Implementation of Concurrent Online MBIST for RFID Memories

SUNIL KUMAR CHALAMACHERLAB, Dr. K. PADMAPRIYA
PG Scholar, Assistant Professor
Department of ECE, JNTUACEA, Anantapur, 515002, India
Email:sunil4ece4a9@gmail.com, Email:kesaripadmapriya@yahoo.com

Abstract

Concurrent online testing is a memory test mechanism where the memory can be tested concurrently with the system operation. Thus, it has instant error detection. Radio Frequency Identification (RFID) devices rely on the correct operation of their memory for identification of objects and delivery of transponder’s information. This paper presents the implementation of concurrent online test scheme for RFID memories based on Memory Built in Self Test (MBIST) architecture. This paper also presents the finite state machine (FSM) of the transponder access scheme, Symmetric transparent version of March c- algorithm, implementation of memory BIST. The solution was implemented using VHDL and was, in turn, verified on Xilinx ISE 9.2i simulator, and synthesized using Spartan 3E kit.

1. Introduction

Digital systems are made of data paths, control paths and memories. Defects in memory arrays are generally due to shorts and opens in memory cells, address decoder and read/write logic. These defects can be modeled as single and multicell memory faults. The dominant use of embedded memory cores along with emerging new architectures and technologies make providing low cost test solution for these on-chip memories a very challenging task. The addition of extra circuitry to facilitate testing of memory chips, called design for testability (DFT), or to allow the test mechanism to be completely contained within the chip, called built-in self-test. Memory BIST has been proven to be one of the most cost-effective and widely used solutions for memory testing for the following reasons: no time, on-chip test pattern generation to provide higher controllability and observability, on-chip response analysis, test can be on-line or off-line, adaptability to engineering changes, easier burn in support. The paper is organized as follows: section 3 describes the finite state machine (FSM) of the transponder access scheme. Section 4 describes the March algorithm, section 5 describes the implementation of Memory BIST, section 6 describes the simulation results, the conclusions are offered in section 7.

2. Background

The two types of BIST are On-line BIST and Off-line BIST. On-line BIST has tests implemented on-chip. It has shorter test time but an area overhead of one to three percent. Off-line BIST, on the other hand has tests implemented off-chip. It has longer test time but no area overhead. On-line BIST can further be classified into three subgroups: Concurrent BIST, Non-Concurrent BIST and Transparent BIST. Concurrent BIST is performed during normal use of the chip. This means that faults occurring during normal use can be detected and, even be corrected. A non-concurrent test is not active during normal use of the chip but only in test mode. The advantage of this form of BIST is that the test does not have to preserve the data which is stored on the chip, thus allowing a maximum freedom in the test data to be used. The disadvantage is the circuit cannot detect faults that are not covered by the faults models used. Transparent BIST scheme is very similar to the Non-Concurrent scheme except the memory contents are preserved. With memory BIST, the entire memory testing algorithm is implemented on-chip, and operates at the speed of the circuit, which are 2 to 3 orders of magnitude faster than a conventional memory test. Most memory BIST schemes exploit the parallelism within memory device to achieve a massive reduction in test time.

A radio-frequency identification system uses tags, or transponder attached to be identified [1]. Two-way radio transmitter and receiver called interrogators or readers send signal to the tag and read its response. The readers generally transmit their observations to a computer system running RFID software or RFID middleware.
The tag’s information is stored electronically in a non-volatile memory. The RFID tag includes a small RF transmitter and receiver. An RFID reader transmits an encoded radio signal to interrogate the tag. The tag receives the message and responds with its identification information. This may be only a unique tag serial number, or may be product-related information such as a stock number, lot or batch number, production date, or other specific information.

2.1 Operation of Transponder

Following a top-down approach, the transponder protocols are defined in three different layers: application, communication and physical.

In the application layer, the transponder receives commands from the interrogator that are valid only when the tag has been singled out. These commands generally consist of writing, reading or locking the tag’s internal memory. At this layer, an interrogator may be able to terminate indefinitely the tag’s operation by issuing a password-protected command.

The communication layer allows an interrogator to manage tag populations while embracing an anti-collision protocol. A great number of tags may be controlled by supervising tag’s data collisions. A regular scheme to avoid collisions employs a two-part scheme where an interrogator, first, selects a broad number of tags and, subsequently, forces them to randomly choose access slots. This access mechanism is employed within the EPC C1G2 protocol [2] and is based in the Dynamic Frame Slotted ALOHA algorithm (DFSA) [3]. To support access from several interrogators, transponders provide session flags allow interrogators to organize groups of tags and force them to enter a particular inventory round.

Transponder memory follows a division in banks according to the function of the memory portion as follows:

- Reserved memory, which includes passwords for accessing special tag functions.
- Product identification memory, which is a code used to identify the object containing the tag.
- Tag Identifier memory, which is the unique identification number of the tag.
- User memory, which is an application specific bank.

3. FINITE STATE MACHINE (FSM) of the Transponder access.

The normal operation of an interrogator, when accessing a set of transponders, relies on subsequent selections of smaller groups of tags and random assignment of access slots. This selection procedure is time-consuming and does not involve reading or writing the memory for transponders that are in the interrogator queue. A selection command issued by the interrogator impels a tag or group of tags to set or unset their internal flags according to a comparison mask. In this way, an interrogator is able to split in smallest sets a larger group of tags in order to access them easily. Typically, an interrogator starts a new inventory pointing towards a previously selected set of tags. Transponders matching the interrogator’s flags selection must generate an internal random Queue Position Number (QPN) which represents its assigned slot in the DFSA algorithm. The maximum QPN available for the transponders is determined by the interrogator each time an inventory starts. In order to establish a direct link interrogator-transponder, the interrogator sends a command which is answered only by transponders which QPN is equal to zero. Meanwhile, the other transponders involved in the inventory should decrement their own QPN by one, until their turn to answer the interrogator comes. The success of the anti-collision scheme relies in the effectiveness of the interrogator to select an appropriate maximum value for the QPN which avoids picking the same time slot by more than one transponder. Every transponder is accessed individually while the others remain in an Arbitrate state waiting for their access slot. In the Arbitrate state, transponders are fully powered by the interrogator signal but no particular operation is being executed. The concurrent online access scheme proposed exploits this waiting state to perform the test of the memory and is based on the anti-collision mechanism of EPC C1G2 standard.

3.1 Selection Stage

Every transponder works in one of four sessions and has separate inventoried flag for each. These flags determine whether the transponder may respond to the interrogator or not within an inventory round. A Selected flag (SL) also exists which purpose is to ensure a greater accuracy during management of large transponder populations. The proposed scheme introduces a Test flag which can be asserted by the interrogator to force transponders to a testing state while being accessed. An interrogator issues a Select command to select a particular transponder population by asserting or deasserting their flags. This command
aims at a particular flag and forces its value, e.g., a SL flag is asserted. Within the proposed scheme, the interrogator chooses the population of tags to be tested by asserting its Test flag with the Select command.

### 3.2 Testing Stage

![Finite State Machine Diagram]

Figure-1: shows the proposed finite state machine (FSM) of the transponder access scheme.

Once a transponder is within the range of an interrogator, it reaches the Ready state. The Ready state is a holding state for energized transponders that are not participating in an inventory round. A transponder that is in Ready state accepts Select commands from the interrogator that force it to set or unset session flags.

The transition from the Ready to the Arbitrate state is done when the interrogator broadcasts a Query command with a session flag as a parameter. Transponders matching the session flag transit to Arbitrate, the others stay in Ready and do not participate in the inventory round. Every transponder, \( t_i \), going to Arbitrate chooses randomly a QPN, \( QPN_i \). The access scheme allows the interrogator to adaptively choose an adequate interval of QPN in order to consider the number of transponders available in the inventory round or the time needed to finish the memory test. Consequently, by issuing commands to transponders, the interrogator forces them to pass from Arbitrate to Ready back and forward until the QPN interval is appropriate for the current inventory round. QPN, \( QPN \),’s valid values are defined as: \( QPN \in [0, 2^0 - 1] \), with Q being chosen by the interrogator for each inventory round. Regular operation of the interrogator-transponders interaction consists of command-based transitions from the Arbitrate state to the Reply state by transponders which QPN is equal to zero. The interrogator has full access to the transponder and its memory within the Reply state. The proposed testing approach includes a new state for testing, MemTest, which sends a signal to a BIST controller to start the test of a given memory block and keeps track of its result. To prevent unwanted behavior, a transponder \( t_i \) in the MemTest state reacts only to the QueryRep command which forces the decrement of QPN, i.e., changes to the next time slot. An extra 32-bit register is implemented in the transponder to be used as a memory block counter during the test process. The information regarding the memory block to test is sent through data lines towards the BIST.

A transponder within Ready state which receives a Query command with matching flags, and with the test flag asserted, should go to MemTest state and should compute its QPN. In this case, QPN should be selected to allow the whole test of the memory, thus, the QPN value randomly chosen within the regular interval is increased by a fixed offset equal to the number of memory blocks to test. Concurrently, the memory block counter is loaded with the number of the first memory block. When the test is finished, the transponder transits to the Arbitrate state to continue with the regular operation related to accessing its information. In order to inform the interrogator that an error has been detected, the transponder should transit to the Reply state while sending a temporary random identifier accompanied with an error code. The error code describes the nature of the error and the place where it has been detected as well. In case of no error detection or while in regular operation, the transponder should backscatter only the temporary identifier.

### 4. March Test Algorithm

Many algorithms have been developed for testing semiconductor memories, from which the most popular and advantageous are the March tests [4]. A march test contains a sequence of march elements which is composed by a March tests are able to detect several fault models such as Stuck-at Faults (SAF), Address Faults (AF) and some Coupling Faults (CF).

The operations that can be executed in the cells may be: write zero (w0), write one (w1), read zero (r0) and

---

**ISSN:2229-6093**

*IJCTA | July-August 2012*

Available online@www.ijcta.com
read one \((r1)\). The read operation checks if the value inside the cell is the expected one. The order in which cells are considered can be ascending or descending. A typical march test used to test RAMs is MATS++ which can be adapted to test EEPROMs. The MATS++ algorithm is described as follows:

\[
\downarrow (w0); \uparrow (r0,w1); \downarrow (r1,w0,r0).
\]

Word-oriented memories, such as the ones found in an RFID, need a slightly different approach. By extending the 0 or 1 to 16 bits, March algorithm can be easily applied to RFID’s word-oriented memories with a reduction on the coverage of CF.

### 4.1 Implementation of Symmetric Transparent Test

Regular march tests produce the erase of the contents in the memory. To prevent losing data a transparent approach is introduced. The transparent method avoids traditional comparison and, instead, uses a signature analysis mechanism based on a feedback shift register [5]. Well-known march tests can be easily extended to transparent versions by replacing values 0 and 1, in the read and write operations, by \(a\) and \(\overline{a}\), respectively, where \(a\) refers to original content and \(\overline{a}\) to its complement. Besides this modification, the initialization part in the original march test should be removed. A symmetric transparent test poses a constraint on the symmetry of the march test, e.g., it should have the same number of reading for the original and the complement content, since the signature mechanism computes the signature when fed by the original content and computes the reciprocal signature when fed by the complementary content. By doing so, the initial state of the signature mechanism should be found at the end of the test when the memory is fault free.

In this paper, we consider the March C- algorithm for testing the EEPROM. The March C- algorithm has a higher complexity than MATS++ and is described in the following transparent version:

\[
\uparrow (ra'); \uparrow (ra,wa'); \downarrow (ra',wa); \downarrow (r'a,wa'); \downarrow (r'a',wa); \downarrow (ra).
\]

The notations are described as follows:

\(\uparrow\): address n-1 to 0
\(\downarrow\): address 0 to n-1

ra: read memory cell and feed result into signature analyzer, expected value is \(a\).
ra': read memory cell and feed result into signature analyzer, expected value is \(a'\).
Wa: write a into memory cell.
wa': write \(a'\) into memory cell.

Figure 2 shows the architecture of the BIST module composed by six entities: offset generator, memory input multiplexer, output multiplexer, BIST controller, signature analyzer and test pattern generator.

The function of the input multiplexer is to choose which signals input to the memory according to the BIST mode. The output multiplexer provides constant values and the ready/busy (RB) signal is set to zero throughout all the test period. The offset generator is a module that modifies incoming address depending on the bank selected for the memory during regular operation. The BIST controller captures the init signal from the transponder’s FSM and starts the test procedure.

The test pattern generator is responsible for generating the test vectors to be introduced to the memory. It contains the sequence and directions of the march test in a configuration array. Its implementation consists of a FSM which takes information from the configuration array and performs its instructions, while the complement of the data read from the memory is used as input when needed.

The signature analyzer is a Multiple Input Shift Register (MISR) with a flow signal that sets its direction of propagation. This implementation avoids the use of two different shift registers for the signature...
and the reciprocal signature computation. To reduce the probability of error masking, an irreducible polynomial was selected for the MISR; it has the following form:

\[ h(y) = 1 + y^7 + y^9 + y^{12} + y^{16}. \]

Additional methods to avoid error masking involves hardware solutions, e.g., additional check parity, the use of hamming codes or larger MISRs, which are undesirable for the constrained RFID system due their overhead.

6. Simulation and Synthesis results

VHDL Design of Testable RFID memories are done with Xilinx ISE Simulator. The design is simulated with the same tool ISE simulator. Simulation and synthesis results are shown below.

Figure-3 simulation results of Transponder in normal mode of operation.

Figure-4: Simulation results for RFID Memories in TEST mode of operation.

Figure-5: Simulation results for Golden Signature.

Figure-6: synthesis report for Transponder in normal mode of operation.

Figure-7: RTL Schematic for Transponder in normal mode of operation.

7. Conclusion

The transparent March c- algorithm has been generated to test the RFID memories has been successfully implemented. 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.
Acknowledgment

Sunil Kumar would like to thank Dr. K. Padma Priya, who had been guiding throughout to complete the work successfully, and would also like to thank the HOD, ECE Department and other Professors for extending their help & support in giving technical ideas about the paper and motivating to complete the work effectively & successfully.

References


