Open Access Library Journal
2020, Volume 7, e6294
ISSN Online: 2333-9721
ISSN Print: 2333-9705

Multi-Memory Grouping Wrapper with Top Level BIST Algorithm

Narek Mamikonyan[1], Suren Abazyan[1], Vakhtang Janpoladov[2]

[1]Yerevan State University, Yerevan, Armenia
[2]Russian-Armenian University, Yerevan, Armenia
Email: su.abazyan@gmail.com

How to cite this paper: Mamikonyan, N., Abazyan, S. and Janpoladov, V. (2020) Multi-Memory Grouping Wrapper with Top Level BIST Algorithm. *Open Access Library Journal*, **7**: e6294.
https://doi.org/10.4236/oalib.1106294

Received: April 2, 2020
Accepted: April 21, 2020
Published: April 24, 2020

Abstract

This algorithm integrates second level Built in self-test (BIST) into multiple memory grouping wrapper. Second level BIST brings additional reliability into the memory system while fastening testing time. Main approach is to test whole memory modules from top level by numerous step count of which can be modified based on power consumption requirement and overheat conditions. The worst case of the algorithm can be observed by the time when the number of steps is equal to the number of memory modules, otherwise the testing time will be relative to 1/N (N is the number of steps). The main advantage of memory wrapping methodology is the possibility of increasing variety of the number of bits and the number of cells in the memory, while using limited memories provided by foundry.

Subject Areas

Computer Engineering, Electric Engineering

Keywords

BIST, Memory Wrapper, Memory Bit

1. Introduction

This paper represents combination of two main units:

- Memory wrapper for increasing flexibility of designing memory-integrated designs.
- Top level BIST method which can reduce testing time of the whole system in one wrapper.

The memory wrapping method commonly used in the VLSI [1] [2] [3]: the main disadvantage of existing methods [1] [4] is that the memory wrapper con-

catenates only memories with the same bit sizes. In other words, the old memory wrappers extend the memory size (capacitance) but not bit size [5]. The standard memory wrapper structures combine concatenation of memory address and simple memory selecting method and the same selecting method is used for memory input and output data pins (**Figure 1**).

Basic memory wrapper consists of few elements:

- U1 unit is combined memories provided from foundry,
- U2 and U3 are the address decoder and memory select decoder,
- U4 is the logical unit which controls data input distribution between memories,
- U5 is the logical unit which controls data output distribution between memories.

The schematic model of U2 and U3 is address decoder, U4 and U5 functionality is the same, both modules selecting connection between wrapper input/output and memory which have been selected based on U3.

The input of U2 and U3 can be combined as address select. With correct concatenation and distribution of the memory pins (clock, write enable, read enable, etc.), the wrapper can be fully functional unit. After above mentioned transformation, the memory wrapper can be used as single memory in the different designs.

Basic BIST for memories has been created for finding broken cells in the memory by writing and reading data bits into/from memory cells [5]. Moreover, mapping of the broken cell address into the reserved memory block addresses is the biggest advantage of the BIST. In other words, the memory BIST directly increases yield of the memory module by additional memory cell mapping. The main disadvantage of the BIST is the power consumption while testing memory cells [6]. BIST core is combination of testing engine and reserved memory (**Figure 2**).

Following is the unit (marked as U) and pin (marked as P) description for **Figure 2**.

1) MUT—The memory which is under test.

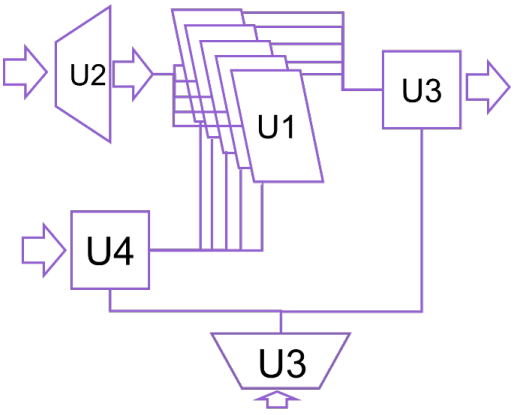

Figure 1. Basic memory wrapper.

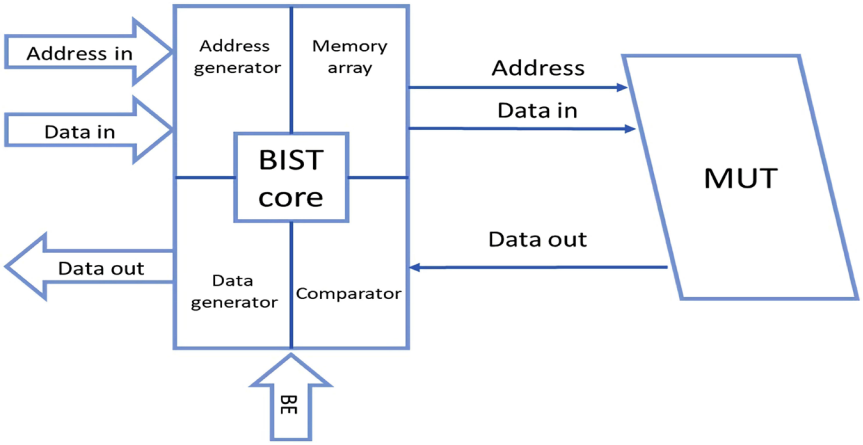

Figure 2. BIST core block diagram.

2) BIST consists of few units:

a) Address generator—Unit for address list generation, which will be given to memory under test (MUT);

b) Data generator—Unit for generating test data to be given to MUT as input;

c) Comparator—Unit to compare generated data with MUT outputted data from under test address;

d) Memory array—Unit for storing data lines which were supposed to be written in broken addresses of MUT;

e) BIST core—Unit for controlling all above mentioned 4 units' works and mapping broken data line addressed of MUT to memory array addresses.

This module has 2 main working modes:

1) BIST mode—to test MUT and remap broken addresses. This mode is self-working mode and does not have connection with external pins.

2) Use mode—to unable the BIST test/repair functionality and leave mapping functionality only. In this mode external pins are being used as Address input and Data input/output.

Modes can be switched with BIST Enable (BE) input signal.

The main aim of having BIST is having ability to check memory cells by writing and reading in/from each cell. For this matter, BIST is writing and reading different values (either logic 0 or logic 1) to be able to cover cases of stacked 0 and stacked 1 in cell. If broken cell is found and there is empty (not linked) cell address in BIST's memory array, BIST core will link broken address of proposed memory cell to BIST memory array address.

Based on memory size, upper mentioned BIST run process can take long initialization time. In memory dominant designs, parallel BIST process can lead to die overheating, which can cause to broken chip.

To overcome overheating and long runtime issues under BIST mode, quasi-parallel BIST implementation can be used. Technic is called quasi-parallel as at each period of BIST, some counts of memories are in test mode, while others are in waiting mode. Main constraint of quasi-parallel BIST is to ensure that on each period of BIST, under-test memories are enough far from each other, in

point of physical placement, to prevent chip from local overheating.

2. BIST Control-Based Wrapper

Above mentioned issue can be solved using BIST control logic with multiple memories, which are already integrated with BIST. BIST control logic will enable BIST mode on some amount of memories base on wrapper compile process selected option (Figure 3).

Wrapper contains 3 main units:

- BIST control—Unit to control BIST mode selection process and concatenate memories and addresses, data input/output (makes system as one memory);
- Standard stand-alone BIST—Upper mentioned standard memory BIST;
- Memory under test—Foundry provided memories.

The option which is selected in compilation time (the amount of parallel-checked memories) will bring to internal grouping of memories, which must be under parallel BIST run. Memories from one internal group must be placed with bigger distance while physical placement process and this will prevent local overheating.

3. Top Level BIST Algorithm

Top level BIST algorithm has two main components (Figure 4):

1) Commonly used BIST methodology for one memory that is integrated with already wrapped memories. This method is the same as memory BIST but addresses and data sizes are extended (wrapped memories). Difference with common BIST is that in top level BIST algorithm BIST is divided into two separate process: checking and repairing.

a) Checking—Checking if the memory cell is working.

b) Repairing—Mapping memory address to BIST backup array.

2) Internal memory BIST enablement core—After checking process this algorithm gets memories with non-working cells as input and enables internal BIST

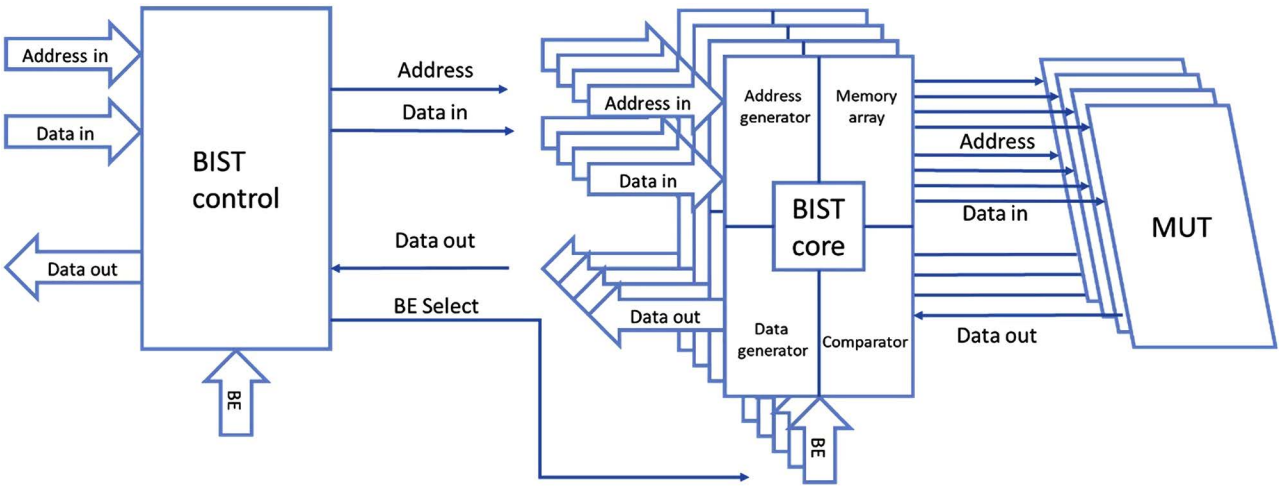

Figure 3. Wrapper block diagram.

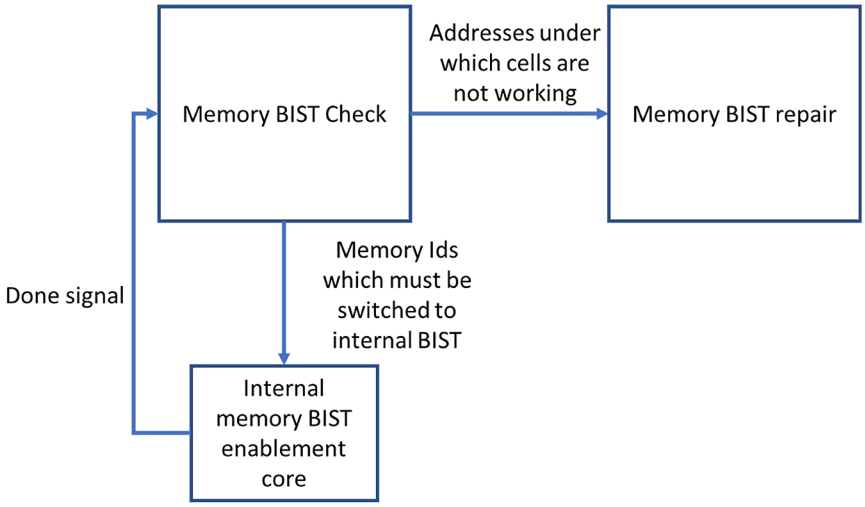

Figure 4. Workflow for top level BIST.

for them. Once internal BIST is done the TOP level BIST is running again, but this time combined with repairing process.

In **Figure 4** can be possible two scenarios:

1) After Top level BIST check all memories are working correctly. In this case no need more action, this scenario is the best in terms of testing time;

2) After Top level BIST check there are memories which need cell address mapping. This scenario will pass through all units mentioned in **Figure 4** starting from memory BIST check until memory BIST repair (memory BIST check –> internal memory BIST enablement core –> memory BIST check –> memory BIST rapier).

4. Memory Grouping Wrapper with Top Level BITS Algorithm

Memory grouping wrapper with top level BIST algorithm is the integration of top level BIST algorithm and BIST control-based wrapper. Main difference of this method compared with one memory BIST method is that BIST core logic combines wrapper grouping methodology and internal memory BIST enablement core (**Figure 5**).

On the other hand, top level BIST can be interpreted as single memory BIST, if memories address and data are combined. Only need to be under consideration BE enablement signal of memories on top level BIST.

Memory grouping wrapper with TOP level BIST contains:

1) Memories under test,

2) Single memory BIST units,

3) TOP BIST unit with BIST core.

5. Testing Results

For testing BIST, selected MARCH C algorithm as a reference and implemented MARCH C for memory and top-level BIST. Implementation has been tested in

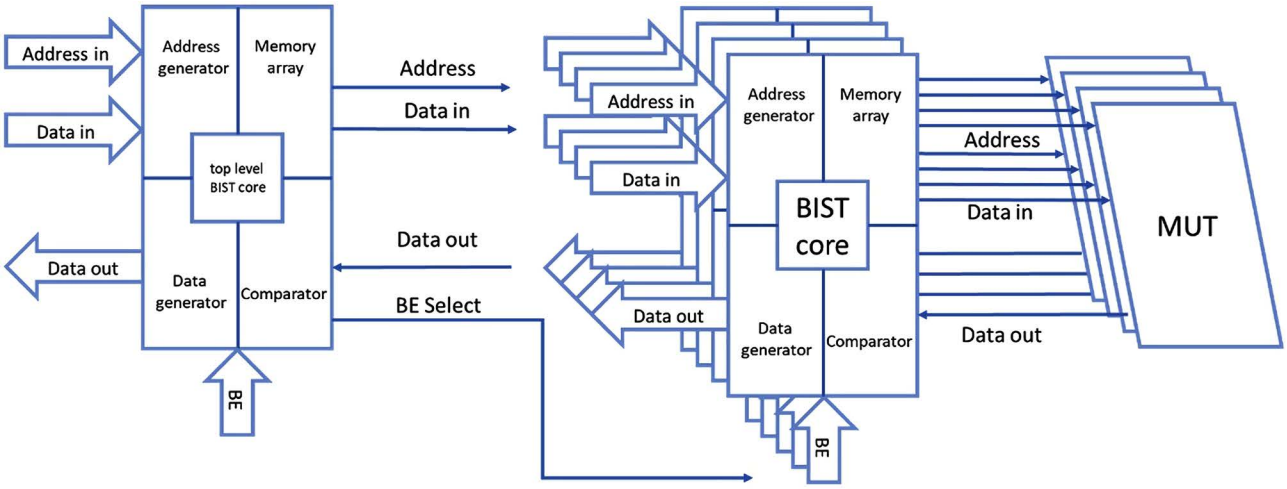

Figure 5. Memory grouping wrapper with top level BIST block diagram.

Table 1. Comparison results.

Memory cell Count	Testing time	Testing time without top level BIST	Redundancy address count only memory	Redundancy address count difference Top level
5	23.7 ms	19.4 ms	50	2
10	27.6 ms	22.7 ms	100	4
20	31.4 ms	24.5 ms	200	6

the Xilinx Inc spartan 6 FPGA with constricted same 64x8 memory cell. Results are shown in Table 1.

New architecture has approximately 13% more STD cells and memory testing time increases by 22% and added approximately 3% more redundancy addresses. The redundancy addresses bring additional reliability in the memory System.

6. Conclusion

This technique brings additional relatability in the memory system. According to the test results using this method memory testing time will be increased, but the memory system will get additional top-level memory BIST. Based on test results top-level BIST brings 3% more redundancy addresses. Results are expected as we are adding more redundancy addresses.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Gharsalli, F., Meftali, S., Rousseau, F. and Jerraya, A. (2002) Automatic Generation of Embedded Memory Wrapper for Multiprocessor SoC. *Proceedings* 2002 *Design Automation Conference* (*IEEE Cat.* No.02CH37324), New Orleans, LA, 10-14 June

2002, 596-601. https://doi.org/10.1109/DAC.2002.1012695

[2] Kukner, H. (2010) Generic and Orthogonal March Element based Memory BIST Engine. Master Thesis, Delft University of Technology, Delft, CE-MS-2010-01.

[3] Cheng, A.C. (2002) Comprehensive Study on Designing Memory BIST Algorithms, Implementations and Trade-Offs. The University of Michigan, Ann Arbor, MI 48109-2122.

[4] Miyazaki, T. and Fujiwara, H. (2006) A Memory Grouping Method for Sharing Memory BIST Logic. *Asia and South Pacific Conference on Design Automation*, Yokohama, 24-27 January 2006, 10-25. https://doi.org/10.1145/1118299.1118457

[5] Pavani, P., Anitha, G., Bhavana, J. and Praneet Raj, J. (2016) A Novel Architecture Design of Address Generators for BIST Algorithms. *International Journal of Scientific & Engineering Research*, **7**, 3-8.

[6] Ojha, S.K., Singh, O.P., Mishra, G.R. and Vaya, P.R. (2019) An Efficient Use of Memory Grouping Algorithm for Implementation of BIST in Self. *Test Journal of Engineering and Applied Sciences*, **14**, 2695-2700. https://doi.org/10.36478/jeasci.2019.2695.2700