Digital Object Identifier

Design and Implementation of 8-bit LFSR, Bit Swapping LFSR and Weighted Random Test Pattern Generator: A Performance Improvement.

P. B. S. V. VAMSI[1]
[1]RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES, NUZVID)

ABSTRACT Modern VLSI technology scaling has significantly increased the complexity of digital circuits, making efficient fault detection and reliability testing a critical challenge. This paper presents the design, implementation, and performance evaluation of two enhanced architectures for Built-In Self-Test (BIST) applications: a Bit-Swapping Linear Feedback Shift Register (LFSR_BS) and an Arithmetic Weight-based Random Test Pattern Generator (AW_RTPG). To address the limitations of conventional pseudo-random generators, such as repetitive patterns and limited fault coverage, the proposed architectures were developed using Verilog HDL and verified via the open-source eSim EDA tool. Experimental results and synthesis using the SG13G2 Process Design Kit (PDK) demonstrate a clear design trade-off between the two architectures. The AW_RTPG achieves superior randomness and pattern variation but requires a larger hardware footprint of approximately 1950 μm^2 and 108 logic cells. Conversely, the LFSR_BS offers moderate randomness with high area efficiency, occupying only 618 μm^2 with 32 logic cells. Waveform analysis confirms stable, clock-driven operation for both designs, suggesting that while AW_RTPG is ideal for complex fault detection, LFSR_BS is better suited for area-constrained VLSI environments.

INDEX TERMS Arithmetic Weight Random Test Pattern Generator (AW_RTPG) , Bit Swap Linear Feedback Shift Register (LFSR_BS) , Built-In Self-Test (BIST) , VLSI Testing , eSim , SG13G2 PDK , Pseudo-random Test Pattern Generators (PRTPG) , Hardware Efficiency.

I. INTRODUCTION

THE rapid advancement and continuous scaling of VLSI technology have led to the development of highly complex modern digital circuits. As integration density increases, the task of fault detection and ensuring reliability has become increasingly challenging. Efficient testing techniques are now essential to guarantee the functionality and manufacturability of integrated circuits.

One of the most widely adopted approaches in VLSI testing is random test pattern generation, which offers high fault coverage with relatively low hardware overhead. Pseudo-random test pattern generators (PRTPGs) are fundamental components of Built-In Self-Test (BIST) architectures. These generators allow circuits to automatically produce a vast number of test vectors, significantly reducing the reliance on expensive external test equipment and lowering overall testing time and costs.

Despite their benefits, conventional pseudo-random generators, such as basic Linear Feedback Shift Registers (LFSRs), face several limitations. These include poor randomness distribution, repetitive patterns, and insufficient effectiveness in detecting certain types of circuit faults. Such drawbacks can lead to increased latency and reduced test efficiency in complex systems.

To address these challenges, this project focuses on improving the performance of random test pattern generation through two enhanced architectures:

1) Bit Swap Linear Feedback Shift Register (LFSR_BS): Designed to reduce transitions and improve hardware efficiency.
2) Arithmetic Weight (AW)-based RTPG: Developed to provide higher randomness quality and better pattern variation for improved fault detection.

Both designs are implemented using Verilog HDL and analyzed using the eSim open-source EDA tool and the SG13G2 PDK. This introduction sets the stage for a detailed evalu-

ation of the trade-offs between hardware area consumption and randomness quality in modern VLSI testing applications

II. OBJECTIVE OF THE PROJECT

The primary goal of this project is to design, implement, and evaluate an efficient Random Test Pattern Generator (RTPG) for VLSI testing, specifically focusing on performance improvements. The specific objectives include:

1) Implementation: Designing Bit Swap LFSR and AW-based RTPG architectures using Verilog HDL.
2) Analysis: Comparing both designs in terms of randomness quality and hardware efficiency.
3) Verification: Simulating the architectures and verifying the generated patterns through waveform analysis.
4) Assessment: Determining the suitability of these designs for Built-In Self-Test (BIST) and VLSI applications.

III. PROBLEM STATEMENT

As VLSI circuits grow in complexity, achieving high fault coverage within a limited test time is a significant challenge in digital testing. Conventional generators, such as basic LFSRs, are hardware-efficient but suffer from several drawbacks:

- Randomness Issues: They often exhibit poor randomness distribution and repetitive patterns.
- Detection Gaps: They can be ineffective at detecting certain complex fault types, reducing overall test efficiency.
- Control Limitations: There is often a lack of sufficient controllability over the generated patterns, making them less ideal for advanced BIST applications.

This project addresses these issues by implementing the Bit Swap LFSR and Arithmetic Weight (AW)-based RTPG to produce more diverse and effective test patterns without significantly increasing hardware complexity.

IV. PROPOSED ARCHITECTURE AND METHODOLOGY

This section details the design and internal logic of the two implemented architectures: the Bit Swap LFSR (LFSR_BS) and the Arithmetic Weight-based RTPG (AW_RTPG). Both designs were implemented in Verilog HDL and optimized for synthesis using the SG13G2 PDK.

A. BIT SWAP LFSR

The LFSR_BS architecture is designed to reduce the number of transitions in the generated patterns, thereby improving hardware efficiency while maintaining moderate randomness

- Internal LFSR: The core is an 8-bit shift register with a specific feedback polynomial.
- Feedback Logic: The feedback bit is generated by XORing specific bits of the current state.
- Bit-Swapping Mechanism: A multiplexer-based logic is used where the MSB acts as a selection line.
- Conditional Swap: When the selection bit is 0, adjacent bit pairs are swapped before being stored in the next clock cycle.

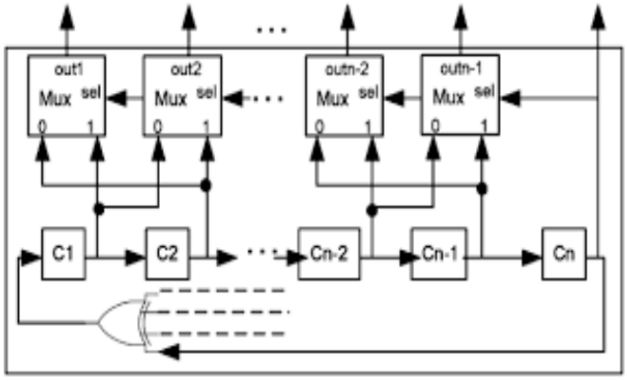

FIGURE 1. ARCHITECTURE OF BIT_SWAP_LFSR.

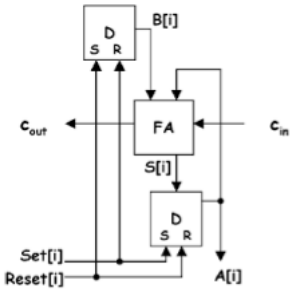

FIGURE 2. ARCHITECTURE OF AW_RTPG

B. ARITHMETIC WEIGHT-BASED RTPG (AW_RTPG)

The AW_RTPG provides superior randomness quality by using an arithmetic approach to generate weighted test patterns. This architecture is more complex and consists of several interconnected modules.

- Modular Design: It utilizes a hierarchy consisting of a 1-bit Full Adder, D-Flip-Flops with asynchronous reset, and a BS-LFSR as the primary data source.
- Bit-Level Logic: For each bit position, the circuit includes Weight and Drive registers along with a Full Adder.
- Carry Chain: An 8-bit ripple carry chain propagates through each bit to ensure proper arithmetic pattern generation.

C. SIMULATION AND SYNTHESIS ENVIRONMENT

- Toolchain: The designs were developed and simulated using the open-source eSim EDA platform.
- Target Library: Synthesis was performed using the SG13G2 Process Design Kit (PDK) to extract area and cell count statistics.
- Verification: Correctness was verified through waveform analysis, ensuring continuous, clock-driven pattern generation under specific reset and enable conditions.

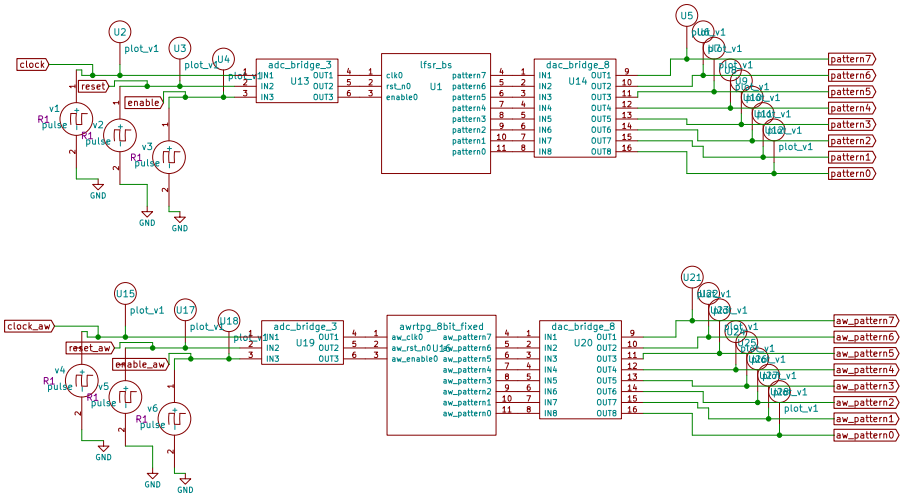

FIGURE 3. SCHEMATIC OF BS_LFSR AND AW_RTPG

V. SIMULATION RESULTS AND WAVEFORM ANALYSIS

This section presents the results obtained from simulating the proposed architectures using the eSim open-source EDA tool. The primary objective is to verify that both the LFSR_BS and AW_RTPG designs generate continuous, clock-driven random test patterns under the specified control conditions.

A. LFSR_BS SIMULATION RESULTS

The Bit-Swapping LFSR was simulated with an active-high enable and an active-low reset. The output waveform confirms the generation of pseudo-random patterns with reduced transitions due to the bit-swapping mechanism.

- Continuous Operation: The generated patterns are clock-driven, meaning a new pattern is produced at every rising edge of the clock when the enable signal is high.
- Transition Analysis: As observed in the waveform, the LFSR_BS provides a moderate level of randomness suitable for area-constrained environments.

B. AW_RTPG SIMULATION RESULTS

The Arithmetic Weight-based RTPG was simulated to assess its randomness quality and pattern variation.

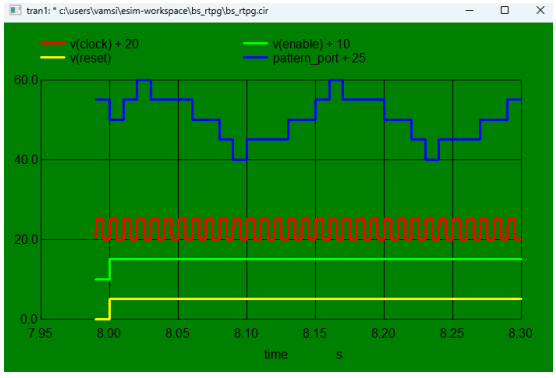

FIGURE 4. Simulation waveforms of bs_lfsr

1) Randomness Quality: Compared to the LFSR_BS, the AW_RTPG exhibits a higher degree of randomness and better pattern variation, as seen in the output waveform.
2) Fault Detection Capability: The increased complexity of the generated patterns makes this architecture more effective for detecting faults in complex digital circuits.
3) Verification: The simulation confirms that the combination of the internal LFSR, full adders, and carry chains functions correctly to produce the expected

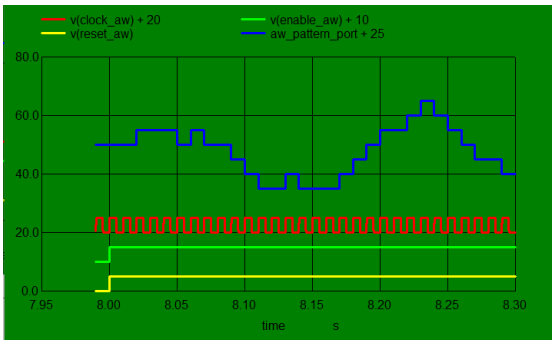

FIGURE 5. Simulation waveforms AW_RTPG

```
7. Printing statistics.

=== lfsr_bs ===

   Number of wires:                 39
   Number of wire bits:             67
   Number of public wires:           8
   Number of public wire bits:      29
   Number of memories:               0
   Number of memory bits:            0
   Number of processes:              0
   Number of cells:                 32
     sg13g2_a21o_1                   1
     sg13g2_dfrbp_1                  8
     sg13g2_mux2_1                   2
     sg13g2_nand2_1                  2
     sg13g2_nand2b_1                 1
     sg13g2_nor2_1                   3
     sg13g2_nor2b_1                  3
     sg13g2_o21ai_1                  9
     sg13g2_or2_1                    1
     sg13g2_xnor2_1                  1
     sg13g2_xor2_1                   1

   Chip area for module '\lfsr_bs': 618.710400
```

FIGURE 6. Cell Count Of LFSR_BS Design

weighted patterns.

C. COMPARATIVE SUMMARY

From the waveform analysis, it is clear that a trade-off exists between the two designs:

- AW_RTPG is superior in terms of randomness but requires more hardware.
- LFSR_BS is highly area-efficient with acceptable randomness for simpler applications.

VI. SYNTHESIS RESULTS AND HARDWARE EFFICIENCY

In this section, the hardware performance of both the LFSR_BS and AW_RTPG architectures is evaluated through logic synthesis. The designs were synthesized using the SG13G2 Process Design Kit (PDK) to extract precise data regarding cell usage and silicon area.

A. AREA AND CELL COUNT ANALYSIS

The synthesis results reveal a significant difference in the hardware footprint of the two architectures:

- AW_RTPG Performance: This architecture consumes 108 logic cells. The total estimated chip area for the awrtpg_8bit_fixed module is 1950.48 μm^2.
- LFSR_BS Performance: In contrast, the lfsr_bs module is much more compact, utilizing only 32 logic cells. It occupies a chip area of 618.71 μm^2.

```
Number of wires:                555
Number of wire bits:            754
Number of public wires:         147
Number of public wire bits:     262
Number of memories:               0
Number of memory bits:            0
Number of processes:              0
Number of cells:                108
  sg13g2_a21oi_1                 12
  sg13g2_dfrbp_1                 24
  sg13g2_inv_1                    8
  sg13g2_mux2_1                   3
  sg13g2_nand2b_1                 9
  sg13g2_nor2_1                  16
  sg13g2_nor2b_1                 12
  sg13g2_o21ai_1                  8
  sg13g2_xnor2_1                 16

Chip area for top module '\awrtpg_8bit_fixed': 1950.480000
```

FIGURE 7. Cell Count Of AW_RTPG Design

```
==========lfsr_bs : New Iteration==========
Instance : 0

Inside foo before eval.....
clk=0
rst_n=1
enable=1
pattern=42

Inside foo after eval.....
clk=1
rst_n=1
enable=1
pattern=106
==========awrtpg_8bit_fixed : New Iteration==========
Instance : 0

Inside foo before eval.....
aw_clk=0
aw_rst_n=1
aw_enable=1
aw_pattern=255

Inside foo after eval.....
aw_clk=1
aw_rst_n=1
aw_enable=1
aw_pattern=254
```

FIGURE 8. Random Test patterns of AW_RTPG and LFSR_BS

B. COMPONENT BREAKDOWN

The synthesis tool utilizes specific standard cells from the SG13G2 library to implement the Verilog logic:

- Storage Elements: Both designs use sg13g2_dfrbp_1 (D-Flip-Flops) for state retention.
- Arithmetic and Logic: The AW_RTPG heavily relies on sg13g2_xnor2_1, sg13g2_nor2_1, and sg13g2_a21oi_1 to implement the 8-bit ripple carry chain and weighting logic.
- Switching Logic: Both modules utilize sg13g2_mux2_1 for the bit-swapping mechanisms.

C. PERFORMANCE TRADE-OFF SUMMARY

The data confirms a clear engineering trade-off between randomness quality and hardware cost.

TABLE 1. Comparison of AW_RTPG and LFSR_BS Architectures

Design Type	Randomness	Area Consumption (μm^2)	Cell Count
AW_RTPG	More	1950.48	108
LFSR_BS	Moderate	618.71	32

VII. CONCLUSION

The project successfully demonstrated the design, implementation, and performance evaluation of two distinct random test pattern generators: the Bit Swap LFSR (LFSR_BS) and the Arithmetic Weight-based RTPG (AW_RTPG). Simulation results obtained through the eSim tool verified that both architectures function correctly under clock-driven conditions.

The comparative analysis revealed a definitive trade-off between randomness quality and hardware cost:

- AW_RTPG is highly effective for complex fault de- tection as it produces superior pattern variation and higher randomness. However, this comes at the cost of increased area, occupying approximately 1┗50 μm^2 and utilizing 10↤logic cells.
- LFSR_BS serves as a hardware-effcient alternative, occupying only 61↤μm^2 with 32 logic cells. While it offers moderate randomness, it is the superior choice for area-constrained VLSI testing environments.

VIII. REFERENCES

[1] A. Bagalkoti, S. B. Shirol, R. S, P. Kumar and R. B. S, →Design and Implementation of ↤bit LFSR, Bit-Swapping LFSR and Weighted Random Test Pattern Generator: A Performance Improvement,→ 201↦International Conference on Intelligent Sustainable Systems (ICISS), Palladam, India, 201↳; pp. ↤2-↤6, doi: 10.110↳/ISS1.201↳.↤↦0↤063.

[2] FOSSEE Team, →eSim: Open Source EDA Tool,→ IIT Bombay. [Online]. Available: https://esim.fossee.in.

Note:- Please verify my project in eSim verision 2.3 tool