# Benchmarking Test Generation Algorithms for Functional Testing

Wai Cheong Tsoi
Department of Electrical and Computer Engineering
University of California, Davis
Davis, California, United States
wctsoi@ucdavis.edu

Mina Neronde
Department of Electrical and Computer Engineering
University of California, Davis
Davis, California, United States
mneronde@ucdavis.edu

Yuhua Pan
Department of Electrical and Computer Engineering
University of California, Davis
Davis, California, United States
yhpan@ucdavis.edu

***Abstract*—Over the years, there has been much research focusing on the processes and procedures surrounding testing circuits of modern complexity and the ways they may be improved. The research we are conducting is of the same mindset, specifically concentrating on the subset of functional testing. Functional testing includes various types of testing methods, such as Exhaustive, Pseudoexhaustive, and Pseudorandom. Some of the testing methods listed above are applicable on specific modules that could be sequential or combinational. Modern microprocessors are a mixture of the two module types, and the complexity of them has become enormous. Due to this reason, the speed of generating test sets, employing them, and detecting the faulty functions had bogged down. Therefore, researchers focused on the improvement in speed, such as, by dividing the functions based on instructions [4] or partitioning a microprocessor, which sections it to several testable areas. In this paper, we discussed the fault coverage of the test sets generated by the three testing algorithms mentioned. As fault coverage also corresponds to the speed of fault detection, we were able to observe the differences and similarities between the distinct methods (Exhaustive, Pseudoexhaustive [1], and Pseudorandom) via the results depending on the design of the modules that we designed.**

***Keywords - Functional Testing, Test Generation, Exhaustive Testing, Pseudoexhaustive Testing, Random Testing, Linear Feedback Shift Register***

## I. Introduction

As the complexity of modules increases exponentially following Moore's Law, structural fault targeting has become near impossible and functional testing has been a necessity in modern day scale. Without targeting or invoking any structural faults, functional testing examines the functionality of a module by comparing the output of the module to a golden reference to discover mismatches, and thus deduce the underlying faults from the mismatches detected. While modern functional testing uses more sophisticated test generation algorithms combined with heuristics analysis to reduce the test set further, it is best to appreciate the speed-up of the testing process by comparing different test generation algorithms. By establishing the premises of the benchmarking process, a better understanding of the different algorithms can be observed.

To compare the speed-up of different test generation algorithms against the Exhaustive method, in the scope of functional testing, two other types of test generation algorithms are introduced in our project: Pseudoexhaustive and Pseudorandom. Pseudoexhaustive is the category of eliminating test vectors based on the information provided in the subject of the modules, knowing that some test vectors are redundant in the sense that some inputs do not affect the output that was currently in test. Random is another popular category of test vector generation for providing fair coverage to combinational circuits, however, it is uncertain on sequential circuits depending on the design. Pseudorandom using Linear Feedback Shift Register (LFSR) is used in our project to establish a comparison to the Exhaustive method. In this setting, our project compares the fault coverage of the algorithms within a given number of cycles and explores the trend of the coverage of the algorithms against the number of test vectors used.

## II. Problem Definition

### A. Coverage of Test Generation Algorithm

Generally speaking, fault coverage denotes that we set faults in particular, such as stuck at a line, and compute the number of detected faults out of the total number of faults. In this paper, we performed functional testing, which verifies the function itself. The purpose of this experiment is to compare the speed-up in malfunctional detection in the three algorithms that can be seen with the fault coverage. Therefore, fault coverage is defined as the detected number of mismatches compared to our reference circuit divided by the total number of mismatches compared to our reference circuit. Instead of discovering a specific fault in the circuit, which is not the scope of our problem, our team focuses on the performance of the algorithm in the scope of how many mismatches it can detect. Transition coverage is also used for sequential circuits to consider the number of transitions the test sequence can cover. Both are valid and fair metrics to compare the performance of the test generation algorithms.

### B. Algorithm Used

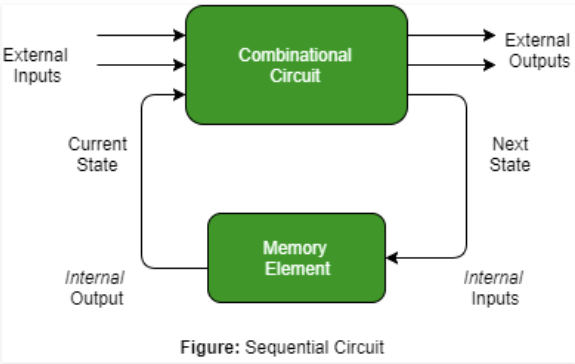
We have implemented three classic functional testing algorithms. They are Exhaustive Testing, Pseudoexhaustive Testing, and Random Testing. Exhaustive Testing is a test approach in which all the possible test combinations are used for testing, basically we need to apply all 2n possible input vectors. Pseudoexhaustive Testing refers to a shortcut to achieve Exhaustive Testing. It can test almost all faults defined by the universal fault model with significantly less than 2nvectors. In this part, we implemented various modules that utilized McCluskey's Method and one that is C-testable.

McCluskey's Method on Pseudoexhaustive test generation proposed that the test vectors become small progressively as the paper [2] applies partitioning and merging steps. According to the paper [2], if certain inputs do not affect an output, vector combinations regarding those inputs should be removed from the test set, thus reducing the number of test vectors. McCluskey's Method based on partitioning are perfectly suited for circuit structure as iterative logic arrays(ILAs). Some ILAs have the useful property that they can be pseudoexhaustively tested with a number of tests that do not depend on the number of cells in the ILA. In this case, we just need to test the whole circuit with constant test vectors.

For Random Testing, we used the LFSR algorithm to generate a specific number of random test vectors and then used these vectors to analyze our modules.

### C. Combinational Module and Sequential Module

We implemented the algorithms in both combinational module and sequential module. Combinational circuits are defined as the time dependent circuits which do not depend upon previous inputs to generate any output. Sequential circuits are those which are dependent on clock cycles and depend on present as well as past inputs to generate any output. The relationship can be shown as below:




*<Figure 1. the structure of a generic sequential circuit>*

## III. Related Work

### A. Test Pattern Generation using LFSR

VinodChandra and Ramasamy's work focused on comparing fault coverages of different test generation algorithms on discovering structural faults in sequential circuits [1]. They compared the D Algorithm, Path-Oriented Decision Making Algorithm (PODEM), Fan-Out Oriented Algorithm (FAN) and Pseudorandom test generation using LFSR. They compared the area, power consumption and fault coverage of the circuits using the stuck at fault model. They used the ISCAS 89 Benchmark Circuits for their tests and thus determined the comparison data for small and medium sequential circuits using the stuck at fault model, LFSR algorithm yielded maximum coverage; for larger circuits, Tetramax ATPG that they used outperformed the LFSR.

While our project does not consider structural faults, and thus could not use most of the algorithms used in their comparison, they proposed the LFSR as a good candidate for small and medium size sequential circuits [1]. Our team would like to replicate LFSR and investigate the performance of it in both combinational and sequential and find whether their suggestion is generally true.

### B. Test Pattern Generation using LFSR

McCluskey's work on Pseudoexhaustive test generation proposed how the size of the test set can be reduced exponentially when testing against a combinational circuit [2]. Four years later in 1988, he collaborated with Wang and further investigated exhaustive and Pseudoexhaustive techniques on combinational circuits using modified LFSRs [3]. By partitioning the outputs, its dependencies on the inputs can be utilized to reduce redundant test vectors. By modifying the LFSRs, test vectors can test some outputs concurrently, further reducing the size of the test set.

Since most Pseudoexhaustive test generation algorithms are built from the idea of reducing test vectors while diminishing its effect on fault coverage, our team would like to investigate the simplest form of this category and consider its behavior in generating test vectors for sequential circuits, the type of circuit absent in McCluskey's paper.

## IV. Goals

Our goal is to compare the performance of the three test generation algorithms by measuring its fault coverage as defined against the number of tests applied, using both combinational and sequential modules designed according to our specifications. Our team would like to investigate the potential speed-up of the algorithms, and how the type of circuit affects the fault coverage of test generation algorithms. Our team would also want to compare our results to previous works to verify VinodChadra and Ramasamy's and McCluskey's claims, on whether LFSR is a good algorithm to be used for small sequential circuits as claimed by VinodChandra and Ramasamy [1], or Pseudoexhaustive test generation provides similar speed-up comparable to that in combinational circuits.

## V. Methods

### A. Module Specification

To accomplish our goal, we decided to create three combinational and three sequential modules to test all those algorithms. The reason why we separate the modules to combinational and sequential is to investigate the difference of the testing generation algorithm on different types of modules. Additionally, we set the following restrictions for the fair testing environment: Input needs to be 12-13 bits, which has the range from 0 to 8191. The limitation of the number of test vectors is 5000, which is slightly above a half of the input range.

### B. Testing

We set a fault stuck at a line to produce malfunctions from the modules. By running the Exhaustive algorithm on the faulty-free module and faulty module, we obtained the total

number of the wrong outputs in 8191 cases. The maximum number of test sets, limited to 5000, is generated via the algorithms. Exhaustive algorithm generated 5000 test cases in sequential, Pseudorandom algorithm produced the test sets with the random number generator that is built in MATLAB. In case of Pseudorandom, LFSR was chosen, which excludes the zero case since zero always stays at zero. Pseudoexhaustive algorithm yielded the final test sets by reducing redundant test cases [2]. These test sets were applied to the modules. This resulted in the number of faults detected by the test sets, and we were able to calculate the fault coverage and speed-up.

## VI. Results and Discussion

By applying the three test generation algorithms on the three combinational modules and three sequential modules in accordance with our specifications, our team has reported the fault coverage of the circuits shown below.

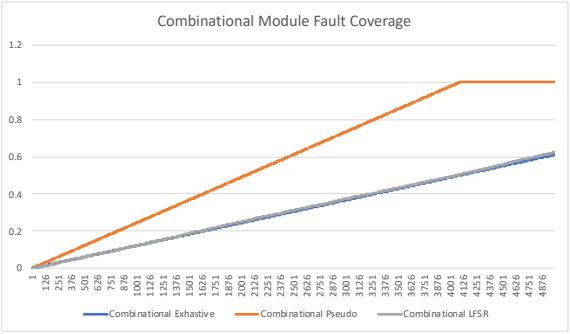
| Combinational | | | |
|---|---|---|---|
| Fault Coverage \ Module | 1 | 2 | 3 |
| Exhaustive | 62% | 60% | 61% |
| Pseudoexhaustive | 100% | 100% | 99.90% |
| Pseudorandom | 62% | 58% | 61% |

*<Table 1. Fault Coverages on Different Combinational Modules after 5000 Test sets>*

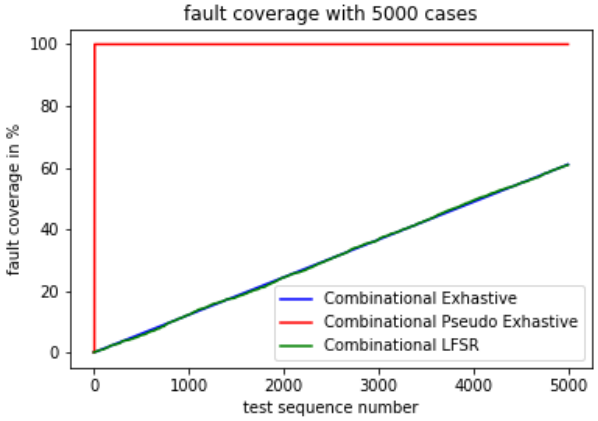
| Sequential | | | |
|---|---|---|---|
| Fault Coverage \ Module | 1 | 2 | 3 |
| Exhaustive | 64% | 60% | 40% |
| Pseudoexhaustive | 100% | 100% | 61% |
| Pseudorandom | 66% | 55% | 3.60% |

*<Table 2. Fault Coverages on Different Sequential Modules after 5000 Test sets>*

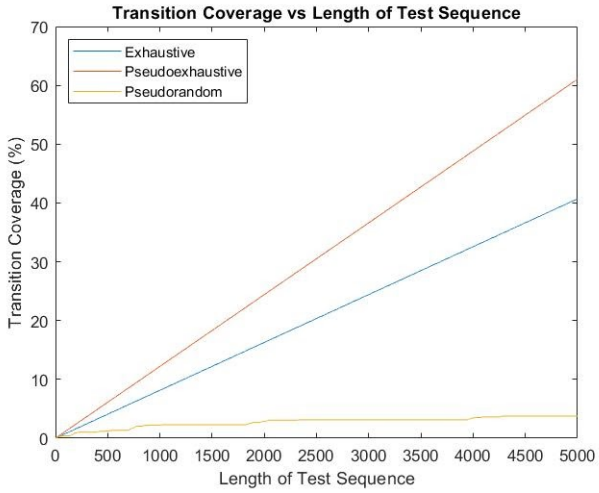
For both types of circuits, it is discovered that the fault coverage of all three methods increase in a linear trend because of our definition of fault coverage. Pseudoexhaustive algorithm provides a speed-up of at least 50% and is generally applicable to most circuits as shown in Figure 2. However, Pseudoexhaustive algorithm requires the knowledge of the behavior of the circuit and optimally, the gate-level design of the circuit, which is difficult to obtain in modern scale circuits. A C-testable circuit, where both the behavior and the structure is known, can achieve 100% fault coverage in a constant number of tests as shown in Figure 3. Exhaustive algorithm provides the baseline of the fault coverage, where every combinatorially possible vector is in the test set. LFSR provides generally similar fault coverage compared to the exhaustive method with the exception of our third sequential circuit, shown in Figure 4. As a one-bit input counter, the test sequence generated by the LFSR would traverse the states of sequential circuit 3 in a wasteful manner, serving as a counterexample for VinodChandra and Ramasamy's statement.




<Figure 2. Fault Coverage on Combinational Module 1>




<Figure 3. Fault Coverage on Combinational Module 2>




<Figure 4. Transition Coverage on Sequential Module 3>

While Pseudorandom test generation algorithms such as LFSR can usually provide a good amount of coverage for simpler circuits of unknown behavior, it is shown that there is at least one case where it failed to provide a comparable transition coverage compared to the exhaustive method. It is also observed that the more information regarding the module we know, the size of test vectors, and hence testing time, can be reduced exponentially.

## VII. Conclusion

Based on the research and analysis we have done, we obtained four conclusions. First, Pseudoexhaustive Testing is overall the best, especially suited for circuits having a regular ILA-type structure. Second, McCluskey's Method can bring us some

potential speed-up. In our simulations, the speed-up ranged from 49.8% to 64.8%. Thirdly, the performance of Pseudorandom testing depends on the design of the module. We were able to observe a similar performance of random testing with exhaustive testing in Module 1, 2. Meanwhile, the random testing in Module 3 contributed little in detecting faults. Finally, as far as we implemented, the fault coverage in functional testing increased linearly, even if they varied in different circuits with different algorithms applied.

## VIII. Future Work

Since we randomly designed three different types of modules to apply the algorithms, the next step should be to find the relationship between a testing algorithm and a circuit type and discover functional testing algorithms that give us better results than Pseudoexhaustive. We have found another functional testing algorithm called Compact Binary Differential Evolution Algorithm (Compact-BinDE), which obtains test vector sequences with an efficient search in binary domain using meta-heuristics method. We did not consider this algorithm in this project because of its complexity. Therefore, Compact-BinDE could be a one of the potential algorithms to analyze, and at the same time we will find the comparison of the results between the algorithms we implemented and other algorithms we will consider in the future.

Other types of algorithms should be added into our benchmark, such as Heuristic Test Generation Algorithms, Meta-Heuristic Algorithms, Genetic Algorithms and more. There are many more categories of test generation algorithms to be compared with more circuits, including the ISCAS 89 Benchmark Circuits, or more complex circuits like microprocessors or even SoCs. It would be a great opportunity to test our hypothesis of the correlation between testing efficiency versus the behavioral and structural information of the device under test.

## References

[1] C. VinodChandra and S. Ramasamy, "Test pattern generation for benchmark circuits using LFSR," 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), Tiruchengode, 2013, pp. 1-6, doi: 10.1109/ICCCNT.2013.6726500.

[2] McCluskey, "Verification Testing—A Pseudoexhaustive Test Technique," in IEEE Transactions on Computers, vol. C-33, no. 6, pp. 541-546, June 1984, doi: 10.1109/TC.1984.1676477.

[3] L. -. Wang and E. J. McCluskey, "Circuits for pseudoexhaustive test pattern generation," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 7, no. 10, pp. 1068-1080, Oct. 1988, doi: 10.1109/43.7806.

[4] Dhananjay Brahme and Jacob A. Abraham, “Functional Testing of Microprocessors”, in IEEE transactions on Computers, vol. C-33, no. 6, June. 1984

Wai Cheong Tsoi is a fourth-year undergraduate student, currently pursuing a B.S. degree in Electrical Engineering at the University of California, Davis.

Yuhua Pan is a second-year graduate student, currently pursuing a M.S. degree in Electrical and Computer Engineering at the University of California, Davis.

Mina Neronde is a first-year graduate student, currently pursuing a M.S. degree in Electrical and Computer Engineering at the University of California, Davis.