ISSN 2348-375X

Unique Journal of Engineering and Advanced Sciences

Available online: www.ujconline.net

Research Article

GENETIC ALGORITHM BASED TEST PATTERN GENERATION FOR ASYNCHRONOUS CIRCUITS WITH HANDSHAKE CONTROLLERS

Jay Kumar SR[1*], Arivazhagan P[2], Saranya K[3], Manikandaprabu N[4]

[1]PG scholar, Sasurie College of Engineering, Vijayamangalam, TN India
[2]PG scholar, Anna university regional centre, Coimbatore,TN India
[3]Assistant Professor, Bannari amman institute of technology, Sathyamangalam, TN India
[4]Lecturer, Dept. of ECE, Senthur Polytechnic College, TN, India

Received: 24-12-2013; Revised: 23-01-2014; Accepted: 23-02-2014

*Corresponding Author: **S.R. Jay Kumar**
PG Scholar, Sasurie College of Engineering Vijayamangalam, TN India, E-mail: jaykumarsr@gmail.com

ABSTRACT

This paper is aimed at generation of automated test pattern for asynchronous circuits based on genetic algorithm. Asynchronous circuits without global clocks are hard to test due to the lack of testing techniques. The testing of asynchronous design involves basic element like C-element, completion detector in handshake controller and logic design. The main contribution is to generate optimized test vector using genetic algorithm for complex asynchronous sequential circuits based on fault simulation. The proposed system based on genetic algorithm is effective in terms of result quality and CPU time requirements. Any stuck-at-fault in the circuit maybe caused due to errors in controller circuit or change in the logic behaviour. To overcome the challenges in the creating test vectors genetic algorithm uses fault simulation process for effective pattern based on the fitness function. Experimental analysis shows that generation of optimal test patterns result with good fault coverage, without the consideration of other methods for increasing the testability.

Keywords: Asynchronous circuits, Genetic Algorithm, Mutation, Pseudo code, Fitness scaling.

INTRODUCTION

Testing is one of the critical processes yet to be effectively solved for asynchronous circuits. Synchronous circuits performing operation based on clock signal can be easily tested, whereas asynchronous circuits are a larger class of circuits with additional controlling units. Asynchronous circuits contain more state holding elements than synchronous circuits, resulting in generation of test vectors harder with a higher area overhead. Additional to stuck-at-fault, delay faults and hazards in the circuit are also a serious concern in finding test patterns. But the behaviour of asynchronous circuits makes them easy to test, because a stuck at fault in the handshake makes the logical design to remain in the same state for a longer time which is easily detectable[1-5].

Moreover, asynchronous circuits have more feedback paths and additional state holding blocks resulting in the need of techniques like DFT. Due to this Automated Test Pattern Generation (ATPG) with efficient algorithms are important topics in the present scenario. Fault simulation model provides good result compared to topological and symbolic designs, which is suited for limited number of state holding elements.

The topological approach needs more complex backtracking resulting in more CPU time. Simulation based approach results in more test length. All these methods results in low fault coverage. The objective of this paper is to generate effective test vectors based on GA's[7].

GA's is used as effective ATPG technique in[6,7] which generates test vectors based on objective function. Comparing with other the approaches it results in the creation of short test sequences.

GA follows three phase approach namely selection, generation and fault simulation of test sequence. Our approach aims at providing an effective solution for all the standard benchmark circuits with acceptable CPU time requirements.

The rest of the paper is organized as follows. Section 2 contains description and working of handshake controller. Overview of genetic algorithm is briefed in Section 3. Section 4 is dedicated to the new GA based TPG method. Section 5 analyzes the achieved results and Section 6 concludes paper.

HANDSHAKE CONTROLLERS

A handshake circuit is a collection of handshake components, connected Across various stages, where all communication takes place via handshaking.

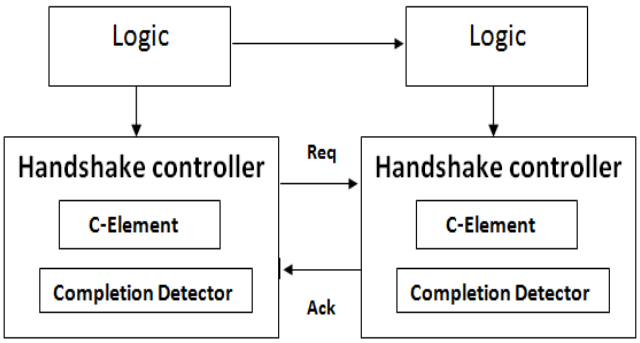

Figure 1: Handshake controller with logic design

The asynchronous circuit can be partitioned into a control block and a data path as illustrated in Figure 1. The control block performs operation based on handshake signals, while the data path works on Boolean signals. The link between the control block and the data path works based on the arrival of data input.

The control block analyzes the conditional signal in logical design to determine the next process of Boolean logic in the data path. The local signals are generated by the control block to enable a datapath register to obtain a new data value. The handshake controller consists of a completion detector (CD) and a C-element. The handshake protocol used in the design is the four phase dual-rail protocol, in which the request signal is encoded

into the data. The m pairs of wires are required to encode m-bit data. For example, one-bit information, denoted by x, can be encoded with a pair of wires x.t and x f . The condition of x.t and x.f is said to be valid if it has the code word (1,0) or (0,1). The (0,0) condition is said to be an empty token. The communication of data between the sender and the receiver involves the four steps:

1) the sender generates a valid codeword on the datapath;
2) the receiver accepts the valid codeword from the channel, and then acknowledges the signal;
3) the sender reply's by sending an empty codeword; and
4)the receiver then completes the communication process.

Thus, the communication in the asynchronous design changes the signal between valid and empty token. That is, protocol contains an empty token between two consecutive valid tokens in the datapath. The completion detector goes from LOW to HIGH when the input to stage Si becomes a valid codeword, and transits from HIGH to LOW when the input to stage Si becomes an empty codeword. The C-element is used as a state holding element which holds the value until new data is arrived.

GENETIC ALGORITHM

A genetic algorithm (GA) is the process of natural selection used to generate effective solutions to various problems. Genetic algorithms generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover.

In order to solve a problem genetic algorithm must have following components:

1) An initial chromosome representation of solution,
2) An effective way to create an initial population,
3) A fitness function that generates effective solution.
4) Genetic operators that produces the structure of "children" during reproduction
5) The total population size and the iteration process.

GA's [7] have been used in generating a possible solution for many search and optimization problems. Each individual is assigned a fitness value based on the evaluation function. The evaluation function calculates how close the individual is to the optimum solution. A group of individuals forms a population that transforms from one generation to the other. The GA starts with an initial population generated randomly. Evolution can take two forms.

Crossover: It is the main operator used for reproduction. It combines portions of two parents to create offsprings, which consists of features of the parents. The crossover is performed with probability Pc .In uniform crossover each chromosome position is crossed equally.

Parent 1 : 10100011
Parent 2 : 00100111
Offspring 1 : 10100111
Offspring 2 : 00100011

Mutation: It is a incremental change made to the member of the population with probability Pm. In binary coded GA mutation performed by flipping a bit. The is effective when the offspring generated after crossover results in the same data.

Before Mutation: 11010011
After Mutation : 11000011

Fitness Scaling:

For effective selection GA selects individual based on the fitness. If selection is not based on scaling or normalization then selected copies consists of minimum good individuals with worst individuals. In the worst case the fitness value are too close or too far apart. The scaling is used to scale the raw fitness so that GA selects good individuals.

Test pattern generation using GA:

Pseudo Code:

1. Generate initial population
2. Perform crossover for p1,p2 to get c1,c2
3. Mutate c1 and c2
4. Select a target fault
5. Apply generated sequence to detect the fault
6. If no fault to detect then stop the process, else fault simulate to generate new sequence and repeat step 2 to 5.

The three phases of the test generation process are

Phase 1: Selecting a target fault
Phase 2: Generating a test for target fault
Phase 3: Fault simulation of the test sequence and search for undetected faults The three phases are repeated until either all the faults have been processed, or a predefined maximum number of iterations been reached.

The test generation is used to generate input sequences that will identify a fault in the circuit. A test consists of two phases. During test generation fault must be excited and it must be made to propagate the error to the output. Even when test sequences are being generated, a sequence may not be found to improve the fault coverage if the initial population does not contain the good combination. GA is then initiated with new population calculated using fault simulation. Many faults are detected by individual test vectors and short sequence, only most difficult faults remain for long sequence.

RESULTS

The figure 2 shows the test pattern generated using GA. From the initial chromosome generated from fault simulation crossover is performed with high probability to get new offspring. These patterns inherit the combination of the parent test sequence. Then mutation is performed with low probability to get different test sequence. The test pattern generated results in the high fault coverage compared with other ATPG test generations.

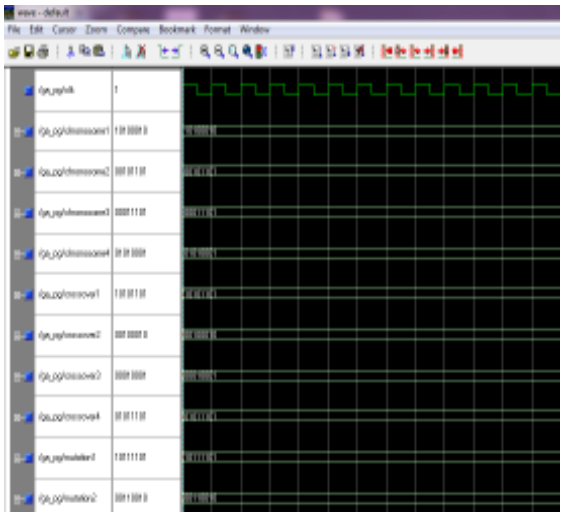

Figure 2: Simulation result for test pattern generated using GA

CONCLUSION

In this work the test pattern generation based on genetic algorithm for self timed logic circuits based on handshake controller is presented. The test pattern generated for asynchronous circuits effectively analyze the communication between control logic for handshake design and the datapath of logical design. The result shows that test pattern produced through GA results in high fault coverage with very low testing time. Additionally it also identifies the path delay fault which occurs in larger asynchronous design.

REFERENCES

1. Handshake solutions. http:www.handshakesolutions.com.
2. Gill G, Agiwal A, Singh M, Shi F, Makris Y. Low overhead testing of delay faults in high-speed asynchronous pipelines. In Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems. 2006, 46.
3. Nowick S, Jha N, Cheng FC. Synthesis of asynchronous circuits for stuck-at and robust path delay fault testability. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.1997; 16(12): 1514.
4. Miron A, Melvin A, Breuer, Arthur D. Friedman. Digital Systems Testing and Testable Design. Computer Science Press. 1990.
5. Unger SH. Asynchronous Sequential Switching Circuits. WileyInterscience, John Wiley & Sons, Inc. New York. 1969.
6. Beerel PA, Meng THY. Testability of asynchronous timed control circuits with delay assumptions. In Proceedings of the 28th ACM/IEEE Design Automation Conference. 1991; 446.
7. Kelsey TP, Saluja KK, Lee SY. An efficient algorithm for sequential circuit test generation, IEEE Trans. Comput. 1993; 1361.

Source of support: Nil, Conflict of interest: None Declared