Automatic Test Pattern Generation

Shashidhara H. R, Ashish S, Pooja P, Suhas S

***Abstract*—Testing is an important part of any digital design process. It is important to determine the faults and to diagnose the fault in a**

that one fault, is different than the expected output. The generated patterns are used to test semiconductor devices after manufacture, and

**product or circuit. A test vector is a set of inputs provided to a system** in some cases to assist with determining the cause of failure (failure

**in order to test that system. In development test vectors are a methodology of testing, verification and validation. Test vector generation is a program used to automatically generate test data for use in automated testing circuits. This can generate many individual test vectors. Manual testing can be done for smaller designs, in which inputs can be given to the system by forcing the values and observing the output. As the complexity of the design increases the testing becomes tedious and hence automation is required. In VLSI testing we need Automated Test Pattern Generator (ATPG) to get input test vectors for the Device Under Test (DUT). Generated test sets are usually compacted to save test time which is not good for failure diagnosis.**

analysis) the effectiveness of ATPG is measured by the number of modeled defects, or fault models, that are detected and the number of generated patterns. It is influenced by the fault model under consideration, the type of circuit under test, the level of abstraction used to represent the circuit under test and the required test quality. Fault models abstract the behavior of manufacturing defects so that test vectors can be generated to detect them.

* Functional Defects: Stuck-at Fault Model.
* Current defects: Pseudo Stuck-at Fault Model (IDDQ).
* Speed defects: At-speed Fault Model, Path Delay Fault Model.

**Automated Test pattern Generation, or ATPG, is a process used** Stuck-at Faults is the most common fault model used in industry.

**in electrical testing wherein the vectors or input patterns** The single stuck-at-fault model has been widely accepted as a

**required to check a device for faults are automatically generated** standard target modelto generate a set of test patterns to detect all the

## by a program to compare between the correct and faulty circuit

stuck faults in the circuit. A single stuck-at fault represents a l ; line

**behavior caused by defects. These generated patterns are used** in the circuit that is fixed to logic value 0 or 1. The single-stuck fault **to test integrate circuits after manufacture, and to assist for** model is also referred to as the classical or standard fault model **determining the reason for failure. ATPG can generate** because it has been the first and the most widely studied and used.

## patterns. It can also find redundant circuit logic and can prove one implementation matches another. We are designing ATPG using frequency determination method.

1. INTRODUCTION

The ATPG process for a targeted fault consists of two phases

1. Fault Activation
2. Fault Propagation

Fault activation establishes a signal value at the fault model site that is opposite of the value produced by the fault model. Fault

Testing is the practice of making objective judgements regarding propagation moves the resulting signal value, or fault effect, the extent to which the system (the device) meets the required forward by sensitizing a path from the fault site to a primary objectives. To test any system a set of test vectors are generated output.It models manufacturing defects which occurs when a circuit and the system is a computable function 𝑦 = 𝑓(𝑥),Where y is the node is shorted to VDD (stuck-at-1 fault) or GND (stuck-at-0 fault) output and x is the input to the system. The inputs could be permanently. The fault can be at the input or output of a gate. Thus, multidimensional in the form of 𝑦 = 𝑓(𝑥1, 𝑥2).The generalized a simple 2-input AND gate has six possible stuck-at faults. In the

equation is of the form 𝑌 = 𝐹(𝑥). circuit shown in Figure 1, suppose we have a stuck-at-0 fault at the

Manual testing can be done for smaller designs, in which inputs can output of an AND gate. Note one important thing, there are three be given to the system by forcing the values and observing the input ports in the circuit, thus we can have a combination of eight output. As the complexity of the design increases the testing different inputs or patterns {000, 001, 010, 011, 100, 101, 110, becomes tedious and hence automation is required. Test vector size 111}; out of the eight patterns, only one pattern {011} will be able is the big issue in the today’s technology. As size of circuit increases to detect this fault, as with rest of the patterns the expected output the size of test vector also increases. The effectiveness and efficiency will be same as the actual circuit output in the presence of this s- a- is increased by automated testing. An automated testing tool compare 0 fault. This is a small circuit so we can easily find the pattern that the expected behavioral results of the circuit and reports the success can detect this fault, but what about much bigger circuits? Well we or failure, also playback the predefined and pre-recorded actions. The don’t have to worry about it as the CAD tools (ATPG tools) will do automated testing is becoming an essential component because of that for us. The ATPG tools will try to generate the stuck-at fault their repetition and extensions for the tasks which are impossible in patterns required to test all the possible fault locations using manual testing. ATPG is an automation method, which is used to find complex algorithms, but if it is unable to find patterns for few faults, an input (or test) sequence applied to a digital circuit and enables the then it will classify those faults as untestable. This is the most test equipment to compare between the correct and faulty circuit common fault model used in industry. It models manufacturing

behavior caused by defects.

A defect is an error caused in a device during the manufacturing process. A fault model is a mathematical description of how a defect alters design behavior. The logic values observed at the device's primary outputs, while applying a test pattern to some device under test (DUT), are called the output of that test pattern. The output of a test pattern, when testing a fault-free device that works exactly as designed, is called the expected output of that test pattern. A fault is said to be detected by a test pattern if the output of that test pattern, when testing a device that has only

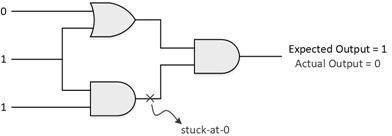
defects which occurs when a circuit node is shorted to VDD (stuck-

at-1 fault) or GND (stuck-at-0 fault) permanently. The fault can be at the input or output of a gate. Thus a simple 2-input AND gate has six possible stuck-at faults. In the circuit shown in Figure 1, suppose we have a stuck-at-0 fault at the output of an AND gate. Note one important thing, there are three input ports in the circuit, thus we can have a combination of eight different inputs or patterns {000, 001, 010, 011, 100, 101, 110, 111}; out of the eight patterns, only one pattern {011} will be able to detect this fault, as with rest of the patterns the expected output will be same as the actual circuit output

in the presence of this s- a-0 fault. This is a small circuit so we can fault activation and fault propagation. Fault activation

easily find the pattern that can detect this fault, but what about much establishes a signal value at the fault model site that is opposite

bigger circuits? Well we don’t have to worry about it as the CAD of the value produced by the fault model. Fault propagation tools (ATPG tools) will do that for us. The ATPG tools will try to moves the resulting signal value, or fault effect, forward by generate the stuck-at fault patterns required to test all the possible sensitizing a path from the fault site to a primary output

fault locations using complex algorithms, but if it is unable to find Designer generated functional test patterns provides 70- 75 patterns for few faults, then it will classify those faults as untestable. percent of coverage. The ATPG supplements to get test

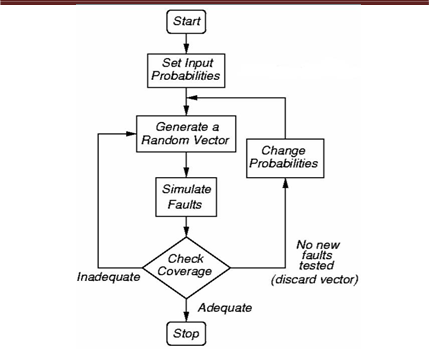
coverage less than 98 percent.

Figure 1: Stuck at 0 model

Once the patterns are generated, the expected response of the circuit for each pattern is obtained in pre-silicon. The expected responses along with the patterns are then stored in the memory of Automatic Test Equipment (ATE). In post-silicon, the manufactured chip is tested using the ATE, which loads the pattern and compares it with the expected response for pass or fail status.

ATPG (Automatic Test Pattern Generation and Automatic Test ATPG (Automatic Test Pattern Generation and Automatic Test Pattern Generator) is an EDA (Electronic design automation) method/technology used to find an input or test sequence. When applied to a digital circuit, ATPG enables automatic test equipment to distinguish between the correct circuit behavior and the faulty circuit behavior caused by defects. The generated patterns are used to test semiconductor devices after manufacture, or to assist with determining the cause of failure (failure analysis).

The effectiveness of ATPG is measured by the number of modelled defects, or fault models, detectable and by the number of generated patterns. These metrics generally indicate test quality (higher with more fault detections) and test application time (higher with more patterns).

ATPG efficiency is another important consideration that is influenced by the fault model under consideration, the type of circuit under test (full scan, synchronous sequential, or asynchronous sequential), the level of abstraction used to represent the circuit under test (gate, register-transfer, switch), and the required test quality. Pattern Generator) is an EDA (Electronic design automation) method/technology used to find an input or test sequence. When applied to a digital circuit, ATPG enables automatic test equipment to distinguish between the correct circuit behavior and the faulty circuit behavior caused by defects. The generated patterns are used to test semiconductor devices after manufacture, or to assist with determining the cause of failure (failure analysis). The effectiveness of ATPG is measured by the number of modelled defects, or fault models, detectable and by the number of generated patterns. These metrics generally indicate test quality (higher with more fault detections) and test application time (higher with more patterns).

ATPG efficiency is another important consideration that is influenced by the fault model under consideration, the type of circuit under test (full scan, synchronous sequential, or asynchronous sequential), the level of abstraction used to represent the circuit under test (gate, register-transfer, switch), and the required test quality.

The ATPG process for a targeted fault consists of two phases:

Figure 2: Schematic showing how testing works

Literature survey enables adaptation of the knowledge required for any project from the problem deﬁnition to ﬁnding a solution and forb its execution. In the following section we summarize the literature survey carried out for this paper.

*Tracy Larrabee, A Framework for Evaluating Test Pattern Generation Strategies, March 1990(1)*, this paper presents a formal approach for the analysis of heuristics used in automatic test pattern generation for combinational circuits. We start with a test pattern generation system that constructs a satisfying assignment for a Boolean formula describing the legal set of tests. We then describe heuristics as modifications to the formula or to the satisfier acting on the formula. It provides experimental results for the system as a whole, and for the effects of four heuristics.

*Tracy Larrabee, Efficient Generation of Test Patterns Using Boolean Difference, March 1990(2),* Most automatic test pattern generation systems for combinational circuits generate a test for a given fault by directly searching a data structure representing the circuit to be tested. This paper describes a new system that divides the problem into two parts: First, it constructs a formula expressing the Boolean difference between the proper and faulted circuits. Second, it applies a Boolean satisfiability algorithm to the resulting formula. The new system can incorporate any of the heuristics used by structural search techniques. It is not only quite general, but is able to test or prove untestable every fault in the popular Brglez-Fujiwara benchmark system. Most automatic test pattern generation systems for combinational circuits generate a test for a given fault by directly searching a data structure representing the circuit to be tested. This paper describes a new system that divides the problem into two parts: First, it constructs a formula expressing the Boolean difference between the proper and faulted circuits. Second, it applies a Boolean satisfiability algorithm to the resulting formula. The new system can incorporate any of the heuristics used by structural search techniques. It is not only quite general, but is able to test or prove untestable every fault in the popular Brglez-Fujiwara benchmark system.

*Tracy Larrabee, “Efficient Generation of Test Pattern Using Boolean Difference (3*), A combinational circuit can be tested for the presence of a single stuck-at fault by applying a set of inputs that excite a variable output response in that circuit. If the fault is present, the output will be different than it would be if the fault were not present. Given a circuit, the goal of an automatic test pattern generation system is to generate a set of input sets that will detect every possible single stuck-at fault in the circuit. This approach differs from most programs now in use, which directly search the circuit data structure instead of constructing a formula from it. The new method is quite general and allows for the addition of any heuristic used by the structural search methods.The Boolean satis ability method has produced excellent results on popular test pattern generation benchmarks.

*Automatic Test Generation For Digital Circuits,S. Hemalatha(4*), Current VLSI manufacturing processes suffer from larger defective parts ratio, partly due to numerous emerging defect types. While traditional fault models, such as the stuck at and transition delay fault models are still widely used, they have been shown to be inadequate to handle these new defects.

The main aim is to develop a complete behavioral fault simulation and automatic test pattern generation (ATPG) system for digital circuits modeled in Verilog and VHDL. An integrated Automatic Test Generation (ATG) and Automatic Test Executing/Equipment (ATE) system for complex boards is developed here. An ATG technique called Behavior-Based Automatic Test Generation technique (namely BBATG) is developed. BBATG uses the device behaviour fault model and represents a circuit board as interconnection of devices. The other method used here is novel test pattern generator (TPG) for built-in self-test. A multiplexer is developed to generate a class of minimum transition sequences. The entire hardware is realized as digital logical circuits and the test results are simulated in Xilinx and Model sim software. Gate level simulation is not an effective solution for complex microcircuits. The low cost, versatile and reconfigurable FPGA-based ATE is implemented called FATE to support in ASIC development phase. The results of this research show that behavioral fault simulation will remain as a highly attractive alternative for the future generation of VLSI and system-on-chips (SoC). Design and Implementation of Test Vector Generation Using Random Forest Technique

*Naveen T V, Dr. M V Latte, Mr. Sathish Shet, Mr. Shashidhara H R (5),* Testing is an important part of any digital design process. This paper presents an approach to test DUT's in an automated manner using ATPG by Implementing the Random Forest Technique. In this we will use the Random Approach to conduct ATPG tests. This involves reading large no. of inputs, arranging the binary values in terms of n-bits and storing it in register.

These N - bits are then applied to ATPG and test vector is generated based on the algorithm, and the generated vector is stored

*Test Vector Generation Using Different Logic Regression Method Shashidhara H. R. (6),* Automatic Test Pattern Generation (ATPG) is used to determine test input sequence for digital circuit which distinguishes for the correct and faulty circuit behaviour. Different fault models of physical defects of the circuits are facilitates test generation. The

Stuck-at faults and fault transition models are extensively used because of their expediency. In this paper two methods of logic regression have been designed in order to obtain random test patterns. In static method, test patterns are generated using a characteristic polynomial, in bit separation method, test patterns are obtained by separating odd and even bits.

*Yu Zhang, Diagnostic Test Pattern Generation and Fault Simulation for Stuck-at and Transition Faults (7)*, In this work a Diagnostic Automatic Test Pattern Generation (DATPG) system is constructed by adding new algorithmic capabilities to conventional ATPG and fault simulation programs. The DATPG aims to generate tests to distinguish stuck-at fault pairs, i.e., two faults must have different output responses. This will help diagnosis to pin point the failure by narrowing down the fault candidates which may be the reason for this particular failure.

*J. A. A. Raghuram S. Tupuri, A novel functional test generation method for processors using commercial atpg.(8),*As the sizes of general and special purpose proctors increase rapidly, generating high quality manufacturing tests for them is becoming a serious problem in industry. This paper describes a novel method for hierarchical functional test generation for processors which targets one embedded module at a time and uses commm-cia1 ATPG tools to derive tests for faults within the module. Applying the technique to benchmark processor designs, we were able to obtain test efficiencies for the embedded modules of the processors which were extremely close to what the commercial ATPG could do with complete access to the module.

*John Sunwoo, Vishwani D. Agrawal, A New ATPG Algorithm for 21st Century: The Simplest but Powerful (9),* Very high-level algorithm and its decision tree is being used in this paper. The output of the ATPG program is the set of test vectors generated by the new algorithm. The significance of the algorithm is in the initializing PIs (Primary Inputs) technique. Unlike like other algorithms that start off with the unknown (pre-defined as X) PIs values or random (or pseudo-random) PIs values [1], the new algorithm assigns all PIs values to be non-controlling values according to the gate which the PI fan-in to this suggestion seemed to be an interesting solution for simple but powerful algorithm.

1. METHADOLOGY

A cycle of ATPG can generally be divided into two distinct phases:

1. Creation of the test
2. Application of the test

During the creation of the test, appropriate models for the device circuit are developed at gate or transistor level in such a way that the output responses of a faulty device for a given set of inputs will differ from those of a good device. The generation of test is basically a mathematical process that can be done in three ways:

1. Manual methods
2. Algorithmic methods
3. Pseudo-random methods.

Random ATPG basically involves three steps:

1. Generate random pattern.
2. Determine how many faults are detected by the random pattern (fault simulation).
3. Continue the above two steps till no (or few) new faults are detected by the random pattern.

To test all the faults defined by the universal fault model in a combinational circuit with n PIs, we need to apply all 2n possible input vector. The exponential growth of the required number of vectors limits the practical applicability of the testing method only to circuits with less than 20 PIs. Pseudo- random techniques for automatically generating simulation vectors are widely in use. Simulation tools may use random vector generators, and randomly generate 1s and 0s, collect a vector of inputs and apply them to the design and perform the simulation.

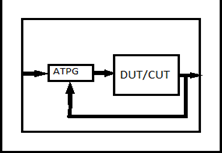


Figure 3: ATPG Module

## 5 BITS

Figure 5: 3-BIT DUT

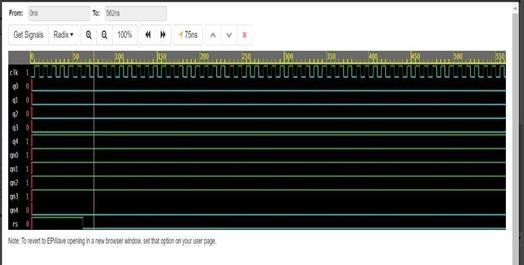
Figure 6: 5-BIT ATPG OUTPUT

Figure 7: 5-BIT DUT

## 3 BITS

1. SIMULATION

## 8 BITS

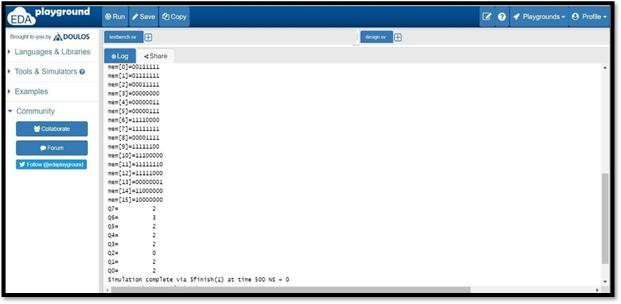
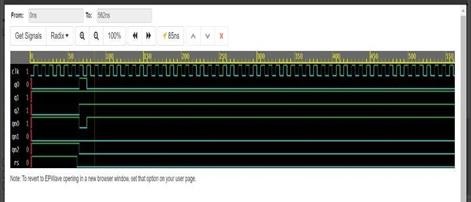
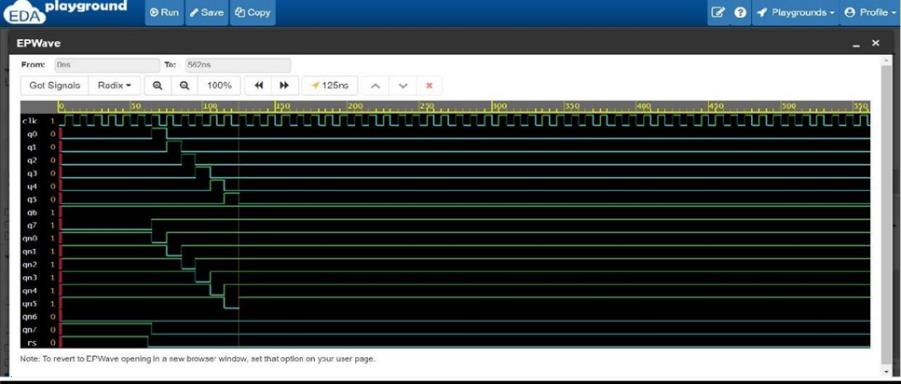


Figure 8: 8-BIT ATPG OUTPUT

Figure 4: 3-BIT ATPG OUTPUT



Figure 9: 8-BIT ATPG OUTPUT

## 16 BITS

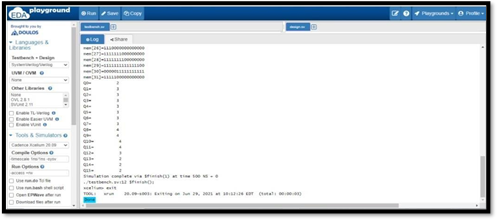
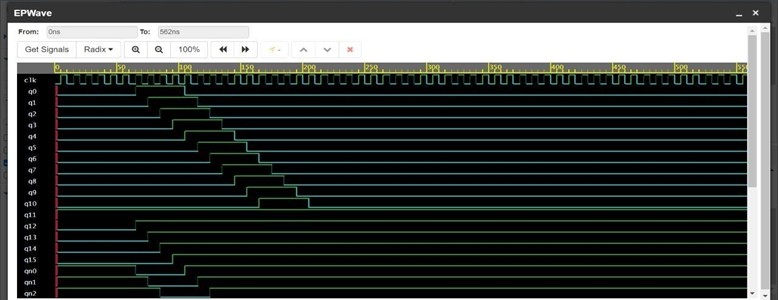
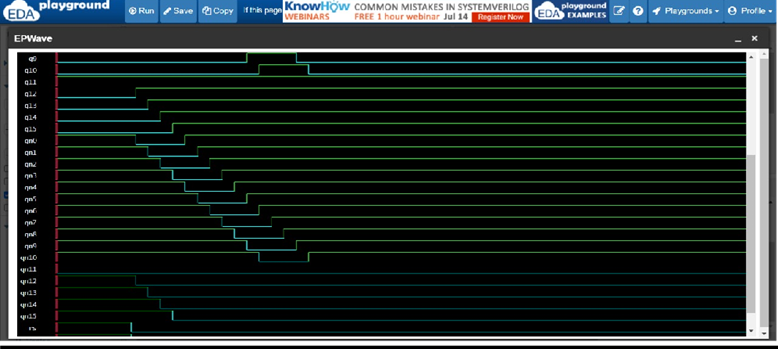


Figure 10: 16-BIT ATPG OUTPUT



Figure 11: 16-BIT DUT

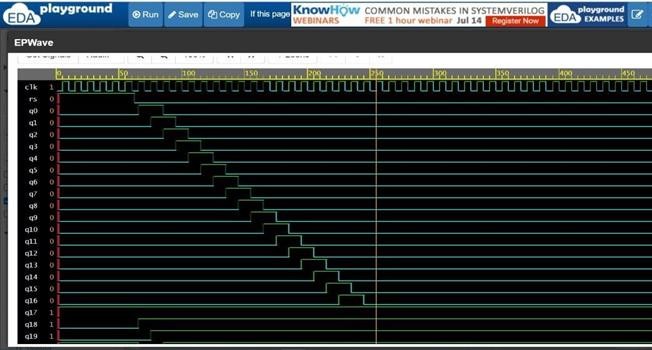


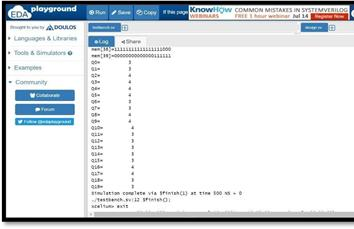
Figure 14: 20-BIT DUT



Figure 15: 20-BIT DUT

## 20 BITS

Figure 12: 16-BIT DUT

Figure 13: 20-BIT ATPG OUTPUT

1. CONCLUSION

Testing is an important part of any digital design process. Testing is used in semiconductor technology to distinguish between the correct and faulty behaviour. Testing and its application and has undergone tremendous development and growth. ATPG is one of the most popular techniques of testing VLSI design. In this paper, The proposed work shows higher efficiency in implementing fault detection through the highest frequency component. Successfully implemented and derived the ATPG algorithm to determine highest frequency bit to detect fault in circuit that is Johnson Counter. We have synthesized the same using EDA Playground. For future enhancements we can vary the group size of vectors in ATPG algorithm

REFERENCES

1. Tracy Larrabee, A Framework for Evaluating Test Pattern Generation Strategies, March 1990
2. Tracy Larrabee, Efficient Generation of Test Patterns Using Boolean Difference, March 1990.
3. Tracy Larrabee, “Efficient Generation of Test Pattern Using Boolean Difference
4. S. Hemalatha, “Automatic Test Generation For Digital Circuits”
5. Naveen T V, Dr. M V Latte, Mr. Sathish Shet, Mr. Shashidhara H R, “Design and Implementation of Test Vector Generation Using Random Forest Technique”
6. Shashidhara H. R**., “**Test Vector Generation Using

Different Logic Regression Method”

1. Yu Zhang,” Diagnostic Test Pattern Generation and Fault Simulation for Stuck-at and Transition Faults”
2. J. A. A. Raghuram S. Tupuri, “A novel functional test generation method for processors using commercial atpg.”
3. John Sunwoo, Vishwani D. Agrawal ,A New ATPG Algorithm for 21st Century: The Simplest But Powerful”

# 8