ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology
Volume 1, Issue 3, May2012

Implementation of March Algorithm Based MBIST Architecture for SRAM

M. Radha Rani[1], G. Rajesh Kumar[2], G. Prasanna Kumar[3]
Vijetha Institute of Technology and Sciences[1], Vishnu Institute of Technology[2,3]
E-mail: sreeni.smily@gmail.com[1], kavi.aeiou@gmail.com[2], prasanna.godi@gmail.com[3]

Abstract – This paper presents the implementation of March Algorithm based Memory Built-In Self Test (MBIST) architecture for Static Random Access Memory (SRAM). A Finite State Machine (FSM) is designed to implement March – based Test algorithm. Also SRAM block and the interfacing modules are presented. There is a standard March Test Algorithm with 22N where N is the number of memory words, read/write operations is discussed. The proposed March test algorithm with 13N can achieve full diagnosis for SRAM. This digital system is described in Verilog HDL and is simulated and synthesized using Xilinx Spartan 3 FPGA. The proposed scheme greatly simplifies the testing process. Besides, the proposed scheme is more efficient in terms of circuit size and test data to be applied, and it requires less time to test SRAM chip. Power consumption of a digital system is nearly fifteen times more in test mode when compared with normal mode. Power consumption is greatly reduced in this proposed architecture by reducing number of read/write operations. Experimental results show that the proposed method achieves a good flexibility with smaller circuit size.

Keywords- SRAM, MBIST, FSM, March Test Algorithm.

I. INTRODUCTION

SOC designs contain a variety of components, including digital, memory, as well as analog and mixed-signal circuits, all of which need to be tested. DFT is essential for reducing test costs and improving test quality.

Scan is widely used for digital logic testing. However, one of the challenges in the nanometer design era is dealing with the rapidly growing amount of scan data required for testing complex SOC designs. The bandwidth between an external ATE and the device under test is limited, creating a bottleneck on how fast the chip can be tested. This has led to the development of test compression and logic BIST architectures, which reduce the amount of data that needs to be transferred between the ATE and device under test. Another key issue for scan testing is applying at speed tests, which are crucial for detecting delay faults. Chapter 2 reviews the basics of scan testing and presents test compression and logic BIST architectures as well as architectures for applying at-speed tests.

When small-scale integration (SSI), medium-scale integration (MSI), and large scale integration (LSI) were the dominant levels of component integration, large systems were often partitioned so that data flow paths and control circuits were placed on separate printed circuit boards (PCBs). Most PCBs in a given design contained data flow circuits that were not difficult to test using an ATPG. A lesser number contained the more complex control logic and handshaking protocols. Test programs for control logic would be created by requiring a logic designer or test engineer to write vectors that were then fault simulated to determine their effectiveness. Since the complex PCBs made up a smaller percentage of the total, test creation was not excessively labor-intensive. The task of writing tests for these boards was further simplified by the fact that sequential transitions in control logic could often be observed directly at I/O pins rather than indirectly through observation of their effects on data flow logic. The evolution of technology has brought about an era where individual ICs now possess hundreds of thousands to millions of gates. RAM and ROM often reside on the same IC with complex logic. Individual I/O pins serve multiple purposes, acting both as inputs and as outputs. The increasing gate to pin ratio results in fewer I/O pins with which to gain access to the logic to be tested. Architecturally, many chips have complex arbitration sequences that require several exchanges of signals before anything meaningful happens inside the chip. All of these factors contribute to potentially long test programs that strain the resources of available test equipment and point to the conclusion that test issues must be considered early in the design cycle.

Advances in semiconductor memory technologies have resulted in a very high density embedded memories. The number of memory cell per chip has increased rapidly. From 1982 to 2006, the number of memory cells per chip has increased more than 1 Gigabyte in size resulting in a more complicated architecture [1]. Paradoxically, the larger capacity of RAM will introduce more physical failure as a result due to crosstalk between closely packed cells. Thus, high capacity memory size will produce more defects during system-on-chip (SOC) manufacturing that can lead to yields reduction. Therefore, the semiconductor industry is facing with a new challenge to test and diagnose failures defects that occurs in high capacity memories. Hence, SRAM memories is one of the most used in the embedded system due to the faster operation compared to other memory types such as DRAM and DDRAM. Investigation on SRAM testing is needed to be explored in terms of their fault models such as coupling faults (CFs) and test algorithms such as March Test Algorithm (MTA). The efficiency of MTA is based on how many write (w) and read (r) operation are applied to the SRAM to detect and diagnose the CFs. With this efficiency, it promises better results on faults detection and diagnosis.

II. BUILT IN SELF TEST

Built-in self-test (BIST) is a design technique in which parts of a circuit are used to test the circuit itself. Built-in self-test is the capability of a circuit (chip, board, or system) to test itself. BIST represents a merger of the concepts of built-in test (BIT) and self-test, and has come to be synonymous with these terms. The related term built-in-test equipment (BITE) refers to the hardware and/or software incorporated into a unit to provide DFT or BIST capability.

The choice of BIST architecture is a function of several factors, some of which are listed next.

1. Degree of test parallelism: Distributed BIST provides a higher degree of test parallelism, since several CUTs can be tested at the same time.
2. Fault coverage: Distributed BIST usually leads to higher fault coverage since TPG and ORA circuits can be customized to each CUT. For example, a BIST technique for a block of combinational logic may not be suitable for a RAM.
3. Level of packaging: At higher levels, centralized BIST becomes more natural. For example, micro diagnostics testing is only applicable at the level where a Micro programmable controller exists. This controller can then be used to test many components of a system.
4. Test time: Distributed BIST usually leads to a reduction in test time.
5. Physical constraints: Size, weight, power, cooling costs, and other factors influence the design. Often embedded and separate BIST architectures require more hardware and degrade performance.[8]
6. Complexity of replaceable units: If a board is a replaceable unit and is to be self-testable, then it must contain TPG and ORA circuitry. If a system is the lowest level of replaceable unit, then its constituent boards need not have TPG and ORA circuitry and a more centralized BIST architecture can be used.
7. Factory and field test-and-repair strategy: The type of ATE and degree to which it is used to test and diagnose failures influences and is influenced by BIST. For example, because of the increasing use of surface-mounted devices, in-circuit orbed-of-nails testing is becoming more difficult to use and hence the need for BIST and the use of boundary scan.
8. Performance degradation: Adding BIST hardware in critical timing paths of a circuit may require a reduction in the system clock rate.

Figure 1 shows a typical logic built-in self-test (BIST) system. The test pattern generator (TPG) automatically generates test patterns for application to the inputs of the circuit under test (CUT). The output response analyzer (ORA) automatically compacts the output responses of the CUT into a signature. Specific BIST timing control signals, including scan enable signals and clocks, are generated by the logic BIST controller for coordinating the BIST operation among the TPG, CUT and ORA. The logic BIST controller provides a pass/fail indication once the BIST operation is complete. It includes comparison logic to compare the final signature with an embedded golden signature, and it often encompasses diagnostic logic for fault diagnosis. As compaction is commonly used for output response analysis, it is required that all storage elements in the TPG, CUT, and ORA will be initialized to known states before self-test and no unknown (X) values be allowed to propagate from the CUT to the ORA. In other words, the CUT must comply with more stringent BIST-specific design rules [3] in addition to those scan design rule required for scan design.

For BIST pattern generation, in-circuit TPGs are commonly constructed from linear feedback shift registers (LFSRs) [9] or cellular automata to generate test patterns or test sequences for exhaustive testing, pseudo-random testing, and pseudo-exhaustive testing [3]. Exhaustive testing always guarantees 100% single-stuck and multiple stuck fault coverage. This technique requires all possible 2n test patterns to be applied to an n-input combinational CUT, which can take too long for combinational circuits where n is huge; therefore, pseudo-random testing [6]is often used for generating a subset of the 2n test patterns and uses fault simulation to calculate the exact fault coverage. The TPG is often referred to as a pseudo-random pattern generator (PRPG). In some cases, this might become quite time consuming, if not infeasible. To eliminate the need for fault simulation while at the same time maintaining 100% single-stuck fault coverage, we can use pseudo-exhaustive testing [7] [3] to generate 2w or2^k-1 test patterns, where $w<k<n$, when each output of the n-input combinational CUT at most depends on w inputs. For testing delay faults, hazards must also be taken into consideration.
For output response compaction, the ORAs are commonly constructed from multiple-input signature registers

(MISRs). The MISR is basically an LFSR that uses an extra XOR gate at the input of each LFSR stage for compacting the output responses of the CUT into the LFSR during each shift operation. Oftentimes, to further reduce the hardware overhead of the ORA, a linear phase compactor comprised of a network of XOR gates is connected to the MISR inputs.

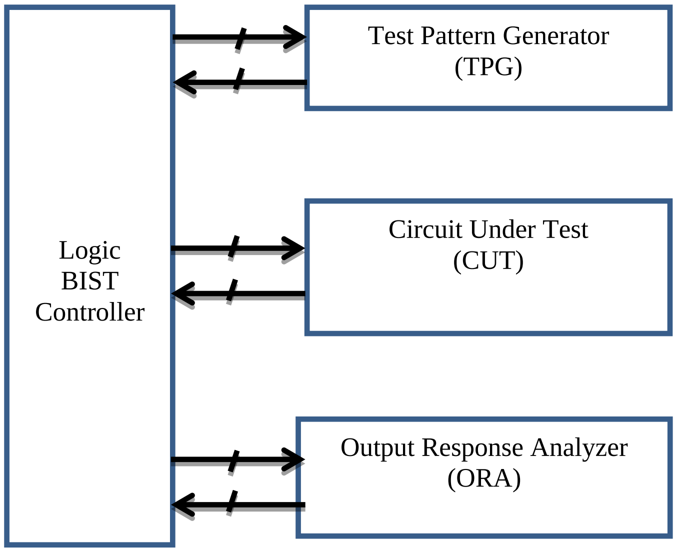

Fig: 1 BIST System

III. MEMORY FAULTS

As memories grow larger, with more memory cells packed into an ever-shrinking die area, the cost to manufacture a die remains fairly constant, while the time it takes to apply test programs increases exponentially[5]. It is variously estimated that the cost to test a memory chip runs from 50% to 70% of the total cost of the finished product. The first step in reducing the cost of memory test is to understand what fault mechanisms are most likely to occur and then develop test programs that target those faults. With this approach, the manufacturer and the end-user can determine their priorities, balancing cost versus DPM (defects per million) that they can tolerate in their applications. A number of different failure types can occur in semiconductor memories, affecting memory cell contents, cell addressing, and the time required to read out data. Some of the more common failures include the following[5]:

Cell opens or shorts
Address non uniqueness
Cell/column/row disturb sensitivity
Sense amplifier interaction
Slow access time
Slow write recovery
Data sensitivity
Refresh sensitivity
Static data losses

Opens and shorts within semiconductor memory cells may occur because of faulty processing, including misaligned masks or imperfect metallization.

These failures are characterized by a general randomness in their nature. Opens and shorts may occur at the chip connections to a printed circuit board. In a km × n memory system containing km words of n bits each, and made up of memory chips of size m × 1, a fault that occurs in bit position i of m consecutive bits is indicative of either a totally failed chip or one in which an open or short exists between the chip and the PCB on which it is mounted. Address nonuniqueness results from address decoder failures that may either cause the same memory cell to be accessed by several different addresses or several cells may be addressed during a single access. These failures often cause some cells to be physically inaccessible. An effective test must insure that each read or write operation accesses one, and only one, memory cell. Disturb sensitivity between adjacent cells or between cells in the same row or column can result from capacitive coupling. Slow access time can be caused by slow decoders, overloaded sense amplifiers, or an excessive capacitive charge on output circuits. Slow write recovery may indicate a saturated sense amplifier that cannot recover from a write operation in time to perform a subsequent read operation.

A memory cell can be affected by the contents of neighboring cells. Worse still, the cell may be affected only by particular combinations on neighboring cells. This problem grows more serious as the distance between neighboring cells shrinks. Refresh sensitivity in dynamic RAMs may be induced by a combination of data sensitivity and temperature or voltage fluctuations. Static RAM cells are normally able to retain their state indefinitely. However, data may become lost due to leakage current or opens in resistors or feedback paths. When discussing faults in random logic, that fault models other than the stuck-at model were examined. The one trait these models had in common was a susceptibility to combinatorial explosion. For very small circuits, the number of faults grew so quickly that it was simply not feasible to consider them. Memory circuits, because of their density and the close proximity of cells to one another, exhibit this problem of combinatorial explosion to a far greater degree. Hence, it becomes necessary to restrict consideration to faults that are most likely to occur.

IV. TESTABLE SRAM DESIGN

Fault diagnosis and location in semiconductor Random Access Memories (RAM) are of prime importance in connection with the increasing density and dominating portion of embedded memories in system-on-chips (SOC). Manufacturing defects should be detected, diagnosed and located for further repair in order to improve the product quality, reliability and yield. Embedded memories in SOC are often designed with various redundancy elements (spare rows and / or columns) intended for fault repair. To increase the manufacturing yield as much as possible, not only the redundancy allocation algorithms, but also the test algorithms should apply efficient procedures for effective diagnosis and location of defective cells. Often these two procedures were considered separately, and usually fault location procedure followed the diagnosis procedure.

This paper proposes a March-based fault location algorithm for repair of bit-oriented and word oriented Static RAMs. A March CL algorithm of complexity 12N is defined for fault detection and partial diagnosis. A 3N or 4N March-like algorithm is used for location of the aggressor word of inter-word CFs. Then another March-like algorithm of complexity 9(1+logB), B is the number of bits in the word, is applied to locate the aggressor bit in the aggressor word. Finally, a March algorithm of complexity 6(1+logB)N is used to detect and locate inter-word stuck-at, transition faults, as well as state, idempotent, inversion, write-disturb CFs. The proposed algorithm has higher fault location ability and lower time complexity than other known algorithms developed for fault location in SRAMs.

V. MARCH ALGORITHM

In this section some classical, or legacy, memory test algorithms will be examined. Memory test algorithms fall into two categories: functional and dynamic[6]. A functional Test targets defects within a memory cell, as well as failures that occur when cell contents are altered by a read or write to another cell. A dynamic test attempts to find access time failures. The All 1s or All 0s tests are examples of functional tests[12].

These tests write 1s or 0s into all memory cells in order to detect individual cell defects including shorts and opens. However, these tests are not effective at finding other failure types.

Like most of the algorithms, begins by writing a background of zeroes. Then it reads the data at the first location and writes a 1 to that address. It continues this read/write procedure sequentially with each address in memory. When the end of memory is reached, each cell is read and changed back to zero in reverse order. The test is then repeated using complemented data. Execution time is of order N. It can find cell opens, shorts, address uniqueness, and some cell interactions.

March-AB Algorithm which is generally used for designing testable SRAMs are described below.

{↕(*w*0)↓(*r*0,*w1*,*r1*,*w1*,*r1*)↓(*r*1,*w0*,*r0*,*w0*,*r0*)↑(*r*0,*w*1,*r*1,*w*1,*r*1) ↑(*r*1,*w0*,*r0*,*w0*,*r0*) ↕(*r*0)}

The basic notations used in algorithm are as follows:

↑: address n-1 to 0
↓: address 0 to n-1
↕: either way
w0: Write 0 to the word
w1: Write 1 to the word
r0: Read a cell whose value should be 0
r1: Read a cell whose value should be 1

The selected March Cd for bit-oriented RAMs is interpreted as follow:

{↕ (w0); ↑ (r0,wl); ↑ (rl); ↑ (r1,w0); ↑ (r0); ↓(r0,w1) ↓ (r1); ↓ (r1,w0); ↕ (r0)}

↑: address 0 to n-1
↓: address n-1 to 0
↕: either way
W0: Write 0 to the word
W1: Write 1 to the word
R0: Read a cell whose value should be 0
R1: Read a cell whose value should be 1

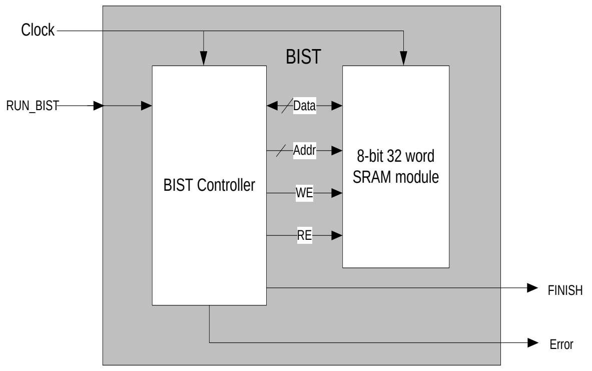

Fig: 2 Block Diagram Representing Testable SRAM

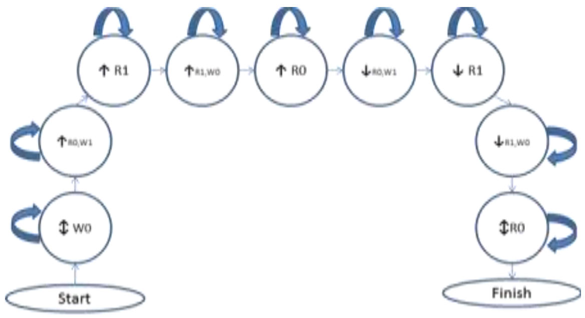

Fig: 3 March Algorithm Flow Graph

Verilog HDL Design of Testable SRAM is done with Xilinx ISE Simulator. The design is simulated with the same tool ISE simulator. Simulation results are shown below. For different levels of the algorithm the simulated results we can observe here. Figure 4 shows the simulation results for the SRAM in Normal mode of operation. When the SRAM it is operating in normal mode, read write operations can be observed. Figure 5 shows the simulation results when the FSM is in (R0, W1) stage. At this stage BIST controller is responsible for generating control signals and test patterns. Figure 6 shows the simulation results when the FSM is in (R0) stage. After this stage "finish" goes high indicating testing process completed. After this stage error signal is "low" indicating it is error free chip.

In order to view the working of the algorithm with a faulty chip, here is a faulty chip designed. BIST architecture is applied for faulty chip also. Figure 7 shows the simulation results for the faulty chip. In the R0 stage it found an error and asserts error signal. When error signal is asserted, immediately test is completed and is indicated by finish signal.

VI. CONCLUSION

The March algorithm has been generated to detect and diagnose the faults in SRAM chip has been successfully implemented. With the proposed March algorithm all the general occurring faults identified and are diagnosed. March test algorithms and the simulation results have shown that 100% fault coverage and 100% diagnostic resolution has been achieved. Regarding computation time we have achieved a great advantage. March AB algorithm computation time is nearly 0.9ns for a 32 byte SRAM chip where as with March Cd algorithm the computation time is 0.08ns only. It is noted that the computation time is calculated for a Fault free chip only. It is strongly believed that this BIST can be widely used for the embedded memory testing especially under the **SOC** design environment due to the superior flexibility and extendibility in applying different combination of memory test algorithms. Future work will consider optimization algorithms other than the March algorithm used in this work to obtain better solutions

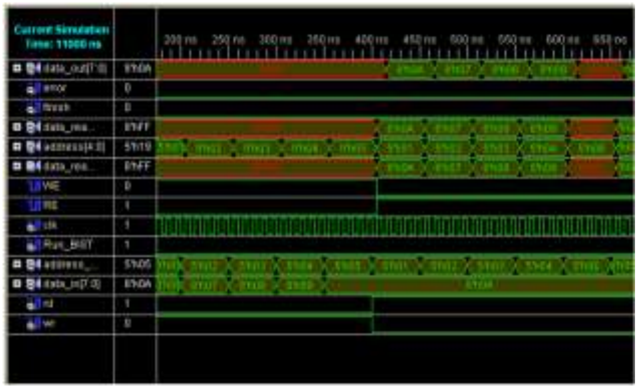
Fig: 4 Simulation results of SRAM chip in Normal Mode of Operation

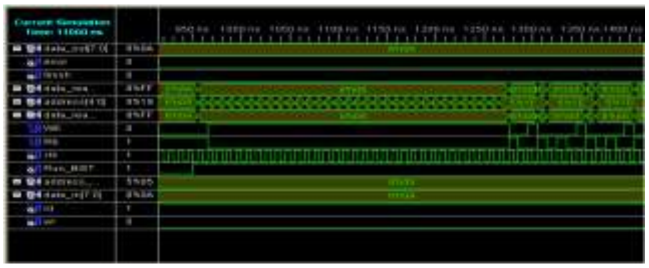
Fig: 5 Simulation results of SRAM chip in Test Mode

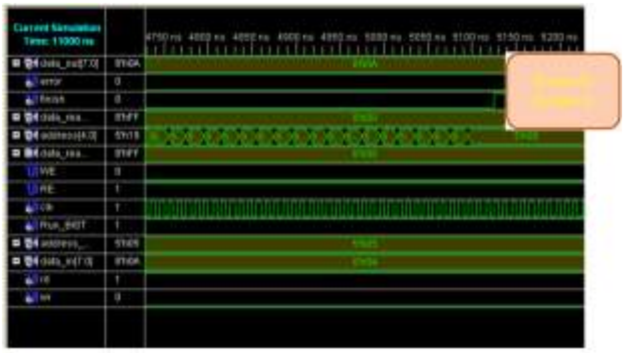
Fig: 6 Simulation Results of the Faulty free SRAM chip

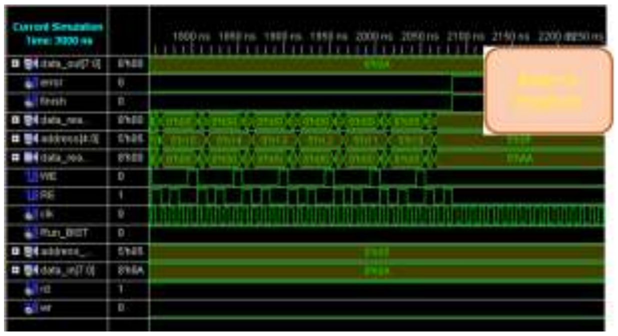
Fig: 7 Simulation Results of the Faulty SRAM chip

REFERENCES

[1] A.J. Van De Goor (2001) Testing Semiconductor Memories, Theory and Practice, ComTex Publishing, Gouda Netherland.

[2] A.J. Van De Goor (1993) Using March Test to test SRAMs, IEEE Design of Computers, USA.

[3] C.W. Wang, C.F Wu, J.F. Li, C.W. Wu, T. Teng k. Chiu, H.p. Lin(2000) A Built-in Self-test and Diagnosis Scheme For Embedded SRAM, ATS, Taiwan.

[4] M. Abramovici, M. A. Breuer, and A. D. Friedman, Digital Systems Testing and Testable Design, IEEE Press, Revised Printing, Piscataway, NJ, 1994.

[5] F. Brglez and H. Fujiwara, A neutral netlistof 10 combinational benchmark circuits and a target translator in

Fortran, in Proc. Int. Symp. On Circuits and Systems, pp. 663–698, June 1985.
[6] P. H. Bardell, W. H. McAnney, and J. Savir, Built-In Test for VLSI: Pseudorandom Techniques, John Wiley & Sons, Somerset, NJ, 1987.
[7] R.-M. Chou, K. K. Saluja, and V. D. Agrawal, Power constraint scheduling of tests, in Proc. Int. Conf. on VLSI Design, pp. 271–274, January 1994.
[8] F. Corno, M. Rebaudengo, M. SonzaReorda, and M. Violante, A new BIST architecture for low power circuits, in Proc. European Test Workshop, pp. 160–164, May 1999.
[9] S. W. Golomb, Shift Register Sequence, Aegean Park Press, Laguna Hills, CA, 1982.
[10] V. Iyengar, and K. Chakrabarty, Precedence-based, preemptive, and power constrained test scheduling for system-on-a-chip, in Proc. VLSI Test Symp., pp. 42–47, May 2001.
[11] S. Manich, A. Gabarro, M. Lopez, J. Figueras, P. Girard, L. Guiller, C. Landrault, S. Pravossoudovitch, P. Teixeira, and M. Santos, Low power BIST by filtering non-detecting vectors, JETTA J. Electronic Testing: Theory and Applications, 16(3), pp. 193–202, June 2000.
[12] Vardanian, V.A.; Zorian, Y, A March-based fault location algorithm for static random access memories, Proc. of 2002 IEEE international workshop, pp. 62-67, 2002.

M. Radha Rani: Asst. Professor in Vijetha Institute of technology and sciences. Major working areas are wireless communications, Linear and Digital ICs and VLSI. Has seven years of teaching experience. Presented research papers in two national conferences.

G. Rajesh Kumar: Asst. Professor in Vishnu institute of technology. Has five years of teaching experience. Major working areas are Digital electronics, Embedded Systems and VLSI. Presented research papers in four international conferences and two national conferences.

G. Prasanna Kumar: Asst. Professor in Vishnu institute of technology. Has four years of teaching experience. Major working areas are Digital Signal Processing, Wireless communications and Embedded Systems. Presented research paper in one national conference.