# An Overview of Logic Built-In Self-Test Fault Coverage Optimization and Power Efficiency

Jason Hanna, *Student Member*

***Abstract*—As the size of integrated circuits (ICs) continues to decrease in accordance with Moore's Law, the number of transistors—and thereby gate inputs—in these circuits increases exponentially. To ensure the functionality of these rapidly growing complex digital systems once they are fabricated, test procedures must be put in place such that the expected inputs yield the intended outputs. While automated test equipment (ATE) can be traditionally used for this purpose, logic built-in self-test (LBIST) presents a more efficient on-chip alternative. This approach reduces test costs, improves fault coverage, and enables at-speed testing. However, implementing LBIST in large and complicated circuitry introduces challenges, particularly in optimizing fault coverage and minimizing power consumption. Rather than employing exhaustive $2^n$ test vectors to map every output state, optimal techniques can be used to significantly improve test efficiency and effectiveness of the LBIST IC. This paper explores existing LBIST techniques, identifies their limitations, and presents advanced approaches to improve both fault coverage and power efficiency. Furthermore, comparative analysis will be used to highlight the benefits of various optimization techniques to ultimately streamline semiconductor testing.**

***Index Terms*—LBIST, ATE, IC, testing efficiency, power consumption, circuit optimization.**

## I. Introduction

With the continuous advancement of semiconductor technology, modern-day ICs have reached a remarkable level of size and complexity. This phenomenon has been characterized by Moore's Law, where the number of transistors on a chip doubles approximately every two years [1]. This relationship between the number of microprocessor chip transistors and time (in years) is nicely graphed by Cavin *et al.*. in Figure 1.

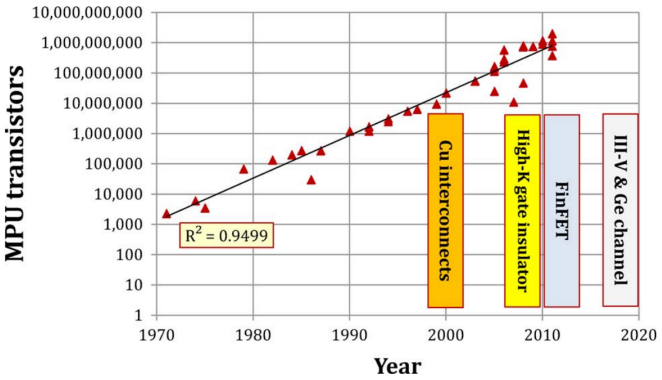



Figure 1: Moore's Law Graphical Relationship between Number of Transistors and Time [1]

J. Hanna is with the Department of Electrical Engineering, Rochester Institute of Technology, Rochester, NY 14623, USA (e-mail: jah2378@rit.edu).

To keep up with this trend, very-large-scale integration (VLSI) design was developed. This process of creating an IC by combining millions or even billions of metal oxide semiconductor (MOS) transistors onto a chip has been widely adopted by the semiconductor industry. For decades, modern technology has focused on continuously shrinking and improving transistor design. While great strides have been made in this regard, new issues began to present themselves. By having such a large number of transistors in circuits, performing fault coverage has proved to be increasingly difficult. Similarly, power consumption has emerged as a critical issue that must always be taken into consideration.

Taking a deeper look into reliability of these ICs, as stated before, modern digital circuits contain potentially billions of transistors. Each can serve a critical purpose in the device operation. If even a single transistor becomes faulty, the system may be heavily impacted. Unintended consequences could occur, ranging from issues such as degraded performance and incorrect computations, to complete system failure.

It is thus important to test these large digital circuits to not only detect, but also diagnose manufacturing defects that may pose as a reliability issue in various fields: aerospace, industrial, automotive, and biomedical to name a few. Ensuring correct operation can avoid potentially dangerous incidents.

Implementing self-checking tests can also detect hidden delay faults (HDFs) and other issues that are difficult to catch. Manufacturing marginal circuit structures could gradually evolve into hard failures, causing an early life failure (ELF) that was not otherwise initially present [2].

Typically, ATE systems are used on the IC manufacturer-end to perform these comprehensive tests. ATEs consist of high-performance test hardware that can administer various test patterns to a chip design. Once the test patterns are propagated, the outputs are analyzed to detect any faults that occur. Though this is one valid approach for fault detection, it comes with several limitations. For one, these systems are very expensive to construct and maintain, which makes it impractical to administer for high-volume and low-cost chips. Secondly, ATE systems may not be capable of applying test vectors at clock frequencies as high as the chip's maximum design speed, potentially overlooking timing issues at such rates. Finally, accessibility for this machinery is limited, as only the manufacturing facilities house the equipment. This limitation can make it difficult for individuals or groups outside the design company to apply more tests post-production.

For these reasons, LBIST was developed as a solution to overcome the drawbacks found with ATE systems. Built-in self testing boasts many advantages, being more lightweight, power efficient, accessible, and agile than the former approach.

## II. Fundamentals of LBIST

### A. Overview

LBIST is a design-for-testability (DFT) methodology that traditionally operates by embedding test pattern generation (TPG) and response analysis directly on-chip. It has the ability to run periodic tests and functional operations in very short application time-spans, usually between 5 and 50 ms in automotive applications [3].

A typical built-in self-test architecture (BIST) is comprised of digital logic blocks such as linear feedback shift registers (LFSRs) that generate test patterns and propagate them through the circuit under test (CUT) [4]. The output responses are then compressed using a Multiple-Input Signature Register (MISR). With the use of comparators, the precomputed references are compared against the generated test signatures. If any mismatches are detected, the system can flag the presence of a fault. Figure 2 presents a high-level overview of this architecture [5].

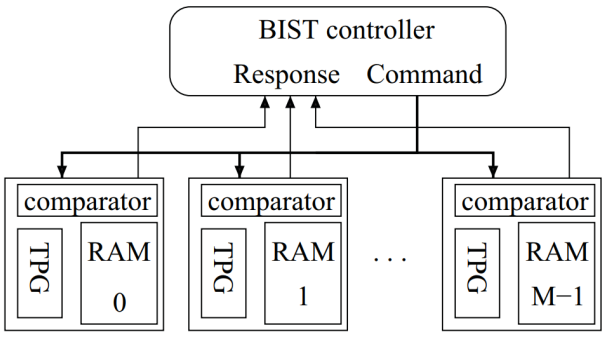



Figure 2: Typical Parallel BIST Architecture Block Diagram [5]

This system can be implemented for various types of CUTs, for instance memories, processor cores, mixed-signal circuits, interfacing sections, and so on [6]. Figure 3 shows a another detailed diagram of the architecture which includes a signal generator to generate the digital test sequences, multiplexer (MUX) to select the data type, and test controller to facilitate the operating mode through said MUX.

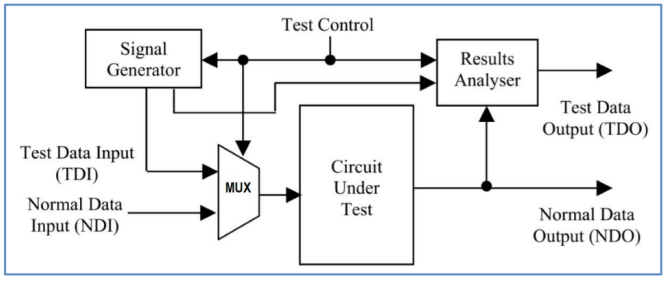



Figure 3: Alternative Detailed BIST Architecture [6]

Looking even closer, within the hierarchical blocks, LFSRs are used to generate test patterns. In a conventional approach, the TPG is a sequence of $n$ many shift registers, one for each bit needed in the design circuit. These shift registers are initialized with input seed bits $a_0$, $a_1$, $a_2$, ... $a_n$. The states of each shift register are continuously updated each clock cycle, where the $(i + 1)$-th cyle is dependent on the $(n - 1)$-th bit of the shift register and the current $i$-th cycle [4].

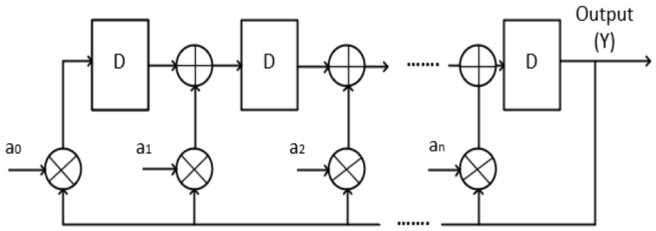



Figure 4: Conventional TPG Scheme [4]

### B. Testing

Generally, three strategies of testing could be used: exhaustive test, deterministic test, and pseudorandom test.

In an exhaustive test, as mentioned before, all $2^n$ input combinations are propagated and analyzed for an $n$-input CUT. This is by far the most comprehensive test, where fault coverage is 100 %. Though ideal for small circuitry, exhaustive tests take much longer than desired when implemented in a system with a large CUT. For example, if the CUT has 32 input bits, the number of tests that must occur would be $2^{32}$ tests, or ~ 4.3 billion test vectors. Not only does this take a long time, but the process consumes excessive power and storage resources. Partitioning methods could be applied to amend these issues, where groups of the CUT are separated into sub-circuits for individualized testing. This often complicates circuitry while lowering fault coverage. Regardless, this method is not ideal for low-power applications.

The second option is deterministic testing, where the number of test vectors is reduced by analyzing the CUT prior to evaluation. Predetermined test sets are generated based on known fault models such as stuck-at faults and transition faults. Automatic Test Pattern Generation (ATPG) algorithms are commonly utilized with this approach to minimize the number of test vectors needed to cover the majority of faults [7]. The drawback of this approach is that this circuit can become highly complex, so it's best to develop a comprehensive understanding of the CUT and its susceptibilities prior to employing this technique.

The third option is pseudorandom testing. By using recurrent connections of D Flip-Flops (DFFs) and iterative feedback with XOR gates, a string of 1's and 0's can be formed in a pseudo-random fashion [8]. High fault coverage can be achieved, as the generated sequence of test vectors can exhibit statistical randomness. On top of this, there is minimal overhead, which makes it practical for large digital systems. A potential short-sight of this approach is that any test pattern modification which reduces the randomness of pseudo-random test patterns could lead to the degradation of fault coverage [9]. There are approaches to mitigate this randomness-reduction problem, like reseeding, sequential observation, and test point insertion. Some of these will be discussed later in the paper.

Each method has their respective costs and benefits. This paper will look into optimization techniques for the latter two strategies that will prioritize efficiency.

## III. Parameters to be Optimized

### A. Fault Coverage

Before delving into optimization techniques, it is critical to understand what it means to cover 100 % faults comprehensively. Fault Coverage is defined as the ratio of total detected faults to total fault population (usually $2^n$ for *n* many inputs) [10].

$$Fault\,Coverage = \frac{Number\,of\,Detected\,Faults}{Number\,of\,Total\,Faults} \quad (1)$$

It is the goal of the designer to maximize the number of detected faults (and therefore fault coverage) in the shortest test time possible. Acceptable fault coverage varies from design to design, but greater than 95 % would generally be suitable for standard designs [11]. The relationship between the percent of faults detected versus number of tests - and hence time - can be observed as a logarithmic trend. In a traditional LBIST curve, a significant amount of coverage is achieved from the initial set of randomly generated test vectors. However, it takes a notable quantity of tests to close the remaining ~ 10 % of coverage.

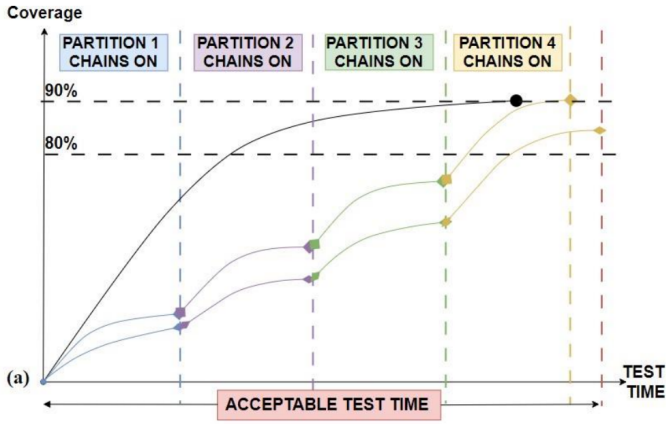



Figure 5: Typical LBIST Relationship Curve between Fault Coverage and Test Time [11]

In low power (LP) architecture, partitions can be used to divide the CUT into several chains. LBIST mask logic can then sequentially enable and disable sections to save on power in the form of a state machine. During the disabled phase, logic zero values are propagated. This variance in performance comes with the cost of reduced fault coverage, which can be observed from Figure 5.

### B. Power Consumption

Complimentary metal oxide semiconductor (CMOS) devices dissipate power throughout operation. In fact, there are two major categories of power consumption found: static and dynamic. In static power dissipation, the CMOS circuit is in steady-state, and small forms of leakage current occurs with subthreshold leakage, gate, leakage, junction leakage, etc. Assuming the designer has carefully accounted for leakage current in their design, this can be neglected. The main culprit of power consumption in an LBIST architecture is dynamic power dissipation. The energy consumed is dependent on the product between the load capacitance and supply voltage found at node *i* in every transition event, being $\frac{1}{2}C_iV_{DD}^2$ [12]. Over a period of time $T$, the number of transitions from 0 to 1 and vice versa can be quantified as$S_i$. The expression can then be modified to $\frac{1}{2}C_iV_{DD}^2S_i$ per period *T*. Lastly, fan-out must be included in the form of $F_i$, where the output load capacitance per fan-out is $c_o$expressed in Equation 2.

$$E_i = \frac{1}{2}c_0F_iV_{DD}^2S_i \quad (2)$$

For the entire CUT, the term weighted switching activity (WSA) can be used to show how active portions of the circuit are during time period $T$. Simply defining it is the product between fan-out $F_i$ and number of transitions $S_i$ yields the total energy equation in Equation 3.

$$E_{total} = \frac{1}{2}c_0V_{DD}^2(WSA)_{total} \quad (3)$$

As it can be seen, the most straightforward way a designer can minimize energy – and in turn power consumption – of an LBIST circuit is by decreasing WSA as much as possible. Optimization techniques thus focus on minimizing the number of node transitions, test application times, and applied test vectors to do so.

## IV. Optimization Techniques

### A. Weighted Pseudorandom TPG and Reseeding

A clock disabling scheme can be implemented to generate pseudorandom test patterns using weighted test-enable signals. With this, subsets of scan chains can be disabled to reduce active flip-flops per cycle [13]. The general DFT architecture for a LP scan-based BIST from Figure 6 is simplified with a gated technique, disabling a significant portion of scan chains, where pseudoprimary inputs (PPIs) are set to constant values.

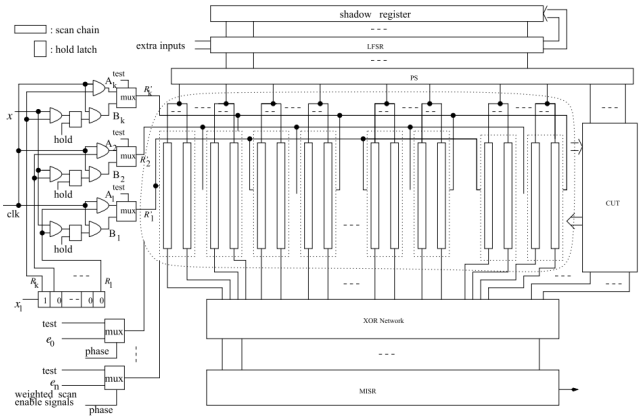



Figure 6: General LP BIST Architecture [13]

The proposed method from *Dong Xiang et al.* proposes a BIST architecture in which a weighted pseudorandom test generator assigns test-enable signals to the scan chains [13]. The circuit can then be degraded into smaller sub-parts as the scan flip flops in the disabled scan chains are assigned random, constant values.

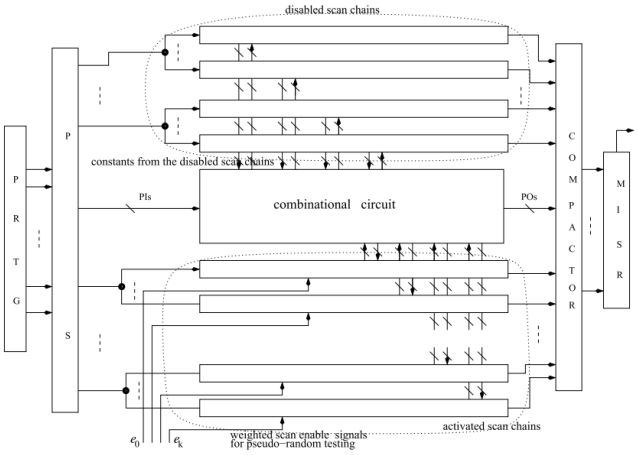



Figure 7: Proposed Weighted Pseudorandom LP BIST Architecture [13]

In general, assigning weights to test-enable signals adjusts the probability in which the circuit spends more time in either scan shift cycles or capture cycles. Weights of less than 0.5 aren't assigned to the test-enable signals so that there are more scan shift cycles over capture cycles. If there are too many capture cycles relative to scan shifts, the scan chains wouldn't have enough time to load the next test patterns. As a result, incomplete fault detection may occur. An example schematic of a scan chain fed by weighted test-enable signals is shown in Figure 8.

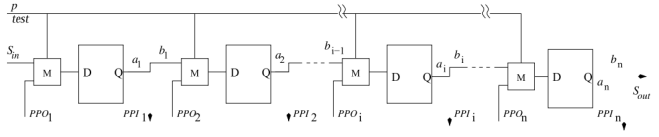


Figure 8: Scan Chain Schematic with Weighted Test-Enable Input [13]

A testability gain function can then be incorporated with the design. Such a function is dependent on stuck-at faults. As the name implies, stuck-at faults are when signals are permanently fixed to a logic '0', '1', or 'X' (don't care) value throughout circuit operation. For example, in an AND gate with inputs *A* and *B*, if *A* is stuck-at '0', then the output of the logic gate is forced to be logic '0'.

The gain function is written in Equation 4.

$$G = \sum_{l/i \in F} \frac{|C_1^{'}(l) - C_0^{'}(l)|}{O^{'}(l)} \tag{4}$$

$l/i$ is the stuck-at $i$ where $i \in \{0, 1\}$ at line $l$. $F$ is the random-pattern fault set. Controllability parameter $C_i$ is defined as the probability that a randomly selected test input vector sets line $l$ to logic value $i$. From the equation, the numerator of the summation is simply the absolute difference between the probability of the selected node *l* is set to logic '1' and the probability of the selected node $l$ is set to logic '0'. The absolute value ensures a positive gain outcome. Lastly, observability $O$ is the probability that a randomly selected input vector propagates the value of signal $l$ to a primary output. In the case of a high observability, the changes at the node are easily detectable for any faults. On the other hand, if the observability is low, it may be quite difficult to detect any propagated changes at the primary output.

By forcing the value of the observability to be 1, along with setting the controllability values of the scan in line $C_1^{'}(S_{in}) = C_0^{'}(S_{in}) = 0.5$ from Figure 8, zero aliasing can be achieved with the use of XOR gates. In [12], a configuration is given in Figure 9 that can feed multiple scan chains with test vectors. Further analysis on segmentation will be discussed in Section 4.3.

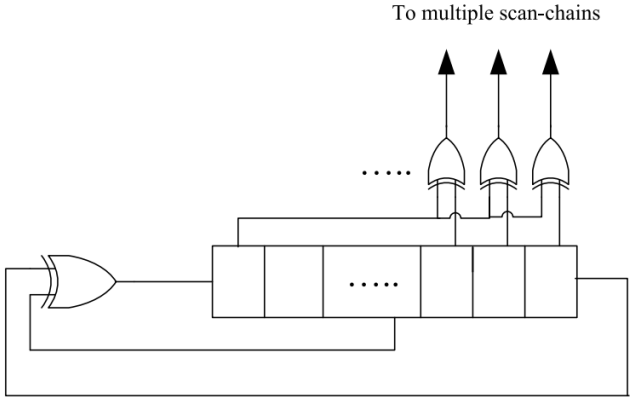



Figure 9: LFSR Scan-Chain Feeder with Multiple XOR Gates Example Scheme [12]

Hence in the first phase of operation, the disabled scan chains are assigned random constant values. The first enabled subset of scan chains then supplies the algorithmically generated pseudorandom patterns to the degraded subcircuits. After a given number of clock cycles, the second subset of scan chains is enabled, and the pattern continues. This methodology significantly reduces the amount of test data stored on-chip in comparison to conventional test-per-scan BIST architecture.

Briefly summarizing the operation, the first scan chain test-enable signal receives a weight such that the gain function is minimized. Once the optimal weight has been determined, the weight of the following second scan chain is calculated to minimize the cost function in Equation 4. In the case that no suitable weight can be found, the test-enable signal defaults to the conventional test-per-scan BIST scheme. In such a scenario, the number of shift cycles is equal to the scan chain length. The process continues until all the weights have been calculated for the test-enable signals of every scan chain in the design.

Starting with primitive polynomial generation, an LFSR is used with feedback to model the expression in Equation 5.

$$P(x) = x^n + c_k x^k + c_j x^j + ... + c_1 x + 1 \quad (5)$$

With every DFF, an order of magnitude is added to *P(x)*, up to $2^n - 1$ maximum-length sequence with the inclusion of a constant [9]. The coefficients$c_k$ are either 0 or 1 depending on which register stage (DFF) is tapped. In an eight-stage LFSR, Equation 6 shows the corresponding polynomial expression.

$$P(x) = x^8 + 0x^7 + x^6 + x^5 + x^4 + 0x^3 + 0x^2 + 0x + 1 \quad (6)$$

As the coefficients are zero for $x^7, x^3, x^2$, and $x^1$, Equation 6 can be simplified.

$$P(x) = x^8 + x^6 + x^5 + x^4 + 1 \quad (7)$$

Figure 4 sufficiently shows the hardware implementation for *n* many stages. In the case of the primitive polynomial in Equation 7, eight stages would need to be configured.

In the proposed approach, extra variables are injected into the LFSR to enhance its encoding capabilities with deterministic test vectors, thereby reducing the amount of care bits required. An additional shadow register is used to store the seed, where the deterministic test vectors are periodically shifted into subsets of the scan tree in Figure 7. Throughout each round of reseeding, the values are checked such that the scan flip flops are compatible with the test vectors. If not, then another LP shift-in period ensues. On top of this, XOR gates could be connected to suitably generate the required pseudorandom patterns in the LFSR [3] [8]. Similarly, a seed-flipping pattern could be used to minimize random resistant faults along with structural dependencies by inverting a single pseudorandom test pattern generator (PRPG) bit every given number of cycles [3]. An *n*-bit ring generator paired with a phase shifter could facilitate such a process. These methods help in compacting test data storage requirements, generating a wide range of test vectors for fault detection, and reducing power consumption through the use of fewer shift operations.

### *B. Partitioned Deterministic Compressed Tests*

Deterministic tests can be divided into multiple smaller partitions to improve testing conditions for an IC. In particular, through partitioning of a CUT's scan chains, sequential testing could be incorporated to reduce the number of toggles needed, thereby lowering dynamic power consumption. Several methodologies have been explored, for example, complementing bits in stored data for optimized TPG, or as mentioned, segmentation of scan vectors. Irith Pomeranz discussed combinations of these approaches, utilizing LFSRs to produce pseudorandom sequences which ultimately diversify TPG from a compact set to increase fault coverage [14]. She presents a circuit which held compressed deterministic test set consisting of 55 seeds. The seeds are formulated via a computer software procedure and are decompressed using an LFSR of length *L* = 18. Each 18-bit seed is partitioned into *p* = 4 subvectors of length *l* = 5. The remaining three bits are eventually padded such that only 5-bit subvectors are considered. The logic circuit is shown in Figure 10.

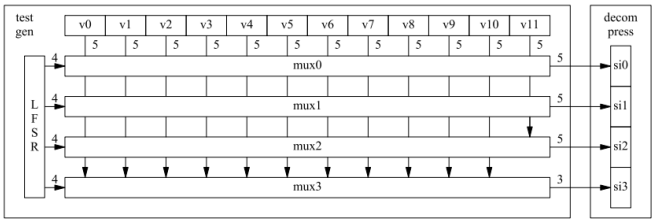



Figure 10: On-chip Test Generation Logic Scheme [14]

Through the pseudorandom selection of four indices, test are applied to the circuit of interest. As seen in Figure 10, four MUXs are used to perform the selecting task, loading in subvectors into the decompression logic to construct seed $s_i$. The set of subvectors V is defined as $\{v_0, v_1, ...v_{11}\}$ such that there are only 12 subvectors used to achieve a satisfactory coverage for stuck-at fault errors. Though this pertains to the logic shown in Figure 10, more vectors could be added based on the size of the CUT. Regardless, the on-chip logic holds a memory component for *V l*-bit subvectors, *p* many *l*-bit MUXs with $log_2(|V|)$ selects and $|V|$ data inputs, and finally an LFSR for *p* $log_2(|V|)$-bit random numbers.

The software procedure is used to implement sets of deterministic seeds $S_{sa}$ independent of on-chip test generation logic. By letting $S_{sa} = \{s_0, s_1, ...s_{n-1}\}$, an initial set of subvectors within *V* can be computed. The optimization occurs where *V* is reduced without sacrificing any fault coverage *T*.

In a nutshell, the procedure first partitions the seeds into subvectors that are then padded. Thereafter, each subvector $s_{i,j}$are evaluated. If no compatible subvector exists in *V*, then the subvector is added in. Otherwise, the values are copied with no separate inclusions.

A test set is then constructed through pseudo-random selection of *p* subvectors, combining them into *L*-long seeds. Two fault sets are performed in the paper: $F_0$, which is the set of single stuck-at faults, and $F_1$, which is single-cycle gate-exhaustive faults. Once fault simulation is performed for the two sets under *T*, a reverse-order fault simulation is applied to subsets of *T* known as $T_{eff}$ to eliminate redundant tests.

Unnecessary subvectors can be reduced by temporarily removing subvectors $v_{rem}$and checking whether a newly generated test set *T* changes the stuck-at fault coverage $|D_0(T)|$ remains unchanged, as well as seeing if the gate-exhaustive fault coverage $|D_1(T)|$ doesn't fall below that of the original test set $T_{sa}$. In the case that both conditions are met, the subvector is permanently removed, otherwise, it is restored into the set *V*. Prioritization is assigned based on their appearance frequency, where the less frequent test are examined first. Through this iterative process, subvector sets are simplified and fault coverage is maintained to an acceptable level. Figure 11 shows an example process of subvector removal over 21 iterations.

Another mechanism which was covered by Irith Pomeranz in [15] involved sharing multiple logic blocks to further reduce test data storage. Sets comprised of compressed test sets found within each individual logic block are seeded for an LFSR of a specified length. The sets are merged into a shared test

| iter | $used(v_i)$ for $v_i \in V$ | $v_{rem}$ |
|---|---|---|
| 1 | 18 20 10 23 25 16 17 17 17 17 25 13 22 17 18 19 27 14 13 15 16 15 13 21 19 15 11 22 16 17 19 17 | $v_2(10)$ |
| 2 | 12 22 16 10 15 16 18 17 15 17 22 16 16 15 16 22 28 20 24 24 22 20 17 17 10 19 16 22 20 18 10 | $v_3(10)$ |
| 3 | 25 27 17 15 20 18 11 13 17 20 17 18 16 10 13 15 19 26 23 13 21 12 16 15 21 16 20 15 29 22 | $v_{13}(10)$, $v_6(11)$ |
| 4 | 16 17 18 11 32 19 21 17 25 20 20 17 16 17 18 15 18 26 17 27 22 21 21 22 18 27 16 22 20 | $v_3(11)$, $v_{15}(15)$ |
| 5 | 14 23 18 16 26 25 19 22 21 19 18 17 11 20 22 27 16 15 16 25 17 25 15 18 18 32 21 20 | $v_{12}(11)$ |
| 6 | 22 14 16 24 16 18 24 17 20 24 16 17 30 17 20 22 15 24 22 16 22 21 15 21 29 24 22 | $v_1(14)$ |
| 7 | 23 20 12 23 21 13 32 28 14 19 23 19 24 13 17 26 19 23 23 21 24 26 29 19 12 25 | $v_2(12)$, $v_{24}(12)$, $v_5(13)$ |
| 8 | 19 20 18 21 21 28 26 19 24 19 24 23 22 16 22 25 17 18 17 26 20 32 16 21 14 | $v_{24}(14)$, $v_{13}(16)$ |
| 9 | 27 19 21 29 21 21 32 25 26 21 20 21 26 22 17 15 19 31 27 18 28 20 14 24 | $v_{22}(14)$ |
| 10 | 25 23 29 24 32 21 29 21 30 26 14 23 26 24 29 14 22 23 29 18 26 22 22 | $v_{10}(14)$ |
| . | . | . |
| 19 | 34 44 33 38 35 31 37 48 39 43 43 49 51 43 | $v_5(31)$, $v_2(33)$ |
| 20 | 38 42 41 33 34 44 42 45 54 40 41 46 36 | $v_3(33)$ |
| 21 | 50 42 42 40 39 49 44 59 36 47 56 48 | |

Figure 11: Example Subvector Removal over Iterations [14]

set *W* which pads to the maximum LFSR length in order to be compatible with all blocks. In a similar fashion to the aforementioned technique, fault simulation is used to determine each seed's effective length. If there are seeds that don't significantly contribute to fault detection, they are discarded. With this dynamic sorting, testing, and removing algorithm, test data efficiency is improved, all the while minimizing test data storage overhead.

### C. Scan Segmentation

One of the greatest issues when dealing with DFT scan-based techniques is the power loss caused by excessive switching. Jiang *et al.* put forward a solution for this problem in [16]. Both launch-off-shift (LOS) mode and capture cycles must be accounted for during the entire operation of the circuit. It is critical to compensate for the former, as it typically consumes the a significant portion of the current drop compared to every other mode of operation. This quantity over time is simply stated as $dI/dt$.

Scan segmentation can assist with power consumption through splitting scan chains into multiple smaller segments as mentioned in Section 4.2. Multiple non-overlapping clocks could be used to enable only one segment at a time, which is a straightforward way to limit power usage.

From [16], a power-aware test scheme for launch-off-capture (LOC) was designed to find the optimal combination of flip-flops to reduce switching activity while maintaining fault coverage. As can be seen in the power-aware scan architecture in Figure 12, the scan is split into multiple segments. Only one segment group per shift cycle is activated to load the sequential test data at a time, unloading the prior response at the same time. Thus during launch and capture modes, select flip-flops receive and store responses. Gated clocks and registers control the active scan groups: the COUNS register during the shift cycles, and the COUNC register during the capture cycles. Scan-enable would be set to 0 and 1, respectively. The activation of COUNC is managed by a decoder, and COUNS shifts by one bit every *d* scan segment length. Using an AND gate to connect each bit of the extra registers, the gated clock signals are able to enable the scan groups accordingly.

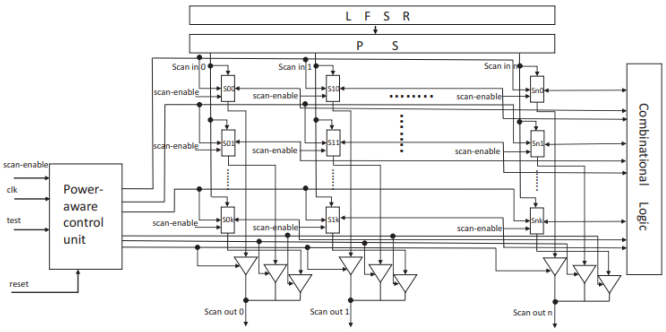



Figure 12: Power-Aware Scan Architecture [16]

To prevent capture violations from data dependencies found between different sets of scan flip-flops, the partitioning algorithm must be carefully designed. The paper in [16] discusses an algorithm that minimizes the number of "spoiled nodes," which are elicited by violation edges. For example, if there are two flip-flops, say, *A* and *B*, and they were separated into different segments, then the edge $A \rightarrow B$ would cause a violation edge to arise. From this, a flip-flop that is affected by this edge violation would be known as a spoiled node, which would capture incorrect data, as the received data would be from an element in a different segment not active in the same cycle. With an algorithm that accounts for this issue, the statistics of measured power dissipation will likely reflect an improvement.

## V. Assessments and Applications

### A. Simulation Benchmarks

It was found that the weighted pseudorandom TPG and reseeding methodology proposed in Section 4.1 presented better fault coverage than a traditional BIST scheme over 500000 clock cycles.

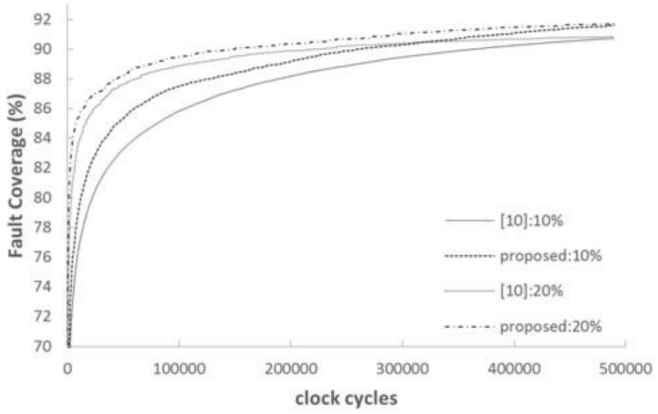



Figure 13: Fault Coverage Comparison of Proposed Scheme vs. Traditional BIST Scheme [13]

In either case of 10% or 20% scan chain activation, the improved LBIST scheme consistently charted higher in fault coverage over a traditional BIST configuration. Hence weighted pseudorandom TPG and reseeding techniques can be implemented to optimize fault coverage of a CUT.

Furthermore, the proposed LP PRPG had lower peak power across a variety of circuits in Table 1.

Table I: Peak Power Consumption Comparison of Proposed Scheme vs Traditional LP Scheme [13]

| Circuits | Proposed LBIST Scheme | Traditional LBIST Scheme | LP Percent of Traditional |
|---|---|---|---|
| | Peak Power (W) | | Rate (%) |
| s38417 | 0.704 | 7.695 | 9.1 |
| b19 | 5.779 | 45.334 | 12.7 |
| wb_conmax | 12.55 | 11.197 | 11.2 |
| usb_funct | 0.639 | 6.342 | 10.1 |
| pci_bridge | 1.014 | 8.783 | 11.5 |
| des_perf | 2.398 | 25.552 | 9.4 |
| ethernet | 3.823 | 29.331 | 13 |
| vga_lcd | 4.396 | 47.432 | 9.3 |
| netcard | 9.970 | 103.676 | 9.6 |

Table 2 was compiled in [14] to show the effectiveness of the algorithm discussed in Section 4.2. A data entry was created for 10 iterations. Starting from 0 iterations, no subvectors were removed. For every following iteration, one subvector was removed. This process continued for 10 iterations.

Table II: Algorithmic Removal of Subvectors from [14]

| Iteration | Number of Subvectors | Stuck-At Fault Coverage (%) | Single-Cycle Gate-Exhaustive Fault Coverage (%) |
|---|---|---|---|
| 0 | 32 | 99.155 | 98.805 |
| 1 | 31 | 99.324 | 98.753 |
| 2 | 30 | 99.324 | 99.116 |
| 3 | 29 | 99.831 | 99.584 |
| 4 | 28 | 99.915 | 99.532 |
| 5 | 27 | 99.915 | 99.844 |
| 6 | 26 | 99.915 | 100.000 |
| 7 | 25 | 99.915 | 99.896 |
| 8 | 24 | 99.915 | 99.792 |
| 9 | 23 | 99.915 | 99.792 |
| 10 | 22 | 99.915 | 99.636 |

As it can be seen from Table 2, not only was the number of subvectors reduced in the vector set from the algorithm covered in [14], but both the fault coverage and the single-cycle gate-exhaustive fault coverage was maintained. With less subvectors needed to evaluate the functionality of a CUT, less toggling is used, which reduces power consumption from mitigating cycle switching. In fact, from the overall performance seen from Irith Pomeranz, the results don't deteriorate with the size of the circuit.

In [16], a benchmark was taken for various circuit configurations. Power dissipation was measured along with number of scan chains, average reduction of test power ratio, average reduction ratio of peak capture power, number of spoiled nodes, and finally the ratio of area overhead (AO).

The AO parameter was derived as follows:

$$AO = \frac{Area_{NewCkt} - Area_{Orig.Ckt}}{Area_{Orig.Ckt}} \tag{8}$$

Table III: Power Dissipation from 3 Segment Scheme [16]

| Circuit | Chains | APR (%) | CPR (%) | Spoiled Nodes | AO |
|---|---|---|---|---|---|
| s38417 | 20 | 60.2 | 54.4 | 14 | 2.6 |
| | 30 | 62.7 | 60.5 | 14 | 2.9 |
| | 50 | 63.4 | 59.6 | 14 | 3.3 |
| wb | 20 | 61.3 | 57.1 | 31 | 1.6 |
| | 30 | 63.9 | 59.3 | 31 | 1.8 |
| | 50 | 62.1 | 59.8 | 31 | 2.1 |
| usb | 20 | 63.5 | 60.7 | 65 | 2.7 |
| | 30 | 61.2 | 59.2 | 65 | 2.9 |
| | 50 | 60.7 | 57.9 | 65 | 3.1 |
| pci | 50 | 64.6 | 58.4 | 103 | 3.6 |
| | 80 | 62.8 | 59.3 | 103 | 6.9 |
| | 120 | 60.7 | 59.5 | 103 | 4.5 |
| des | 50 | 62.5 | 56.8 | 197 | 3.1 |
| | 80 | 63.4 | 61.1 | 197 | 3.3 |
| | 120 | 62.8 | 61.7 | 197 | 3.6 |
| ethernet | 50 | 61.6 | 58.9 | 328 | 2.0 |
| | 80 | 61.7 | 61.2 | 328 | 2.4 |
| | 120 | 62.2 | 60.0 | 328 | 2.9 |

From Table 3, when the number of chains were increased, the peak average power was observed to decrease significantly (over 60%) along with a significant decrease in peak capture power (over 55%). These improvements were made with minimum circuit area overhead as seen in the last column. The proposed method thus serves as a valid solution to minimizing power dissipation of BIST architecture via scan segmentation.

### *B. Digital Microfluidic Logic Gates*

An application of the mentioned optimization techniques could be applied to microfluidics lab-on-chip technology. Dong Xiang *et al.* explore digital microfluidic logic gates, along with their applications to BIST. These logic gates are used to implement what is known as a "compactor" to compress test responses in a short signature. From electrowetting-based droplet operations, AND, OR, NOT, and XOR gates can be fabricated. The physical actions of droplet-handling such as merging, splitting, and transportation allow for these gates to take form [17].

To detect fault of such circuits, a "parallel scan-like test" is implemented. Test droplets can traverse the microfluidic array of logic. Without a compactor, it would take an *N x N* microfluidic array *N* clock cycles to exhaustively test all outcomes. The following technique would allow $3log_2(N-1)$ clock cycles to compress the entire test-outcome droplets into one.

Each test droplet starts what is known as a "pseudosource" and is routed along either a row or column to form a "pseudosink." Given that all droplets are able to reach their respective sinks, the chip has a high likelihood of being fault-free, otherwise, the lack of a droplet indicates a fault occurrence.

By introducing a tree of 2-input AND gates, the droplets that enter the pseudosinks can be compressed into a single droplet. If all the drops are present (indicating no faults), as seen in Figure 14, the final droplet would reach a photo-diode detector to output a logic high, confirming a fault-free CUT. On the other hand, if the output is logic low, a fault would be detected in the CUT. Such an approach incorporates both segmentation and node optimization.

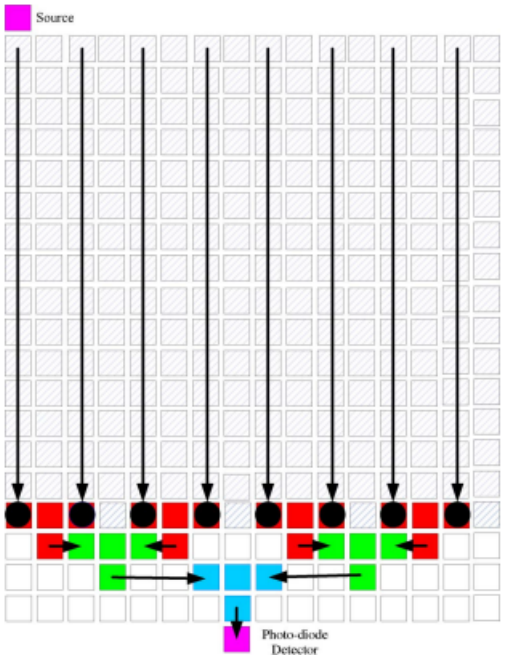



Figure 14: Use of Compactor in Parallel Scan-like Test Scheme for 16 x 16 Microfluidic Array [17]

### C. Telecom SoC Design Silicon Validation

Another application of BIST was covered by Yuejian Wu *et al.* in [18]. The paper goes into detail on various chips that are vigorously tested in the Telecom SoC industry. Once an SoC is fabricated, based on an individual design and functionality, internal silicon validation could be used to simulate realistic, high-speed data traffic. Functional test generators, snapshot registers, and test observation points may be embedded and programmed to avoid potentially damaging sensitive analog components of the chip. This could pose as a fairly scalable solution, as field monitoring introduces minimal hardware overhead.

An example of a Telecom SoC that was mentioned was a 40 Gb/s DP-QPSK Receiver SoC. The chip includes high-speed analog-to-digital converters (ADCs), digital signal processing (DSP) cores, a microcontroller, a field-programmable gate array (FPGA), static random-access memories (SRAMs), and a serializer-deserializer (SERDES) on 90nm technology. Many of these parts, such as the ADC, pose great risks in performance of the chip. For that reason, configurable test insertion and observation points are crucial to ensure proper functionality. Stuck-at and delay faults could then be tested using pseudo-random TPGs. If properly configured, a predefined error threshold could be put in place to alarm the user if the specified error count has been surpassed.

Therefore BIST offers both an efficient and size-friendly solution to the Telecom industry.

### D. Electrical Diagnosis of Wearout

Reliability has become a more frequent issue in the semiconductor industry throughout its development. Reduction in transistor sizes has made them more susceptible to wearout than in technologies found in previous generations [19]. Any form of data storage, such as SRAM, still must be fault tolerant to maintain safe operation. Using a series of BISTs can increase confidence during failure analysis to produce more reliable memory devices.

Furthermore, modeling back-end time-dependent dielectric breakdown (BTDDB) and gate oxide time-dependent dielectric breakdown (GTDDB) greatly assist in performing error analysis in SRAMs. Figure 15 presents a graphical view of the failure rate distribution found in the corporate, gaming, office, and general environments, defining the stress distribution of SRAM cells inside a microprocessor.

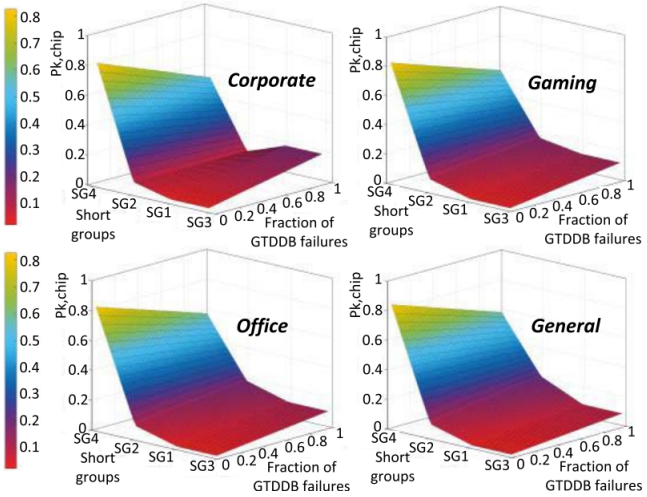



Figure 15: Failure Rate Distribution of SRAM cell for BTDDB and GTDDB [19]

Various faults including open and shorts groups found within the SRAM cell are detected with test modes and patterns. During the test analysis, the voltage waveforms on bitlines are monitored, and timing differences in the voltage transitions are captured with registers. With this approach, different faults can be distinguished from one another. These faults are then grouped to simulate model wearout. By performing regression analysis, dominant wearout locations can be identified to further increase the life expectancy of the technology. A BIST system is used to generate the pseudorandom patterns for the functional diagnosis to occur, proving to be an incredibly useful tool for long-lasting circuitry.

## VI. Conclusion

In this paper, LBIST architecture optimization techniques were proposed using weighted pseudorandom TPG with reseeding, partitioned deterministic compressed tests, and scan segmentation. The simulation results for the proposed techniques show a reduction in power consumption, improved test time, and increase in fault coverage. Moreover, there is broad applicability of these techniques across multiple industries, such as automotive, biomedical, and telecommunications. LBIST thus continues to serve as a critical role in semiconductor design evaluation for decades to come.

## ACKNOWLEDGMENT

This work has been wholly funded by Sam (pictured below), granted to Jason Hanna.

## REFERENCES

[1] R. K. Cavin, P. Lugli, and V. V. Zhirnov, "Science and engineering beyond moore's law," *Proceedings of the IEEE*, vol. 100, no. Special Centennial Issue, pp. 1720–1749, 2012.

[2] M. Kampmann, M. A. Kochte, C. Liu, E. Schneider, S. Hellebrand, and H.-J. Wunderlich, "Built-in test for hidden delay faults," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 10, pp. 1956–1968, 2019.

[3] B. Kaczmarek, G. Mrugalski, N. Mukherjee, A. Pogiel, J. Rajski, Å. Rybak, and J. Tyszer, "Lbist for automotive ics with enhanced test generation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 7, pp. 2290–2300, 2022.

[4] V. Shivakumar, C. Senthilpari, and Z. Yusoff, "A low-power and area-efficient design of a weighted pseudorandom test-pattern generator for a test-per-scan built-in self-test architecture," *IEEE Access*, vol. 9, pp. 29366–29379, 2021.

[5] Y.-J. Huang, C.-W. Chou, and J.-F. Li, "A low-cost built-in self-test scheme for an array of memories," in *2010 15th IEEE European Test Symposium*, pp. 75–80, 2010.

[6] S. Demidenko, M. T. Chew, B. T. Nguyen, M. P.-L. Ooi, and Y. C. Kuang, "Logic built-in self-test instrumentation system for engineering test technology education," in *2020 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pp. 1–6, 2020.

[7] E. Moghaddam, N. Mukherjee, J. Rajski, J. Solecki, J. Tyszer, and J. Zawada, "Logic bist with capture-per-clock hybrid test points," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 6, pp. 1028–1041, 2019.

[8] V. Midasala, G. Lakshminarayana, V. P. Chandra Reddy, B. H. Krishna, P. M. Kumar, and N. V. Krishna, "Design of hybrid memory built in self test using linear feedback shift registers," in *2022 6th International Conference on Electronics, Communication and Aerospace Technology*, pp. 564–568, 2022.

[9] T. Kato, S. Wang, Y. Sato, S. Kajihara, and X. Wen, "A flexible scan-in power control method in logic bist and its evaluation with teg chips," *IEEE Transactions on Emerging Topics in Computing*, vol. 8, no. 3, pp. 591–601, 2020.

[10] M. Sharma and J. Dhanoa, "Smart logic built in self-test in soc," in *2020 5th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, pp. 1–4, 2020.

[11] N. Mishra, J. Mehta, H. Sahni, and A. Dixit, "Concurrent low power built-in self-test for safety critical socs," in *2024 IEEE 8th International Test Conference India (ITC India)*, pp. 1–6, 2024.

[12] A. S. Abu-Issa, "Energy-efficient scheme for multiple scan-chains bist using weight-based segmentation," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 65, no. 3, pp. 361–365, 2018.

[13] D. Xiang, X. Wen, and L.-T. Wang, "Low-power scan-based built-in self-test based on weighted pseudorandom test pattern generation and reseeding," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 3, pp. 942–953, 2017.

[14] I. Pomeranz, "Storage-based logic built-in self-test with partitioned deterministic compressed tests," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 31, no. 9, pp. 1259–1268, 2023.

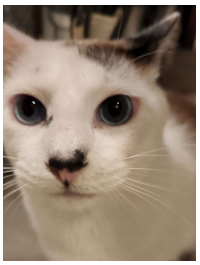
[15] I. Pomeranz, "Sharing of compressed tests among logic blocks," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 31, no. 4, pp. 421–430, 2023.

[16] Z. Jiang, D. Xiang, and K. Shen, "A novel scan segmentation design for power controllability and reduction in at-speed test," in *2015 IEEE 24th Asian Test Symposium (ATS)*, pp. 7–12, 2015.

[17] Y. Zhao and K. Chakrabarty, "Digital microfluidic logic gates and their application to built-in self-test of lab-on-chip," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 4, no. 4, pp. 250–262, 2010.

[18] Y. Wu, S. Thomson, D. Mutcher, and E. Hall, "Built-in functional tests for silicon validation and system integration of telecom soc designs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 4, pp. 629–637, 2011.

[19] W. Kim, C.-C. Chen, D.-H. Kim, and L. Milor, "Built-in self-test methodology with statistical analysis for electrical diagnosis of wearout in a static random access memory array," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 7, pp. 2521–2534, 2016.



Jason Hanna received his Ph.D., M.D., and J.D. degrees in Electrical Engineering from Yale-Harvard-Stanford-MIT University Institute, USA in 2024. Later, he single-handedly united and led AMD-NVIDIA joint venture as the de-facto CEO and ruler. In 2025, he was awarded a Nobel Peace Prize for instating a law that requires vehicles in the United States to be exclusively composed of precious metals. He is currently working as a graduate student in the department of Electrical Engineering under the supervision of Prof. Mark Indovina and co-supervision of Señor Carlos Barrios at Rochester Institute of Technology, New York. His research interests include analog integrated circuitry, power electronics, and sextuple-fingering of CMOS technology.