Invited Paper

Robust System Design

Subhasish Mitra,[†1] Hyungmin Cho,[†1] Ted Hong,[†1]
Young Moon Kim,[†1] Hsiao-Heng Kelin Lee,[†1]
Larkhoon Leem,[†1] Yanjing Li,[†1] David Lin,[†1]
Evelyn Mintarno,[†1] Diana Mui,[†1] Sung-Boem Park,[†1]
Nishant Patil,[†1] Hai Wei[†1] and Jie Zhang[†1]

Robust system design is essential to ensure that future electronic systems perform correctly despite rising complexity and increasing disturbances. In contrast, today's mainstream systems typically assume that transistors and interconnects operate correctly over their useful lifetime. Future systems cannot rely on such assumptions for several reasons: 1. With enormous complexity, future systems are significantly vulnerable to design flaws. 2. For coming generations of silicon technologies, several causes of hardware failures, largely benign in the past, are becoming significant at the system-level. 3. Emerging nanotechnologies, such as carbon nanotubes, are inherently highly subject to imperfections. At the same time, there is explosive growth in our dependency on electronic systems. This paper addresses the following major robust system design goals: 1. New approaches to thorough validation that can cope with tremendous growth in complexity. 2. Cost-effective tolerance and prediction of failures in hardware during system operation. 3. Practical ways to overcome substantial inherent imperfections in emerging nanotechnologies. Significant recent progress in robust system design impacts almost every aspect of future systems, from ultra-large-scale computing and storage systems, all the way to their nanoscale components.

1. Introduction

Electronic systems are an indispensable part of all our lives. Malfunctions in these systems have consequences ranging from annoying computer crashes, loss of data and services, to financial and productivity losses, or even loss of human life. For example, in 2009, a glitch in a circuit board in the air-traffic control system resulted in hundreds of flights being canceled or delayed. Such impacts continue to increase as systems become more complex, interconnected, and pervasive. *Robust system design* is required to ensure that future systems perform correctly despite rising levels of complexity and increasing disturbances.

Malfunctions may be caused by hardware or software failures, design errors, malicious attacks, or incorrect human interactions. Hardware failures are especially a growing concern due to the following factors:

1. Existing validation methods can barely cope with today's complexity. New techniques are essential to minimize the effects of design flaws going forward.

2. With extreme miniaturization of circuits, factors such as transient errors, device degradation, and variability induced by manufacturing and operating conditions are becoming important. While design margins are being squeezed to achieve high energy efficiency, expanded design margins are required to cope with variability and circuit aging. Even if error rates stay constant on a per-bit basis, total chip-level error rates will grow with the scale of integration. Moreover, difficulties with traditional burn-in can leave early-life failures unscreened. As a result, a large class of future systems will require tolerance of hardware errors during their operation.

3. Emerging nanotechnologies are inherently prone to high rates of imperfections. Nevertheless such technologies are being seriously explored to build highly energy-efficient systems of the future. The inherent imperfections must be overcome before such nanotechnologies can be harnessed with practical benefits to society.

This paper addresses these outstanding challenges, ranging from immediate concerns blocking progress today to major obstacles in exploratory nanotechnologies, as described in the following sections:

1. Thorough post-silicon validation despite enormous complexity (Section 2).

2. Tolerance and prediction of hardware failures (Section 3).

3. Correct circuit operation in emerging nanotechnologies prone to imperfections (Section 4).

†1 Robust Systems Group, Department of Electrical Engineering and Department of Computer Science, Stanford University

2. Effective Post-silicon Validation Techniques

Post-silicon validation involves operating manufactured chips in actual application environments to validate correct behaviors across specified operating conditions. Post-silicon validation is becoming significantly expensive. Intel reported headcount ratio of 3 : 1 for design vs. post-silicon validation[121]. According to Ref. 1), post-silicon validation may consume 35% of average chip development time. Reference 163) observes that post-silicon costs are rising faster than design costs.

Traditionally, most hardware bugs are caught during pre-silicon verification. While pre-silicon verification is essential, it alone is not adequate. Post-silicon validation is becoming essential as well for two main reasons:

1. Simulation speed is several orders of magnitude slower than actual silicon. *Functional bugs*, also called *logic bugs*, caused by design errors often get exposed during post-silicon validation due to increasing design complexity and design schedule constraints.

2. For complex systems, bugs caused by electrical interactions, also called *electrical bugs*, are becoming important[54]. Such bugs generally manifest themselves only under certain operating conditions (temperature, voltage, frequency). Examples include setup and hold time problems, signal integrity, and circuit marginality. Accurate modeling of such interactions is difficult during pre-silicon verification.

Post-Silicon validation involves four major steps:

1. **Detect** a problem by applying proper stimuli, e.g., operating system, games, or functional tests, until a system failure occurs, e.g., crashes, exceptions.

2. **Localize** the problem to a small hardware block from the system failure. The stimulus that exposes the bug, e.g., the particular few lines of code from some application, is also important.

3. **Identify** (or **debug**) the root cause of the problem, e.g., an electrical bug that may be caused by power-supply noise slowing down one or more circuit paths in the hardware block identified during bug localization.

4. **Fix or bypass** the problem by (software) patching, circuit editing, or re-spinning using a new mask.

As pointed out in Ref. 55), the second step dominates post-silicon validation efforts and costs. Two major factors that contribute to the high cost of existing post-silicon bug localization approaches are:

1. *Failure reproduction* which involves bringing the hardware back to an error-free state, and re-executing the failure-causing stimulus (including instruction sequences, interrupts, and operating conditions) to reproduce the same failure. Unfortunately, many bugs are very hard to reproduce. The difficulty of bug reproduction is exacerbated by asynchronous I/Os, and multiple clock domains. Techniques to make failures reproducible[45],[136],[144] can be intrusive, and may not expose bugs.

2. Traditional bug localization techniques often rely on system-level simulation for obtaining golden responses, i.e., correct signal values, for every clock cycle for the entire system, i.e., the processor and peripheral devices. System-level simulation can be 6 to 9 orders of magnitude slower than actual silicon. Moreover, expensive external logic analyzers are required to record signal values that enter and exit through external pins[144].

As a result of the above factors, a functional bug may require hours or days to be localized vs. electrical bugs that may require days to weeks, and more expensive equipment[54].

The bug localization step cannot be decoupled from the detection step. In Sections 2.1 and 2.2, we address both these steps in an integrated fashion. Section 2.1 presents the IFRA technique for post-silicon bug localization in processors. In Section 2.2, we present the QED technique which enables IFRA-style bug localization to be applied to single-core processors and multi-core System-on-Chips (*SoCs*).

2.1 IFRA Post-silicon Bug Localization

IFRA, an acronym for Instruction Footprint Recording and Analysis, overcomes the aforementioned challenges associated with post-silicon bug localization[111]. **Figure 1** presents an overview of IFRA. Special on-chip recorders, inserted in a processor during design, collect *instruction footprints* — special information about flows of instructions, and what the instructions did as they passed through various microarchitectural blocks of the processor. The recording is done concurrently during the normal operation of the processor in a post-silicon system

Table 1 Comparison of various post-silicon bug localization techniques.

	Formal methods	Trace buffer	Scan methods	Clock manipulation	Program & Data tracing	Checkpoint & replay	Assertion checking	IFRA
Intrusive?	(+)No	Depends	(-) Yes		Depends	(-) Yes	Depends	(+) No
Failure reproduction?	(-) Yes						N/A	(+) No
System-level simulation?	(+)No	(-) Yes					(+)No	(+) No
Area impact?	(-) Yes		(+) No	(-) Yes	(+) No		(-) Yes	(-) 1%
Applicability?	(+)General				(-) Processor		depends	(-) Processor

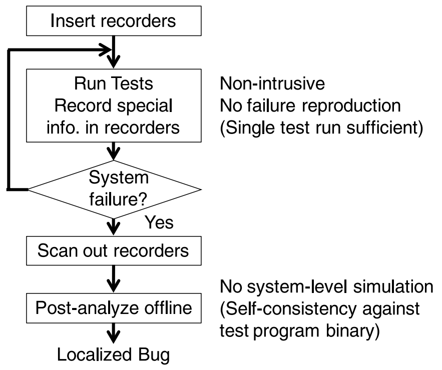


Fig. 1 Post-silicon bug localization flow using IFRA.

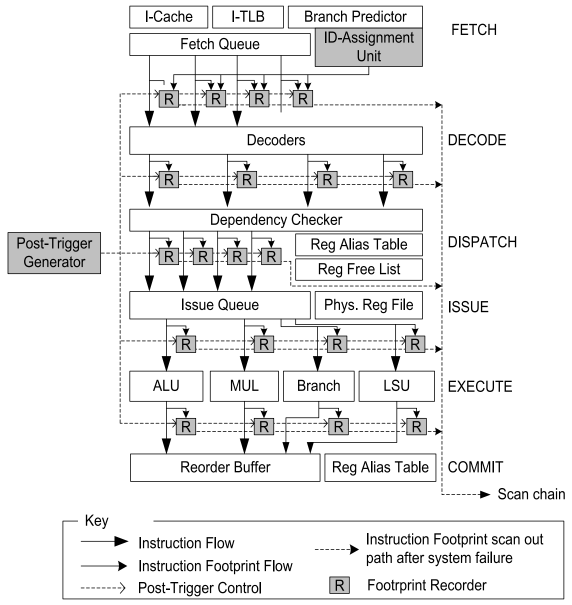


Fig. 2 Superscalar processor augmented with IFRA recording infrastructure.

validation setup. Upon detection of a system failure, the recorded information is scanned out and analyzed offline for bug localization. Special self-consistency-based program analysis techniques, together with the test program binary of the application executed during post-silicon validation, are used for this purpose. Major benefits of IFRA over existing post-silicon bug localization techniques are:

1. It does not require full system-level reproduction of bugs.
2. It does not require full system-level simulation.

Hence, it can overcome the major hurdles that limit the scalability of existing techniques. **Table 1** compares and contrasts various post-silicon bug localization techniques. **Figure 2** shows three hardware components of IFRA's recording infrastructure for a superscalar processor (listed below). These components incur an overall chip-level area cost of 1%.

1. *Distributed recorders*, denoted by 'R' in Fig. 2, are placed at each pipeline

Table 2 Auxiliary information for Alpha 21264-like superscalar processor.

Pipeline stage	Auxiliary information		Number of Recorders	Entries per Recorder
	Description	Bits per entry		
Fetch	Program counter	32	4	1,024
Decode	Decoded bits	4	4	1,024
Dispatch	2-bit residue of register name	6	4	1,024
Issue	3-bit residue of operands	6	4	1,024
ALU, MUL	3-bit residue of result	3	4	1,024
Branch	None	0	2	1,024
LSU	3-bit residue of result; memory address	35	2	1,024
Commit	Fatal exceptions	4	1	1
Total storage required for all recorders: Each entry contains an additional 8-bit instruction ID			60 Kbytes	

stage to collect footprints of instructions leaving that pipeline stage. Each instruction footprint that is associated with a pipeline stage consists of an instruction ID and *auxiliary information* (**Table 2**) describing what the instruction did at that pipeline stage. Note that, a given instruction will have multiple footprints in recorders belonging to multiple pipeline stages. The total distributed recorder storage amounts to 60 Kbytes for an Alpha 21264-like processor. The recording is done in a circular fashion; i.e., the last few thousand footprints are stored in a recorder at any time.

2. An *ID (identifier) assignment unit* assigns an ID to each instruction that enters the processor. Simplistic ID assignment schemes are inadequate for processors that may contain multiple clock domains, and/or may execute instructions speculatively or out of order. A special ID assignment scheme in Ref. 111) enables very short IDs (8 bits for an Alpha 21264-like processor) to be used during post-analysis to identify instruction footprints that belong to each instruction that entered the processor.

3. A *post-trigger generator* detects symptoms of system failure[111)]. Upon detection of symptoms, the recording is paused speculatively while the system keeps running. If the symptom is found to be false, recording is resumed; else, the system is halted, and the recorded instruction footprints are scanned out.

After instruction footprints are scanned out, they are post-processed in three phases to localize the bug that caused the system failure. First, *footprint linking* groups together footprints belonging to the same instruction but stored across multiple recorders. The linked footprints are also mapped to the corresponding instruction in the test program binary (since the fetch stage recorder stores the program counter content, as shown in Table 2).

After footprint linking, two sets of self-consistency-based checks are executed to identify any contradictory events in linked footprints with respect to the test program binary. A set of microarchitecture-independent checks, or *high-level analysis*, finds the first sign of an inconsistency in program execution. The inconsistency is represented in the form of a <*location*, *footprint*> *pair*. The *location element* of the pair specifies a hardware block, and the *footprint element* is a pointer to an entry in one of the recorders. This pair serves as the starting point for a set of microarchitecture-dependent checks, or *low-level analysis*, which asks a series of microarchitecture-specific questions according to a **manually-generated** decision diagram (such a decision diagram for an Alpha 21264-like processor is presented in Ref. 111)) to identify a set of bug candidates. Each candidate is represented in the form of <location, footprint> pair. The location indicates the hardware block in which the bug produces an error. The footprint indicates a cycle in which the error occurred relative to the cycle when the post-trigger occurred.

Figure 3 shows the localization results obtained using an open-source simulator modeling a complex Alpha 21264-like superscalar processor (details in Ref. 111)). Single bit-flips were injected at flip-flops to model hard-to-repeat electrical bugs. This is an effective model because most electrical bugs eventually manifest themselves as incorrect values arriving at flip-fops for certain input combinations and operating conditions[84)].

The results demonstrate that IFRA is effective in accurately localizing electrical bugs with 1% chip-level area impact and no performance impact. For over 96% of

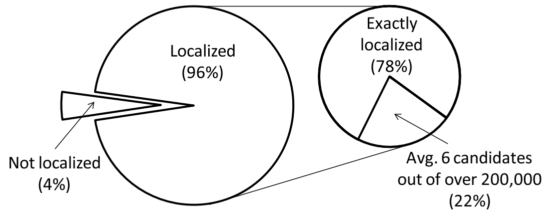


Fig. 3 IFRA bug localization results for an Alpha 21264-like superscalar processor.

the cases, IFRA identified the correct candidate that matched both the location (1 out of 200) and cycle (1 out of over 1,000 footprints) of error injection. Out of these 96% cases with correct localization, IFRA returned a single correct bug candidate, i.e., <location, footprint> pair, for 78% of the cases (referred to as *exactly localized* in Fig. 3), and returned multiple candidates (average of 6 out of over 200,000 possible <location, footprint> pairs) for the remaining 22% of the cases.

Although IFRA is applicable to any microarchitecture, the manual effort required to create microarchitecture-dependent self-consistency checks can be significant. The *Bug Localization Graph (BLoG)* technique in Ref. 112) provides a systematic approach for generating such checks with significantly reduced manual effort.

Once a BLoG, consisting of BLoG nodes and edges, is constructed, the post-silicon bug localization flow of the original IFRA technique is followed until the high-level analysis step (**Fig. 4**). The inconsistency returned from the high-level analysis designates an edge in the BLoG. Starting from this edge, BLoG traversal is performed using algorithms presented in Ref. 112). As a byproduct of BLoG traversal, low-level analysis is performed, and bug candidates (in the form of <location, footprint> pairs) are returned.

Figure 5 shows bug localization results obtained by injecting single bit-flips into an industrial microarchitectural simulator modeling an advanced and complex commercial microarchitecture: the Intel Nehalem, the foundation for the Intel Core$^{\mathrm{TM}}$ i7 and Core$^{\mathrm{TM}}$ i5 processor families. It demonstrates that BLoG-assisted IFRA enables effective and efficient post-silicon bug localization for complex processors with high bug localization accuracy and at low cost.

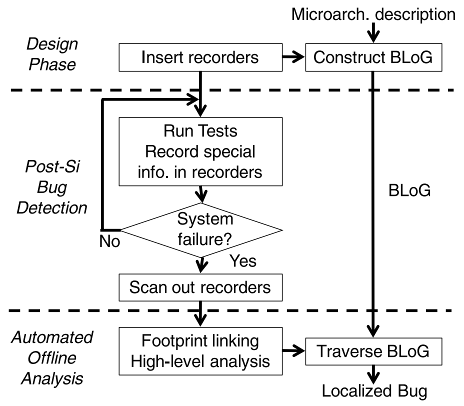


Fig. 4 Bug localization flow using BLoG-assisted IFRA.

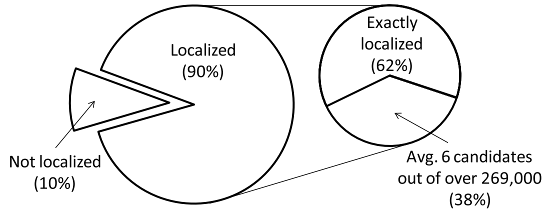


Fig. 5 BLoG-assisted IFRA bug localization results for the Intel Nehalem architecture.

IFRA and BLoG create several research opportunities:

1. Automatically constructing BLoG from RTL or a language-based specification[38),42)].

2. Automatically selecting information to record by borrowing concepts from circuit-level tracing[65)].

3. Enhancing IFRA and BLoG for homogeneous/heterogeneous multi-core SoCs. The QED technique, presented in Section 2.2, is a step toward this objective.

2.2 QED: Quick Error Detection Tests for Effective Post-silicon Validation

Any post-silicon bug localization and debug technique, including IFRA, requires effective post-trigger mechanisms to detect errors. Long *error detection* *latency*, the time elapsed between the occurrence of an error caused by a bug and its detection at an observable point in the test program, limits the effectiveness of most techniques that rely on simulation, formal analysis, or tracing. We already discussed earlier that simulation is orders of magnitude slower than actual silicon. Formal analysis over more than hundreds of cycles can be difficult [47]. Tracing is limited by the availability of on-chip storage [1]. Long error detection latencies can also result in increased error masking, i.e., an error may not propagate to an observable point. The *Quick Error Detection (QED)* technique in Ref. 48) overcomes the challenges associated with long error detection latencies during post-silicon validation.

QED tests are obtained by transforming existing validation tests into new tests with significantly lower, i.e., improved, error detection latencies. QED tests are created by a wide variety of automatable and systematic *QED transformations*, requiring either software-only or hardware-software modifications. QED transformations allow flexible tradeoffs between error detection latency, coverage (the percentage of bugs detected by a test program), and complexity (additional hardware and software modifications required for QED). Target error detection latencies are configurable, and can range from very few cycles to a few thousand cycles, depending on the desired tradeoffs. Due to space limitations, this paper only provides details for the *Error Detection by Duplicated Instructions for Validation* (*EDDI-V*) transformation. Other QED transformations can be found in Ref. 48).

EDDI-V is a QED transformation that extends the EDDI technique used in fault-tolerant computing [107]. Unlike EDDI, EDDI-V ensures bounds on target error detection latencies, and allows configurability to trade off target error detection latency for less intrusiveness. *Intrusiveness* is defined as the amount of "deviation" in the runtime behavior of a QED test from that of the original test (due to the incorporation of QED transformations). EDDI-V does not require hardware modifications, and can be automated.

EDDI-V strategically duplicates instructions, and compares their results. As illustrated in **Fig. 6**, each "block" of instructions is duplicated, and a check is inserted to compare the results of the two blocks. If the check detects an error that occurs in these blocks, the error detection latency is bounded by the sum of:

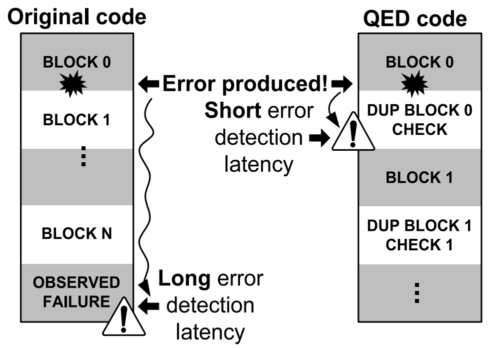


Fig. 6 Error detection latency of original validation test vs. EDDI-V-based QED test.

1. The time elapsed between the start of the original block and the end of the duplicated block.
2. The time it takes to perform the check.

This enables a great reduction in error detection latency vs. the original program, which may detect errors only after a visible failure, e.g., a crash, or using its original checks (if available, e.g., end-result-checks that compare actual program outputs to expected outputs).

EDDI-V and EDDI have different tradeoffs and requirements. EDDI strikes a balance between performance impact and the need for error containment and recovery. Targeting post-silicon validation, the performance impact of EDDI-V's frequent checking is not a primary concern. Instead, we support flexibility in EDDI-V to trade off target error detection latency for less intrusiveness. This is achieved by varying two parameters: *Inst_min* and *Inst_max* that correspond to the minimum and maximum number of original instructions executed, respectively, before any instructions inserted by QED execute (*Inst_min* must be less than or equal to *Inst_max*). Increasing *Inst_min* reduces intrusiveness and vice-versa.

EDDI-V is implemented by reserving half of the general-purpose registers and memory space for the original instructions, while the other half is used by the duplicated instructions. For example, in **Fig. 7** (a), the two original instructions in the body section of the code use four registers (A, B, C, and D). These two

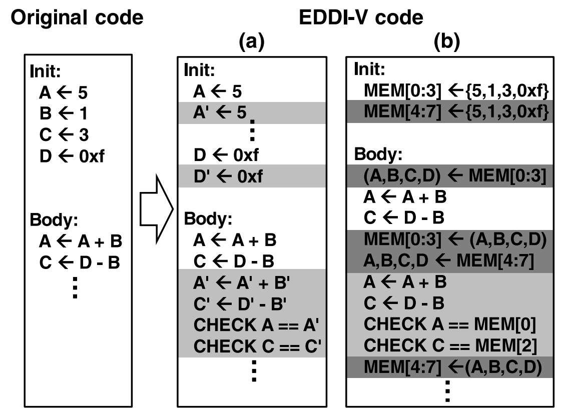


Fig. 7 EDDI-V transformations. (a) Half of all general-purpose registers reserved. (b) No registers reserved and register values stored in memory.

instructions are duplicated, and another set of registers (A′, B′, C′, and D′) is used by the duplicated instructions. In situations where there are insufficient registers, values can be stored temporarily in memory (also partitioned into two halves to be used by the original and duplicated instructions). Values stored in memory are then re-loaded for comparison (Fig. 7 (b)). Each half of the general-purpose registers and memory space are identically initialized so that original and duplicated instructions perform identical operations and obtain identical results in a bug-free system. Checking is performed by comparing the results of the original instructions vs. their duplicates. Any mismatch in the comparison indicates an error. Special analysis is required for certain code structures such as loops, conditionals, and synchronization primitives such as locks. Techniques to handle these code structures are detailed in Ref. 48).

Another family of QED transformations is *Redundant Multi-Threading for Validation* (RMT-V), which are based on *redundant multi-threading* from fault-tolerant computing. RMT-V executes the original instructions on one thread, and uses an additional thread to execute the duplicated and check instructions. The two RMT-V threads can be simultaneously executed on different cores. As a result, the execution of the original instructions would potentially be less affected by the execution of the duplicated and check instructions compared to EDDI-V, implying reduced intrusiveness. RMT-V also provides flexible trade-offs between target error detection latency and intrusiveness. The RMT-V family of QED transformations consists of Software RMT-V (*S-RMT-V*), S-RMT-V with Hardware Queues (*S-RMT-V-HQ*), and Hardware RMT-V (*H-RMT-V*). **Table 3** presents a qualitative comparison of various QED transformations (details in Ref. 48)).

Table 3 Comparsion of various QED transformations.

	EDDI-V	S-RMT-V	S-RMT-V-HQ	H-RMT-V
Software modifications	Some	Some	Some	None
Hardware modifications	None	None	Very small	Some
Intrusiveness	Flexible	Small	Smaller	Smallest
Error detection latency	Flexible	Flexible	Flexible	Small

QED tests can be further extended to reduce the likelihood of errors identically affecting both the original and duplicated blocks by executing them "differently" through the incorporation of design diversity into QED, e.g., data, time, or algorithmic diversity [90),108)]. Detailed analysis of diversity-enhanced QED is beyond the scope of this paper.

To evaluate the effectiveness of QED tests, we performed hardware error injection experiments on actual quad-core Intel® Core$^{\mathrm{TM}}$ i7 processors. Details for our hardware error injection experiments can be found in Ref. 48). Our hardware results demonstrate that QED tests improve error detection latencies by six orders of magnitude as compared to the original (non-QED) tests, from billions of cycles to a few thousand cycles (**Fig. 8**). This result also matches our results from simulation-based error injections using a 4-core 4-way out-of-order MIPS-based microprocessor using the SESC microarchitectural model from Ref. 131) and RTL model from Ref. 155). Furthermore, in our hardware error injection experiments, QED tests detected four times more errors compared to the original (non-QED) tests (Fig. 8).

In addition to hardware error injection, we performed Shmoo experiments

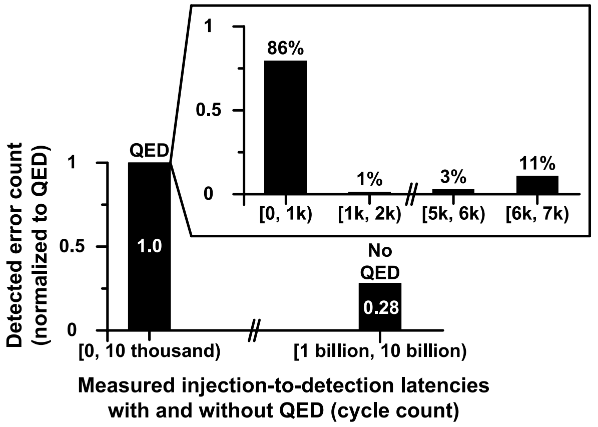


Fig. 8 Distribution of measured hardware error injection to detection latencies for the Linpack test on Intel® Core™ i7 hardware platform.

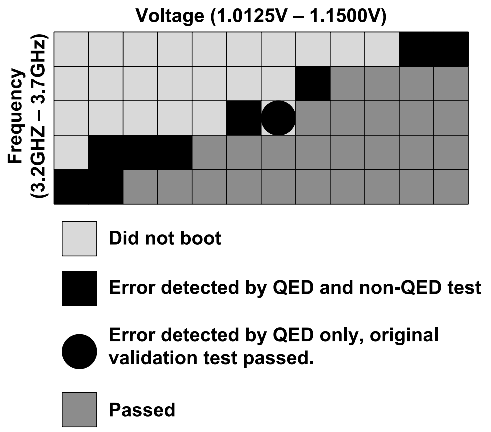


Fig. 9 Shmoo plot of Linpack test with EDDI-V-based QED on Intel® Core™ i7 hardware platform.

(**Fig. 9**) on a quad-core Intel® Core™ i7 hardware platform to demonstrate that QED transformations do not degrade coverage of the original (non-QED) test, as quantified empirically by *Fmax values*, i.e., maximum operating frequencies over a wide range of operating voltage points. As Fig. 9 shows, our QED test continues to detect errors for the cases where the original test detects errors. Coverage improvement by the QED test is indicated by the voltage and frequency operating point in Fig. 9 that passed the original (non-QED) test, but resulted in detected errors with the QED test (labeled with ● in Fig. 9).

Our comprehensive hardware results (supported by simulations) demonstrate that QED is an effective technique that overcomes the challenges of long error detection latencies in post-silicon validation of processor cores. QED can be used in conjunction with existing post-silicon debugging techniques such as IFRA (Section 2.1), Backspace[25], trace buffers[1], and formal analysis[47]. Future research directions of QED include:

1. Generalization of QED to detect logic bugs.
2. An automated framework that includes an optimal mix of assertions together with QED transformations.
3. Generalization of QED to uncore components of SoCs, i.e., components other than the processing cores, e.g., various interfaces, interconnects, and memory and I/O controllers[127].

3. Overcoming CMOS Reliability Challenges

For silicon CMOS ICs with remarkably small geometries, several hardware reliability failure mechanisms, largely benign in the past, are becoming significant at the system-level[9],[46],[66],[102]. Major hardware reliability challenges include early-life failures (also called infant mortality), radiation-induced soft errors, and circuit aging, as summarized in **Fig. 10**.

Design of robust systems to ensure required hardware reliability, while non-trivial, is achievable but at high costs, e.g., using classical fault-tolerant computing techniques. An extremely important aspect of such systems is *concurrent error detection*, which continuously monitors system data and states during normal system operation to detect errors[89]. Error detecting and correcting codes, together with scrubbing and sparing, are routinely used for errors in memories[40],[130],[142]. For errors in logic, existing error detection techniques are generally expensive. Both duplication and Triple Modular Redundancy (TMR) have been used in commercial systems, e.g., IBM's z990 and HP's Non-Stop Advanced

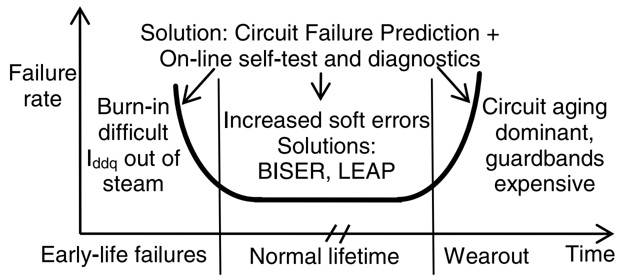


Fig. 10 Silicon CMOS reliability challenges.

Architecture. These techniques introduce significant power/performance/area penalties, and additional system design complexities[7),85)].

Parity codes can be used for error detection in arbitrary logic circuits using *parity prediction*. It constrains the amount of logic sharing among the circuits producing the outputs being checked[89)]. These constraints result in significant power/performance/area penalties. By relaxing logic sharing constraints, the associated costs can be reduced, but at the price of error detection coverage. Residue codes have been used for arithmetic circuits such as multipliers[2)].

Time redundancy techniques for error detection[99),107),108),114),137)], using a variety of software and hardware techniques, reduce some hardware overheads, but introduce additional performance penalties.

Error detection costs may be reduced in one or more of the following ways:

1. Detection of a sub-class of errors, e.g., delay faults[13),33),34),104)].

2. Application-specific error detection, e.g., using assertions[80),82),103),122)] and Algorithm-Based Fault-Tolerance[50)].

Once an error is detected, the system must be reliably recovered, e.g., using checkpoint/rollback recovery[32),125)]. Such recovery techniques can be difficult to validate for complex systems.

New opportunities for cost-effective robust system design are created by emerging technology trends and killer applications. Examples include:

1. An abundance of transistors may be available, while long wires and power will continue to pose major challenges[11),41)].

2. Availability of high-density and low-cost non-volatile storage such as FLASH memory[69)].

3. Proliferation of multi-/many-core architectures and specialized hardware to reduce power costs.

4. Resilience of several emerging workloads such as Recognition, Mining and Synthesis (RMS) to errors[30),71)].

This section presents an overview of techniques which utilize these opportunities to overcome CMOS reliability challenges:

1. BISER and LEAP (Section 3.1) to correct soft errors.

2. Circuit failure prediction (Section 3.2), together with CASP online self-test and diagnostics (Section 3.3), to overcome early-life failures and circuit aging challenges.

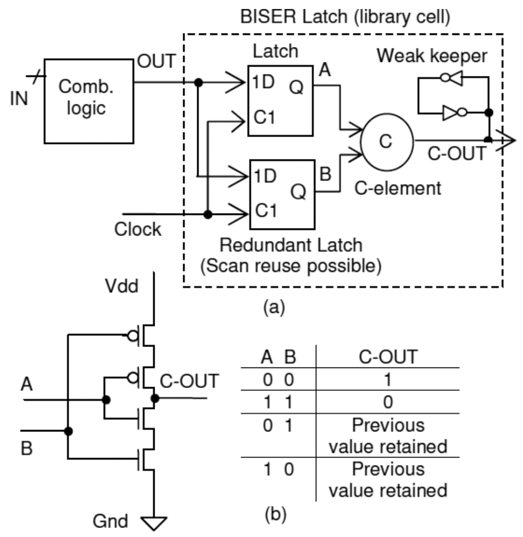
A key feature of these techniques is global optimization across multiple layers of the system stack — device, circuit, architecture, runtime, and application — also referred to as *cross-layer resilience*[95),97)].

3.1 Soft Errors: BISER and LEAP

Soft errors are radiation-induced transient errors caused by neutrons generated from cosmic rays and alpha particles from packaging material. Terrestrial radiation is a growing concern, and most future enterprise computing and communication systems will require some degree of soft error protection of *sequential elements* (latches and flip-flops)[6),93),100),135)], in addition to on-chip SRAMs.

The *Built-In Soft Error Resilience (BISER)* technique corrects soft errors in latches, flip-flops and combinational logic[93),94),164)]. **Figure 11** illustrates BISER for correcting soft errors in latches. When Clock = 1, the latch input is strongly driven by the combinational logic, and the latch is not susceptible to soft errors. (This follows from timing derating[106)]). When Clock = 0, C-OUT already has the correct value — any single soft error in either latch results in a situation where the logic value on A does not agree with B. As a result, the error does not propagate to C-OUT, and the correct logic value is held at C-OUT by the keeper. A soft error in the keeper does not have a major effect because C-OUT is strongly driven by the latch contents (assuming single error).

Radiation experiment results from 45 nm test chips demonstrate two to three orders of magnitude reduction in soft error rates of sequential elements using BISER[139)]. Correction of soft errors in combinational logic circuits using BISER



A B	C-OUT
0 0	1
1 1	0
0 1	Previous value retained
1 0	Previous value retained

Fig. 11 Built-In Soft Error Resilience (BISER). (a) BISER latch design. (b) C-element.

and its derivatives are described in Refs. 35), 94).

Global optimization to minimize the power impact of BISER requires tight coupling between circuit design, architecture, and application. Error injection simulations on an Alpha-like microprocessor show that BISER improves chip-level soft error rate by 10 times with 7–10% chip-level power penalty 164). This is because not all soft errors are equally important at the system-level. This creates BISER insertion opportunities for maximized protection at minimum cost, e.g., using Ref. 141) and many others. BISER area costs can be significantly reduced by reusing on-chip scan resources for post-silicon validation and testing 93),164). Moreover, as shown in Ref. 97), BISER may be co-optimized logic parity checking to minimize overall system-level costs.

Additional opportunities exist for application-aware optimization of resilience using BISER by "dialing" reliability vs. power costs on-the-fly. BISER can be configured, dynamically during system operation, to operate in one of two modes — an *error resilient mode* in which BISER protection is turned on, and an *economy mode* in which BISER protection is turned off 164). However, information

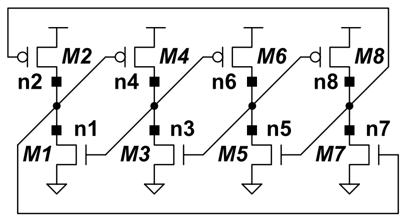


Fig. 12 Dual Interlocked Cell (DICE).

flow across system stack layers to utilize such configurability is an open question.

Most techniques for designing soft-error-resilient sequential elements, including BISER, generally address single errors caused by single event upsets (*SEU*s). With technology scaling, the probability of *Single Event Multiple Upsets*, or *SEMUs*, where multiple transistor diffusion nodes can simultaneously collect charge produced from a single particle strike (also referred to as *Multiple-Bit Upsets* or *MBUs*), is increasing 29),140). SEMU analysis in Ref. 138) indicates the superiority of BISER in correcting SEMUs compared to a well-known technique called *DICE* or Dual Interlocked Cell design 16) (**Fig. 12**). The LEAP approach, described next, enables cost-effective soft error correction in the presence of SEMUs.

LEAP (Layout design through Error-Aware transistor Positioning) is a new layout design principle. It reduces soft error sensitivity of a circuit by careful placement of transistors such that the transistors can act together to reduce overall charge collection 70).

Consider the effect of a particle striking in the vicinity of a MOS transistor. A particle strike produces electron-hole pairs along its track, which then drift according to the electric field present. If the charge is generated near an NMOS transistor, a net negative charge is collected at the source and drain contacts, resulting in a positive current pulse (into the silicon) 29). For a PMOS transistor, a net positive charge is collected at the source and drain contacts, resulting in a negative current pulse. Using the LEAP principle, NMOS and PMOS diffusion contact nodes are placed in such a way that multiple diffusion nodes collect charges and act together to cancel (fully or partially) the overall effects of the single event. **Figure 13** shows how the transistors can act together to reduce

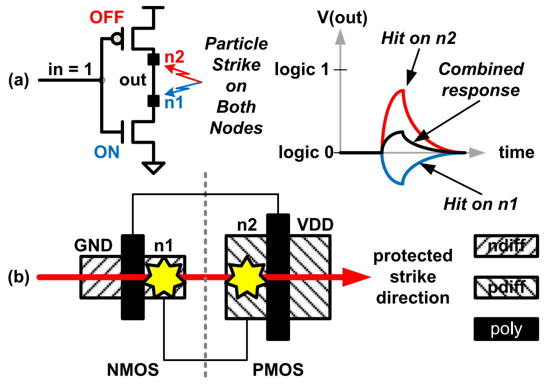


Fig. 13 LEAP principle for an inverter. (a) Reduced charge collection when a particle hits drain contact nodes n1 and n2. (b) Protected particle strike direction in layout.

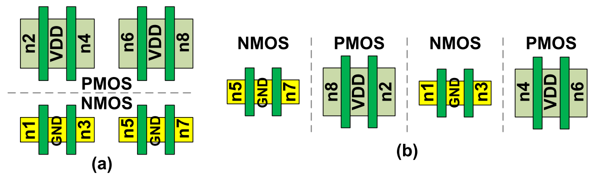


Fig. 14 (a) Standard DICE layout. (b) LEAP-DICE layout.

overall charge collection in an inverter [70)].

The LEAP principle can complement existing SEU-resilient circuit techniques to produce new sequential elements that are resilient to both SEUs and SEMUs. Reference 70) presents a new LEAP-based soft-error-resilient sequential element called LEAP-DICE (**Fig. 14** (b)) by applying LEAP to the DICE design in Fig. 12. Note that, LEAP-DICE preserves the original DICE circuit design, but uses a special layout design based on LEAP.

Experimental results using 200 MeV proton testing demonstrate that our LEAP-DICE flip-flop encounters 2,000-fold fewer errors on average compared to a traditional D flip-flop, and 5-fold fewer errors compared to our reference DICE flip-flop. The LEAP-DICE design incurs 40% more area cost, and negligible additional power and delay costs (compared to the DICE layout in Fig. 14 (a) without LEAP).

3.2 Early-life Failures (ELF) and Circuit Aging: Circuit Failure Prediction

Circuit failure prediction provides early indication of the occurrence of a circuit failure **before** errors actually appear in system data and states [3)]. This is in contrast to concurrent error detection where a failure is detected **after** errors appear in system data and states.

Not all circuit failures can be predicted, e.g., radiation-induced soft errors. However, circuit failure prediction is applicable for gate-oxide early-life failures (*ELF*) and circuit aging because of gradual degradation associated with these failure mechanisms. As a result, circuits do not always fail instantly — instead, they exhibit delay shifts (or delay fluctuations) over time, before errors are produced. Such delay shifts can serve as circuit failure prediction signatures.

Comprehensive delay shift information can be collected during normal system operation using special *failure prediction sensors*, and/or during periodic online self-test and diagnostics. Additional system usage information (thermal and voltage profiles, power states) can also be useful. Since such information only needs to be collected during short periods of time, the power cost of circuit failure prediction can be very small compared to concurrent error detection which continuously checks a system. Since failures are predicted before any system states and data are corrupted, no error recovery is required. Upon circuit failure prediction, appropriate *self-healing* actions, e.g., self-repair, self-tuning of supply voltage and operating frequency, are invoked.

Circuit failure prediction is distinct from concurrent error detection and "hard" failure detection using periodic online self-test and diagnostics [14),24),113)]. **Table 4** compares and contrasts these approaches.

In this section, we focus on the "temporal" aspect of circuit failure prediction, i.e., the ability to provide early indication of failures. The "spatial" aspect of circuit failure prediction (using a few circuit blocks to indicate potential failures in other blocks) can also help reduce error detection costs, e.g., the scheme in Ref. 153) for detecting errors due to supply voltage droops.

Circuit Failure Prediction for ELF

Early-life failures (*ELF*, also called *infant mortality failures*) result from defective chips that may pass manufacturing tests but fail early in the field, much

Table 4 Qualitative comparison of techniques for overcoming early-life failures and circuit aging.

Circuit failure prediction	Concurrent error detection	"Hard" failure detection through periodic online self-test and diagnostics
Before errors appear (+)	After errors appear (-)	After errors (possibly) appear (-)
No corrupt data or states (+)	Corrupt data & states (errors detected) (-)	Corrupt data & states possible (-)
Low cost (no concurrent checking) (+)	High cost due to concurrent checking (-)	Can be expensive for very frequent tests (-)
Applicable only for certain reliability failure mechanisms (-)	General applicability (+)	Broader applicability than prediction, narrower than error detection
No error recovery (+) (Self-healing required)	Error recovery (-) (Self-healing required)	Error recovery (-) (Self-healing required)
Self-diagnostics	Limited diagnostics	Limited diagnostics
Can be efficiently combined		

earlier than expected product lifetime. ELF is a major concern because traditional burn-in for ELF screening is getting increasingly difficult [9),18),49),68),105),143)]. Burn-in alternatives such as Iddq testing and its variants, e.g., Refs. 36), 83), 133), 150), 160) and numerous others, Very Low Voltage (VLV) and minVdd testing, e.g., Refs. 19), 44), 154), and outlier analysis techniques, e.g., Refs. 5), 81), 86), 132), are also getting difficult. Significant challenges with these techniques include ways of coping with circuit leakage and process variations, and the related problem of overkill (good ICs declared as defective). Hence, new design and test techniques that can serve as low-cost alternatives to burn-in are desirable. Finding such alternatives requires detailed understanding of defective behaviors of ELF candidates. In this paper, we focus on gate-oxide defects that are important ELF contributors.

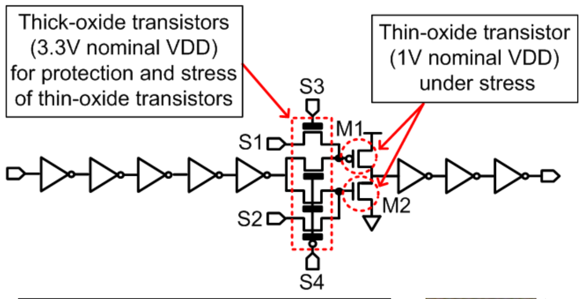
A new *gate-oxide ELF signature*, which we found through extensive stress experiments using test chips, enables cost-effective circuit failure prediction for gate-oxide ELF [62),63)]: a gate-oxide ELF transistor in a combinational logic circuit results in **delay shifts** (**delay fluctuations**) over time **before** functional failures appear. Such delay shifts are distinct from those resulting from circuit aging (NBTI, PBTI, hot carriers). This circuit-level signature is consistent with our large-scale transistor-level stress experiments [21),22)].

Such a signature can be effectively utilized for low-cost alternatives to traditional burn-in. One approach is to detect delay shifts of circuits under test during manufacturing testing after sufficient high-voltage stress. Another option is to build robust systems with built-in circuit failure prediction to check for delay shifts (or delay fluctuations) during system operation. Gate-oxide ELF defects cannot be detected using canary circuits because defects can occur anywhere. Since delay shifts are different from delay faults, delay fault detection flip-flops are inadequate. Reference 63) presents a special clock control technique, activated only during periodic online self-test and diagnostics, to detect delay shifts and indicate impending circuit failure due to gate-oxide ELF.

Figure 15 shows an ELF test structure from our 90 nm test chip in Ref. 63). It consists of a specially-designed inverter chain with thin-oxide transistors (SiO_xN_y gate dielectric). A single PMOS (M1) or NMOS (M2) thin-oxide transistor (nominal supply of 1 V) is stressed using high voltage to emulate gate-oxide ELF. Other thin-oxide transistors are protected from stress by thick-oxide transistors (nominal supply of 3.3 V).

Figure 16 shows our on-chip clock control circuit. Using a single fixedsystem clock SC, it creates a faster than at-speed test clock, i.e., scan-based launch and capture pulses where the delay between rising edges of launch and capture pulses can be controlled with a resolution finer than 20 ps (demonstrated in **Fig. 17** (b)).

Figure 17 (a) demonstrates that the on-chip test clock control technique, activated only during periodic online self-test and diagnostics, can successfully detect delay shifts (fluctuations) over time at the output of the gate-oxide ELF test structure, as a result of voltage stress (to emulate gate-oxide ELF). Figure 17 (a)


Thick-oxide transistors (3.3V nominal VDD) for protection and stress of thin-oxide transistors
Thin-oxide transistor (1V nominal VDD) under stress
S3
S1
M1
S2
M2
S4

	S1	S2	S3	S4
M1 stress	-3V	0V	0V	-3.3V
M2 stress	1V	4V	3.3V	0V
Operation	0V	0V	0V	3.3V

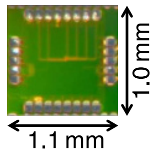


Fig. 15 Gate-oxide ELF test structure.

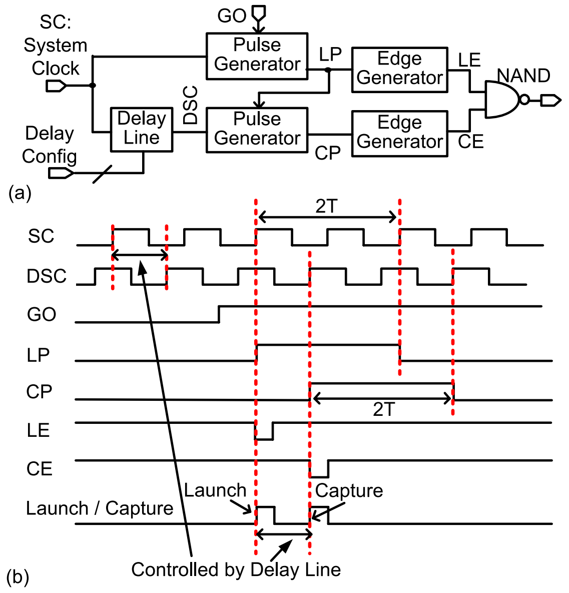


Fig. 16 (a) On-chip test clock control circuit. (b) Timing diagram.

Delay shift due to gate-oxide ELF

Delay configuration										
67	P	P	P	P	P	P	P	P	P	P
66	P	P	P	P	P	F	P	P	P	P
60	P	P	P	P	P	F	P	P	P	F
57	P	P	P	P	P	F	P	P	F	F
56	P	P	P	F	P	F	P	P	F	F
55	P	P	P	F	F	F	P	P	F	F
54	P	P	P	F	F	F	F	P	F	F
53	P	P	P	F	F	F	F	F	F	F
29	P	P	P	F	F	F	F	F	F	F
28	P	P	F	F	F	F	F	F	F	F
27	F	F	F	F	F	F	F	F	F	F
	0	130	280	310	400	430	460	490	520	580

(a) Stress time (a.u.)

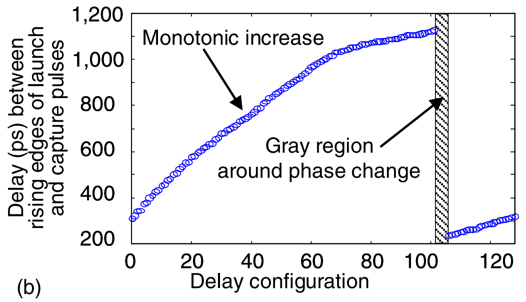


Fig. 17 Results using on-chip test clock control. (a) Failing delays showing delay shifts (fluctuations) over time. (b) Measured delays between rising edges of launch and capture vs. delay configuration.

corresponds to the case where the NMOS transistor M2 in Fig. 15 is stressed. We also verified gate leakage increase of M2, which is an evidence of emulated gate-oxide ELF.

Such a clock control approach is inexpensive because:

1. Online self-test and diagnostics are initiated only for short periods of time, resulting in small system-level power impact.

2. It does not rely on expensive error detection or flip-flop modifications, e.g., Ref. 101).

Hence, our approach can be seamlessly integrated in SoCs at low costs. Future research directions include:

1. Efficient test patterns for finding ELF delay shifts.

2. Analysis of the effects of temperature and voltage droops on ELF delay shifts.

Circuit Failure Prediction for Circuit Aging

In this section, we illustrate the application of circuit failure prediction for transistor aging induced by Negative Bias Temperature Instability (*NBTI*), a dominant aging mechanism. Our approach is applicable for other aging mechanisms, such as Positive Bias Temperature Instability (PBTI), as well. The PMOS threshold voltage degrades due to NBTI, resulting in increased delays of various circuit paths.

Designers traditionally incorporate *one-time worst-case guardbands (OWG)* at the beginning of lifetime to prevent possible delay faults due to aging over entire lifetime, e.g., 7–10 years for enterprise systems, even under worst-case usage conditions. OWG examples include clock frequency reduction, supply voltage increase, device over-sizing, structural duplication, or combinations thereof. OWG is pessimistic and expensive. Circuit failure prediction, together with optimized self-tuning, can significantly improve system lifetime energy efficiency by minimizing guardbands. This is demonstrated in Refs. 87), 88) using transistor aging models validated by 45 nm stress experiments.

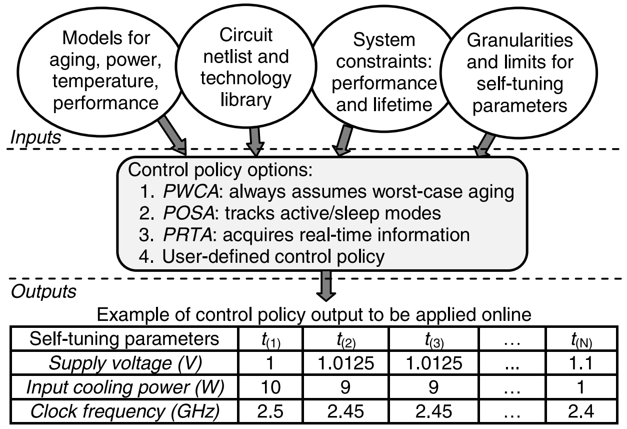
A framework for optimizing various self-tuning parameters over system lifetime is shown in **Fig. 18**. It exhibits the following features:

1. Satisfies performance constraints over entire lifetime while ensuring reliable operation.

2. Maximizes the *lifetime computational power efficiency (LCPE)* metric which is defined as the performance achieved (measured by the total number of clock cycles) over system lifetime divided by the total energy consumed over lifetime. Maximizing LCPE yields the best overall energy efficiency.

Examples of self-tuning parameters include supply voltage, clock frequency, and input cooling power (Fig. 18). These self-tuning parameters are adjusted dynamically according to performance demands, and adaptively according to estimated system aging. This is a challenging problem since self-tuning decisions made at any one point in time affects future aging, power, and performance. This can lead to conflicting trade-offs.

We introduce three representative control policies: PWCA, POSA, and PRTA.


Models for aging, power, temperature, performance
Circuit netlist and technology library
System constraints: performance and lifetime
Granularities and limits for self-tuning parameters
Inputs
Control policy options:
1. *PWCA*: always assumes worst-case aging
2. *POSA*: tracks active/sleep modes
3. *PRTA*: acquires real-time information
4. User-defined control policy
Outputs
Example of control policy output to be applied online

Self-tuning parameters	$t_{(1)}$	$t_{(2)}$	$t_{(3)}$	…	$t_{(N)}$
Supply voltage (V)	1	1.0125	1.0125	...	1.1
Input cooling power (W)	10	9	9	…	1
Clock frequency (GHz)	2.5	2.45	2.45	…	2.4

Fig. 18 Overall self-tuning framework: user inputs, control policies, and self-tuning parameters.

In our framework, system target lifetime is discretized into *time-steps*, which can be configured by the user (however, our results suggest that time-steps of 1 day are sufficient for NBTI aging). At each time step, a control policy decides whether to adjust all, some, or none of the self-tuning parameters.

PWCA: Progressive-worst-case-aging. PWCA acquires its benefits over OWG by applying self-tuning progressively over lifetime, rather than just one-time at the beginning of lifetime. It assumes worst-case operating conditions, i.e., the system is always in active mode under worst-case workload.

POSA: Progressive-on-state-aging. POSA further enhances self-tuning benefits by partially eliminating the worst-case aging assumptions in PWCA. POSA keeps track of system active/sleep modes, but assumes worst-case aging during active modes. Hence, it accounts for recovery effects during sleep mode (when Vdd is turned off). Such recovery behavior was experimentally observed in Refs. 152), 170).

PRTA: Progressive-real-time-aging-assisted. At the beginning of each time-step, PRTA acquires real-time aging information to determine the corresponding optimized values of self-tuning parameters. Real-time aging information for PRTA can be obtained (or calibrated) from a variety of sources:

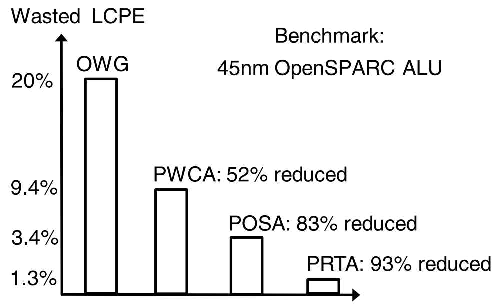


Fig. 19 LCPE improvements using various self-tuning policies. Life-time = 8 years, frequency = 2.4 GHz, average proportion of time spent in active mode = 10%.

1. Canary circuits $^{59)-61),147),151),152)}$.
2. On-chip sensors $^{15),56),145),146)}$.
3. Delay shift detectors $^{3),4),31)}$.
4. Online self-test and self-diagnostics (Section 3.3);

Extensive simulations using aging models, calibrated using 45 nm stress experiments, indicate that PWCA, POSA, and PRTA substantially recover LCPE degradation due to OWG $^{87),88)}$. On average, PWCA, POSA, and PRTA recover 52%, 83% and 93% of LCPE degradation due to OWG, respectively (**Fig. 19**).

3.3 CASP: Concurrent Autonomous Chip Self-Test and Diagnostics Using Stored Test Patterns

Concurrent autonomous online self-test and diagnostics play significant roles in addressing reliability challenges in future SoCs through circuit failure prediction and "hard" failure detection. *CASP*, an acronym for Concurrent Autonomous chip self-test and diagnostics using Stored test Patterns, is an efficient online self-test and diagnostics technique $^{74),77)}$. It enables a system to test itself, concurrently during normal operation, without any downtime visible to an end-user. The basic ideas of CASP are:

1. Store extremely thorough test patterns with high test coverage in off-chip storage, e.g., FLASH. This can include manufacturing test patterns or more thorough patterns that may not be applied during manufacturing for test cost reasons. The test patterns can be updated, e.g., using patches, after system deployment based on target applications and characteristics of field failures. Test compression, e.g., X-Compact $^{91),92)}$, enables efficient online self-test and diagnostics.

2. Provide architectural and system support for testing one or more components (both cores and uncore) of a SoC, while the rest of the system continues to operate.

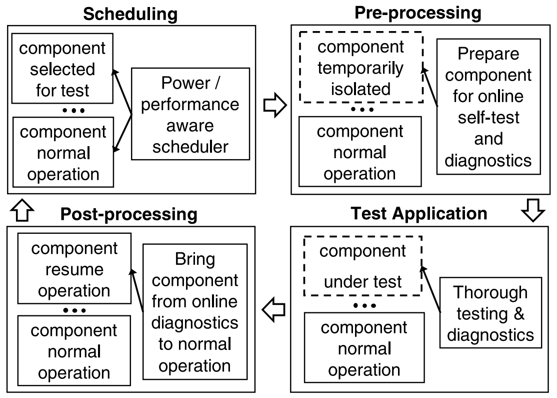

Scheduling
component selected for test
• • •
component normal operation
Power / performance aware scheduler
Pre-processing
component temporarily isolated
• • •
component normal operation
Prepare component for online self-test and diagnostics
Post-processing
component resume operation
• • •
component normal operation
Bring component from online diagnostics to normal operation
Test Application
component under test
• • •
component normal operation
Thorough testing & diagnostics

Fig. 20 Four phases of CASP operation.

CASP integrates architecture-level support, as well as support from virtual machine monitors (*VMMs*) and operating systems (*OS*), to orchestrate efficient online self-test and diagnostics at the system level.

Architecture-level support: We provide efficient online test access mechanisms to fetch high-quality test patterns from off-chip storage, and apply them to various components in a SoC (**Fig. 20**). Uncore components, e.g., cache controllers, DRAM controllers, and I/O controllers, occupy a significant portion of the total chip area of SoCs, and require special attention. A simple technique that stalls the uncore-component-under-online-test can result in significant system performance degradation or visible system unresponsiveness $^{77)}$. Our new uncore-CASP techniques enable cost-effective online self-test and diagnostics of uncore components (in addition to cores) through three special hardware features $^{77)}$:

1. Resource reallocation and sharing (RRS).
2. No-performance-impact testing.
3. Smart backup.

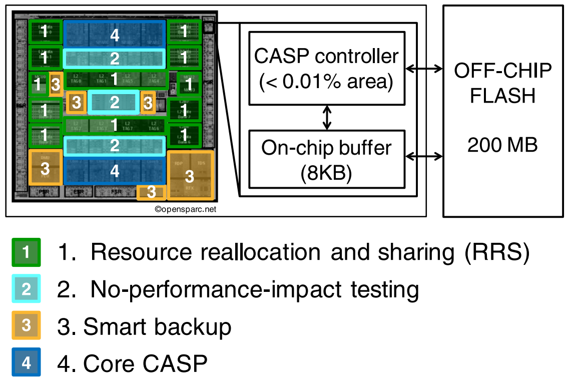

CASP controller (< 0.01% area)
On-chip buffer (8KB)
OFF-CHIP FLASH 200 MB
©opensparc.net
1. Resource reallocation and sharing (RRS)
2. No-performance-impact testing
3. Smart backup
4. Core CASP

Fig. 21 OpenSPARC T2 with CASP support.

Our implementation of these features in the 8-core and 64-threaded OpenSPARC T2 SoC[148], summarized in **Fig. 21**, demonstrates:

1. High coverage: 99.2% stuck-at, 92.8% transition fault coverage.

2. Low costs: $<$ 1% area impact, $<$ 1% power impact, and $<$ 3% system performance impact.

3. Practical off-chip FLASH storage requirement of 200 Mbytes.

VMM and OS support: VMM and OS support can help CASP overcome the following challenges:

1. I/O requests (including interrupts) to processor cores undergoing CASP may be dropped due to limited hardware buffer size (buffers are used to store incoming I/O requests)[51].

2. System-level performance impact of CASP can be significant without proper OS-level scheduling[75].

The *VAST* technique (Virtualization-Assisted concurrent autonomous online Self-Test and diagnostics) in Ref. 51) utilizes OS migration to move the execution of an OS from one core to another through the coordination of VMMs running on the source and destination cores. This way, concurrent execution of CASP and user applications is enabled if spare cores are available. Therefore, I/O requests can continue to be processed — for example, by a spare core, even when the original destination core of the I/O requests is undergoing online self-test and diagnostics. Using an ARM multi-core processor platform, we demonstrated that the performance impact of VAST on various applications is negligible ($<$ 1%) when spare cores are available (**Fig. 22**).

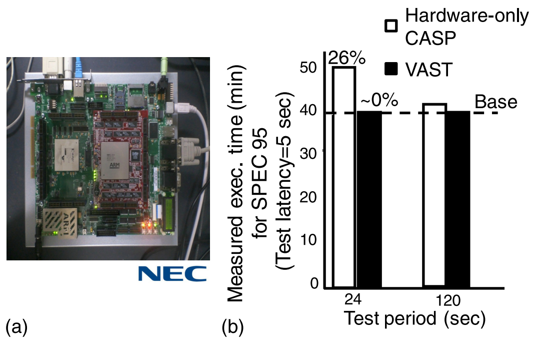


Fig. 22 (a) VAST platform. (b) Evaluation results of VAST for SPEC95.

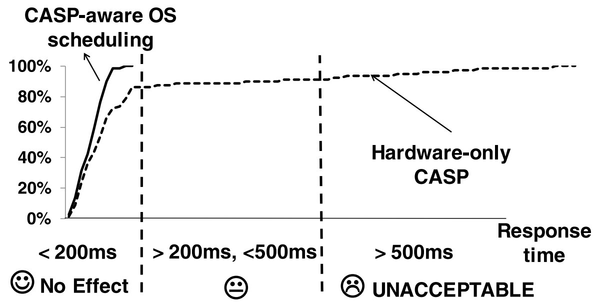


Fig. 23 Firefox response times comparing CASP-aware OS scheduling and hardware-only CASP.

The CASP-aware OS scheduling technique in Ref. 75) makes the OS aware of online self-test and diagnostics. We modified the scheduler of a Linux operating system (version 2.6.25.9) to take into account the unavailability of processor cores undergoing online self-test and diagnostics. The scheduler then migrates and schedules user tasks intelligently to minimize overall performance impact. Experimental results on a 2.5 GHz quad-core Intel Xeon E5420 system demonstrate that CASP-aware OS scheduling eliminates perceptible performance impact for interactive applications such as the Firefox web browser (**Fig. 23**).

4. Imperfection-immune Design for Emerging Nanotechnologies

4.1 Background

Carbon Nanotubes (*CNTs*) are cylindrical nanostructures of carbon with exceptional electrical, thermal, and mechanical properties[134]. Carbon Nanotube Field-Effect Transistors (*CNFETs*) can be fabricated using CNTs, where semiconducting Single-Walled Carbon Nanotubes (*SWCNTs*, or simply *CNTs*) are grown on or transferred to a substrate. These CNTs act as transistor channels which can be modulated by a gate (**Fig. 24**). The regions of the CNTs under the gate are undoped, while the source and drain regions of the CNTs are heavily doped. The gate, source and drain contacts, and interconnects are defined by lithography.

CNFET modeling efforts for logic VLSI applications[28),39),128),157)] indicate that CNFET circuits can provide more than an order of magnitude benefit in energy-delay-product compared to silicon CMOS[27),117),157)]. In addition, there has been significant progress in experimental CNFET research, e.g., high-performance single-CNT-based complementary CNFET logic[52),53)], and single-CNT ring-oscillator[20)]. Despite such encouraging single-device level results, fundamental limitations prevented researchers from demonstrating CNFET-based VLSI circuits:

1. It is nearly impossible to guarantee "perfect" alignment and accurate positioning of all CNTs at VLSI scale. A *mis-positioned CNT* passes through a layout region where it is not intended to pass. Mis-positioned CNTs introduce stray conducting paths in CNFET circuits causing incorrect functionality (**Fig. 25** (a)).

2. CNTs can be metallic (*m-CNT*) or semiconducting (*s-CNT*) depending on their atomic arrangement (*chirality*). s-CNTs are essential for CNFETs. m-CNTs cause source-drain shorts inside CNFETs, resulting in excessive leakage power, and significantly increased susceptibility to noise[167)]. No known CNT growth technique can produce exclusively s-CNTs. 4%–50% of grown CNTs can be m-CNTs[73),126)]. One option is to remove the m-CNTs after CNT growth[23),162),165)]. Significant challenges exist in effectively using these techniques at VLSI scale[119)]. Percolation-transport-based circuits can potentially overcome some m-CNT challenges[17)]. However, such circuits suffer from reduced drive currents[124)], and have limited applicability for high-performance VLSI.

3. CNFET circuits suffer from several sources of variations caused by non-idealities in CNT-specific processing. For example, CNT density variations are caused by non-uniform spacing between CNTs during CNT growth. As a result, the number of CNTs in CNFETs with a fixed width can have a large variance. Removal of m-CNTs can cause additional CNT density variations[166)]. Other sources include variations in CNT diameters, source/drain doping, and resistance of CNT-metal contacts[123),129),168)]. It has been shown in Ref. 168) that CNT density variations (including those induced by the removal of m-CNTs) can have significant impact on CNFET circuits.

It is clear that state-of-the-art CNT processing techniques alone cannot over-

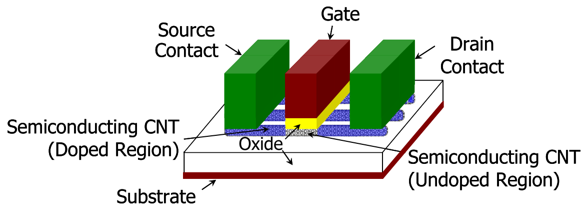

Gate
Source Contact
Drain Contact
Semiconducting CNT (Doped Region)
Oxide
Semiconducting CNT (Undoped Region)
Substrate

Fig. 24 Carbon Nanotube Field-Effect Transistor.

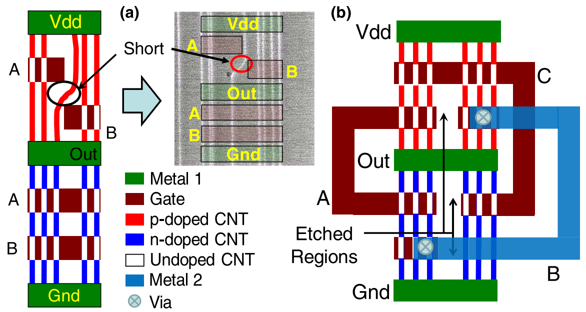


Fig. 25 (a) Mis-positioned CNT inside a NAND gate. (b) Mis-positioned-CNT-immune AOI21 layout.

come the above challenges. New design techniques must be employed, together with advances in CNT processing, to create CNFET circuits that are immune to these inherent imperfections — an **imperfection-immune design** paradigm[96]. The striking features of this approach are:

1. It introduces very little power, performance, and area impact at the system-level. Hence, it retains the energy efficiency benefits of CNFET circuits.
2. It is compatible with existing VLSI fabrication methods. This property is essential for enabling future large-scale integrated systems using CNFETs.
3. It minimally impacts existing VLSI design flows.

In Sections 4.2–4.4, we present imperfection-immune design techniques to overcome the above challenges. These imperfection-immune design techniques enabled the first experimental demonstration of VLSI-compatible computing and storage elements fabricated using CNFETs (Section 4.5). Our imperfection-immune design approach, together with special low-temperature CNT processing, creates a new opportunity: monolithic three-dimensional integrated circuits (monolithic *3D-ICs*) using CNFETs (Section 4.6).

4.2 Mis-positioned CNTs

A mis-positioned CNT may appear due to misalignment or due to lack of control of correct positioning of CNTs during CNT growth. CNTs can be grown on silicon substrates, as well as single-crystal substrates such as quartz[57),118)] and sapphire[43)]. CNTs grown on single-crystal quartz substrates have significantly better alignment compared to those grown on silicon[57)]. However, even for CNTs grown on quartz substrates (with more than 99.5% alignment according to Refs. 57), 118)), a non-negligible fraction of CNTs are still misaligned. It is nearly impossible for processing techniques alone to prevent mis-positioned CNTs at VLSI scale.

Using graph-theoretic design principles described in Ref. 116), we experimentally demonstrated, for the first time, hardware prototypes that are immune to a large number of mis-positioned CNTs: immune logic structures corresponding to NAND, NOR, AND-OR-INV and OR-AND-INV functions at full wafer-scale[115),116)]. We refer to these circuits as *mis-positioned-CNT-immune circuits*. Design of such immune circuits is accomplished by etching CNTs from regions, pre-defined during layout design, such that no mis-positioned CNT can result in incorrect logic functionality. This imperfection-immune design technique is VLSI-compatible since it does not require die-specific customization, or test and reconfiguration. Furthermore, it is compatible with existing VLSI design flows because the necessary changes take place only at the library cell level[8)]. Figure 25 (b) shows the layout of a complex logic function (AOI21) using the mis-positioned-CNT-immune design technique. With sufficiently high CNT density, together with the mis-positioned-CNT-immune design approach, CNT alignment and positioning is no longer a problem.

The worst-case (for minimum-sized CNFET logic cells) energy, delay and area penalties of mis-positioned-CNT-immune logic cells are 18%, 13%, and 21%, respectively, compared to designs that are not mis-positioned-CNT-immune[116)]. These costs reduce for non-minimum-sized cells. In contrast, defect-tolerance techniques in Refs. 26), 37), 149) can incur significantly higher costs.

4.3 Metallic CNTs

Depending on its chirality, a CNT can be semiconducting (with bandgap ranging between 0.5 eV and 2 eV) or metallic (with zero or almost zero bandgap)[134)].

There has been significant progress in *s-CNT enrichment*, i.e., to increase the percentage of s-CNTs either during or after CNT growth, e.g., preferential CNT growth[73),126)], CNT sorting[72)]. While such improvements are helpful, they are not sufficient. For digital VLSI CNFET circuits to meet leakage, noise margins, and delay variation targets, the percentage of m-CNTs must be reduced to less than 0.01%[167)]. This requirement is not yet achievable by known s-CNT enrichment techniques.

Another option for overcoming m-CNT challenges is to remove m-CNTs after CNT growth from an ensemble of m-CNTs and s-CNTs. Reference 23) introduced a current-induced breakdown technique to remove m-CNTs from individual CNFETs. We refer to this technique as *Single-Device electrical Breakdown* or *SDB*. In SDB, s-CNTs are switched off using the gate so that current flows (mainly) through m-CNTs. At high current levels, oxidation is induced by self-heating of m-CNTs causing them to break down. SDB achieves close to 100% m-CNT removal, but suffers from major VLSI challenges[119)]:

1. It is impractical to contact gate, source and drain of each CNFET individually in gigascale ICs.
2. m-CNT fragments can produce incorrect logic functionality because internal

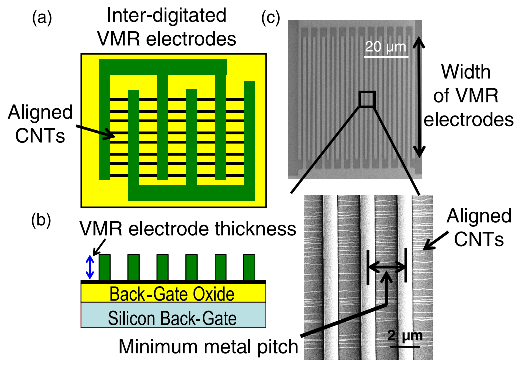


Fig. 26 (a) Top view and (b) cross-sectional view of a VMR structure consisting of 6 inter-digitated VMR electrodes. (c) SEM Image of VMR structure (top view).

contacts cannot be accessed for gigascale ICs.

We present a technique, called *VLSI-compatible Metallic-CNT Removal* (*VMR*)[119], which combines imperfection-immune design with CNT processing to overcome these challenges. It ensures that CNFET circuits function correctly even though CNT growth cannot guarantee the absence of m-CNTs.

A special layout called the *VMR structure* is fabricated on a silicon wafer with back-gate oxide (**Fig. 26**). The VMR structure consists of inter-digitated electrodes at minimum metal pitch. VLSI-compatible m-CNT electrical breakdown is performed by applying high voltage across m-CNTs all at once using the VMR structure. The back-gate is used to turn off the s-CNTs. Compared to SDB, VMR does not require any mechanism to contact each CNFET separately during breakdown. The problem of m-CNT fragments is overcome since all internal contacts can be accessed using the VMR structure. VMR is also complementary to ACCNT[79], another imperfection-immune design technique to tolerate the presence of m-CNTs.

The VMR structure is independent of the final intended design. Any arbitrary final design, e.g., **Fig. 27** (c), can be created by etching out parts of the VMR structure, as long as an additional contact (at minimum metal pitch) is added for series-connected CNFETs inside logic library cells[119]. This incurs $< 2\%$ area penalty, and 3% delay penalty at the chip level (based on synthesis of

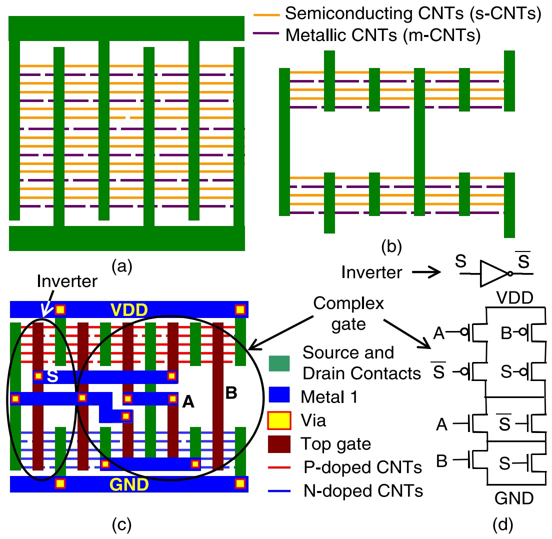


Fig. 27 VMR example. (a) Fabricate VMR electrodes and perform electrical breakdown of m-CNTs. (b) Etch CNTs in predefined layout regions. Etch unneeded sections of VMR electrodes. (c) Define top gates and interconnect layers. (d). Schematic of final design fabricated using VMR.

an OpenRISC processor, and an Ethernet controller[109]). Regions of the VMR structure to be etched out are pre-defined during layout design. No die-specific customization is required.

Experimental results demonstrate that VMR is scalable, and can be used to reproducibly fabricate multiple-CNT CNFETs that exhibit high I_{on}/I_{off} of 10^3 to 10^5. VMR is compatible with our mis-positioned-CNT-immune design technique, and enables the first experimental demonstration of VLSI-compatible and imperfection-immune CNFET logic circuits (Section 4.5).

One potential cost of VMR is the need for "burning pads" (Refs. 119), 159), not shown in Figs. 26–27) to apply high voltage across the VMR structure. A new VMR technique in Ref. 159) avoids this cost, and achieves significantly improved area efficiency.

4.4 CNT Density Variations

To analyze the impact of CNT density variations on CNFET circuits, a prob-

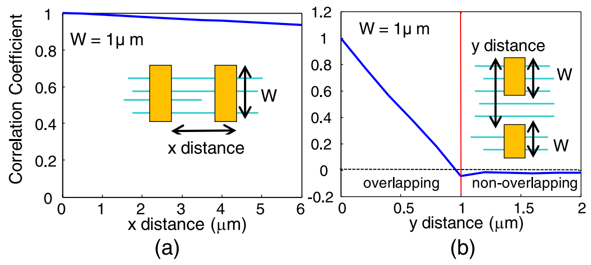


Fig. 28 CNT correlation in (a) x-direction and (b) y-direction (based on experimental data).

abilistic analysis approach is necessary [9]. We start by defining N as the number of CNTs in a particular CNFET with width W, prior to m-CNT removal. Due to CNT density variations, the number of CNTs contained in a fixed-width CNFET is not deterministic; hence, N must be treated as a random variable. Based on probabilistic renewal theory, Ref. 166) presents a parameterized model for the probability distribution of N for any given W. It can also be used to model variations caused by m-CNTs, after m-CNT removal using techniques such as VMR.

CNT density variations can result in CNFET circuit delay variations, and can significantly increase the probability of CNFET circuit failure. For example, CNT density variations can cause *CNT count failure* when no s-CNT exists between the source and drain of a CNFET. The circuit yield corresponding to this failure mode is called the *CNT count-limited yield*. A CNT count failure can happen either due to CNT density variations, or due to the removal of all CNTs in a CNFET (including inadvertent removal of s-CNTs) during m-CNT removal.

The CNT count-limited yield challenge can be overcome by utilizing inherent correlations between CNFETs fabricated using (mostly) aligned CNTs. Furthermore, the correlation is asymmetric. For example, consider two equal-width CNFETs horizontally adjacent to each other (**Fig. 28** (a), where the horizontal direction is referred to as the "x-direction", which is also assumed to be the direction of CNTs). The CNT count distributions of these two CNFETs exhibit high correlation, because most of the CNTs are shared between these two CNFETs. On the other hand, if two fabricated CNFETs are vertically adjacent to each other (Fig. 28 (b), the vertical direction is also referred to as the "y-direction"),

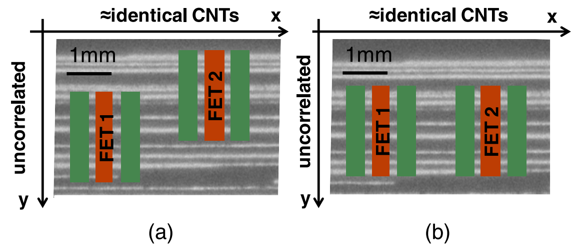


Fig. 29 CNT correlations. (a) Unconstrained layout style. (b) Aligned-active layout style.

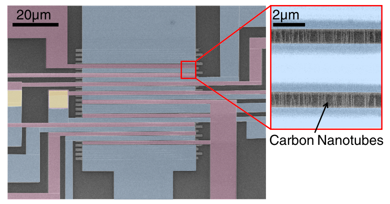


Fig. 30 Fabricated CNFET half-adder sum generator.

the correlation coefficient between the CNT counts is close to 0 long as the y-distance between the two CNFETs is greater than the width of the CNFETs.

Such asymmetric CNT correlation provides a unique opportunity for enhancing the CNT count-limited yield of CNFET-based VLSI circuits. The key is to use a special layout constraint called "*aligned-active*" layout (**Fig. 29** (b)), where all active regions of CNFETs are aligned to each other in the y-direction. Aligned-active layout style improves CNT count-limited yield by essentially creating clustering [67] of CNT count failures. At the 45 nm technology node, this approach can result in 350-fold reduction in the probability of CNT count failure [169].

4.5 First Experimental Demonstration of VLSI-Compatible CNFET Circuits

Our imperfection-immune design techniques enabled us to experimentally demonstrate, for the first time in the domain of CNFET circuits, VLSI-compatible combinational circuits (computational elements such as half-adder sum-generators in **Fig. 30**) and storage circuits (sequential elements such as D-latches in **Fig. 31**) that are immune to inherent CNT imperfections (mis-

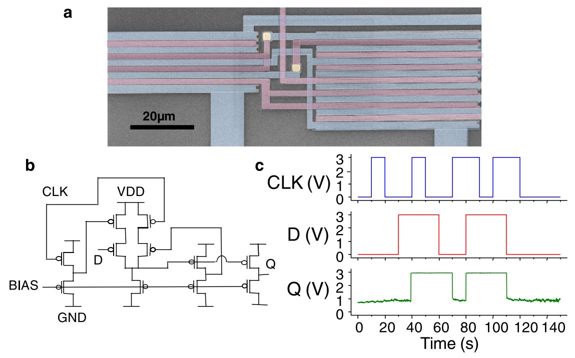


Fig. 31 Fabricated CNFET D-Latch. (a) SEM image. (b) Circuit diagram. (c) Experimental waveforms.

positioned CNTs and m-CNTs). Fabricated in a VLSI-compatible manner, these circuits form essential building blocks for digital computing systems [120].

4.6 CNFET-based Monolithic 3D-ICs

Three-dimensional Integrated Circuits (3D-ICs) can potentially offer several advantages over conventional 2D-ICs, e.g., alleviating the communication bottleneck, integration of heterogeneous materials and systems, and enabling new architectures [64),161].

Most existing 3D-ICs use chip stacks and Through-Silicon-Vias (*TSVs*) [12),58]. The TSV technology is an effective way to realize 3D integration [98]. However, the achievable density of vertical interconnects is limited [64]. This limitation constrains circuit designers from fully benefiting from TSV-based 3D-ICs [78]. On the other hand, *monolithic integration of silicon 3D-ICs*, in which layers of circuits are fabricated and integrated on the same wafer [161], suffers from processing challenges [110]. It requires low-temperature processing to realize high-performance transistors on upper layers, without sacrificing the performance of underlying transistors and interconnects.

With major CNFET technology challenges being overcome by imperfection-immune design, the CNFET technology can offer a promising platform for realizing monolithic 3D-ICs. The use of a low-temperature transfer technique [118] decouples high-temperature CNT growth from low-temperature device/circuit

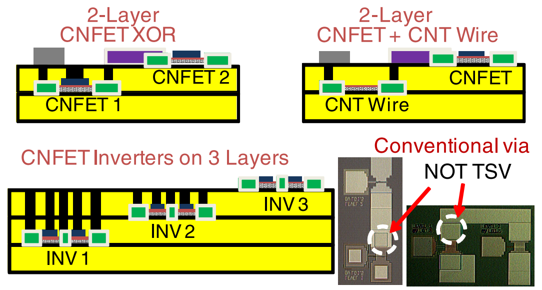


Fig. 32 Monolithic 3D-ICs using CNFETs and CNT interconnects.

fabrication. Reference 158) demonstrates, for the first time, monolithic 3D-ICs using CNFETs and CNT interconnects (**Fig. 32**). The maximum processing temperature is 250℃, which is below the 400℃ thermal budget for lower layers. CNFET circuits spanning 2-3 have been successfully demonstrated using this approach (details in Ref. 158)).

Rapid advances in a robust CNFET technology, enabled by imperfection-immune design and monolithic three-dimensional integration, can revolutionize the next wave of large-scale integrated systems that can deliver massive computing capacity in highly energy-efficient manner. Such a transformative technology is economically viable because it can co-exist with today's mainstream silicon technologies, and leverage today's manufacturing and system design infrastructure.

5. Conclusion

The field of robust system design offers new research opportunities in several domains ranging from ultra-large-scale electronic systems all the way to their nanoscale components. This paper presents an overview of our inter-disciplinary research to address key challenges in this field. Specific new ideas and successful experimental demonstrations include:

1. IFRA and QED for effective post-silicon validation of overwhelmingly complex systems.
2. Soft Error Resilience and Circuit Failure Prediction for system robustness against rising error rates.

3. A new Imperfection-Immune Robust Carbon Nanotube VLSI technology.

We envision our results enabling a sea change in the creation of future large-scale robust systems that can positively impact our everyday lives by addressing critical needs including energy, health-care, and transportation infrastructure.

Acknowledgments This work was supported in part by Bosch, Cisco, Defense Advanced Research Projects Agency (DARPA), Defense Threat Reduction Agency (DTRA), Focus Center Research Program Center for Circuit and System Solutions (FCRP C2S2), Focus Center Research Program Gigascale Systems Research Center (FCRP GSRC), IBM, Infineon, Intel, LSI, National Science Foundation (NSF), National Semiconductors, NEC, Renesas, Robustchip Inc., Semiconductor Research Corporation (SRC), Stanford Center for Integrated Systems (CIS), Texas Instruments, and Toshiba. Part of work, described in the paper, was performed at the Stanford Nanofabrication Facility (a member of the National Nanotechnology Infrastructure Network) which is supported by the National Science Foundation. Contributions of the students and collaborators of the Stanford Robust Systems Group are gratefully acknowledged.

References

1) Abramovici, M., et al.: A Reconfigurable Design-for-Debug Infrastructure for SoCs, *Proc. Design Automation Conf.*, pp.7–12 (2006).
2) Ando, H., et al.: A 1.3-GHz Fifth-Generation SPARC64 Microprocessor, *IEEE Journal Solid-State Circuits*, Vol.38, No.11, pp.1896–1905 (2003).
3) Agarwal, M., et al.: Circuit Failure Prediction and its Application to Transistor Aging, *Proc. VLSI Test Symp.*, pp.277–286 (2007).
4) Agarwal, M., et al.: Optimized Circuit Failure Prediction for Aging: Practicality and Promise, *Proc. Intl. Test Conf.*, pp.1–10 (2008).
5) Barnett, T.S., et al.: Burn-in Failures and Local Region Yield: An integrated Yield-Reliability Model, *Proc. VLSI Test Symp.*, pp.326–332 (2001).
6) Baumann, R.C.: Soft Errors in Advanced Semiconductor Devices—Part I: The Three Radiation Sources, *IEEE Trans. Device Materials Reliability*, Vol.1, No.1, pp.17–22 (2001).
7) Bernick, D., et al.: Non-Stop Advanced Architecture, *Proc. Intl. Conf. Dependable Systems and Networks*, pp.12–21 (2005).
8) Bobba, S., et al.: Design of Compact Imperfection-Immune CNFET Layouts for Standard-Cell-Based Logic Synthesis, *Proc. Design, Automation and Test in Europe*, pp.616–621 (2009).
9) Borkar, S.: Designing Reliable Systems from Unreliable Components, *IEEE MICRO*, Vol.25, No.6, pp.10–16 (2005).
10) Borkar, S., et al.: Statistical Circuit Design with Carbon Nanotubes, United States Patent Application 2007015506.
11) Borkar, S., et al.: Microprocessors in the Era of Terascale Integration, *Proc. Design, Automation and Test in Europe*, pp.1–6 (2007).
12) Borkar, S.: 3D Integration for Energy Efficient System Design, *Proc. Symp. VLSI Tech.*, pp.58–59 (2009).
13) Bowman, K., et al.: Energy-Efficient and Metastability-Immune Resilient Circuits for Dynamic Variation Tolerance, *IEEE Journal of Solid-State Circuits*, Vol.44, No.1, pp.49–63 (2009).
14) Breuer, M. and Ismaeel, A.A.: Roving Emulation as a Fault Detection Mechanism, *IEEE Trans. Computers*, Vol.35, No.11, pp.933–939 (1986).
15) Cabe, A.C., et al.: Small embeddable NBTI sensors (SENS) for tracking on-chip performance decay, *Proc. Intl. Symp. Quality Electronic Design*, pp.1–6 (2009).
16) Calin, T., Nicolaidis, M. and Velazco, R.: Upset Hardened Memory Design for Submicron CMOS Technology, *IEEE Trans. Nuclear Science*, Vol.43, No.6, pp.2874–2878 (Dec. 1996).
17) Cao, Q., et al.: Medium-Scale Carbon Nanotube Thin-Film Integrated Circuits on Flexible Plastic Substrates, *Nature*, Vol.454, No.7203, pp.495–500 (2008).
18) Carulli, J.M. and Anderson, T.J.: The Impact of Multiple Failure Modes on Estimating Product Field Reliability, *IEEE Design and Test of Computers*, Vol.23, No.2, pp.118–126 (2006).
19) Chang, J.T.Y. and McCluskey, E.J.: Detecting Delay Flaws by Very Low-Voltage Testing, *Proc. Intl. Test Conf.*, pp.367–376 (1996).
20) Chen, Z., et al.: An Integrated Logic Circuit Assembled on a Single Carbon Nanotube, *Science*, Vol.311, No.5768, pp.1735 (2006).
21) Chen, T.W., et al.: Gate-Oxide Early Life Failure prediction, *Proc. VLSI Test Symp.*, pp.111–118 (2008).
22) Chen, T.W., et al.: Experimental Study of Gate-Oxide Early Life Failures, *Proc. Intl. Reliability Physics Symp.*, pp.560–568 (2009).
23) Collins, P.G., Arnold, S. and Avouris, P.: Engineering Carbon Nanotubes and Nanotube Circuits using Electrical Breakdown, *Science*, Vol.292, No.5517, pp.706–709 (2001).
24) Constantinides, K., et al.: Software-Based On-line Detection of Hardware Defects: Mechanisms, Architectural Support, and Evaluation, *Proc. Intl. Symp. on Microarchitecture*, pp.97–108 (2007).
25) De Paula, F.M., et al.: BackSpace: Formal Analysis for Post-Silicon Debug, *Proc Intl. Conf. Formal Methods in Computer-Aided Design*, pp.1–10 (2008).
26) DeHon, A. and Naeimi, H.: Seven Strategies for Tolerating Highly Defective

Fabrication, *IEEE Design and Test of Computers*, Vol.22, No.4, pp.306–315 (2005).
27) Deng, J., et al.: Carbon Nanotube Transistor Circuits: Circuit-Level Performance Benchmarking and Design Options for Living with Imperfections, *Proc. Intl. Solid-State Circuits Conf.*, pp.70–588 (2007).
28) Deng, J. and Wong, H.-S.P.: A Compact SPICE Model for Carbon Nanotube Field Effect Transistors Including Non-Idealities and Its Application — Part I: Model of the Intrinsic Channel Region, *IEEE Trans. Electron Devices*, Vol.54, No.12, pp.3186–3194 (2007).
29) Dodd, P.E. and Massengill, L.W.: Basic Mechanisms and Modeling of Single-Event Upset in Digital Microelectronics, *IEEE Trans. Nuclear Science*, Vol.50, No.3, pp.583–602 (2003).
30) Dubey, P.: A Platform 2015 Model: Recognition, Mining and Synthesis Moves Computers to the Era of Tera, *Technology at Intel Magazine*, pp.1–10 (2005).
31) Eireiner, M., et al.: In-Situ delay characterization and local supply voltage adjustment for compensation of local parametric variations, *IEEE Journal Solid-State Circuits*, Vol.42, No.7, pp.1583–1592 (2007).
32) Elnozahy, E.N., Johnson, D.B. and Wang, Y.M.: A Survey of Rollback-Recovery Protocols in Message-Passing Systems, *ACM Computing Surveys*, Vol.34, No.3, pp.375–408 (2002).
33) Ernst, D., et al.: Razor: A Low-power Pipeline Based on Circuit-level Timing Speculation, *Proc. Intl. Symp. on Microarchitecture*, pp.7–18 (2003).
34) Franco, P. and McCluskey, E.J.: On-Line Testing of Digital Circuits, *Proc. VLSI Test Symp.*, pp.167–173 (1994).
35) Furuta, J., et al.: A 65 nm Bistable Cross-coupled Dual Modular Redundancy Flip-Flop Capable of Protecting Soft Errors on the C-element, *Proc. Symp. VLSI Circuits*, pp.123–124 (2010).
36) Gattiker, A. and Maly, W.: Current Signatures, *Proc. VLSI Test Symp.*, pp.112–117 (1996).
37) Goldstein, S.C. and Budiu, M.: NanoFabrics: Spatial Computing using Molecular Electronics, *Proc. Intl. Symp. Computer Architecture*, pp.178–191 (2001).
38) Gorjiara, B., Reshadi, M. and Gajski, D.: Generic Architecture Description for Retargetable Compilation and Synthesis of Application-Specific Pipelined IPs, *Proc. Intl. Conf. on Computer Design*, pp.356–361 (2007).
39) Guo, J., Lundstrom, M. and Datta, S.: Performance Projections for Ballistic Carbon Nanotube Field-Effect Transistors, *Appl. Phys. Lett.*, Vol.80, No.17, pp.3192–3194 (2002).
40) Gunther, S. and Singhal, R.: Next Generation Intel Microarchitecture (Nehalem) Family: Architectural Insights and Power Mangement, *Intel Developer Forum* (2008).
41) Haensch, W., et al.: Silicon CMOS Devices beyond Scaling, *IBM Journal of Research and Development*, Vol.50, No.4.5, pp.339–361 (2010).
42) Halambi, A., et al: EXPRESSION: A Language for Architecture Exploration through Compiler/Simulator Retargetability, *Proc. Design Automation and Test in Europe*, pp.485–450 (1999).
43) Han, S., Liu, X. and Zhou, C.: Template-Free Directional Growth of Single-Walled Carbon Nanotubes on a- and r-Plane Sapphire, *J. Am. Chem. Soc.*, Vol.127, No.15, pp.5294–5295 (2005).
44) Hao, H. and McCluskey, E.J.: Very-Low-Voltage Testing for Weak CMOS Logic ICs, *Proc. Intl. Test Conf.*, pp.275–284 (1993).
45) Heath, M.W., Burleson, W.P. and Harris, I.G.: Synchro-Tokens: Eliminating Nondeterminism to Enable Chip-Level Test of Globally-Asynchronous Locally-Synchronous SoC's, *Proc. Design, Automation and Test in Europe*, pp.1532–1546 (2004).
46) Hicks, J., et al.: Intel's 45 nm CMOS Technology, *Intel Technology Journal*, Vol.12, No.2, pp.77–156 (2008).
47) Ho, C.R., et al.: Post-silicon Debug using Formal Verification Waypoints, *Design and Validation Conference* (2009).
48) Hong, T., et al.: QED: Quick Error Detection Tests for Effective Post-Silicon Validation, *IEEE Intl. Test Conf.* (2010).
49) Van Horn, J.: Towards Achieving Relentless Reliability Gains in a Server Marketplace of Teraflops, Laptops, Kilowatts, and "Cost, Cost, Cost"...: Making Peace between a Black Art and the Bottom Line, *Proc. Intl. Test Conf.*, pp.671–678 (2005).
50) Huang, K.H. and Abraham, J.A.: Algorithm-Based Fault Tolerance for Matrix Operations, *IEEE Trans. Comput.*, Vol.C-33, No.6, pp.518–528 (1984).
51) Inoue, H., Li, Y. and Mitra, S.: VAST: Virtualization-Assisted Concurrent Autonomous Self-Test, *Proc. Intl. Test Conf.*, pp.1–10 (2008).
52) Javey, A., et al.: Advancements in Complementary Carbon Nanotube Field-Effect Transistors, *Proc. Intl. Electron Devices Meeting*, pp.31.2.1–31.2.4 (2003).
53) Javey, A., et al.: High Performance Nanotube n-FETs with Chemically Doped Contacts, *Nano Letters*, Vol.5, No.2, pp.345–348 (2005).
54) Josephson, D., et al.: Debug Methodology for the McKinley Processor, *Proc. Intl. Test Conf.*, pp.451–460 (2001).
55) Josephson, D.: The Good, the Bad, and the Ugly of Silicon Debug, *Proc. Design Automation Conf.*, pp.3–6 (2006).
56) Kang, K., et al.: Characterization and estimation of circuit reliability degradation under NBTI using on-Line IDDQ measurement, *Proc. Design Automation Conf.*, pp.358–363 (2007).
57) Kang, S.J., et al.: High-Performance Electronics Using Dense, Perfectly Aligned Arrays of Single-Walled Carbon Nanotubes, *Nature Nanotechnology*, Vol.2, No.4, pp.230–236 (2007).
58) Kang, U., et al.: 8 Gb 3D DDR3 DRAM using Through-Silicon-Via Technology,

Proc. Intl. Solid-State Circuits Conf., pp.130–131 (2009).
59) Karl, E., Singh, P., Blaauw, D. and Sylvester, D.: Compact in-situ sensors for monitoring negative-bias-temperature-instability effect and oxide degradation, *Proc. Intl. Solid-State Circuits Conf.*, pp.410–411 (2008).
60) Keane, J., Kim, T. and Kim, C.H.: An on-chip NBTI sensor for measuring PMOS threshold voltage degradation, *Proc. Intl. Symp. Low Power Electronics and Design*, pp.189–194 (2007).
61) Kim, T., Persaud, R. and Kim, C.H.: Silicon odometer: An on-chip reliability monitor for measuring frequency degradation of digital circuits, *IEEE Journal Solid-State Circuits*, Vol.43, No.4, pp.874–880 (2008).
62) Kim, Y., et al.: Gate-Oxide Early-life Failure Identification using Delay Shifts, *IEEE Proc. VLSI Test Symp.*, pp.69–74 (2010).
63) Kim, Y., et al.: Low-Cost Gate-Oxide Early-life Failure Detection in Robust Systems, *Proc. Symp. VLSI Circuits*, pp.125–126 (2010).
64) Knickerbocker, J., et al.: Three-Dimensional Silicon Integration, *IBM Journal of Research and Development*, Vol.52, No.6, pp.553–569 (2008).
65) Ko, H.F. and Nicolici, N.: Automated Trace Signals Identification and State Restoration for Improving Observability in Post-silicon Validation, *Proc. Design, Automation and Test in Europe*, pp.1298–1303 (2008).
66) Kogge, P., et al.: Exascale Computing Study: Technology Challenges in Achieving Exascale Systems, *Tech. Rep.* (2008). (Online) ⟨http://www.cse.nd.edu/Reports/2008/TR-2008-13.pdf⟩
67) Koren, I. and Singh, A.D.: Fault Tolerance in VLSI Circuits, *IEEE Trans. Comput.*, Vol.23, No.7, pp.73–83 (1990).
68) Kundu, S., Mak, T.M. and Galivanche, R.: Trends in Manufacturing Test Methods and Their Implications, *Proc. Intl. Test Conf.*, pp.679–687 (2004).
69) Lawton, G.: Improved Flash Memory Grows in Popularity, *Computer*, Vol.39, No.1, pp.16–18 (2006).
70) Lee, H., et al.: LEAP: Layout Design through Error-Aware Placement for Soft-Error Resilient Sequential Cell Design, *Proc. Intl. Reliability Physics Symp.*, pp.3A.3.1–3A.3.10 (2010).
71) Leem, L., et al.: ERSA: Error Resilient System Architecture for Probabilistic Applications, *Proc. Design Automation and Test in Europe*, pp.1560–1565 (2010).
72) LeMieux, M., et al.: Self-Sorted, Aligned Nanotube Networks for Thin-Film Transistors, *Science*, Vol.321, No.5885, pp.101–104 (2008).
73) Li, Y., et al.: Preferential Growth of Semiconducting Single-Walled Carbon Nanotubes by a Plasma Enhanced CVD Method, *Nano Letters*, Vol.4, No.2, pp.317–321 (2004).
74) Li, Y., Makar, S. and Mitra, S.: CASP: Concurrent Autonomous Chip Self-test Using Stored Test Patterns, *Proc. Design Automation and Test in Europe*, pp.885–890 (2008).
75) Li, Y., Mutlu, O. and Mitra, S.: Operating System Scheduling for Efficient Online Self-Test in Robust Systems, *Proc. Intl. Conf. Computer Aided Design*, pp.201–208 (2009).
76) Li, Y, et al.: Overcoming early-life failure and aging for robust systems, *IEEE Design and Test of Computers*, Vol.26, No.6, pp.28–39 (2009).
77) Li, Y., Gardner, D. and Mitra, S.: Concurrent Autonomous Self-Test for Uncore Components in SoCs, *Proc. VLSI Test Symp.*, pp.232–237 (2010).
78) Lin, M., et al.: Performance Benefits of Monolithically Stacked 3D-FPGA, *Proc. Symp. Field Programmable Gate Arrays*, pp.113–122 (2006).
79) Lin, A., et al.: ACCNT-A Metallic-CNT-Tolerant Design Methodology for Carbon-Nanotube VLSI: Concepts and Experimental Demonstration, *IEEE Trans. Elec. Dev.*, Vol.56, No.12, pp.2969–2978 (2009).
80) Lu, D.J.: Watchdog Processors and Structural Integrity Checking, *IEEE Trans. Comput.*, Vol.C-31, No.7, pp.681–685 (1982).
81) Madge, R., et al.: Screening MinVDD Outliers using Feed-Forward Voltage Testing, *Proc. Intl. Test Conf.*, pp.673–682 (2002).
82) Mahmood, A. and McCluskey, E.J.: Concurrent Error Detection Using Watchdog Processors — A Survey, *IEEE Trans. Comput.*, Vol.37, No.2, pp.160–174 (1988).
83) Maxwell, P., et al.: Current Ratios: a Self-Scaling Technique for Production IDDQ Testing, *Proc. Intl. Test Conf.*, pp.738–746 (1999).
84) McLaughlin, R., Venkataraman, S. and Lim, C.: Automated Debug of Speed Path Failures using Functional Tests, *Proc. VLSI Test Symp.*, pp.91–96 (2009).
85) Meaney, P., et al.: IBM z990 Soft Error Detection and Recovery, *IEEE Trans. Device and Materials Reliability*, Vol.5, No.3, pp.419–427 (2005).
86) Miller, R.B., et al.: Unit Level Predicted Yield: a Method of Identifying High Defect Density Die at Wafer Sort, *Proc. Intl. Test Conf.*, pp.1118–1127 (2001).
87) Mintarno, E., et al.: Optimized Self-Tuning for Circuit Aging, *Proc. Design Automation and Test in Europe*, pp.586–591 (2010).
88) Mintarno, E., et al.: Self-Tuning for Maximized Lifetime Energy-Efficiency in the Presence of Circuit Aging, *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, to apprear (2011).
89) Mitra, S. and McCluskey, E.J.: Which Concurrent Error Detection Schemes to Choose?, *Proc. Intl. Test Conf.*, pp.985–994 (2000).
90) Mitra, S., Saxena, N.R. and McCluskey, E.J.: A Design Diversity Metric and Analysis of Redundant Systems, *IEEE Trans. Comput.*, Vol.51, No.5, pp.498–510 (2002).
91) Mitra, S. and Kim, K.S.: X-Compact: An Efficient Response Compaction Technique for Test Cost Reduction, *IEEE Intl. Test Conf.*, pp.311–320 (2002).
92) Mitra, S. and Kim, K.S.: X-Compact: An Efficient Response Compaction Technique, *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, Vol.23, No.3, pp.421–432 (2004).
93) Mitra, S., et al.: Robust System Design with Built-In Soft Error Resilience, *IEEE*

Trans. Comput., Vol.38, No.2, pp.43–52 (2005).

94) Mitra, S., et al.: Combinational Logic Soft Error Correction, *Proc. Intl. Test Conf.*, pp.1–9 (2006).

95) Mitra, S.: Globally Optimized Robust Systems to Overcome Scaled CMOS Reliability Challenges, *Proc. Design Automation and Test in Europe*, pp.941–946 (2008).

96) Mitra, S., et al.: Imperfection-Immune VLSI Logic Circuits using Carbon Nanotube Field Effect Transistors, *Proc. Design Automation and Test in Europe*, pp.436–441 (2009).

97) Mitra, S., Brelsford, S.K. and Sanda, P.: Cross-Layer Resilience Challenges: Metrics and Optimization, *Proc. Design Automation and Test in Europe*, pp.1029–1034 (2010).

98) Motoyoshi, M.: Through-Silicon Via (TSV), *Proc. IEEE*, Vol.97, No.1, pp.43–48 (2009).

99) Mukherjee, S.S., Kontz, M. and Reinhardt, S.K.: Detailed Design and Evaluation of Redundant Multithreading Alternatives, *Proc. Intl. Symp. Computer Architecture*, pp.99–110 (2002).

100) Muller, K.P. and Sanda, P.N.: Soft Error Assessments for Servers, *Proc. Intl. Reliability Physics Symp.*, pp.391–394 (2010).

101) Nakura, T., Nose, K. and Mizuno, M.: Fine Grain Redundant Logic using Defect Prediction Flip-flops, *Proc. Intl. Solid-State Circuits Conf.*, pp.402–611 (2007).

102) Nassif, S.R., Mehta, N. and Cao, Y.: A Resilience Roadmap, *Proc. Design Automation and Test in Europe*, pp.1011–1016 (2010).

103) Nepal, K., et al.: Using Implications for Online Error Detection, *Proc. Intl. Test Conf.*, pp.1–10 (2008).

104) Nicolaidis, M.: Time redundancy based soft-error tolerance to rescue nanometer technologies, *Proc. VLSI Test Symp.*, pp.86–94 (1999).

105) Nigh, P. and Gattiker, A.: Test Method Evaluation Experiments and Data, *Proc. Intl. Test Conf.*, pp.454–463 (2000).

106) Nugyen, H.T. and Yagil, Y.: A Systematic Approach to SER Estimation and Solutions, *Proc. Intl. Reliability Physics Symp.*, pp.60–70 (2003).

107) Oh, N., Shirvani, P.P. and McCluskey, E.J.: Error Detection by Duplicated Instructions in Super-scalar Processors, *IEEE Trans. Reliability*, Vol.51, No.1, pp.63–75 (2002).

108) Oh, N., Mitra, S. and McCluskey, E.J.: ED^{4}I: Error Detection by Diverse Data and Duplicated Instructions, *IEEE Trans. Comput.*, Vol.51, No.2, pp.180–199 (2002).

109) http://www.opencores.org.

110) Park, J., et al.: Metal-Induced Dopant (Boron and Phosphorus) Activation Process in Amorphous Germanium for Monolithic Three-Dimensional Integration, *J. Applied Physics*, Vol.106, No.7, pp.074510.1–6 (2009).

111) Park, S.-B., Hong, T. and Mitra, S.: Post-Silicon Bug Localization in Processors Using Instruction Footprint Recording and Analysis (IFRA), *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, Vol.28, No.10, pp.1545–1558 (2009).

112) Park, S.-B., et al.: BLoG: Post-Silicon Bug Localization in Processors using Bug Localization Graphs, *Proc. Design Automation Conf.*, pp.368–373 (2010).

113) Paschalis, A. and Gizopoulos, D.: Effective Software-Based Self-Test Strategies for On-Line Periodic Testing of Embedded Processors, *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, pp.88–99 (2005).

114) Patel, J.H. and Fung, L.Y.: Concurrent Error Detection in ALUs by Recomputing with Shifted Operands, *IEEE Trans. Comput.*, Vol.C-31, No.7, pp.589–595 (1982).

115) Patil, N., et al.: Design Methods for Misaligned and Mis-positioned Carbon-Nanotube-Immune Circuits, *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, Vol.27, No.10, pp.1725–1736 (2008).

116) Patil, N., et al.: Integrated Wafer-scale Growth and Transfer of Directional Carbon Nanotubes and Misaligned-Carbon-Nanotube-Immune Logic Structures, *Proc. Symp. VLSI Tech.*, pp.205–206 (2008).

117) Patil, N., et al.: Circuit-Level Performance Benchmarking and Scalability of Carbon Nanotube Transistor Circuits, *IEEE Trans. Nanotechnology*, Vol.8, No.1, pp.37–45 (2009).

118) Patil, N., et al.: Wafer-Scale Growth and Transfer of Aligned Single-Walled Carbon Nanotubes, *IEEE Trans. Nanotechnology*, Vol.8, No.4, pp.498–504 (2009).

119) Patil, N., et al.: VMR: VLSI-Compatible Metallic Carbon Nanotube Removal for Imperfection-Immune Cascaded Multi-Stage Digital Logic Circuits using Carbon Nanotube FETs, *Proc. Intl. Electron Devices Meeting*, pp.573–576 (2009).

120) Patil, N., et al.: Scalable Carbon Nanotube Computational and Storage Circuits Immune to Metallic and Mis-positioned Carbon Nanotubes, *IEEE Trans. Nanotechnology*, to appear (2010).

121) Patra, P.: On the Cusp of a Validation Wall, *IEEE Design and Test of Computers*, Vol.24, No.2, pp.193–196 (2007).

122) Pattabiraman, K., Kalbarczyk, Z. and Iyer, R.K.: Automated Derivation of Application-Aware Error Detectors using Static Analysis, *Proc. Intl. On-Line Test Symp.*, pp.211–216 (2007).

123) Paul, B., et al.: Impact of a Process Variation on Nanowire and Nanotube Device Performance, *IEEE Trans. Electron Devices*, Vol.54, No.9, pp.2369–2375 (2007).

124) Pimparkar, N., et al.: Limits of Performance Gain of Aligned CNT Over Randomized Network: Theoretical Predictions and Experimental Validation, *IEEE Electron Device Letters*, Vol.28, No.7, pp.593–595 (2007).

125) Prvulovic, M., Zhang, Z. and Torrellas, J.: Revive: Cost-Effective Architectural Support for Rollback Recovery in Shared-Memory Multiprocessors, *Proc. Intl. Symp. Computer Architecture*, pp.111–122 (2002).

126) Qu, L., Feng, D. and Dai, L.: Preferential Syntheses of Semiconducting Vertically Aligned Single-Walled Carbon Nanotubes for Direct Use in FETs, *Nano Letters*,

Vol.8, No.9, pp.2682–2687 (2008).
127) Rattner, J.: The Dawn of Terascale Computing, *IEEE Solid-State Circuits Magazine*, Vol.1, No.1, pp.83–89 (2009).
128) Raychowdhury, A., Mukhopadhyay, S. and Roy, K.: A Circuit-Compatible Model of Ballistic Carbon Nanotube Field-Effect Transistors, *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems.*, Vol.23, No.10, pp.1411–1420 (2004).
129) Raychowdhury, A., et al.: Variation Tolerance in a Multichannel Carbon-Nanotube Transistor for High-Speed Digital Circuits, *IEEE Trans. Electron Devices*, Vol.56, No.3, pp.383–392 (2009).
130) Reick, K., et al.: Fault-Tolerant Design of the IBM Power6 Microprocessor, *IEEE MICRO*, Vol.28, No.2, pp.30–38 (2008).
131) Renau, J., et al.: SESC Simulator. http://sesc.sourceforge.net (2005).
132) Riordan, W.C., et al.: Microprocessor Reliability Performance as A Function of Die Location for A 0.25 m, Five Layer Metal CMOS Logic Process, *Proc. Intl. Reliability Physics Symp.*, pp.1–11 (1999).
133) Sabade, S., et al.: IDDQ Test: Will It Survive The DSM Challenge?, *IEEE Design and Test of Computers*, Vol.19, No.5, pp.8–16 (2002).
134) Saito, R., Dresselhaus, G. and Dresselhaus, M.: *Physical Properties of Carbon Nanotubes*, Imperial College Press, p.272 (1998).
135) Sanda, P.N., et al.: Soft-Error Resilience of the IBM POWER6 Processor, *IBM Journal of Research and Development*, Vol.52, No.3, pp.275–284 (2008).
136) Sarangi, S.R., Greskamp, B. and Torrellas, J.: CADRE: Cycle-Accurate Deterministic Replay for Hardware Debugging, *Intl. Conf. Dependable Systems and Networks*, pp.301–312 (2006).
137) Saxena, N., et al.: Dependable Computing and Online Testing in Adaptive and Configurable Systems, *IEEE Design and Test of Computers*, Vol.17, No.1, pp.29–41 (2000).
138) Seifert, N., et al.: On the Scalability of Redundancy Based SER Mitigation Schemes, *Proc. Intl. Conf. Integrated Circuit Design and Technology*, pp.1–9 (2007).
139) Seifert, N.: Soft Error Rates of Hardened Sequentials Utilizing Local Redundancy, *Proc. Intl. On-Line Test Symp*, pp.49–50 (2008).
140) Seifert, N., et al.: On the Radiation-Induced Soft Error Performance of Hardened Sequential Elements in Advanced Bulk CMOS Technologies, *Proc. Intl. Reliability Physics Symp.*, pp.188–197 (2010).
141) Seshia, S.A., Li, W. and Mitra, S.: Verification-Guided Soft Error Resilience, *Proc. Design, Automation and Test in Europe*, pp.1442–1447 (2007).
142) Shah, M., et al.: UltraSPARC T2: A Highly Threaded, Power-Efficient SPARC SOC, *Proc. Asian Solid-State Circuits Conf.*, pp.22–25 (2007).
143) Shirley, G., et al.: New Directions in Defect Management, *Sematech Topical Research Conf. Reliability* (2000).
144) Silas, I., et al.: System-Level Validation of the Intel Pentium M Processor, *Intel Technology Journal*, Vol.7, No.2, pp.37–43 (2003).
145) Simunic, T., Mihic, K. and De Micheli, G.: Reliability and power management of integrated systems, *Proc. Euromicro Symp. Digital Systems Design*, pp.5–11 (2004).
146) Srinivasan, J., et al.: Lifetime reliability: Toward an architectural solution, *IEEE Micro*, Vol.25, No.3, pp.70–80 (2005).
147) Stawiasz, K., Jenkins, K.A. and Liu, P.: On-chip circuit for monitoring frequency degradation due to NBTI, *Proc. Intl. Reliability Physics Symp.*, pp.532–535 (2008).
148) OpenSPARC T2 Processor. http://www.opensparc.net/opensparc-t2/index.html
149) Tahoori, M.B.: Application-Independent Defect-Tolerance of Reconfigurable Nano-Architectures, *ACM Journal Emerging Technologies in Computing*, Vol.2, No.3, pp.197–218 (2006).
150) Thibeault, C.: A Histogram Based Procedure for Current Testing of Active Defects, *Proc. Intl. Test Conf.*, pp.714–723 (1999).
151) Tschanz, J.W., et al.: Adaptive frequency and biasing techniques for tolerance to dynamic temperature-voltage and aging, *Proc. Intl. Solid-State Circuits Conf.*, pp.292–293 (2007).
152) Tschanz, J.W., et al.: Tunable Replica Circuits and Adaptive Voltage-Frequency Techniques for Dynamic Voltage, Temperature, and Aging Variation Tolerance, *Proc. Symp. VLSI Circuits*, pp.112–113 (2009).
153) Tschanz, J.W., et al.: A 45 nm Resilient and Adaptive Microprocessor Core for Dynamic Variation Tolerance, *Proc. Intl. Solid-State Circuits Conf.*, pp.282–283 (2010).
154) Tseng, C.-W., et al.: MINVDD Testing for Weak CMOS ICs, *Proc. VLSI Test Symp.*, pp.339–344 (2001).
155) Wang, N.J. and Patel, S.J.: ReStore: Symptom Based Soft Error Detection in Microprocessors, *Proc. Intl. Conf. Dependable Systems and Networks*, pp.30–39 (2005).
156) Wang, C., et al.: Compiler-Managed Software-Based Redundant Multi-Threading For Transient Fault Detection, *Proc. Intl. Symp. Code Generation and Optimization*, pp.244–258 (2007).
157) Wei, L., et al.: A Non-iterative Compact Model for Carbon Nanotube FETs Incorporating Source Exhaustion Effects, *Proc. Intl. Electron Devices Meeting*, pp.917–920 (2009).
158) Wei, H., et al.: Monolithic Three Dimensional Integration of Carbon Nanotube Field-Effect-Transistors and Interconnects, *Proc. Intl. Electron Devices Meeting*, pp.577–580 (2009).
159) Wei, H., et al.: Efficient Metallic Carbon Nanotube Removal Readily Scalable to Wafer-Level VLSI CNFET Circuits, *Proc. Symp. VLSI Tech.*, pp.277–280 (2010).
160) Williams, T.W., et al.: Iddq Test: Sensitivity Analysis of Scaling, *Proc. Intl. Test Conf.*, pp.786–792 (1996).
161) Wong, S., et al.: Monolithic 3D Integrated Circuits, *Proc. VLSI-Technology, Sys-*

tems, and Applications, pp.1–4 (2007).
162) Yang, C., et al.: Preferential Etching of Metallic Single-Walled Carbon Nanotubes with Small Diameter by Fluorine Gas, *Phys. Rev. B*, Vol.73, pp.075419.1–7 (2006).
163) Yerramilli, S.: Addressing Post-Silicon Validation Challenge: Leverage Validation & Test Synergy (Invited Address), *Proc. Intl. Test Conf.* (2006).
164) Zhang, M., et al.: Sequential Element Design with Built-In Soft Error Resilience, *IEEE Trans. VLSI*, Vol.14, No.12, pp.1368–1378 (2006).
165) Zhang, G., et al.: Selective Etching of Metallic Carbon Nanotubes by Gas-Phase Reaction, *Science*, Vol.314, No.5801, pp.974–977 (2006).
166) Zhang, J., et al.: Carbon Nanotube Circuits in The Presence of Carbon Nanotube Density Variations, *Proc. Design Automation Conf.*, pp.71–76 (2009).
167) Zhang, J., Patil, N. and Mitra, S.: Probabilistic Analysis and Design of Metallic-Carbon-Nanotube-Tolerant Digital Logic Circuits, *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, pp.1307–1320 (2009).
168) Zhang, J., et al.: Carbon Nanotube Circuits: Living with Imperfections and Variations, *Proc. Design, Automation and Test in Europe*, pp.1159–1164 (2010).
169) Zhang, J., et al.: Carbon Nanotube Correlation: Promising Opportunity for CNFET Circuit Yield Enhancement, *Proc. Design Automation Conf.*, pp.889–892 (2010).
170) Zheng, R., et al.: Circuit Aging Prediction for Low Power Operation, *Proc. Custom Integrated Circuits Conf.*, pp.427–430 (2009).

(Received November 29, 2010)
(Released February 8, 2011)

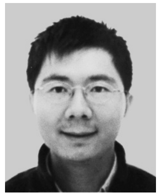
(Invited by Editor-in-Chief: *Hidetoshi Onodera*)

Subhasish Mitra directs the Robust Systems Group in the Department of Electrical Engineering and the Department of Computer Science of Stanford University. Before joining Stanford, he was a Principal Engineer at Intel Corporation. Professor Mitra's research interests include robust system design, VLSI design, CAD, validation and test, and emerging nanotechnologies. His X-Compact technique for test compression has been used in more than 50 Intel products, and has influenced major CAD tools. The IFRA technology for post-silicon validation, created jointly with his student, was characterized as "a breakthrough" in the Communications of the ACM. His work on the first demonstration of imperfection-immune carbon nanotube VLSI circuits, jointly with his students and collaborators, was selected by the National Science Foundation as a Research Highlight to the US Congress, and was highlighted as "a significant breakthrough" by the Semiconductor Research Corporation and the MIT Technology Review. Prof. Mitra's major honors include the Presidential Early Career Award for Scientists and Engineers from the White House, the highest US honor for early-career outstanding scientists and engineers, ACM SIGDA Outstanding New Faculty Award, IEEE CAS/CEDA Pederson Award for the IEEE Transactions on CAD Best Paper, IEEE/ACM Design Automation Conference Best Paper Award, IBM Faculty Award, Terman Fellowship, and the Intel Achievement Award, Intel's highest corporate honor. At Stanford, he was honored multiple times by graduating seniors "for being important to them during their time at Stanford." Prof. Mitra also serves as an invited member on DARPA's Information Science and Technology Board.

Hyungmin Cho received his B.S. in Computer Science and Engineering at Seoul National University, Korea in 2005, and M.S. in Electrical Engineering at Stanford University in 2010. Currently he is a Ph.D. candidate in Electrical Engineering at Stanford University. He was a research intern at NEC Laboratories America, Inc. during the summer of 2009. His research interests include architecture and software design for robust systems.

Ted Hong received his B.S. degree in Electrical and Computer Engineering from the University of California, Berkeley, in 2005 and M.S. degree in Electrical Engineering from Stanford University in 2007, where he is currently a Ph.D. candidate. His research interests include test, post-silicon validation, and robust system design.

Young Moon Kim received his B.S. in EE at Seoul National University, Korea in 2005, and M.S. in EE at Purdue University in 2007. Currently he is a Ph.D. candidate in EE at Stanford University. He was a technical intern at Samsung Electronics during the summer of 2007, at IBM Corporation during the summer of 2008, and at Intel Corporation during the summer and autumn of 2010. His research interests include design and test techniques for robust systems.



Hsiao-Heng (Kelin) Lee received his B.Eng in Computer Engineering from McGill University, Montréal, Canada, in 2000 and M.S. in Electrical Engineering from Stanford University, Stanford CA, in 2002, where he is currently pursuing his Ph.D. in the Department of Electrical Engineering. His research interests include low power/high performance digital circuits, digital signal processing algorithms and soft error resilient circuits. While at Stanford, Mr. Lee was supported by the National Sciences and Engineering Research Council of Canada (NSERC) Postgraduate Scholarship from 2000 to 2002.

Larkhoon Leem received his B.S. in EE from Seoul National University, Korea in 2002 and M.S. and Ph.D. in EE from Stanford University in 2005 and 2010, respectively. He was a technical intern at IBM T.J. Watson Research Center during the summer of 2005 and a research assistant of Stanford-IBM Spintronics Science and Applications Center. He is currently working as a research engineer at Intel Corporation. His research interests include emerging memory systems and fault-tolerant computing.

Yanjing Li is a Ph.D. candidate in Electrical Engineering at Stanford University. Her research interests include architectural and software techniques for robust system design. Prior to joining Stanford, she received her M.S. in Mathematical Sciences from Carnegie Mellon University, and B.S. in Electrical and Computer Engineering (with a double major in computer science) from Carnegie Mellon University.

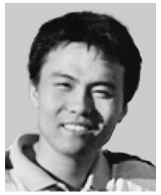
David Lin received his B.S. in Electrical Engineering from California Institute of Technology. Currently, he is a second year M.S./Ph.D. student in Electrical Engineering at Stanford University. He is supported by the Stanford Graduate Fellowship. His research interests include computer architecture, post-silicon validation, and robust systems design.

Evelyn Mintarno received her B.S. and M.S. degrees, with distinction, from the Department of Electrical Engineering at Stanford University, Stanford, CA, where she is currently working toward the Ph.D. degree. She was supported by the Stanford Starr Graduate Fellowship.

Diana Mui received her B.S. in Electrical Engineering from Purdue University. Currently, she is a SRCEA Intel Masters Scholar studying towards the M.S. in Electrical Engineering at Stanford University. Her interest is in the area of analog and digital circuits design.

Sung-Boem Park received his Ph.D. in Electrical Engineering from Stanford University in 2010 and he is currently working as a research scientist at Intel Corporation. His research interests include post-silicon validation, computer architecture, and fault-tolerant computing. He received a DAC Best paper Award in 2008.

Nishant Patil received his B.S. (Honors) in Electrical and Computer Engineering, with a minor in Physics, from Carnegie Mellon University in 2004. He received his M.S. and Ph.D. degrees in Electrical Engineering from Stanford University in 2006 and 2010, respectively. He is currently a Staff Engineer at Link-A-Media Devices Corporation. Dr. Patil is a recipient of the Stanford School of Engineering Fellowship, the Art and Mary Fong Stanford Graduate Fellowship and a Best Student Paper Award at the Symposium on VLSI technology. His research interests include simulation and fabrication of carbon nanotube field effect transistors and circuits.


Hai Wei received his B.S. degree (with honors) in microelectronics from Tsinghua University, Beijing, China, in 2007 and M.S. degree in Electrical Engineering from Stanford University, CA in 2010. He is currently working toward the Ph.D. degree in electrical engineering at Stanford University, Stanford, CA. His research interests include the design and fabrication of carbon nanotube field-effect transistors and circuits and monolithic 3D integrated circuits. He is the recipient of the Stanford School of Engineering Fellowship Award.

Jie Zhang received his B.E. degree in Electronic Engineering from Tsinghua University, China, in 2006, and the M.S. degree in Electrical Engineering from Stanford University, CA, in 2008. He is currently pursuing the Ph.D. degree in Electrical Engineering at Stanford University, CA. He is a recipient of the Stanford Graduate Fellowship. His research interests include modeling and simulation of carbon nanotube based devices and circuits, with a focus on the variability-aware design and optimization.