme. The FET channel is oriented along the length of the rectangle in both horizontal and vertical NWs in the figure; arrows show propagation of data through the tile.

C. *Overview of the WISP-0 Processor*

WISP-0 is a stream processor that implements a 5-stage pipelined streaming architecture. Each stage is implemented in its own nanotile. NWs are used to provide communication between adjacent nanotiles. Each nanotile is surrounded by microwires (MWs) which carry ground, power supply voltage, and some control signals. Additionally, in order to preserve the density advantages of nanodevices, data is streamed through with minimal control/feedback paths. With the help of dynamic Nano-latches [12], intermediate values during processing are stored on the wire without requiring explicit latching. Support is assumed in the compiler to avoid hazards. WISP-0 uses a 3-bit opcode and 2-bit operands. It supports many different arithmetic operations including multiplication.

Fig. 3 shows the layout. A nanotile is shown as a box surrounded by dashed lines. More details about the various circuits used can be found in [12][13] [14]. In this paper, we use WISP-0 to evaluate the efficiency of our fault-tolerance techniques which are added to all circuits.

D. *Manufacturing of NASICs*

NASIC manufacturing can be done with a combination of self-assembly and more conventional top-down manufacturing steps. It is useful to review this before a fault model can be discussed. NASICs do not require reconfigurable devices[2].

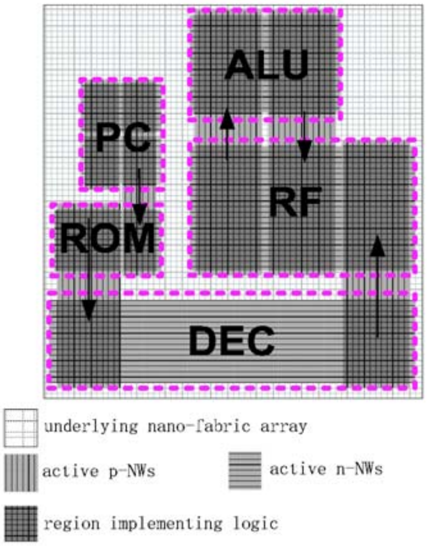

Fig. 3. Floorplan of the WISP-0 Processor.

The interfacing between the micro and the nano components is therefore limited to IO signals as no programming related interfacing and decoders are needed. Nevertheless, there are a number of other key manufacturing challenges that still remain. To manufacture NASIC fabrics, we envision the following main process steps:

Prepare and align NWs:

- Grow NWs to a certain diameter under the control of seed catalysts [1] or by other methods. During the growth NWs are lightly doped for semiconductivity [2]. For single-type FET NASICs, only one type of doping is used for both horizontal and vertical NWs. For NASICs with both types of FETs, each NW set (horizontal vs. vertical) will need to be differently doped.
- NWs can be aligned into parallel horizontal and vertical sets with Langmuir-Blodgett techniques [3]. Depending on the NW pitch assumed, other approaches relying on soft lithographic techniques [37] or based on using grooves to align NWs on a substrate might be possible.

Create FETs, metallic interconnect between FET channels, gate regions, and form 2-D NW grid:

- Regions on both the horizontal and vertical NWs - where there should be no FET channels - are first metalized over with the help of a lithographic mask. The resolution required is 2NW pitches (e.g., 20nmx20nm at a 10nm NW pitch). While this resolution can be fairly demanding depending on the size of the NW pitch, the shape and size of these regions do not have to be precise. A crosspoint area has a rectangular shape proportional with the NW width – as opposed to the typically larger NW pitch. A metalized crosspoint region can, therefore, be of any shape up to a 2NWx2NW square area - beyond that size another crosspoint could be covered causing a defect. This process step is, therefore, likely less challenging than a lithographic process in conventional CMOS with a similar feature size requiring exact shapes,

[2] Some of our earlier papers on NASICs assumed reconfigurable FETs. However, if built-in fault tolerance is added that is not necessary.

sizes, and straight edges. Lithographic techniques with a resolution required for this step have been reported in [4][5]. Nevertheless, we expect this process step to be a key factor in determining the actual NW pitch that can be manufactured. The misalignment of this lithographic mask could generate stuck-short defects, e.g., when some FET channels, that should normally be part of the design, are metalized over. As will be shown in the following sections, these defects can be masked fairly well with a combination of built-in fault-tolerance techniques. In the evaluation section, we also explore the impact of larger NW pitches on the density of the WISP-0 design. A larger NW pitch could facilitate manufacturing designs, even before all process steps are worked out.

- Metallization of the NW gate regions can be done for each set of NWs in conjunction with the previous metallization step. The required resolution for gate regions is fairly low as each logic plane will have either its entire horizontal or all its vertical NWs acting as gates. After being metalized, the gate regions will need to be covered with an oxide shell. Once this step is completed, a 2-D NW grid can be assembled by moving one NW set on top of the other.
- A fine-grained metallization step is essentially responsible for creating the FET channels, creating the metallic interconnects between the FETs, and extending the metallic segments created in the earlier metallization step. Before this step, the assembled 2-D NW grid contains some metallic regions corresponding to (i) crosspoints where no FET channels are needed and (ii) gate regions; other segments of the NWs remain doped as required for the FET channels. FET channels can be distinguished at the crosspoints by using one layer of NWs as a fine-grained mask over the other layer during a final metallization step. This step needs to be completed for both dimensions of a nanogrid – flipping of the structure might be required. After this, channels are formed at grid crosspoints (see, for example, the process in [6] with NiSi), in both dimensions, because the top layer protects the bottom NW from being metalized over; at the same time, the FET channels become automatically connected with small metallic NW segments. Crosspoint regions that have already been metalized in the previous step would remain metallic and would not be affected by this step.

Microwires and contacts:

- Can be added with lithographic process steps.

As discussed in this section, while key individual steps have been demonstrated in laboratory settings (e.g., FETs at NW crosspoints, NW growth and specialization, NW alignment, and fine-grained metallization with the help of NWs to create FET channels), combining the necessary manufacturing steps remains a challenging and unproven process. By working on nanoscale fabrics and architectures, the research community can, however, expose these requirements and tradeoffs between manufacturability and system-level capabilities, fueling more focused research on manufacturing techniques required for assembling nanoscale systems. More on the manufacturing related differences between various proposed nanoscale fabrics is discussed in Section VIII.B.

III. Sources of Error and Fault Model in NASICs

A. *Types and Sources of Error*

Sources of error include permanent defects, process and environmental variation related errors, transient errors, as well as internal and external noise related ones.

Permanent defects are mainly caused by the manufacturing process. The small nanowire dimensions combined with the self-assembly process, driven by the promise of cheaper manufacturing, is expected to contribute to high defect rates in nanoscale designs. Examples of permanent defects in NASIC fabrics would include malfunctioning FET devices, broken NWs, bridging faults between NWs, and contact problems between controlling MWs and NWs. For example, in a process that requires the metallization of segments connecting NASIC FETs, the channels of transistors could be metalized over and therefore stuck-on. The NWs used as gate control have a core-shell structure [22] and, therefore, if a shell is thicker than expected, the FETs controlled by these gates may have no bias applied. Prevalent defect types are also dependent on the types of transistors used. The FET channels will be conducting for depletion mode FETs [19] but will be cut-off for enhancement-mode FETs [20]. This means that when the FET has no bias applied it would be either always conducting (easier to tolerate) or would be cut off (much harder to tolerate) depending on its type.

Process variation related errors are caused by speed deviations due to device parameter variations. These errors occur typically for certain input combinations as a result of larger than expected circuit delays for those input combinations. While the actual parameter variation in NASIC depends on the manufacturing process ultimately used (so this data is currently not available), research from deep sub-micron CMOS technology underlines the seriousness of this problem. We project that delay variations in NASICs would be caused by doping variations on the NWs used for channels and by channel length variations caused by the metallization process that separates FETs from each other (by creating small metallic interconnects between them) and they could be fairly significant.

Internal noise related faults caused by higher frequency and crosstalk between NWs are to be expected in fabrics like NASICs where NWs are placed close to each other. The NASIC control and the dynamic logic used could also affect noise margins. External noise factors such as radiation could be also present: with small dimensions, there might be an increasing likelihood that an α-particle, neutron or proton hitting the chip would cause transient faults. Other noise sources such as electromagnetic interference and electrostatic discharge could cause permanent faults [37].

Overall, we expect that these faults and process variation related ones will be less of a problem in NASICs compared to manufacturing defects, but factors to account for nevertheless. Our objective in the NASIC project is to address all these different sources of errors in a uniform manner with built-in fault tolerance techniques at fabric, circuit, and architecture levels. This paper is a snapshot of our efforts to date.

B. Fault Model Assumed

In NASICs we consider a fairly generic model with both uniform and clustered defects and three main types of permanent defects: NWs may be broken, the transistors at the crosspoints may be stuck-on (no active transistor at crosspoint) or stuck-off (channel is switched off). A stuck-off transistor can also be treated as a broken NW. The initial thinking is that the more common defect type is due to stuck-on FETs as a consequence of the metallization process used. NASIC fabrics require a mask at a 2NW pitch for one of their metallization steps (to avoid channels at crosspoints where no FETs are placed). Stuck-off FETs are also less likely especially in depletion mode fabrics. Recent thinking from [24] suggests that we will be able to control the reliability of NWs fairly well so broken NWs will be likely less frequent than stuck-on FETs.

In this paper we consider defect rates of up to 15%. As suggested by other researchers, the defect levels in nanofabrics are in a few percent range [2]. During our initial work we found that defect rates greater than 15% would likely eliminate the density benefits of nanoscale fabrics compared to projected CMOS, in the context of microprocessor designs. Fabrics with higher defect rates might still be applicable as replacement technology for FPGAs and Structured ASICs: e.g., if lookup-tables for programming of interconnect in FPGAs could be replaced with programmable devices; the lost density due to high-defect rates will likely be offset.

In addition to permanent defects, other error sources such as due to process variation and transient faults are also discussed. Both uniformly distributed and clustered faults are modeled.

IV. BUILT-IN FAULT-TOLERANCE IN NASICS

A. Circuit-Level and Structural Redundancy

Fig. 4 shows a simple example of a NASIC circuit implementing an AND-OR logic function with built-in redundancy: redundant copies of NWs are added and redundant signals are created and logically merged in the logic planes with the regular signals. To make the masking mechanism work, we also modify the dynamic circuit style reported in our prior work [12]. We use different clocking schemes for horizontal and vertical NWs: this, we have found empirically to yield better results. As shown in Fig. 4, horizontal NWs are predischarged to "0" and then evaluated. Vertical NWs are instead precharged to "1" and then evaluated. The circuit implements the logic function $o_1 = ab+c$; *a'* is the redundant copy of *a* and so on. Signal *a* and *a'* are called a NW pair.

A NASIC design is effectively a connected chain of AND-OR (or equivalent) logic planes. Our objective is to mask defects/faults either in the logic stage where they occur or following ones. For example, a break on a horizontal NW in the AND plane (see, for example, position "A" in the figure) causes the signal on the NW to be "0". This is because the NW is disconnected from V_{dd}. The faulty "0" signal can, however, be masked by the following logic OR plane if the corresponding duplicated/redundant NW is not defective.

A NW break at position "B" can be masked by the AND plane in the next stage. Similar masking can be achieved for breaks on vertical NWs. Stuck-off FETs can be modeled as broken nanowires; the defect tolerance would work as described above. For stuck-on FETs, the situation is relatively simpler as each FET has its redundant copy: if one of the two transistors is stuck-on, the circuit still works.

B. Improving Fault-Tolerance by Interleaving NWs

While the previous technique can mask many types of defects, faults at certain positions are difficult to mask. For example, if there is a break at position "C" in Fig. 4, the bottom horizontal NW is disconnected from ground preventing predischarge. The signal on this NW may potentially retain a logic "1" from a previous evaluation. Because of OR logic on the vertical NWs, the two vertical NWs would then be set to logic "1". Since both outputs on the vertical NWs are faulty, the error cannot be masked in the next stage. In Fig. 4, the thicker segments along the horizontal NWs show the locations at which faults are difficult to mask. We call these segments *hard-to-mask segments*.

For nanotiles with multiple outputs, a particular arrangement of output NWs and their redundant copies could significantly reduce the size of hard-to mask segments. This is shown in Fig. 5: 5(a) presents a design in which each output NW and its redundant copy are adjacent to each other. In this arrangement, all segments to the right of the leftmost output NW pair (o_1 and o_1*'* in Fig. 5(a)) are hard-to-mask. Alternatively, the interleaved version in Fig. 5(b), shows an arrangement in which the output NWs and their redundant copies are separated into two groups (o_1 and o_2 form one group; o_1*'* and o_2*'* form another group). In this case, the size of the hard-to-mask segments is reduced. In general, the size of hard-to-mask segments can be reduced in larger scale designs to half, i.e., to half of the region covered by the vertical NWs plus the segment related to the control FET. This latter region is fixed and for most designs adds a negligible area. Interleaving is also helpful in masking clustered defects because duplicated NWs are set apart from one another.

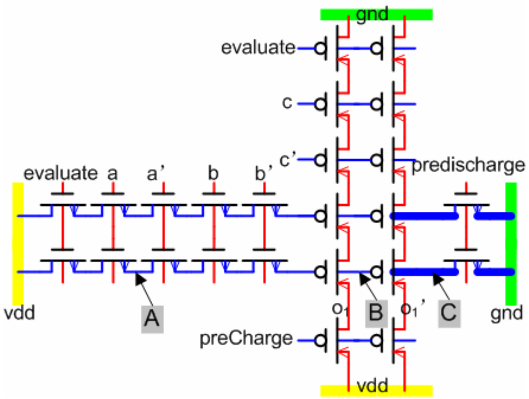

Fig. 4. Simple NASIC circuit with built-in redundancy.

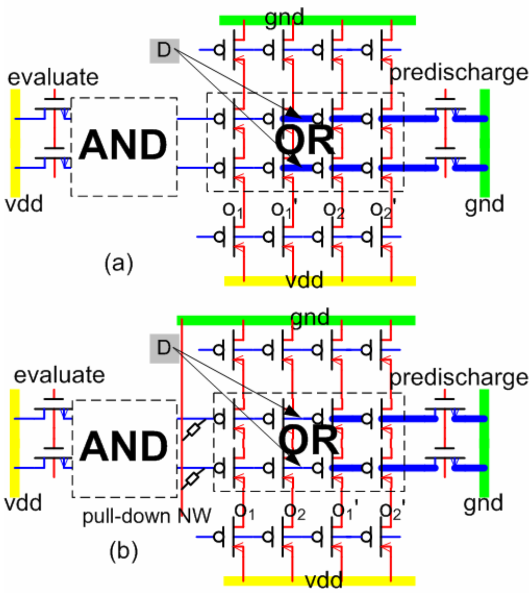

Fig. 5. Interleaving NWs and adding weak pull-up/down NWs to reduce hard-to-mask regions. The bottom circuit has interleaved vertical NWs and weak pull-down NW between the AND and OR planes.

C. *Adding Weak Pull-UP/Down NWs*

Even after built-in redundancy and careful interleaving, there are still some hard-to-mask segments remaining: for example, the thick lines in Fig. 5(b). A possible solution to mitigate this problem is to insert weak pull-down vertical wires between the AND and OR planes. The idea is to pull down (or up depending on logic plane) floating inputs, due to broken NWs, that would cause logic faults: e.g., a floating "1" input to an OR plane that would make the OR logic always compute "1". Modifying floating signals to a preferred logic level would allow masking in following logic planes.

A weak pull-down NW does not affect operation if there are no defects, but introduces a performance tradeoff when there are defects, by slowing the circuit down somewhat. It also contributes to leakage power. At each crosspoint between a vertical pull-down wire and horizontal NWs there is a resistance created. This resistance has to be made larger than the switch-on resistance (estimated to be smaller than 10MΩ according to [2][3]) of a depletion-mode FET and smaller than the switch-off resistance (over 10GΩ). We are currently building a detailed SPICE simulator that would enable us to explore the performance tradeoffs due to these added wires in more detail. To ease manufacturing one could also use MWs instead of the NWs to implement weak pull-up/down wires.

D. *Adding CMOS TMR*

Voting based techniques such as TMR [30] have been used extensively before. To be efficient, voting requires that the probability of a defect in the voting circuit is much smaller than in the design it is applied to. This is clearly the case in conventional technology. TMR is not applicable as is in NASIC designs because at 5-15% fabric defect rates the TMR circuits themselves would be likely defective.

Nevertheless, in pipelined processor designs one could add TMR, e.g., with majority voting, at certain points in a design in CMOS, without affecting throughput significantly. If each nanotile has two extra identical replicas, we could vote either at each stage or on the final outputs. Voting helps where the other nanoscale techniques leave faulty outputs.

E. *Nanoscale Error-Correcting (EC) Circuits*

1) *Hamming Distance*

The Hamming distance between two input codes is defined as the number of bits that is different. For example, the Hamming distance between "000" and "001" is 1. For the simple 1-bit adder design in Fig. 1, the minimum Hamming distance between the input codes is 1. Therefore, in that example, we cannot tolerate any defect on vertical NWs.

By adding redundant bits to the input signals, we are able to increase the minimum Hamming distance of input codes. In the 2-way redundancy example shown in Fig. 4, the input codes are simply duplicated and the Hamming distance is increased to 2. With a minimum Hamming distance of 2, the design with 2-way redundancy can tolerate 1-bit error on the input signals. In the following subsection, we will show the required circuit-level modification to achieve error-correction with built-in error-correcting circuits and redundant code signals, for a more efficient defect masking.

2) *Error-Correcting Code Background*

Achieving a certain Hamming distance between codes with minimum redundant bits is a well-known problem in the communication area. These codes called as error-correcting codes are widely used to correct signal errors in noisy channels. Various kinds of error-correcting codes have been proposed and used; the Hamming code is one of the most popular codes due to its simplicity [23].

Considering a set of 3-bit codes {"000", "001", "010", "011", "100", "101", "110", "111"}, the minimum Hamming distance between these codes is 1. By adding 3 redundant bits to the codes, we can achieve a Hamming distance of 3. The redundant bits (shown in parentheses below) are not unique according to the coding theory. An example of a new code set is {"(000)000", "(011)001", "(101)010", "(110)011", "(110)100", "(101)101", "(011)110", "(000)111"}. Obviously, this code set is more efficient than the one created by a simple signal duplication used in 2-way redundancy - which achieves a Hamming distance of 2 similarly with 3 added redundant bits.

In general, the number of required redundant bits is determined by the desired Hamming distance and the code width. For a given Hamming distance, the error-correcting code rate, defined as the ratio between the original signal width and the width of all signals including redundant ones, approaches 1 as the original signal width gets large [23] - which means the relative overhead goes down. For example, 11-bit wide signals would only need 4 redundant bits to achieve a Hamming distance of 3.

Note that in traditional coding theory, codes for a 1-bit error correcting require a Hamming distance of 3; codes for 2-bit error correcting require a Hamming distance of 5. In general, codes for *n*-bit error correcting require a Hamming distance of 2*n*+1 [23]. In NASICs, however, with Hamming distance of *n* we can tolerate *n-1* defects on vertical NWs. This is because in the case of permanent defects any input combination can only be impacted in the same bit positions.

This paper focuses on Hamming codes; we are currently also exploring a variety of other techniques such as based on BCH codes [44][45].

3) *Error-Correcting in NASICs*

To apply the EC technique in NASICs, redundant bits are added to original input signals for the desired Hamming distance. Next, error-correcting related FETs are added so as to keep the output signals the same as outputs in original designs.

The following simple circuit in Fig. 6 (OR plane is omitted for clarity) shows how to add error-correction to a NASIC circuit. The AND logic outputs ~*c* on the top horizontal NW and *c* on the bottom NW. It is easy to see that one single defect makes the output faulty: e.g., the defect shown on the right vertical NW forces the output on the top horizontal NW to logic "1" (Fig. 6 (a)) for all input values. The output is set during evaluation (*neva* is turned on).

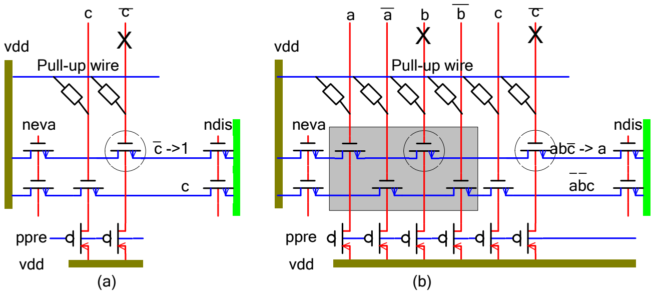

Fig. 6. A simple NASIC circuit: (a) Original design without defect-tolerance. (b) Design with the built-in EC technique.

To apply EC, as shown in Fig. 6 (b), we add 2 redundant bits (*a* and *b* and their complementary forms) to the original input signals *c* and ~*c*. The values of *a* and *b* are related to the value of *c*. In this example, we choose "110" and "001" as possible input combinations with a Hamming distance of 3. We then add redundant vertical NWs for the redundant inputs. At each new crosspoint (shown in the shadowed area in Fig. 6 (b)), we place a FET only if it does not impact the correct outputs. For example, the output signal on the top horizontal NW should be "1" when *c* is "0". Based on the input combinations we choose, *a* and *b* are "1", so we place 2 FETs at the corresponding crosspoints (shown as n-FETs on the top horizontal NW in the shadowed area in Fig. 6 (b)). We can similarly set the crosspoints for the second horizontal NW in the shadowed area. As mentioned, the added overhead is of course smaller for larger designs.

Let us analyze why this design can tolerate 2-bit errors on vertical NWs. For example, assuming the input combination is "001", the output signal on the top horizontal NW should be equal to "0" (~*c*). If we, however, add 2 breaks on the vertical NWs *b* and ~*c* (indicated by "X"s in Fig. 6 (b)), the signals on NWs *b* and ~*c* will be set to faulty "1" because they are disconnected from *Gnd*. As a result, the FETs shown in the circles in Fig. 6(b) will be switched on permanently. Without the added circuits, the output signal on the top horizontal NW would be forced to faulty "1". However, the redundant signal *a* ("0" in the example) forces the output signal on the top horizontal NW to a correct "0". Similar analysis can be made for other input combinations. Clearly, we can guarantee the correct output signals on horizontal NWs even when any two vertical NWs have defects. The key insight here is that *the added FETs in the EC circuit take over the role of any of the original FETs in case they would become faulty or have incorrect input(s) and would because of that not be able to affect the output*.

With a Hamming distance of 3, the circuit in Fig. 6 (b) can tolerate any 2 defects on vertical NWs

4) *1-bit NASIC Adder with EC*

We apply EC on the 1-bit NASIC adder using the method described above. The new adder is shown in Fig. 7.

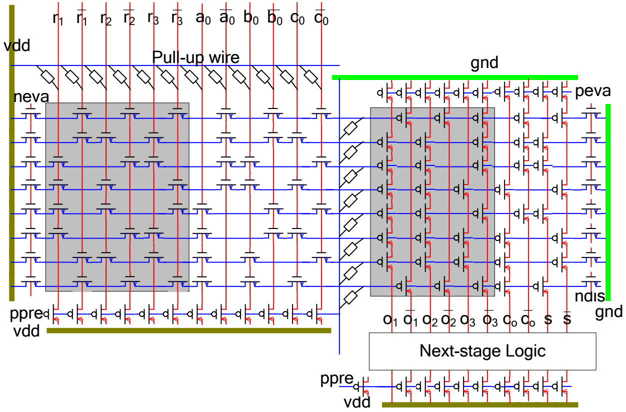
r1 r̄1 r2 r̄2 r3 r̄3 a0 ā0 b0 b̄0 c0 c̄0
vdd
Pull-up wire
gnd
neva
peva
ppre
vdd
o1 ō1 o2 ō2 o3 ō3 co c̄o s s̄
ndis
gnd
Next-stage Logic
ppre
vdd

Fig. 7. 1-bit NASIC full adder with EC. The circuits in the shadowed area are redundant circuits added for the purpose of error correction.

Three redundant bits (r_1, r_2 and r_3) are added for a Hamming distance of 3. Error-correcting related FETs for these 3 redundant bits are shown in the left-side shadowed area. Circuits in the left-side shadowed area are actually helping in providing the correct output on each horizontal NW (input to the OR plane); the right-side shadowed area is used to generate redundant signals for the error-correcting circuits in the next stage. This example also shows how EC can be applied in cascaded circuits.

5) *EC Combined with 2-way Redundancy*

There is one issue with the EC technique: complementary signals are required for proper functionality. However the product-term signals on horizontal NWs are not complementary. Thus, it may not be feasible to apply the EC technique for defects on horizontal NWs. Creating a complementary version for each product-term is not feasible on a 2-D fabric with this type of 2-level logic – we are currently investigating other logic style based on mixed AND/NAND-OR/NOR logic in the same tile where this might be possible. For the time being, we therefore apply 2-way redundancy techniques on horizontal NWs. As will be shown in the next section, the yield of WISP-0 can still be improved considerably with this hybrid approach.

V. EVALUATION

Using the design approaches described in Section IV, we can incorporate the techniques into all circuits of WISP-0 [14]. We used our NASIC CAD tools to modify WISP-0. To verify the efficiency of our fault-tolerance approaches, we developed a simulator to estimate the yield of WISP-0 for different defect rates and also considered other error sources.

A. Yield Evaluation of WISP-0

The simulation results for permanent defects are provided in Fig. 8 (assumes defective FETs) and Fig. 9 (assumes broken NWs). First we present results assuming defects are uniformly distributed. Clustered defects are addressed in separate subsequent subsections.

The notation used is: *RAW* stands for WISP-0 without redundancy (or baseline); *2-way* stands for WISP-0 with 2-way redundancy; *2-way+TMR* stands for 2-way redundancy plus micro-scale TMR on the WISP-0 result; *EC3+2way* denotes a design with EC using a Hamming distance of 3 on vertical NWs and 2-way redundancy on horizontal NWs; and *EC4+2way* denotes EC with a Hamming distance of 4 on vertical NWs and 2-way redundancy on horizontal NWs. While other combinations are possible, we found these to be most insightful and representative. The 2-way redundancy techniques also incorporate the techniques discussed in Sections IV.B and IV.C

From the results, we can see that EC-based techniques achieve the best overall yield. Compared with a 2-way redundancy approach, the improvement of the hybrid approach (EC3+2-way) on the yield of WISP-0 is 12% when the defect rate of transistors is at 2%, 76% at 5% defect rate, and 5X at 10%. Note that the improvement is greater for higher defect rates.

As expected, EC with a Hamming distance of 4 (EC4) on vertical NWs, achieves a better yield compared to EC3. The improvement compared to 2-way is 12% when the defect rate of transistors is at 2%, 103% at 5% defect rate, and 11X at 10%. However, the rate of improvement is not as significant as for the EC3 version - especially when the defect rate of transistors is less than 10%.

One possible explanation is that the likelihood of 3-bit errors on vertical NWs is relatively small compared to 1 or 2-bit errors for these rates, so the approach starts to have diminishing returns despite the greater Hamming distance.

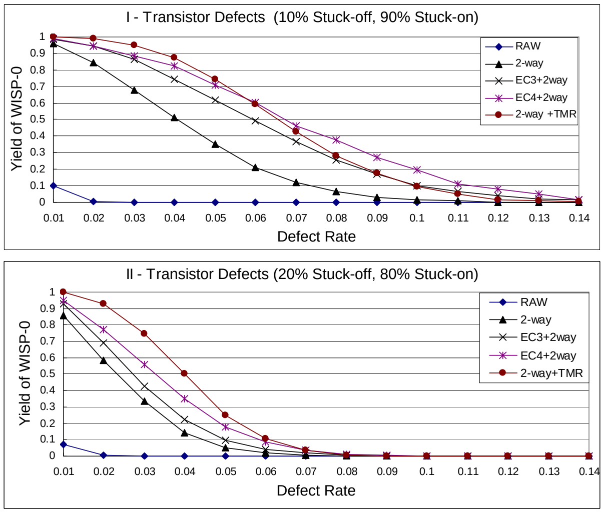

Fig. 8. The yield achieved for WISP-0 with different techniques when only considering defective transistors.

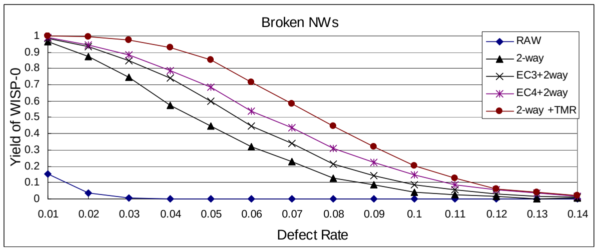

Fig. 9. The yield achieved for WISP-0 with different techniques when only considering broken NWs.

We simulated two different distributions of defective transistors; we assumed that the stuck-on FETs are more prevalent and simulated a relatively smaller fraction of stuck-off defects (10% and 20% respectively) for the reasons we discussed in Section III. In Fig. 8 (bottom graph), we can see that our techniques are more efficient for stuck-on defects than for stuck-off defects. EC based approaches perform well for defects based on broken NWs but not as good as the *2-way +TMR* combination. Similar to the case with 20% stuck-off FETs, broken NWs are difficult to mask. However, as discussed in Section 3, we project stuck-off FET defects and broken NWs to be less prevalent than stuck-on FETs.

Some defect-masking techniques provide good yield improvement but require relatively large area overhead. For example, as shown in Fig. 8 and Fig. 9 micro-scale TMR (implemented in CMOS at the output of WISP-0) combined with 2-way redundancy achieves a somewhat higher yield than *EC3+2-way* in some scenarios. This comes, however, at a cost of a 2.67X larger area than with *EC3+2-way* (density results will be detailed in the following section). Therefore, it is important to understand the area overhead (or impact on density) of the different fault-tolerance techniques in conjunction with their fault masking ability.

TABLE I
TECHNOLOGY PARAMETERS

NW pitch	10nm
NW width	3~4nm
Technology Node (ITRS 2005)	MW pitch
70-nm	170nm
45-nm	108nm
32-nm	76nm
18-nm	42nm

B. Comparison with Equivalent CMOS Processor

The normalized density of WISP-0 for the various scenarios is shown in Fig. 10. Technology parameters used in the calculations are listed in Table I. To get a better sense of what the densities actually mean we show the density of an equivalent WISP-0 processor. We designed this processor in Verilog, synthesized it to 180nm CMOS. We derive the area with the help of the Synopsys Design Compiler tool. Next, we scaled it to various projected technology nodes based on the predicted parameters by ITRS, assuming area scales down quadratically. For the purpose of this paper, we assume that the CMOS version of WISP-0 is defect-free and no fault-tolerance technique is applied.

We can see from the results that the area overhead of adding 2-way redundancy for the nanoscale designs is roughly 3X when MWs in NASICs are assumed to have the same dimensions as MWs would have in 18nm CMOS technology. TMR-related overhead added to the nanoscale design brings an extra 3X overhead because TMR requires 3 copies of nanoscale blocks. A WISP-0 design based on *ECC3+2-way* requires around 20% more area than one based on 2-way redundancy for both horizontal and vertical NWs, but achieves a much better yield.

Overall, the density of a NASIC based WISP-0 remains at least 3X (without EC but with TMR) or 7X (with EC) greater than the density of the corresponding CMOS processor at 18nm.

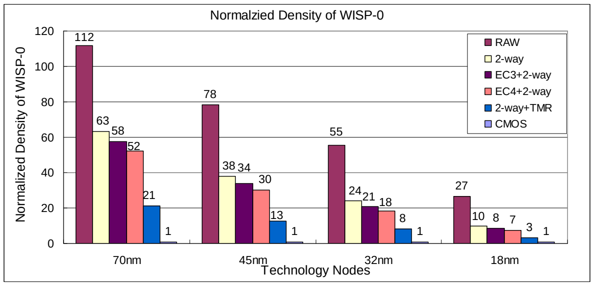

Fig. 10. WISP-0 density with different defect tolerance techniques.

C. *WISP-0 Density-Yield Product Evaluation*

To evaluate the tradeoff between yield improvement and area, we also consider the yield and density together in a combined metric. The yield-density product is a comprehensive indicator for the efficiency of different defect-tolerance techniques; it represents the ratio between the benefit (yield of designs) and its cost (area overhead).

The yield-density product results for various defect rates are presented in Fig. 11 and Fig. 12 respectively. We can see that the EC-based approaches, *EC3+2way* and *EC4+2way*, are significantly more efficient than the other approaches, except for relatively small defect rates. Compared to 2-way redundancy, an approach based on *EC3+2way* improves the yield-density product by 52% when the defect rate of FETs is 5% and by 4.2X for a 10% rate. Clearly, different levels of defect rates may require different defect-tolerance techniques: for defect rates lower than 3%, 2-way redundancy appears to be sufficient.

When defect rates increase beyond 3%, EC with a Hamming distance of 3 is desirable. If the defect rate is larger than 5%, EC with a Hamming distance of 4 is the best choice. Future NASIC CAD tools can take advantage of this and insert appropriate levels of defect tolerance depending on expected defect rates.

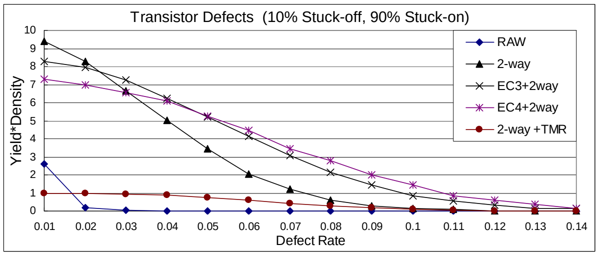

Fig. 11. WISP-0 yield-density products considering defective FETs.

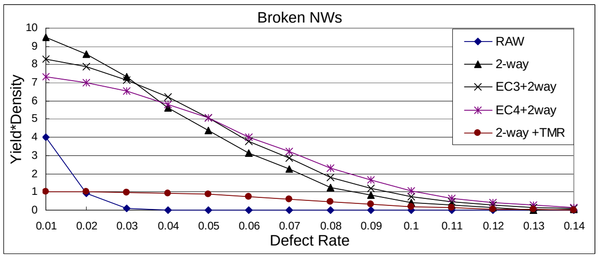

Fig. 12. WISP-0 yield-density products considering broken NWs.

D. *NASICs with Clustered Defects*

In our previous results we assumed that all defects are uniformly distributed. However, defects can also be clustered as a group of adjacent FETs or a group of adjacent NWs could be damaged during the manufacturing process. In a 2-way redundancy scheme, if clustered defects make two redundant signals faulty, these faults cannot be masked. However, if the same two redundant signals are placed far-enough apart, clustered defects will unlikely make them faulty simultaneously.

To evaluate the impact of clustered defects, we first introduce a model for clustered defects. First, we set a probability for defect clusters or *cluster rate*. FETs belonging to clusters would have greater probabilities to be defective than in defect models based on uniformly distributed defects. Intuitively, the probability of a FET being defective decreases with increasing distances from the center of the cluster it belongs to.

Fig. 13 shows how the probability of defects is modeled in a cluster. Parameters of this model include *a*, representing the probability of defects in nodes adjacent to cluster centers, and *n* representing the maximum distance between the outmost defective transistors or NWs and the center; *n* also determines the size of clusters.

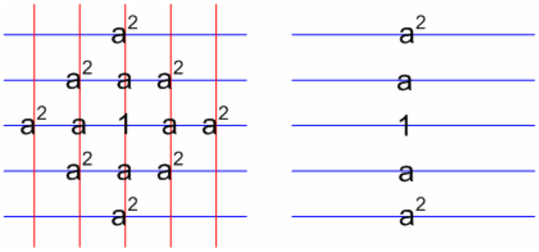

Fig. 13. A simple defect model for clustered defects; shows how defect probabilities are decreasing for FETs and NWs further away from a cluster center.

E. *WISP-0 Yield with Clustered Defects*

Fig. 14 shows the yield of WISP-0 assuming clustered transistor defects; Fig. 15 shows the yield with clustered broken NWs. The results indicate that our defect-tolerance techniques also work for clustered defects/faults: the yield remains at around 20% even when the cluster rate of transistors is 5% for the parameters simulated. Note that each defect cluster may have multiple defects.

The yield-density product of WISP-0 for clustered defects is shown in Fig. 16 and Fig. 17. While the microscale TMR combined with 2-way redundancy (*2-way+TMR*) gives a somewhat higher yield than *EC3+2-way* (see Fig. 14 and Fig. 15), it achieves a lower yield-density product due to its significantly higher area overhead.

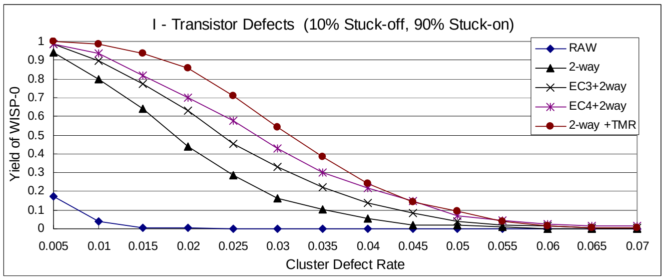

Fig. 14. WISP-0's yield for various cluster rates assuming defective transistors; clustered defects with parameters a=0.2 and n=2.

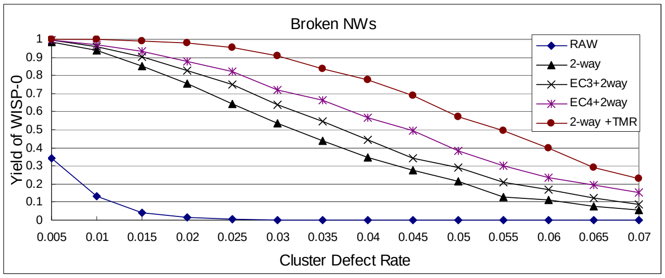

Fig. 15. WISP-0's yield for various cluster rates when considering broken NWs; clustered defects with parameters a=0.2 and n=2.

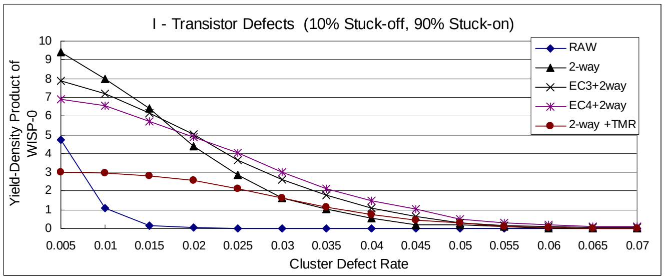

Fig. 16. Yield-density product achieved for WISP-0 considering defective transistors; clustered defects with parameters a=0.2 and n=2.

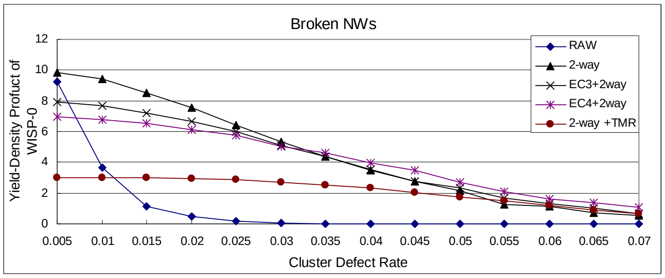

Fig. 17. Yield-density product achieved for WISP-0 when only considering broken NWs; clustered defects with parameters a=0.2 and n=2.

F. *Impact of Transient Errors*

We extended the yield simulator to provide an initial analysis on the benefits of the built-in fault tolerance techniques for transient errors. This is shown in Fig. 18. The results indicate that we could tolerate transient faults fairly well although the masking is less effective than for permanent defects. On the other hand, we expect these errors to be much less frequent than those caused by permanent defects. One insight is that the system-level TMR appears to have the best overall benefit for these types of errors. The reason is that as these errors are random and transient, if an error does not occur at the same time and same position across at least 2 copies, the system-level TMR voting could mask it – assuming that other errors are corrected.

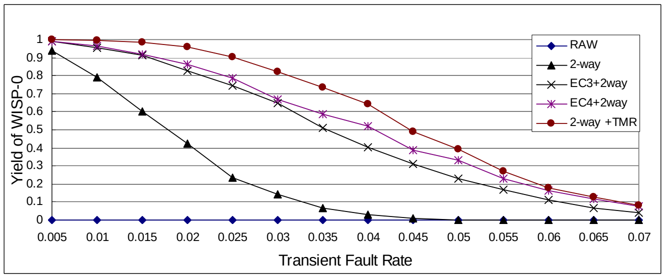

Fig. 18. Yield achieved assuming transient faults.

G. *Impact of Device Parameter Variation*

The actual parameter variation for devices used in NASICs is not known as yet. We can predict, however, based on deep sub-micron CMOS processes, that process variation could cause significant variations in the parameters of semiconductor NW devices. Device parameter variation can impact a circuit's speed/delay, by making certain execution paths longer than expected. Delay variation related faults are in many ways similar to those caused by permanent defects except that they would be limited to certain input combinations (using the circuit paths with longer than acceptable delays). One can argue that the techniques presented in this paper would therefore be able to address such faults. In fact, we estimate that we would be able to mask a higher rate of faults caused by device parameter variations than due to permanent defects, as only a subset of inputs would cause errors as opposed to all inputs. As part of our future work, we plan to model delay variation in NASIC circuits for an exact analysis of the built-in fault tolerance techniques for these types of faults.

VI. Sensitivity Analysis

A. *Impact of NW Pitch on Density*

In the previous analyses, we assumed that the pitch between NWs is 10nm. While this has been demonstrated in the laboratory, it will take time until we can reliably manufacture larger designs at this scale (the same way as it took the semiconductor industry decades to refine lithography to today's resolution). A larger NW pitch may come with lower defect rates and it will also be significantly easier to manufacture. For example, a 20nm pitch design would require the NASIC metallization masks at 40nm resolution: a much more doable undertaking than 20nm. On the other hand, as expected, a larger NW pitch will result in lower overall density so it is important to understand its impact at the system level.

The impact of a 20nm NW pitch on density is presented in Fig. 19. Note that the density of WISP-0 with any of the EC-based approaches is still 2X better than 18nm CMOS technology. This is a result of a high density interconnect structure combined with high-density logic in a NW-based fabric. A plausible option might be to start manufacturing at a relatively lower density and gradually scale with improvements in nano manufacturing.

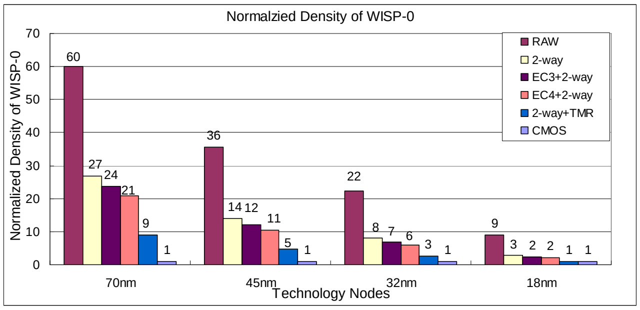

Fig. 19. Density comparison between NASIC WISP-0 assuming a 20-nm NW pitch and an equivalent CMOS WISP-0.

VII. Delay and Power Estimates

Delay and power estimation was done for the WISP-0 processor built on Silicon Nanowires.

TABLE II
PARAMETER VALUES

NW-pitch	10nm
NW-shell thickness (t_{sh})	1nm
NW-width (w)	4nm
Dielectric Constant of SiO_2(ε_r)	2.2
Resistivity of Si (ρ_{Si})	10^{-5} Ωm
Resistivity of NiSi (ρ_{NiSi})	10^{-7} Ωm
NW-MW contact Resistance (R_c)	10 kΩ
Transistor ON Resistance (R_{ON})	4 kΩ
Transistor OFF Resistance(R_{OFF})	10 GΩ
Supply Voltage	3V-4.5V

A NW-MW contact resistance of 10kΩ and resistivity values of 10^{-7}Ω-m and 10^{-5}Ω-m for NiSi and Si respectively were used in these calculations [21]. R_{ON} for a transistor of length 5nm and width 4nm was calculated to be around 4kΩ. An R_{OFF} resistance of 10GΩ was used [7]. A nanowire pitch of 10nm, an oxide layer thickness of 1nm, and a dielectric constant of 2.2 were assumed. Table II summarizes all the parameter values used in these calculations.

A. Delay Calculations

A lumped RC model was used for the worst-case delay analysis. Expressions from [7] were used for capacitance estimation. These calculations take into account NW-NW junction capacitances and relatively realistic coupling scenarios. The coupling capacitance per unit length was found to be 39.04pF/m. The junction capacitance was found to be 0.652aF.

TABLE III
Capacitive Loading (in aF)

	Control NW(H)	Datapath NW(H)		Control NW(V)	Datapath NW(V)	
	pre/eva	pre	eva	pre/eva	pre	eva
PC	14.99	9.78	25.27	11.08	4.56	32.43
ROM	8.48	11.08	33.47	9.78	20.12	82.68
DEC	11.74	20.21	83.33	11.74	55.42	143.1
RF	27.38	26.73	98.21	9.13	42.38	167.6
ALU	29.34	18.26	37.78	16.95	30.64	138.7

Table III indicates the capacitive loading on each tile of WISP-0 for different clock phases. During each phase, there is one control NW and one or more datapath NWs switching. In the table 'Control NW (H)' refers to a Horizontal precharge/evaluate signal. Since the precharge and evaluate control NWs in one plane are geometrically identical, the capacitive loading on these NWs is the same. 'Datapath NW (V)' refers to datapath nanowires in the vertical plane. The capacitive loading during precharge and evaluate is dissimilar for datapaths owing to different lengths and coupling effects.

The lumped capacitance is in the range of ado-Farads, and as expected, larger components such as the RF (Register File) are more heavily loaded. Table IV shows the maximum delay for the tiles of WISP-0 for a MW-NW contact resistance of 10kΩ. 'H-pre' and 'V-pre' stand for horizontal and vertical precharge phases respectively, 'H-eva' and 'V-eva' are horizontal and vertical evaluate phases. All delays are in picoseconds.

TABLE IV
DELAY (ps) – ASSUMES CONTACT RESISTANCE

	H-pre	H-eva	V-pre	V-eva
PC	0.227	0.463	0.141	0.536
ROM	0.215	0.796	0.302	3.785
DEC	0.375	1.485	0.934	2.742
RF	0.596	2.135	0.615	4.778
ALU	0.481	1.415	0.667	3.667

In WISP-0, datapath lengths and the number of transistors on each datapath are different. Consequently the delay varies over a wide range of values. However, the performance of a pipeline is determined by the slowest segment; in this case it is the vertical plane of the RF (delay=4.778ps). The operating frequency assuming a 33% duty cycle (reflecting a clock needed for a precharge-evaluate-hold control) is easily shown to be 69GHz. It is expected that the frequency will be lower in practical designs with longer datapaths and larger bitwidths.

The contact resistance of 10kΩ is a large contributor to the overall delay for all nanotiles. It is expected that with improvements in manufacturing, this value may be significantly reduced. Table V tabulates the delay for all nanotiles without any contact resistance.

TABLE V
DELAY (ps) –NO CONTACT RESISTANCE

	H-pre	H-eva	V-pre	V-eva
PC	0.56	0.186	0.33	0.236
ROM	0.80	0.508	0.96	3.147
DEC	0.155	0.830	0.471	1.674
RF	0.222	1.268	0.260	3.558
ALU	0.153	0.952	0.339	2.593

When compared with the values in Table IV, it is clear that even on the larger nanotiles, a large portion of the delay is due to the contact resistance. For example, for the slowest segment (‘V-eva’ of RF), the contact resistance contributes 25% of the delay. On smaller nanotiles this effect is far more prominent (75% for ‘H-pre’ of the Program Counter tile or PC). The operating frequency for the nanotile without contact resistance is estimated to be 93GHz.

B. Power Estimation

The average dynamic power and the leakage power were estimated for the tiles of WISP-0. Dynamic power calculations were done for a 69GHz operating frequency for a range of typical operating voltages between 3V-4.5V – the voltage is estimated based on the original NW FET papers. The expression used is:

$$P_{dyn} = \sum_{pre,eva} (C_{L1} + N * C_{L2}) * V_{DD}^2 * f$$

Where f is the operating frequency, C_{L1} is the capacitance on the control nanowire and C_{L2} is the capacitance on a datapath nanowire. N is the number of datapath nanowires switching simultaneously. In cases where N is variable (e.g., application specific), an average value is chosen assuming a 50% switching probability.

TABLE VI
DYNAMIC POWER CONSUMPTION (μW)

	3V	3.5V	4V	4.5V
PC	213	290	380	481
ROM	377	509	665	841
DEC	977	1330	1738	2199
RF	2780	3784	4942	6254
ALU	447	609	795	1007

Table VI shows the dynamic power consumption (in μW) for the components of WISP-0 at the 69GHz frequency. It is seen that the Register File consumes maximum average dynamic power. This is due to a relatively large capacitive load owing to the relatively large size of the tile. The power consumption trends on the whole are orders of magnitude lower than those seen in conventional CMOS technologies.

.

TABLE VII
LEAKAGE POWER AT 4.5V (nW)

PC	10.8
ROM	10.1
DEC	24.3
RF	38.6
ALU	14.0

Leakage power consumption of NASIC tiles was estimated for a supply of 4.5V. An R_{OFF} resistance of 10GΩ[7] was used for the calculations. Table VII enumerates the calculated values for WISP-0. The high R_{OFF} implies that the leakage power in these circuits is negligibly small (in the order of nano-Watts).

VIII. Related Work

A. Nanoscale Devices for Computing

Some of the most promising underlying nanodevices today targeting digital applications, potentially applicable in 2-D computing fabrics, are based on semiconductor nanowires (such as in NASICs) and carbon nanotubes (CNTs). The diameters of NWs and CNTs are in the order of a few nanometers, and their density can be as high as 100 billion switches/cm^2 [39]. The electrical characteristics of NWs can be more reliably controlled than those of nanotubes [2]; many researchers believe therefore that NW-based devices are easier to assemble into grids and computing systems in general. Current control in NWs or CNTs is realized by using gates formed in various ways, or by forming diode junctions. FET behavior has been achieved using metallic gates [40][41] and crossing NWs or CNTs [2][41]. By varying the amount of oxide grown at their intersection, crossing CNTs or NWs can be made such that one NW forms a diode with the other, or one acts as a FET gate to the other, or they do not couple at all [2]. Rapid progress is being made in the development of feasible logic devices. Diode resistor logic was demonstrated. At the same time restoring logic was introduced with NW FET-resistor logic [2]. Avouris from IBM made important progress toward low power logic by developing complementary devices on the same nanotube and demonstrated a CMOS-like nano-inverter [43].

B. Nanoscale Computing Fabrics

Table VIII shows the comparison of four recent fabric styles. These include NASICs, NanoPLA [7], CMOL [9], and a fabric proposed by HP/UCLA [31][32]. Hewlett-Packard Research has patented a molecular crossbar latch (Kuekes, patent #6,586,965). NASICs use field-effect transistors (FETs) at nano-crossbar junctions to implement logic, rather than diodes or molecular switches such as proposed by NanoPLA and CMOL. With exception of CMOL - that implements part of the logic functions with CMOS cells connected with vertical pins to a nanogrid implementing wired-OR logic - all other fabrics assume the availability of FETs for either logic or signal restoration. NanoPLA uses the FETs in the decoder logic: this is required for addressing grid crosspoints and for reprogramming the fabric around faults. NASIC is also different from the other fabric schemes in the areas of fault tolerance and applications targeted. While most fabrics rely on reconfigurable devices, defect map extraction, and reconfiguration around defects, NASICs use built-in fault-tolerance techniques at various levels to mask faults. Only the NASIC approach might provide a solution to address faults that are caused by non-permanent defects such as device parameter variation related ones and transient faults.

Most other fabrics are targeted and evaluated for logic applications targeting FPGAs and comparison is often done

TABLE VIII
COMPARISON OF NASIC WITH OTHER NANOSCALE FABRICS

Design	Nano Devices	Targeted Applications	Defect Tolerance	CMOS Roles	Manufacturing Difficulties
NASIC	Single or complementary types of FETs	ASIC-style logic, processors	Built-in defect tolerance at various levels of granularity	Providing Vdd/Gnd.and dynamic logic control signals	Alignment during metallization of crosspoints with no FETs for logic customization
NanoPLA	Diodes + FETs as restoration	FPGA logic	Reconfiguration	Vdd/Gnd, extraction of defect maps, reconfiguration	Decoder imprint implementation or stochastic decoder; addressing all crosspoints
CMOL	Molecular switches	FPGA logic, Memory	Reconfiguration	Logic functions, signal restoration and reconfiguration	Nano-micro interface: pins with different heights required; some alignment between nano grid and CMOS cells
HP/UCLA	Diodes + two types of reconfigurable FETs	Logic	Reconfiguration	Providing Vdd/Gnd	Reconfigurable FETs

with CMOS FPGA logic. In contrast, the NASIC project and fabric focuses on processor designs and datapath. All proposals face various manufacturing difficulties at this time. The CMOL fabric has lower requirements on alignment but uses a somewhat challenging 2-level interconnect solution - with different height vertical pins that need to connect the CMOS cells to the nano grid. The NanoPLA approach requires complex defect map extraction and addressing decoder where all crosspoints need to be reached. All fabrics with exception of NASICs assume the availability of reconfigurable devices. All designs use a variant of 2-level logic as underlying logic family.

C. *Built-In Nanoscale Fault Tolerance*

While there has been little work done on fault-tolerance techniques for nanoscale fabrics, there has been a considerable amount of work done in the field of coding for fault masking in logic in the past. Much of it is based on restoring logic following logic in which faults may occur [25][26][46]. These approaches are problematic when working with crossed nanowire fabrics because the fault rates are expected to be so high that the restoring logic would itself have faults in it. Systems using residue codes either can only be used to detect errors [27], or require complicated iterative processing to correct a limited number of errors [28]. The most representative recent related work (likely developed in parallel with this work) at nanoscale is [18]; it focuses on built-in defect-tolerance at the nano-micro interface. A comprehensive overview of fault tolerance techniques focusing primarily on deep sub-micron CMOS is presented in [36]. In terms of the logic structure proposed, the interwoven logic in [29] is the closest to the one used in our work and the theory regarding critical and non-critical errors in regular logic structures appears applicable.

IX. CONCLUSIONS AND FUTURE WORK

In this paper we demonstrated a variety of built-in fault tolerance techniques on a NASIC-based processor. Our simulation results show that we can tolerate faults from a variety of sources and still achieve considerably higher density than in an equivalent CMOS design at the end of the projected ITRS roadmap. NASIC-based processors show great promise due to the combination of fault-masking, high density, and scalability. The density of NASIC-based designs scales with improvements in nano-manufacturing. Our current focus is on exploring additional techniques for fault tolerance and addressing manufacturability issues. We are working on a second nano processor with a larger bitwidth than WISP-0, incorporating additional NASIC-related architectural innovations and circuit optimizations.

ACKNOWLEDGMENT

We would like to acknowledge the fruitful collaboration on NASIC CAD tools with Drs Pottier and Lagadec from the Universite de Bretagne Occidentale, France. Furthermore, we have also received valuable input from Drs Krishna, Koren, Jackson, Anderson, Cieselski, and Tuominen from the University of Massachusetts in at Amherst, Dr Kiehl, University of Minnesota, Dr Likharev, Stony Brook University, and Dr Mircea Stan, University of Virginia. We would also like to acknowledge the support of Dr Avouris, IBM, who encouraged our early efforts in exploring nanoscale processors.

REFERENCES

[1] Y. Cui, L. J. Lauhon, M. S. Gudiksen, J. Wang, C. M. Lieber, “Diameter-controlled synthesis of single crystal silicon nanowires”, Applied Physics Letters 78, 15, pp.2214-2216, 2001.

[2] Y. Huang, X. Duan, Y. Cui, L.J. Lauhon, K-Y. Kim, and C. M. Lieber, “Logic Gates and Computation from Assembled Nanowire Building Blocks”, Science, 294, 1313 (2001).

[3] Y. Huang, X. Duan, Q. Wei, and C. M. Lieber, “Directed Assembly of One-Dimensional Nanostructures into Functional Networks”, Science 291, 5504 (2001).

[4] A. J. Bourdillon, G. P. Williams, Y. Vladimirsky and C. B. Boothroyd, “22 nm lithography using near field X-rays”, Emerging lithographic technologies VII, Proceedings of the SPIE, vol.5037, pp.622-633, June2003.

[5] A. J. Bourdillon, G. P. Williams, Y. Vladimirsky and C. B. Boothroyd “Near Field X-ray Lithography to 15 nm”, Emerging lithographic technologies VIII, Proceedings of the SPIE, vol.5374, pp.546-557, 2004.

[6] Y. Wu, J. Xiang, C. Yang, W. Lu and C. M. Lieber, “Single-crystal metallic nanowires and metal/semiconductor nanowire heterostructures”, Nature 430, 61-65 (2004).

[7] A. DeHon. “Nanowire-based Programmable Architectures”. ACM Journal on Emerging Technologies in Computing Systems, 1(2), 2005.

[8] S. C. Goldstein and M. Budiu. “Nanofabrics: Spatial Computing Using Molecular Electronics”, the 28th Annual International Symposium on Computer Architecture, ISCA’01, 2001.

[9] K. K. Likharev, “CMOL: Devices, Circuits, and Architectures. Introducing Molecular Electronics”, 2004.

[10] C. A. Moritz and T. Wang, “Towards Defect-tolerant nanoscale architectures”, Invited Paper – Proceedings of the IEEE Nano2006 conference, Cincinnati, Oh, 2006.

[11] C. A. Moritz, “Exploring NASICs and a comparison with CMOL: an architect’s perspective”, Third Advanced Research and Development Agency (ARDA) Workshop, Invited Presentation, Tampa, Fl, 2006.

[12] C. A. Moritz and T. Wang, “Latching on the wire and pipelining in nanoscale designs”, Third Non-Silicon Computing Workshop, NSC-3, organized in conjunction with 31st International Symposium on Computer Architecture (ISCA 2004), Munich, Germany, 2004.

[13] T. Wang, Z. Qi and C. A. Moritz. “Opportunities and Challenges in Application-tuned Circuits and Architectures Based on Nanodevices”, CF '04: Proceedings of the 1st ACM International Conference on Computing Frontiers, Ischia, Italy, 2004.

[14] T. Wang, M. Bennaser, Y. Guo and C. A. Moritz, “Wire-Streaming Processors on 2-D Nanowire Fabrics”, Proceedings of Nanotech 2005, Nano Science and Technology Institute, Anaheim, Ca, 2005.

[15] J. P. Patwardhan, V. Johri, C. Dwyer, A. R. Lebeck, “A Defect Tolerant Self-organizing Nanoscale SIMD Architecture”, ASPLOS’06, San Jose, CA, 2006

[16] M. Leuchtenburg, P. Narayanan, T. Wang, C. A. Moritz, “Single-Type FET Logic on 2-D Nanowire Grid”, submitted in 2006 and UMASS Technical Report.

[17] D. B. Strukov and K .K. Likharev, “Defect-Tolerant Architecture for Nanoelectronic Crossbar Memories”, Journal of Nanoscience and Nanotechnology, special issue on Nanotechnology for Information Storage, 2006

[18] P. J. Kuekes, W. Robinett, G. Seroussi and R. S. Williams, “Defect-tolerant interconnect to nanoelectronic circuits: internally redundant demultiplexers based on error-correcting codes”, Nanotechnology, vol. 16, pp. 869-882, 2005

[19] Y. W. Heo, L. C. Tien, Y. Kwon, D. P. Norton, and S. J. Pearton, “Depletion-mode ZnO nanowire field-effect transistor”, vol. 85, pp. 2274-2276, 2004

[20] S. Koo, M. D. Edelstein, Q. Li, C. A. Richterand and E. M. Vogel, ”Silicon Nanowires as Enhancement-mode Schottky Barrier Field-effect Transistors”, Nanotechnology, vol. 16, pp. 1482-1485, 2005

[21] Y. Wu, J. Xiang, C. Yang, W. Lu and C. M. Lieber, “Single-crystal Metallic Nanowires and Metal/Semiconductor Nanowire Heterostructures”, Nature, vol. 430, pp. 61-65, 2004

[22] Y. Huang, X. Duan, Y. Cui, L.J. Lauhon, K-Y. Kim and C.M. Lieber, “Logic Gates and Computation from Assembled Nanowire Building Blocks”, Science, vol. 1313, No. 294, 2001

[23] A. A. Bruen, M. A. Forcinito, “Cryptography, Information Theory, and Error-Correction”, Wiley-Interscience, 2005

[24] Y. Li, F. Qian, J. Xiang and C.M. Lieber, “Nanowire electronic and optoelectronic devices”, Materials Today 9, 18-27 (2006)

[25] J. Von Neumann, “Probabilistic logics and the synthesis of reliable organisms from unreliable components”, Automata Studies (Annals of Math. Studies No. 34), Ed. C. E. Shannon and J. McCarthy, Princeton, N.J.: Princeton Univ. Press, 1956, pp. 43-98.

[26] D. B. Armstrong, “A general method of applying error correction to synchronous digital systems”, Bell Syst. Tech. J., vol. 40, 1961, pp. 477-593.

[27] I. L. Sayers and D. J. Kinniment, “Low-cost residue codes and their applications to self-checking VLSI systems”, IEE Proceedings, vol. 132, Pt. E, No. 4, July 1985.

[28] H. Krishna and J.D. Sun, “On Theory and Fast Algorithms for Error Correction in Residue Number System Product Codes”, IEEE Transactions on Computers, Vol. 42, No. 7, July 1993.

[29] W. H. Pierce, “Interconnection structure for Redundant Logic, Failure-Tolerant Computer Design”, Academic Press, 1965.

[30] R. E. Lyions and W. Vanderkulk, “The use of triple modular redundancy to improve computer reliability”, IBM Journal of Research and Development, 6(2), 1962.

[31] J. R. Heath, P. J. Kuekes, G. S. Snider, and R. S. Williams, “A defect-tolerant computer architecture: Opportunities for nanotechnology”, Science 280, 5370 (June 1998), 1716--1721.

[32] Y. Luo, C. P. Collier, J. O. Jeppesen, K. A. Nielsen, E. DeIonno, G. Ho, J. Perkins, H.-R. Tseng, T. Yamamoto, J. F. Stoddart, J. R. Heath, “Two-Dimensional Molecular Electronics Circuits”, ChemPhysChem. 2002, 3, 519--525.

[33] Y. Cui, X. Duan, J. Hu1, and C. M. Lieber, “Doping and Electrical Transport in Silicon Nanowires”, Journal of Physical Chemistry, vol. 104, 2000.

[34] A. B. Greytak, L. J. Lauhon, M. S. Gudiksen, and C. M. Lieber, “Growth and transport properties of complementary germanium nanowire field-effect transistors, Applied Physics Letters, 84(21), May 2004.

[35] H. T. Ng, J. Han, T. Yamada, P. Nguyen, Y. P. Chen, and M. Meyyappan, “Single Crystal Nanowire Vertical Surround-Gate Field-Effect Transistor”, Nano Letters 4(7):1247-1252, American Chemical Society, 2004.

[36] T. Lehtonen, J. Plosila, and J. Isoaho, “On Fault Tolerance Techniques Towards Nanoscale Circuits and Systems”, TUCS Technical Report, No 708, Turku Centre for Computer Science, August 2005.

[37] B. D. Gates, Q. Xu, J. Ch. Love, D. B. Wolfe, and G. M. Whitesides, “Unconventional Nanofabrication”, Annual Reviews Mater. Res. 2004.

[38] C. Constantinescu, “Trends and challenges in VLSI circuit reliability”, IEEE Micro, Volume: 23, Issue:4, pages: 930-931, ISSN: 0013-5194

[39] T. Rueckes, K. Kim, E. Joselevich, G.Y. Tseng, C-L. Cheung, and C.M. Lieber, “Carbon Nanotube-Based Nonvolatile Random Access Memory for Molecular Computing”, Science, 289, 94, 2000.

[40] R. Martel, V. Derycke, J. Appenzeller, S. Wind, and Ph. Avouris, “Carbon Nanotube Field-Effect Transistors and Logic Circuits”, DAC 2002, New Orleans, 2002.

[41] A. Bachtold, P. Hadley, T. Nakanishi, C. Dekker, “Logic Circuits with Carbon Nanotube Transistors”, Science, 294, 1317, 2001.

[42] A. B. Greytak, L. J. Lauhon, M. S. Gudiksen, C. M. Lieber, “Growth and transport properties of complementary germanium nanowire field effect transistors,” Applied Physics Letters, 84, 4176, 2004.

[43] Ph. Avouris, R. Martel, V. Derycke, J. Appenzeller, “Carbon Nanotube Transistors and Logic Circuits”, Physica B Condensed Matter, Vol. 323, pp. 6-14, October 2002.

[44] R. C. Rose and D. K. Ray-Chaudhuri, “On a class of error-correcting binary group codes”, Inform. And Contr., vol. 3, pp 68-79, March 1960.

[45] A. Hocquengham, “Codes correcteurs d’erreurs”, Chiffre, vol. 2, pp. 147-156, September 1959.

[46] T. R. N. Rao, “Error Coding for Arithmetic Processor”, Academic Press, ISBN 0-12-580750-4, 1974.

[47] B. W. Johnson, “Design and Analysis of Fault-Tolerant Digital Systems”, Addison-Wesley Publishing, ISBN 0-201-07570-9, 1989.