Asian Journal of Applied Science and Technology (AJAST)
Volume 1, Issue 5, Pages 79-81, June 2017

A Review on Methods of Test Pattern Generation of Flash Memory

V.Mohan Bharath Rao[1] and Farha Kowser[2]

[1]*UG Student, Department of ECE, MVJ College of Engineering, Whitefield, Bengaluru, Karnataka, India.*
[2]*Assistant Professor, Department of ECE, MVJ College of Engineering, Whitefield, Bengaluru, Karnataka, India.*

Article Received: 16 May 2017 *Article Accepted: 03 June 2017* *Article Published: 05 June 2017*

ABSTRACT

There are many methods of testing the flash memory before the memory chip or device gets ready for packaging. These tests are essential to detect the faults in the memory during the manufacturing stage or any other stages of processing. The methods of memory testing are manual, software testing and by obtaining simulation files. These three methods are discussed in this paper and it is compared with the automatic test pattern generation method of flash memory based on LabView.

Keywords: Flash memory, Simulation files, Test pattern and LabView.

1. INTRODUCTION

Execution of Flash memory is predominant, in this manner it ended up noticeably one of the quickest developing semiconductor recollections. At present, it is broadly utilized as a part of many fields, yet there are still a few issues with the trial of Flash memory. The working and testing strategy for memory are intricate, there is no a test example whose blame scope can achieve 100%. The workload of producing a test design by the strategy for utilizing physically is restrictive. Current calculations for testing are for the most part MSCAN, Checkerboard, Walk, Gallop calculation, and so forth. Yet, each blame scope and time unpredictability of calculation have their own particular points of interest and weaknesses. In genuine test, keeping in mind the end goal to accomplish high blame scope, test program can't simply depend on one of these calculations, yet consolidate with two, three or more to produce test design. At present, there are generally three approaches to produce test design, one is finished by hand, it is reasonable for gadgets which rationale capacities are exceptionally basic. The second route is to purchase relating instrument programming, yet this sort of programming is exceptionally costly, and permitting conditions are extremely brutal. Recreation documents which can be gotten from configuration organizations are the third way, those records would be given to test gear provider organizations, however along these lines is extremely time consuming, many outline organization can't give the reproduction records by the reason of the assurance of licensed innovation rights.

2. SOFTWARE TESTING METHOD

Memory diagnostic software programs are low-cost or free tools used to check for memory failures on a PC.They are usually in the form of a bootable software distribution on a floppy disk or CD-ROM.Memory testers are designed to detect two types of faults, non-Permanent faults and permanent faults.There are two types of non-permanent faults, transient fault and intermittent fault .Errors in RAM introduced by transient faults are often called software errors. Factors that will contribute to transient faults are, cosmic ray, alpha particle, pollution, humidity, temperature, pressure, vibration, power supply fluctuations, electromagnetic interference, static electrical discharges.Intermittent faults are caused by non-environmental conditions such as, loose connections, deteriorating or aging components, critical timing, resistance and capacitance variation, physical irregularities, noise.

A. Zero-one algorithm

Also known as Memory Scan (MSCAN).
Step 1: Write 0 in all cells
Step 2: Read all cells
Step 3: Write 1 in all cells
Step 4: Read all cells

The test coverage of Zero -One algorithm is given. Can't detect all address decoder faults. SAF (stuck at faults) can be detected as long as the address decoder is correct. Not all transient faults are detected. Cell may contain 0s before step 1. Not all coupling faults are detected. Cell may contain 0s before step 1. Test length=4*2^N (N=Total Address Bits). It is easy to implement but has little test strength.

B. Checkerboard algorithm

Cells are divided into two groups, cell -a and cell -b forming a checkerboard pattern.
Step 1: Write 1 in all cell-a and 0 in cell-b
Step 2: Read all cells
Step 3: Write 0 in all cell-a and 1 in cell-b
Step 4: Read all cell

Test Coverage of checkerboard Algorithm is given. Can't detect all Address decoder faults. SAF can be detected as long as the address decoder is correct. Not all Transient faults are detected. Initial cell status unknown. Not all coupling faults are detected. Detects shorts between adjacent cells. Worse

© 2017 AJAST All rights reserved.

case test for leakage between adjacent cells. Test length=4*2^N (N=Total Address Bits).

3. OBTAINING SIMULATION FILES

Simulation files which can be obtained from design companies. The files would be provided to test equipment supplier companies. This way is very time consuming, many design company are unable to provide the simulation files by the reason of the protection of intellectual property rights.

Fault coverage and time complexity of each algorithms are different. In actual test, in order to achieve high fault coverage, test program can't just rely on one of these algorithms, but combine with two, three or more to generate test pattern.

Need for the automatic test pattern generation are, effective algorithms can be chosen and used to generate a more general applicable test pattern. Generation of the test pattern with practical low cost. Reduction in test program development difficulty. Improvement in development speed, quality and efficiency of test system.

4. STANDARD TEST INTERFACE LANGUAGE

Standard Test Interface Language (STIL) is a general purpose device pattern interchange format, which defines the functions and parameters tested, supported in the emulator. It is an open pattern depiction specification, which is readable, easy to understand, transport, debug and correct. **EDA-ATE** (Electronic Design Automation) data conversion process can be converted to each big manufacturer's ATE devices by using STIL. This standardization process helps shorten entire cycle from design to testing. It also reduces intermediate links, and the occurence of errors because of incompatible standards. At present, STIL has been widely used in Europe and Japan and other developed countries, In the United States, Intel, IBM, TI, Freesacale, NS and other giants have already adopted STIL as standard data format from integrated circuit design to the test. In Japan, Toshiba and other manufacturers are also actively promoting the STIL become an industry standard.

5. AUTOMATIC TEST PATTERN GENERATION USING LABVIEW

As the LabView method presented in the reference paper, the interface structure for SPI and parallel memory is given.

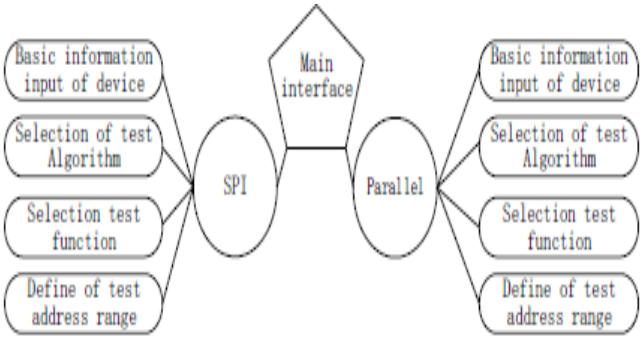

Fig: Interface structure of generator

The proposed method of interface structure is realized using LabView programing. The user interface for both SPI and parallel is developed by LabView environment.

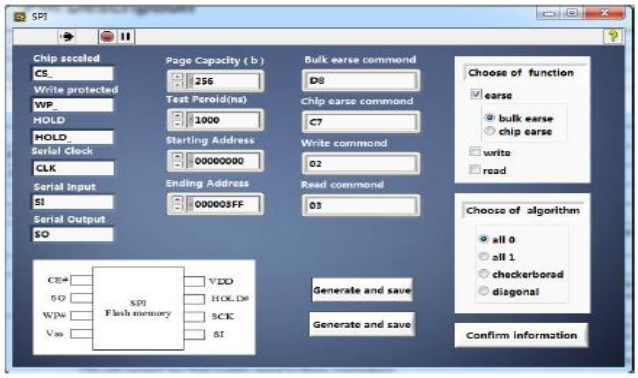

Fig: User interface of SPI flash memory

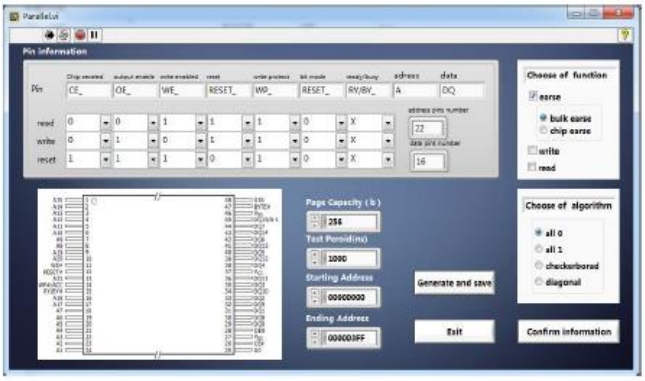

Fig: User interface of Parallel memory

The Pattern file according to the above user interface is generated in the STIL format. Finally, a test program edit interface is obtained based on the algorithm proposed in the reference paper.

6. CONCLUSION

This method of automatic test pattern generation based on LabView is advantageous than other methods because we can choose the algorithm for the generation of test vector. The fault coverage range increases when we combine two or more algorithm for testing the memory. The difficulty level of programming is reduced due to LabView programming.

REFERENCES

[1] S. Tyson, G. Wicker, T. Lowery, S. Hudgens, and K. Hunt, "Nonvolatile, High Density, High Performance Phase-Change Memory," *Proc. of IEEE Aerospace Conf*, vol. 5, pp. 385-390, Mar. 2000.

[2] M. G. Mohammad and K. K. Saluja, "Flash memory disturbances: Modeling and test," in Proc. *IEEE VTS*, Marina Del Rey, CA, pp. 218– 224, Apr. 2001.

© 2017 AJAST All rights reserved.

[3] O. Ginez, J. M. Portal, and H. Aziza, "A High-Speed Structural Method for Testing Address Decoder Faults in Flash Memories," *ITC 2008 IEEE International*, pp.1-10, 2008.

[4] P. Mauroux, A. Virazel, and B. Godard, "NAND Flash testing: A preliminary study on actual defects," *ITC 2009 IEEE International*, vol. 1, pp. 1-6, Nov. 2009.

[5] Y. S. Kim, D. J. Lee, C. K. Lee, and et al , "New scaling limitation of the floating gate cell in NAND Flash Memory," *2010 IEEE International Reliability Physics Symposium (IRPS)*, pp.599-603, May 2010.

[6] C. F. Wu, C. T. Huang, K. L. Cheng, and et al, "Fault simulation and test algorithm generation for random access memories," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21(4), pp.480–490, Apr. 2002.

[7] C. T. Huang, J. C. Yeh, Y. Y. Shih, and et al, "On test and diagnostics of flash memories," *in Proc. 13th IEEE Asian Test Symp (ATS), Kenting, Taiwan*, pp. 260–265, Nov. 2004.

[8] M. Park, K. Kim, J. Park, and et al, "Direct field effect of neighboring cell transistor on cell-to-cell interference of nand flash cell arrays," *IEEE Electron Device Letters*, vol. 30(2), pp. 174-177, Feb. 2009.

[9] J. C. Yeh, K. L. Cheng, Y. Chou, and et al, "Flash Memory Testing and Built-In Self-Diagnosis with March-Like Test Algorithms," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26(6), pp. 1101-1113, June 2007.

[10] T Taylor and G Maston, "Standard Test Interface Language(STIL): A New Language for Patterns and Waveforms," *Proc Int. Test Conf*, pp. 565-570,1996.

[11] S. Motoasca, A. Nicolaide, and E. Helerea, "Simple Analytical Method for Hysteresis Modelling using LabVIEW," *JOAM*, Vol. 10, pp.18471852, July, 2008.

[12] Cheng Gao, Linjiang Hu and Jiaoying Huang "Automatic Test Pattern Generation Method of Flash Memory Based on Labview" *2016 Prognostics and System Health Management Conference (PHM-Chengdu)*, 978-1-5090-2778-1/16/$31.00 ©2016 IEEE.

© 2017 AJAST All rights reserved.