R2D3: A Reliability Engine for 3D Parallel Systems

Javad Bagherzadeh, Aporva Amarnath, Jielun Tan, Subhankar Pal, Ronald G. Dreslinski
University of Michigan, Ann Arbor, MI 48109
{javadb, aporvaa, jieltan, subh, rdreslin}@umich.edu

***Abstract*—This paper proposes a holistic reliability management engine, R2D3, for post-Moore's technology based parallel 3D systems that have low yield and high failure rate. The proposed engine, comprising of a controller, reconfigurable crossbars and detection circuitry, provides concurrent single-replay detection and diagnosis, fault-mitigating repair and aging-aware lifetime management at runtime. We show that R2D3 achieves 96% coverage of defects, repairs faulty cores, and reduces V_{th} degradation by 53%. This leads to a 78% performance improvement over 8 years and a 2.16× longer mean-time-to-failure over a baseline 8-core 3D processor with no reliability management.**

I. Introduction

Monolithic 3D integration presents the opportunity of designing cores and associated networks using multiple tiers by utilizing monolithic inter-tier vias (MIVs) [22]. With the decline of Moore's law, along with the use of hardware specialization [21, 20, 25], architects have leveraged the performance benefits of this vertical stacking to significantly decrease the run-time of compute-intensive workloads by stacking multiple layers of cores/processors to create 3D parallel systems for data accelerators [7, 17] and Network-on-Chip (NOC) architectures [6, 26]. Despite the benefits offered, yield and reliability are still the major obstacles for commercial realization of this technology [13, 12, 1]. Processors in 3D technology are more susceptible to permanent transistor failures, often due to wearout phenomena such as negative bias temperature instability (NBTI) [11] and electromigration (EM) [33]. Recent studies have shown how the elevated temperatures and longer heat dissipation paths in 3D ICs lead to significantly rapid aging and higher fault rates [30, 13].

Previous solutions can be divided into two categories: prevention - methods that slow down aging by decelerating wearout; and treatment - methods that handle failures once they have occurred. A reliable system requires prevention techniques along with three critical capabilities of treatment: 1) detection to identify the presence of a fault, 2) diagnosis to locate the source of the fault, and 3) repair to isolate the failure from the system [8]. Table I includes previous reliability solutions and illustrates the specific feature of reliability they target. The issue with the previous work is that these solutions are proposed in isolation of each other, proposed for specific architecture and problem, and are not compatible, so it is extremely difficult to combine them to create a holistic solution. Even if we ignore the compatibility and consider a *Frankenstein* solution that combines these prior work, the overall performance penalty will be high. Low yield and high fault rates in 3D technology call for the incorporation of an end-to-end solution that provides all four features of reliability at a low overhead. This paper demonstrates how a holistic approach that considers all required aspects creates a solution that is effective, low-overhead, easy to adopt and, hence, practical.

This paper proposes **Reliability by Reconfiguring 3D systems – R2D3**, a holistic, aging-aware reliability engine with fine-grained reconfigurability for parallel streaming systems that can concurrently **detect, diagnose, repair and prevent** failures at runtime. The engine comprising of a controller, reconfigurable crossbars and detection circuitry, takes advantage of the smaller routing delay over vertical layers in 3D circuits to create a fast-reconfigurable fabric that leverages the availability of identical resources in parallel systems. Figure 1 shows high level components of R2D3 incorporated into a in-order core based parallel 3D system where vertical buses that act as crossbars are inserted between consecutive stages. Our controller creates epochs of execution, at the end of which the detection circuitry can access the inputs and outputs of all layers to verify correctness and diagnose faults. The R2D3 controller manages the reconfigurable fabric to concurrently provide single-cycle replay detection and diagnosis, fault-mitigating repair and aging-aware lifetime management during runtime. Furthermore, to prevent faults, the controller considers temperature variation across the chip and activity factor of each stage to adaptively reconfigure functioning stages to create virtual pipelines. This balances out the aging rate, extending the lifetime of the system.

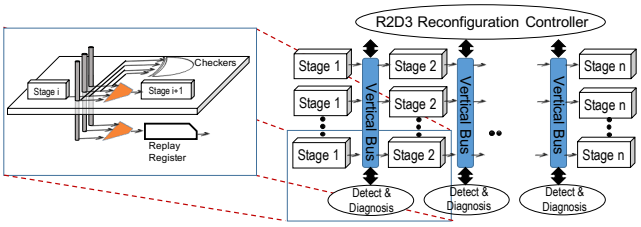


Fig. 1: Schematic showing high level components of R2D3, where corresponding pipeline stages are stacked vertically and vertical buses that function as crossbars are inserted between consecutive stages. At the end of each epoch, detection circuitry can access the inputs and outputs of all layers to check the output and the R2D3 Reconfiguration Controller manages the components.

R2D3 can be adopted to parallel architectures that have large number of identical units, such as systolic arrays, mesh-based systems, many-core or multi-core systems. In this work, we study the effects of R2D3, for vertically-stacked, in-order OpenSPARC T1 pipelines, as such pipelines closely resemble the basic compute units in modern-day streaming accelerators and massively parallel systems [14, 32].

We compare the following 8-core 3D systems: 1) System with R2D3 engine using an adaptive and dynamic reconfiguration policy (**R2D3-Pro**); 2) R2D3 engine using a round-robin dynamic non-adaptive reconfiguration policy (**R2D3-Lite**); 3) System equipped with failure-repairing static reconfiguration policy (**Static**); 4) A 3D-stacked processor with no reconfiguration infrastructure (**NoRecon**). Our evaluation, using a physical design and gem5, shows that our fault detection technique provides 96% coverage of defects. R2D3-Lite and R2D3-Pro, achieve 1.63× and 2.16× improvement in lifetime and 52% and 78% increase in throughput over NoRecon, respectively, while incurring a 7.4% area and 8.2% frequency overhead in comparison to NoRecon. Furthermore, R2D3 reduces V_{th} degradation by 53% over a period of 8 years in comparison to NoRecon and Static. In summary, this paper makes the following contributions:

1) Demonstrate a practical reliability engine, R2D3, for high fault rate monolithic 3D technology systems that can concurrently support prevention, detection, diagnosis and repair during runtime.
2) Propose a mechanism as part of R2D3 to detect and diagnose faults by re-executing instructions on idle units. It detects and distinguishes transient and permanent faults using single-cycle replay and localizes faults at the granularity of a pipeline unit.

Reliability Solutions	Granularity	Detection & Diagnosis		Repair	Lifetime Management		Overhead (%)		
		Detection	Coverage (%)		Technique	Enhancement (%)	Performance	Area	Power
ARGUS [19]	Core	✓	98				3.9	17.0	N.R.
BulletProof [5]	Pipeline stage	✓	89				18.0	5.9	N.R.
ACE [4]	Core	✓	99				5.5	5.8	4.0
CoreCannibal [2]	Pipeline Stage			✓			12.0	3.5	N.R.
3DFAR [3]	Pipeline stage			✓			5.0	7.0	N.R.
StageNet [9]	Pipeline stage			✓			33.0	17.0	16.0
Viper [23]	Pipeline stage			✓			24.0	8.0	N.R.
NBTI 3D [18]	Core				✓	Performance: 2.4	9.0	N.R.	N.R.
Bubblewrap [15]	Core				✓	Frequency: 16% Throughput: 30%	N.R.	N.R.	up to 90.0
NBTI Multicore [29]	Core				✓	Failure: 20% MTTF: 30%	6.0	N.R.	N.R.
Artemis [24]	Core				✓	Performance: 25%	2.0	N.R.	N.R.
R2D3 [this work]	**Pipeline stage**	✓	**96**	✓	✓	**Performance: 78% Lifetime: 116%**	**8.2**	**7.4**	**6.5**

TABLE I: Feature Comparison Matrix: Fault detection and diagnosis, repair and prevention (lifetime management) are the key features required for a reliable solution which is satisfied only by R2D3 at a low-cost. *N.R. stands for Not Reported.

3) R2D3 prevents failures by introducing graceful degradation using smart scheduling policies at the stage-level. It considers the activity factor and temperature variation of each unit to adaptively create virtual pipelines that balance out the aging rate.

II. Related Work

Although there is a plethora of work to address reliability concerns, no work addresses all the four features of reliability concurrently. Hence, we categorize previous work into fault detection and diagnosis, repair or lifetime management solutions as shown in Table I.

A. Treatment Techniques

Prior works leverage the redundant nature of multi-core systems that allow low-cost repair by disabling defective cores [4]. However, adopting core-level mechanisms for high fault-rate technologies can cause multiple failures to discard many cores at once. Therefore, these solutions do not scale well with high fault rates [23]. Some research has focused on developing low-cost fault tolerance for classic pipelined processors that rely on online testing [5], runtime fault detection [19], or defect isolation [9, 23]. StageNet [9] has shown that fine-grained architectures can break apart the hardware units of classic hard-wired pipelines, dissolving them into a sea of redundant hardware components. Bullet-Proof [5] utilizes a microarchitectural checkpointing mechanism that creates epochs of execution, during which distributed on-line built in self-test (BIST) mechanisms validate the integrity of the underlying hardware.

B. Lifetime Management Techniques

Aging is highly dependent on the utilization and operating temperature. System-level techniques take advantage of the application runtime behavior to improve lifetime. Adaptive voltage scaling (AVS) is an architecture-level technique proposed to mitigate aging in modern processors. Facelift [31] is a specific application of dynamic voltage scaling (DVS) in which the supply voltage only adapts once during the lifetime of a processor to switch from a slow aging mode to a high speed mode. Bubblewrap [15] uses techniques based on Facelift to enhance performance in a multi-core processor. Artemis [24] is an aging-aware application mapping and DVS scheduling framework that considers the PDN-aging of 3D NoC-based CMPs.

These works try to mitigate the effects caused by aging early on, instead of reducing it over time. The effect is limited, as when the supply voltage increases to counteract aging, the V_{th} degradation soon converges to that found in the guardbanded case [24]. This calls for an adaptive lifetime management technique deployed during runtime.

C. Frankenstein Solutions

While some prior work may seem to incur a lower overhead in a particular category in Table I, they do not provide all the four features of reliability. Even if we ignore compatibility issues and combine multiple solutions together, in a Frankenstein method, the system will incur a high performance penalty and area overhead. For example, if we hypothetically want to create a combined solution with the highest coverage and best lifetime enhancement, including fault repairing, we would combine [24], [3] and [4]. It would result in a 12.5% performance and 12.8% area overhead without even considering power. Our work motivates the need for a unified solution that can concurrently provide all the features of reliability at a low overhead.

D. Need for a Holistic Solution

First, some solutions lack essential features to be considered practical. For instance, previous reconfigurable architectures such as StageNet [9], Core Cannibal architecture [2], 3DFAR [3] and Viper [23] lack fault detection and diagnosis at a finer granularity which makes them incomplete. **Second**, studies shows that elevated temperatures and longer heat dissipation paths in 3D ICs lead to rapid aging and higher fault rates [30, 13] and this calls for an end-to-end solution that can control aggressive aging and repair the system upon a fault. **Third** is the aforementioned high-overhead *Frankenstein* issue.

To resolve these issues, our solution proposes to utilize the third dimension, to provide a unified reliability solution with concurrent fault detection, diagnosis, repair and graceful V_{th} degradation, compared to the limited and specialized approaches mentioned above. With this, R2D3 delivers a substantially reduced-overhead solution, as shown in Table I, when compared to prior work.

III. Proposed engine

This section describes our baseline architecture that provides fault repair, along with our proposed fault detection and diagnosis mechanism, and two reconfiguration policies to combat aging.

A. Architecture

In the rest of the paper, we use an in-order core-based system as the baseline. However, note that R2D3 is not restricted to in-order cores and can be adopted to any system that has identical units connected in a streaming style, *e.g.* systolic arrays and SIMD pipelines [17, 27]. We refer to in-order pipelines because the microarchitecture is more defined (we can easily generalize a 5-stage pipeline) and easier to benchmark, and also very resembling to most parallel compute units. Figure 1 shows the high level components of R2D3, where corresponding pipeline stages are stacked vertically and vertical buses that function as crossbars are inserted between consecutive

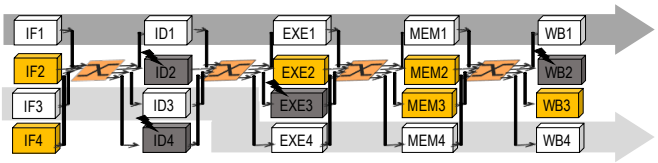


Fig. 2: In this 4 fault situation, R2D3's reconfiguration controller dynamically reconfigures to connect healthy units, providing 2 complete pipelines (wide bands). Functional stages, colored in orange (*leftovers*) can swap out other working stages to balance wearout and detect and diagnose faults. The backward paths, caches and their crossbars are omitted for simplicity.

stages. By replacing the direct connections at each pipeline stage boundary with interconnect switches, we create a network of resources in which each pipeline stage is connected to all instances of the subsequent stage. At the layer closest to the heat-sink, we insert the reconfiguration controller and detection circuitry which consists of two comparators between subsequent stages, for all layers. The reconfiguration controller can dynamically configure the interconnect to route instructions through functional hardware and detour around failed units, as shown in Figure 2.

Interconnect switch design: For the interconnect switch design, we adopted a bus-style implementation presented in [3] that is shown in the zoom-in section of Figure 1, where vertical links containing all signals at stage boundaries run across the entire height of the design, and each layer can multiplex its inputs from the prior stage on either the same layer or other layers with those links. Monolithic 3D fabric allows the placement of identical resources within short vertical distances from each other, providing a low overhead and fast interconnect network using MIVs. Cross-layer interconnect switches do not require any buffering, which greatly simplifies their design and control requirements. The signals are switched at their destination layer to avoid a single point of failure and provide redundancy for the interconnect. To minimize the performance loss from inter-stage communications, we use MUX-based full crossbar switches. The MUX-based crossbar has a fixed channel width and, as a result, an instruction transfer from one stage to the next can occur within the same clock cycle when implemented in 3D. The frequency overhead is $<$8.2% due to the small propagation delays of vertical MIVs.

B. Utilizing Inherent Redundancy

Leftovers are salvageable redundant compute units, i.e. stages that are functional, but could not be used to form a complete pipeline. They provide redundancy to support treatment and prevention forming the foundation of our unified reliability solution. With low yield and high fault rate in a design, there will likely be more *leftovers*. There are two scenarios in which a *leftover* is available:

1) Idle functional stages of faulty pipelines that cannot form a complete core (orange stages in Figure 2).
2) Pipeline stages in other working cores that are power gated because of low utilization or power and temperature constraints [15].

C. Fault Detection and Diagnosis

We consider independent instruction streams (from different threads/tasks) executing on each logical pipeline that do not have the same resource utilization (occupancy) profile. R2D3 controller utilizes a checkpointing mechanism that creates epochs of execution (T_{epoch} cycles), at the end of which it tests the underlying hardware. This provides us an opportunity to re-execute instructions on *leftovers* located in the proximity of the design-under-test (DUT) stage. Hence, detection and diagnosis requires no additional redundancy and therefore incurs no performance penalty by using idle resources. The scheme runs in the background during normal application execution

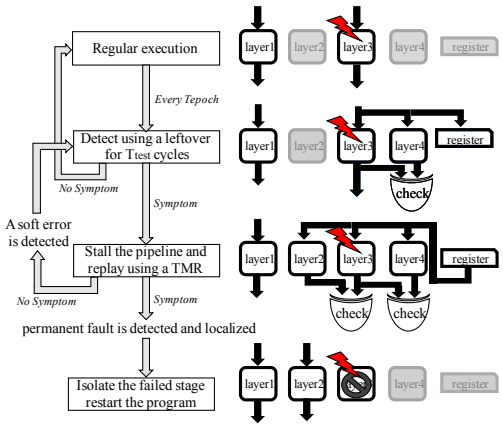


Fig. 3: R2D3 fault detection, diagnosis and repair mechanism in a flowchart (left) and pipeline configurations in a 4-layer design in each phase (right).

and is transparent to software. If there are no leftovers available at the end of a particular epoch, the R2D3 controller temporarily suspends a core to be used for fault detection. While this reduces performance, it is extremely rare because the nature of the workloads as well as thermal issues rarely allow 100% utilization of all cores.

R2D3's detection and diagnosis mechanism has these features:

1) Detection: Creates coarse-grain computational epochs that uses *leftovers* to execute identical instructions of the DUT stage.
2) Diagnosis: Distinguishes between transient vs. permanent fault, and locates the faulty stage using a single replay in one cycle.

Fault Detection: T_{epoch} cycles are used to determine how often a particular stage is tested. At the end of each epoch, we exploit the *leftovers* to verify the functional integrity of each stage of the logical pipelines for T_{test} cycles by parallel execution of the same instructions as those running on the DUT stage. If no fault is detected during the online testing, the next epoch computation starts.

Before running the test on the leftover stage, we warm up the stage by forwarding all the pipeline registers, register file, and branch predictor states from the DUT using the same vertical buses used to create logical pipelines, such that when all states of the leftover stage are same as the DUT stage, we can start the parallel re-execution test. As shown in Figure 3, we use simple inter-stage checkers at the output of the pipeline stages to assess the faulty behavior of the DUT. Having a fast 3D reconfiguration fabric, the output and input of each pipeline stage can be controlled and monitored at the layer with the crossbar switches. If the input of two similar stages in two different layers are the same, the output of the two should be identical too. If not, a fault has been detected. We also use the vertical buses to make the I/Os of each pipeline stage available to the detection circuitry. We copy the inputs of the the stages that are been tested into our replay register for one clock cycle to help diagnose (localize) the fault.

To enhance fault coverage, we can evaluate the underlying hardware for a longer period (higher T_{test}), allowing rigorous testing operations. We can also reduce the epoch duration, testing each stage more frequently (smaller T_{epoch}). Hence, the delay for fault detection can be reduced, which also minimizes the performance penalty caused by using the faulty hardware. However, using the *leftovers* for fault detection adds power overhead and there exists a trade-off between test duration/fault coverage ratio and the added power overhead. Furthermore, we consider the additional NBTI-based wearout of using *leftover* for detection and diagnosis in our evaluations (Section V).

Fault Diagnosis: After a symptom is detected, the fault needs to be diagnosed, *i.e.* distinguish between transient and permanent hardware fault, and in case of a permanent fault, identify the faulty component

to initiate system repair. The diagnosis algorithm of R2D3, as shown in Figure 3, achieves this with the help of another *leftover* and the ability to replay an execution using the inputs of the previous cycles. R2D3 uses a simple step to distinguish between and localize transient vs. permanent faults: it halts the pipeline for one cycle and replays the symptom-generating instruction on the two symptom-generating stages and a new known third good stage using Triple Modular Redundancy (TMR). If the symptom does not recur, a transient fault was detected and the execution resumes after stalling for only one cycle. If the symptom recurs, *i.e.* a hard fault, the third stage is used to vote and determine the fault. Localizing a permanent fault at a pipeline-stage granularity facilitates fine-grained repair and reconfiguration.

D. Repair

By adaptively routing around failed stages, we can salvage working units and repair the system. When a fault occurs, the victim unit (*i.e.* a pipeline stage) is isolated and the controller reconfigures the crossbars to construct logical pipelines based on the latest failure map. In Figure 2, four faults have disabled units on different vertical layers. R2D3 can build two complete pipelines dynamically, shown by the two wide bands. A core-level solution would have only one functioning core. Once reconfigured, we re-execute the task, starting either from a checkpoint or the beginning.

E. Aging-Aware Lifetime Management

Our solution uses the controller to replace any working stage with a *leftover* from another layer to provide an opportunity for the resource under stress to partially recover V_{th}. This enables R2D3 to effectively control and evenly distribute wearout among all working stages, ultimately letting the system degrade gracefully. Based on this, we propose two dynamic reconfiguration methods, **R2D3-Lite** and **R2D3-Pro** that aim to reduce the V_{th} degradation, and minimize its effect on NBTI-based aging as V_{th} degradation is a major cause of failures in systems [11]. Although R2D3 can be used to optimize any wearout mechanisms, we optimize our policy for NBTI-based aging

R2D3-Lite: For this policy, the controller reassigns *leftovers* within a pipeline in a round-robin fashion after a certain reconfiguration time period, T_{sched}. This scheme not only balances the usage of resources in the processors, but also gives the units a chance to be unstressed and partially recover their V_{th} degradation. This policy equalizes the usage of all the pipeline stages, but does not necessarily balance wear-out. Cores farther away from the heat sink that have higher temperatures, a major contributor to aging, would still witness higher wearout than layers closer to the heat sink.

R2D3-Pro: Expanding on R2D3-Lite, R2D3-Pro is specifically designed to address the characteristics of the 3D system and provide uniform aging. R2D3 assigns an **activity factor,** A_i, to each stage based on its temperature profile and location in the stack *i.e.* if the activity index is lower, then a stage is more prone to hot spots and degradation. Using the pre-calculated data obtained based on temperature patterns and ΔV_{th}, the controller updates these indices during execution and adaptively utilizes each pipeline stage based on its activity index while reconfiguring. To balance ΔV_{th}, the relative activity factor of the cores needs to follow the predicted values. The policy is modeled to favor stages that are less likely to wear out and heat up in the near future by increasing their usage probabilities.

The activity factor for each stage is calculated as:

$$A_i = \frac{\alpha_i}{\sum_{j=1}^{n_{live}} \alpha_j}.n_{workload} \quad (1)$$

where α_i is the predicted activity factor for pipeline stage i, n_{live} is the number of pipeline stages of the same type that are functional and available and $n_{workload}$ is the number of cores required to run the workload ($\leq n_{live}$). The activity indices are calculated every period of calibration window (T_{cal}) and each of the available resources are scheduled based on their activity index within T_{cal} as the following:

$$T_{sched,i} = A_i.T_{cal} \quad (2)$$

where $T_{sched,i}$ is the time period over which pipeline stage i is scheduled to be active. Activity factors can either be determined offline based on the steady state temperature of cores for typical workloads (implicitly based on the location of cores), or at runtime based on the temperature and wear-out (ΔV_{th}) history. In this work, we use the steady state temperature method as described in Section IV.

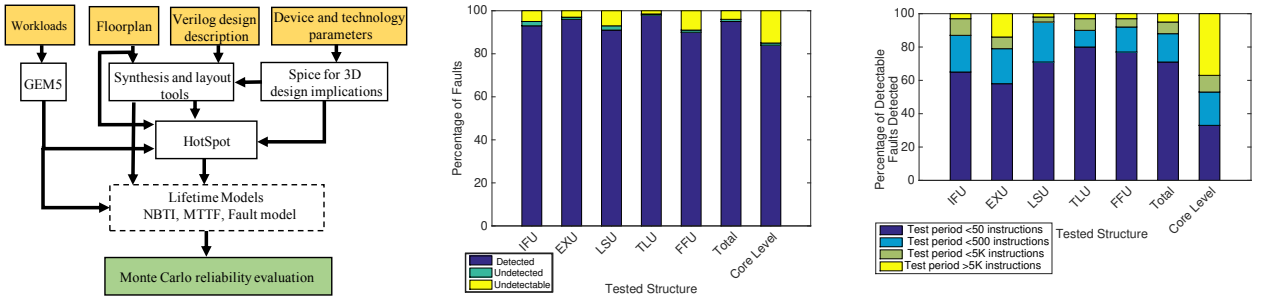
A final advantage is that both R2D3-Lite and R2D3-Pro entail no additional performance overhead, as execution continues even when swapping resources. This can be accomplished by warming up a leftover a few cycles before T_{sched}, *i.e.* by duplicating operations and forwarding all the necessary data such as branch predictor and register file states through vertical buses. Thus, at the time of reconfiguration, the leftover will be substituted in seamlessly and the only action needed would be to power off the swapped out module. For our reconfiguration frequency (calibration window), we optimize for power and performance overhead, and fault coverage, as NBTI degradation is independent of frequency. Hence, we set our calibration window for R2D3 equal to 5 ms (5 M cycles) which is based on the studies that are presented in Section V-B.

IV. Experimental methodology

This section illustrates the main steps of the proposed reliability evaluation mechanisms to address technology failures and NBTI-based wearout failures. We describe our methodology involved in the detection, diagnosis and prevention of faults. For fault simulations, we use the Synopsys TetraMAX ATPG tool to generate test patterns for the synthesized netlist. We use the stuck-at fault model, which is the industry standard model for test pattern generation. It assumes that a circuit defect behaves as a node stuck at 0 or 1. Our fault model injects permanent transistor failures into any design component in any layer, proportional to the area. Once a pipeline unit is detected as faulty, we disable the entire unit and trigger a dynamic reconfiguration. We assume that faults in local caches are handled by ECC. If a fault hits an interconnect switch, we disable the unit connected to the output, and if it hits the pipeline's control logic, we disable the entire pipeline. A brief overview is shown in Figure 4(a). To model the NBTI-based aging mechanisms and mean-time-to-failure (MTTF), we adopt a divide-and-conquer-based reliability evaluation that employs a Monte Carlo based simulation presented in [28].

We synthesized our design on a commercial 45 nm Silicon-on-Insulator (SOI) process technology with Synposys Design Compiler, and place-and-route the design using Cadence Innovus. To create the 3D layout, we follow the specifications and design rules recommended by Dae *et al.* [16], and evaluate power and timing through SPICE simulations. The physical design and parameters, and power consumption estimates are generated using Synopsys sign-off tools.

We use HotSpot v6.0 for thermal modeling. Using the 3D capability available in grid mode, the floorplan of the system is incorporated into the analysis. The output of the HotSpot simulation, temperatures of each sub-block along with the circuit netlists generated using the Cadence tools are utilized to perform sub-block level SPICE simulations. The gem5 simulator was used in this work to simulate the performance of a multicore system with 8 in-order cores. The paramters used for simulating the proposed architecture are shown in Table II. Three popular kernels, general matrix-matrix multiplication



(a) (b) (c)

Fig. 4: (a) The proposed reliability evaluation methodology for NBTI failure mechanism. (b) Breakdown of different types of faults for each unit. Total illustrates the additive results for all pipeline stages in the stage-level approach and core-level shows the coverage for solutions that have fault detection at a core granularity. (c) Breakdown of average percentage of detected faults of detectable permanent faults by length of testing period T_{test}.

TABLE II: gem5 simulation parameters for R2D3.

Module	Parameters
Core	Single-issue, in-order pipeline @ 1.0 GHz
L1 D-Cache	8 kB, 4-way set-associative, private
L2 D-Cache	64 kB, 4-way set-associative, shared
I-Cache	4 kB, 4-way set-associative, private
Main Memory	4-channel DDR4-2400 x64 @ 18.8 GB/s per channel

(GEMM), general matrix-vector multiplication (GEMV) and fast Fourier transform (FFT), were chosen for performance evaluation. FFT is widely used in communication and visual processing systems. GEMM and GEMV are ubiquitous kernels in machine learning, scientific workloads and other big-data applications.

V. Evaluation

This section demonstrates the use of the proposed engine, evaluates fault detection coverage and compares R2D3 that uses an adaptive and dynamic reconfiguration policy (R2D3-Pro) against a system that uses round-robin dynamic non-adaptive reconfiguration policy (R2D3-Lite), a system with failure-repairing static reconfiguration policy (Static), and a 3D-stacked processor without reconfiguration (NoRecon).

A. Physical Design

We evaluate R2D3 on an OpenSPARC T1 processor which implements the 64-bit SPARC V9 architecture [10]. The OpenSPARC T1 processor contains 8 in-order, five-stage pipelined, single-threaded cores. Table III presents the area and power break down based of our 45 nm SOI-technology physical design. Each core has an area of 0.387 mm^2 and operates at a frequency of 1 GHz. The area occupied by the crossbars including MIVs, switching logic and checkers is 7.4%. Frequency decreased by 8.2% because of the crossbar units and checkers. We report a power of 250 mW for a single core excluding the power of register files and caches using Synopsys Primetime which shows a 6.5% overhead compared to the NoRecon design.

B. Fault Detection and Diagnosis

Based on the fault model discussed in the previous section, we assume that we can observe and verify a fault using two identical pipeline stages. We performed a detailed fault analysis based on the number of test pattern instructions to determine the protected area covered by the fault detection mechanism as reported in Table III.

TABLE III: Area and power for a 5-stage pipeline.

Stage	Total area (mm^2)	Crossbar Overhead (%)	Checker Overhead (%)	Protected Area (%)	Power (mW)
IFU	0.056	10.3	0.43	88	115
EXU	0.036	12	0.5	95	23
LSU	0.067	18.8	0.74	98	44
TLU	0.040	5	0.22	91	10
FFU	0.014	35.4	1.24	96	3
Total	0.387	7.4	0.31	93	250

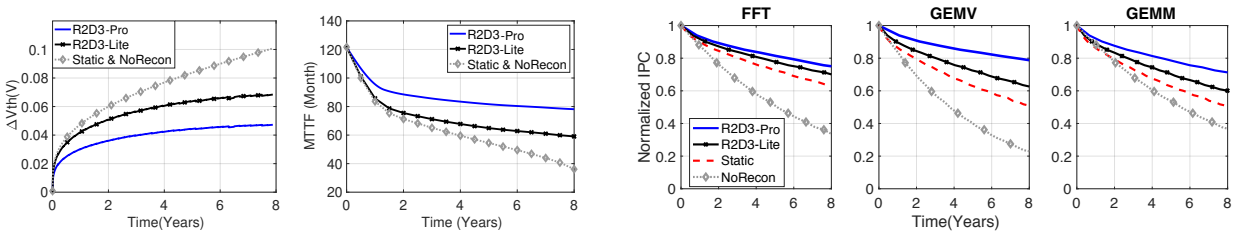
Figure 4(b) breaks down the types of faults for each unit. The column labeled "Total" illustrates the additive results for all pipeline stages in the stage level approach, while the "Core-Level" bar shows the coverage for solutions that target core granularity fault detection. For each case, the bar shows the percentage of detectable faults that are detected, percentage of detectable faults that go undetected when only 10 million ATPG test instructions are run, and the percentage of faults that are undetectable regardless of how many ATPG instructions are run. Figure 4(b) shows the detectable faults (detected plus undetected), which are the target of the detection mechanisms, is 96% for our pipeline-stage-level architecture (Total) compared to 84% for a core-level mechanism. Our solution gives better observability over signals at the stage level, which also faciliates observing faults.

Figure 4(c) gives the breakdown of the percentage of average detectable permanent faults that are detected within a certain number of instructions by running Monte-Carlo fault injection for different applications. For each structure, the bars are divided into several latency stacks from $<$50 to $>$5K instructions. Figure 4(c) shows that, on an average, 96% of detectable permanent faults are detected within 5k clock cycles (bar labeled Total). This is considerably higher than the 63% for a core-level architecture. Although core-level detection is able to achieve greater than 63% coverage rates, this would be at the cost of longer testing period compared to R2D3 pipeline-stage level detection. By increasing T_{test}, the fault detection testing period at the end of each epoch, the coverage rate will increase gradually, but this will also result in a higher power overhead as we are running an otherwise idle leftover stage in parallel. Thus, there is a trade-off between fault coverage and power overhead. Since the proposed fault detection policy is concurrent, T_{test} does not cause any performance overhead. Therefore, to reduce the power overhead and yet achieve a high coverage rate of 96%, we choose 5k cycles as T_{test}.

C. Lifetime Management

1) Thermal Simulation: Although R2D3 is not directly designed for hotspots, it is effective in regulating the overall temperature by managing the usage of hardware resources based on their temperature profile. Figure 6 shows the average temperature map of the hottest layer (farthest from the heat-sink) during execution for Static 3D, R2D3-Lite and R2D3-Pro. R2D3-Pro and R2D3-Lite show up to 33°C and 24°C reduction in average temperature over Static.

2) NBTI-based V_{th} Degradation: We evaluate V_{th} degradation caused by the NBTI effect across NoRecon, Static, R2D3-Lite and R2D3-Pro systems over a period of 8 years (Figure 5(a)). Although Static has the capability to adaptively repair the system by routing



(a) (b) (c)

Fig. 5: Lifetime management evaluation over a period of 8 years a) V_{th} degradation for NoRecon, Static, R2D3-Lite and R2D3-Pro b) Mean Time To Failure for NoRecon 3D, Static, R2D3-Lite and R2D3-Pro c) Instructions Per Cycle (IPC) of NoRecon 3D, Static, R2D3-Lite and R2D3-Pro running different workloads.

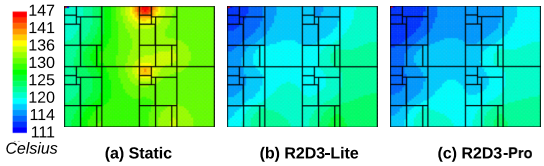


Fig. 6: Average temperature map of the hottest layer (farthest from heat-sink) during execution for (a) Static, (b) R2D3-Lite and (c) R2D3-Pro.

around failed stages, NoRecon and Static degrade identically as neither of them have protection against V_{th} degradation and age at the same rate. In contrast, R2D3-Lite and R2D3-Pro can reduce the NBTI effect by 31% and 53% over NoRecon and Static. Unlike Static that only uses a chosen set of pipeline stages that are reconfigured upon a fault, R2D3-Lite and R2D3-Pro manage to balance the usage of units across the 3D fabric. Since R2D3-Pro differentiates between stages on different layers and assigns an appropriate activity index to each stage, implicitly distinguishing core locations, we attain an additional 30% lower V_{th} degradation over R2D3-Lite.

3) Lifetime and Performance Assessment: Figure 5(b) shows the MTTF comparison between the multicore systems NoRecon, Static, R2D3-Lite and R2D3-Pro over a period of 8 years. On average, R2D3-Pro and R2D3-Lite demonstrate 2.16× and 1.63× improvements in MTTF degradation over NoRecon and Static in 8 years.

Figure 5(c) illustrates the performance in terms of instructions per cycle (IPC) for the three workloads running on different configurations over a period of 8 years. Static, R2D3-Lite and R2D3-Pro show improvements over the NoRecon design, as they are able to reconfigure in case of a fault. Specifically, GEMV achieves a higher performance improvement in comparison to the other two workloads, where R2D3-Pro achieves 3.76× higher performance over NoRecon after 8 years, compared to 2.27× and 1.97× for FFT and GEMM. GEMV is highly parallel compared to the rest. It exhibits higher utilization, power and temperature and consequently incurs higher aging and failure rates. Hence, there are significant benefits from the extra cores that R2D3-Pro can revive by considering temperature and activity profiles.

On average, for the three workloads, R2D3-Pro improves performance by 78% compared to NoRecon over an 8-year period. R2D3-Pro and R2D3-Lite improve performance by 21% and 11% over Static.

VI. Conclusions

R2D3 is the first end-to-end dynamic aging-aware framework built on a fine-grained fault-tolerant reconfigurable architecture to detect faults and eliminate/remedy NBTI effects at a marginal performance cost for monolithic 3D systems. The fact that R2D3 achieves all 4 features of reliability efficiently and incurs low overhead demonstrates how a holistic approach can be used for monolithic 3D or other technologies with more reliability risks. This holistic perspective enables us create a substrate that can be smartly reused and managed and be effective, low-overhead, easy to adopt and practical.

References

[1] A. Amarnath et al. "3DTUBE: A Design Framework for High-Variation Carbon Nanotube-based Transistor Technology". *2019 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*. 2019.

[2] B.Romanescu et al. "Core Cannibalization Architecture: Improving lifetime chip performance for multicore processors in the presence of hard faults". *PACT*. '08.

[3] J. Bagherzadeh et al. "3DFAR: A three-dimensional fabric for reliable multi-core processors". *DATE*. 2017.

[4] K. Constantinides et al. "A Flexible Software-Based Framework for Online Detection of Hardware Defects". *Transactions on Computers* (2009).

[5] K. Constantinides et al. "BulletProof: a defect-tolerant CMP switch architecture". *Proc. HPCA*. 2006.

[6] D. Dutoit et al. "A 0.9 pJ/bit, 12.8 GByte/s WideIO memory interface in a 3D-IC NoC-based MPSoC". *Symp. on VLSI Tech.* 2013.

[7] P. Gadfort et al. "A power efficient reconfigurable system-in-stack: 3D integration of accelerators, FPGAs, and DRAM". *SOCC*. 2014.

[8] S. Gupta et al. "Adaptive online testing for efficient hard fault detection". *International Conference on Computer Design*. 2009.

[9] S. Gupta et al. "StageNet: A Reconfigurable Fabric for Constructing Dependable CMPs". *IEEE Trans. on Computers* (2011).

[10] R. Heald et al. "A third-generation SPARC V9 64-b microprocessor". *IEEE Journal of Solid-State Circuits* (2000).

[11] H. Hong et al. "Lifetime Reliability Enhancement of Microprocessors: Mitigating the Impact of Negative Bias Temperature Instability". *ACM Comput. Surv.* (2015).

[12] A. Islam. "Variability and Reliability of Single-Walled Carbon Nanotube Field Effect Transistors". *Electronics* (2013).

[13] T. Jiang et al. "Through-silicon via stress characteristics and reliability impact on 3D integrated circuits". *MRS Bulletin* (2015).

[14] N. P. Jouppi et al. "In-Datacenter Performance Analysis of a Tensor Processing Unit". *44th ISCA*. 2017.

[15] U. R. Karpuzcu et al. "The BubbleWrap many-core: Popping cores for sequential acceleration". *42nd MICRO*. 2009.

[16] D. Kim et al. "Impact of nano-scale through-silicon vias on the quality of today and future 3D IC designs". *Proc. SLIP*. 2011.

[17] H. Kung et al. "Mapping Systolic Arrays onto 3D Circuit Structures: Accelerating Convolutional Neural Network Inference". *SiPS*. 2018.

[18] C. H. Lin et al. "The effect of NBTI on 3D integrated circuits". *EDAPS*. 2012.

[19] A. Meixner et al. "Argus: Low-Cost, Comprehensive Error Detection in Simple Cores". *In Proc. MICRO*. 2007.

[20] S. Pal et al. "A 7.3 M Output Non-Zeros/J Sparse Matrix-Matrix Multiplication Accelerator using Memory Reconfiguration in 40 nm". *2019 Symposium on VLSI Technology*. 2019.

[21] S. Pal et al. "OuterSPACE: An Outer Product Based Sparse Matrix Multiplication Accelerator". *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. 2018.

[22] S. Panth et al. "Design challenges and solutions for ultra-high-density monolithic 3D ICs". *Proc. S3S*. 2014.

[23] A. Pellegrini et al. "Viper: Virtual Pipelines for Enhanced Reliability". *ISCA*. 2012.

[24] V. Y. Raparti et al. "ARTEMIS: An Aging-Aware Runtime Application Mapping Framework for 3D NoC-Based Chip Multiprocessors". *IEEE TMSCS* (2017).

[25] M. M. Sabry Aly et al. "Energy-Efficient Abundant-Data Computing: The N3XT 1,000x". *Computer* (2015).

[26] A. Sheibanyrad et al. *3D Integration for NoC-based SoC Architectures*. 1st. Springer Publishing Company, Incorporated, 2010.

[27] H. J. Siegel. "Interconnection networks for SIMD machines". *Computer* (1979).

[28] J. Specification. "Failure Mechanisms and Models for Semiconductor Devices". *Paper No. JEP* (2010).

[29] J. Sun et al. "Workload Assignment Considering NBTI Degradation in Multicore Systems". *J. Emerg. Technol. Comput. Syst.* ().

[30] T. Tanaka. "3D-IC technology and reliability challenges". *IWJT*. 2017.

[31] A. Tiwari et al. "Facelift: Hiding and slowing down aging in multicores". *MICRO*. 2008.

[32] Yuan Lin et al. "SODA: A Low-power Architecture For Software Radio". *33rd ISCA*. 2006.

[33] F. M. d'Heurle. "Electromigration and failure in electronics: An introduction". *Proceedings of the IEEE* (1971).