Fault Tolerant and BIST design of a FIFO cell

F. Corno, P. Prinetto, M. Sonza Reorda

Politecnico di Torino
Dipartimento di Automatica e Informatica
Torino, Italy

Abstract*

This paper presents a BIST design of a parametrized FIFO component. The component is currently being used in the standard library of Italtel, the main Italian telecom circuit maker. Design choices have been strongly influenced by industrial constraints imposed by the Italtel design flow. To achieve the desired fault coverage level for faults in the memory and in the control logic, traditional BIST schemes had to be combined with more advanced testing techniques. Different parts of the circuits are tested with different strategies and algorithms to account for their different nature: critical parts of the design, such as the FIFO control unit and the BIST controller, are tested with on-line test techniques. The final implementation shows that a high fault coverage is attained with an acceptable area overhead and no speed penalty.

1. Introduction

This paper describes the design of a FIFO component (Fig. 1) with BIST capabilities. The component is now being used in the Italtel standard library and is exploited in several industrial designs. The main contribution of this paper is to show how the effectiveness of complex BIST design can be improved, and brought to acceptable fault coverage levels, through the coupling with more advanced test architectures developed for on-line testing schemes. In this case study, we adopted fault tolerant schemes and self-checking components to ensure that critical circuitry is correctly working.

The original (i.e., non-BIST) FIFO specifications are around a dual-port static RAM and feature the following characteristics:

- *empty* and *full* indicators are provided, flagging the status of the queue
- response time for read and write operations is one clock cycle
- illegal operations (e.g., reading from an empty FIFO) are correctly ignored
- simultaneous read and write operations are allowed at any time.

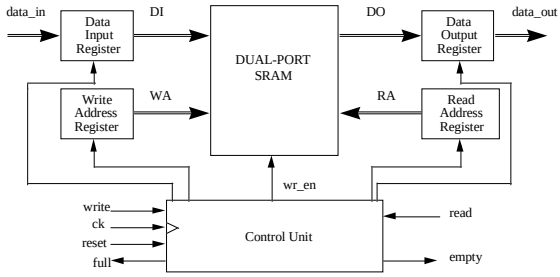

Figure 1: *block diagram of the FIFO component*

Section 2 describes the design goals for the BIST component stemming from company-wide industrial constraints and from testability considerations. Starting from the design goals, section 3 outlines the test strategy, analyzing different possible architectures and justifying the adoption of on-line test schemes. Section 4 analyzes the obtained result from the points of view of testability, area and performance, while section 5 draws some concluding remarks.

2. Design goals

The constraints imposed by the targeted industrial environment strongly limited design freedom and influenced final choices. In particular, the most compelling constraints were:

- the component must follow a standard company-wide BIST protocol, in order to be able to activate the BIST through the standard Boundary-Scan Test Access Port

* This work has been partially supported by Italtel. Contact address: Paolo Prinetto, Dipartimento di Automatica e Informatica, Politecnico di Torino, Corso Duca degli Abruzzi 24, I-10129 Torino (Italy), e-mail **Paolo.Prinetto@polito.it**

EURO-DAC '96 with EURO-VHDL '96
0-89791-848-7/96 $4.00 ©1996 IEEE

- the design must be as independent as possible of the size of the embedded RAM. In particular, the finite state machines (FIFO controller and BIST controller) must depend neither on the depth nor on the width of the FIFO
- no knowledge about the internal structure of the RAM can be assumed, therefore special addressing structures [ShRo85] [VGSZ94] are not applicable
- for manufacturing and testability reasons, according to company-wide design rules, a strictly synchronous design is required.

Furthermore, the following testability goals were settled:

- the embedded RAM should be tested for significant fault models, including at least stuck-at, transition and coupling faults [ABFr90] [AbRe83]
- to reduce overhead, the RAM test must be implemented resorting to the same addressing logic of the non-BIST component
- the data path (counters, comparators, multiplexers, etc.) should be tested for single stuck-at faults
- the control unit responsible for the "normal mode" behavior is to be regarded as a critical component, therefore 100% fault coverage is required
- in the BIST controller itself faults that shorten the length of the test sequence or skip some test procedure have to be covered
- the effects of multiple faults are neglected.

3. Test strategy

Given the above constraints, different design choices concerning the BIST architecture were adopted. In the analysis, the design has been decomposed in:

- the dual-port *memory*, inclusive of all decoding logic
- the *data path*, consisting of all registers, counters, multiplexers, comparators and random logic surrounding the memory
- the *normal-mode control unit*, controlling the FIFO behavior of the component while not under test
- the *BIST controller*, i.e., the FSM responsible for generating the correct sequence of operations composing the test algorithm.

In the following, a test solution for each part of the circuit will be discussed.

Memory

Since one goal of a BIST solution is to thoroughly test the RAM according to significant fault models, a RAM testing algorithm must be applied to the memory. The specialized BIST controller is therefore introduced, generating the required sequence of patterns to be written to and read from the memory.

In the choice of the test algorithm, resource sharing forces us to re-use the same addressing logic used in the normal-mode behavior. In particular, the two memory ports are addressed by up-counters, therefore the RAM should be regarded as single-order addressed (SOA). The March B– test for SOA memories is selected [vdGo93], that meets our fault coverage goals, giving a 100% fault coverage for addressing faults, stuck-at faults, transition faults, coupling faults, and unidirectional linked faults. The March B– algorithm has been modified to account for a *m*-bits wide memory by adding a fifth march element [CPSB95].

Data Path

As far as the data path is considered, a functional test is applied to these components by the BIST controller while the March test is in progress. In particular, registers are tested for stuck-at faults, read and write counters are sometimes operated in parallel and their contents compared, outputs of comparators are always checked, even when their expected value is known, and so on. With these principles, all functions of each functional block in the data path are exercised and verified. Some blocks exhibited a low fault coverage with this functional test, mainly because their state was almost stable (such as the "flag generator", the FSM generating the error signal) or their outputs were not observable (such as the "pattern generator", the FSM generating 1/0 patterns to be written into the RAM. For such components, a self-checking version has been developed, by duplicating the circuit and comparing the outputs.

Normal Mode Control Unit

Being the normal-mode control unit the most important component, special care has been taken to devise its test strategy. Several solutions have been examined and tested. Figure 2 shows alternative architectures that have been examined: an arrow from a block to another is present when the former is in charge of testing the latter.

As a first hypothesis, the normal-mode control unit could be merged with the BIST controller (Fig. 2a), thus potentially reducing the area overhead. This solu-

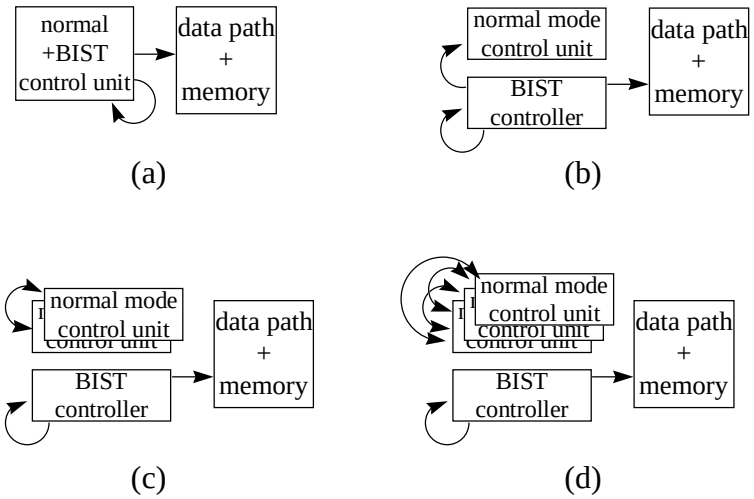

Figure 2: *alternative test architectures*

tion has been discarded since there was no guaranteed way of testing the faults in the portion of the combinational logic computing the normal mode functions while the circuit was in test mode. In other words, the control unit did not excite the normal-mode state transitions while executing the test: the net result was that the normal-mode behavior was not tested at all, except by accident.

From the above considerations, all acceptable solutions must contain separate normal-mode and BIST controllers; three alternatives were examined. One first alternative (Fig. 2b) could employ the BIST controller to apply a functional test to the normal-mode control unit. Since it is very difficult to achieve high fault coverages with functional tests for FSMs, we discarded this option.

In a second alternative (Fig. 2c) one could implement the normal-mode control unit as a self-checking FSM, by duplicating it or by adopting more sofisticated schemes [Lala85]. This would guarantee a very high fault coverage, provided an extensive set of inputs is applied to the FSM inputs; unfortunately, the good/faulty indication would show up during the normal functioning of the circuit and not during the dedicated BIST phase. This in unacceptable, since it would require modifying the system-level test strategy, by requiring a functional test to follow BIST.

A last examined alternative (Fig. 2d) was to build a

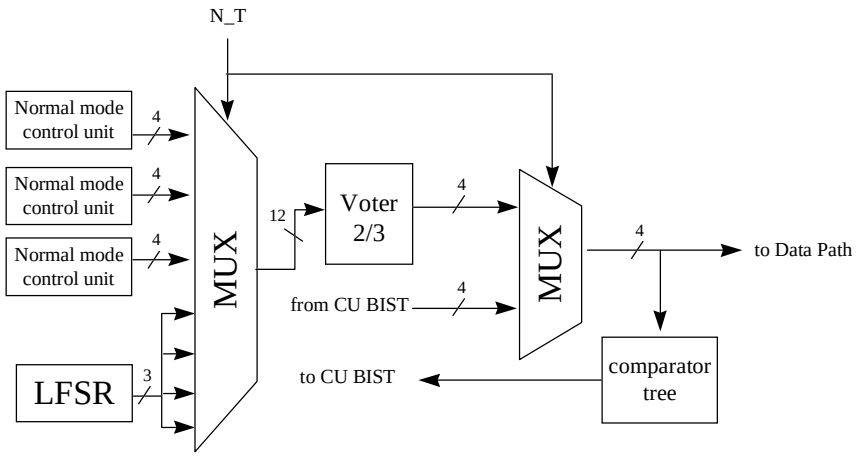

Figure 3: *fault-tolerant Normal-mode Control Unit*

fault-tolerant normal-mode control unit, e.g., by triple redundancy. Since the control unit is small (one flip-flop and some tens of gates), the area overhead for triplication is comparable with other solutions. In the final circuit implementation, this last solution has been adopted (Fig. 3): the outputs of the triplicated normal mode control unit feed a 2/3 voting whose outputs are used in the data path. The critical part of the circuit is represented by the voting and multiplexing logic: the BIST controller therefore performs an exhaustive test on this combinational logic. In particular, an LFSR is used to generate all possible 3-bit combinations for each of the 4 channels of the voter. The correct response is checked by comparing the outputs of different channels.

BIST Controller

The last component that needs to be considered and tested is the BIST controller itself. A self-checking architecture must be implemented to avoid that a fault in the BIST controller could flag a circuit as good without actually performing the test.

In particular, a set of *critical faults* has been identified via fault simulation, where a critical fault is defined as a fault in the BIST logic that could possibly mask some faulty behavior in the non-BIST logic, by misinterpreting the status indications or by skipping some test phase.

Since the goal was to avoid the presence of critical faults in the BIST controller, a MISR has been employed to compute a signature of the flip-flops and outputs during the test sequence, ensuring that the correct sequence of states has been traversed and that the correct values have been applied to the data path. The effect of aliasing proved to be negligible.

4. Implementation Analysis

The BIST component has been designed in the Synopsys environment with the SGS-Thomson ISB24000 technology. The additional area has been measured and Table 1 reports the size of the components (excluding the RAM) for different synthesis strategies. The equations are parametrized by the RAM width (m) and depth (N). The term "patt_gen" is the contribution of a small FSM generating alternating bit patterns [CPSB94], whose size does not depend linearly on the RAM size. Area overhead values are also plotted in Fig. 4.

max testability	$3245 + 136.75\log_2 N + 64.5m$ + patt_gen (=428)
min area	$3168 + 136\log_2 N + 63.75m$ + patt_gen (=424)
min delay	$3173 + 136\log_2 N + 63.75m$ + patt_gen (=428)

Table 1: *area overhead*

Concerning timing overheads, the component has been designed so that the critical path (lying on the path from a clock edge to an empty/full indication) does not contain any component relevant to BIST. In other words, the operating speed of the FIFO does not decrease as a consequence of the insertion of the BIST circuitry.

The attained fault coverage was quite satisfying, and is reported in Table 2. The apparently low fault coverage of the data path is mainly due to multiplexers that exclude primary inputs during test and to output wires, which are not testable under any BIST scheme.

The described component is currently in use at Italtel for FIFO components of various sizes embedded in larger designs. The only parts that depend on the FIFO dimensions are address counters ($\log_2 N$ bits wide), input and output buffers and comparators (m bits wide) and the expected signature for the BIST controller. Critical parts of the design, including control units, multiplexers and voters, are independent of the FIFO size, therefore their size and testability are guaranteed.

Block	TFC%	fault model	test strategy
memory array	100.00	fault classes detailed in section 3	BIST
data path	83.88	stuck-at	functional testing
normal-mode control unit	N/A		fault tolerance by triplication
voters	100.00	stuck-at	BIST exhaustive test
BIST controller	74.65	stuck-at	self-checking via signature analysis

Table 2: *test effectiveness*

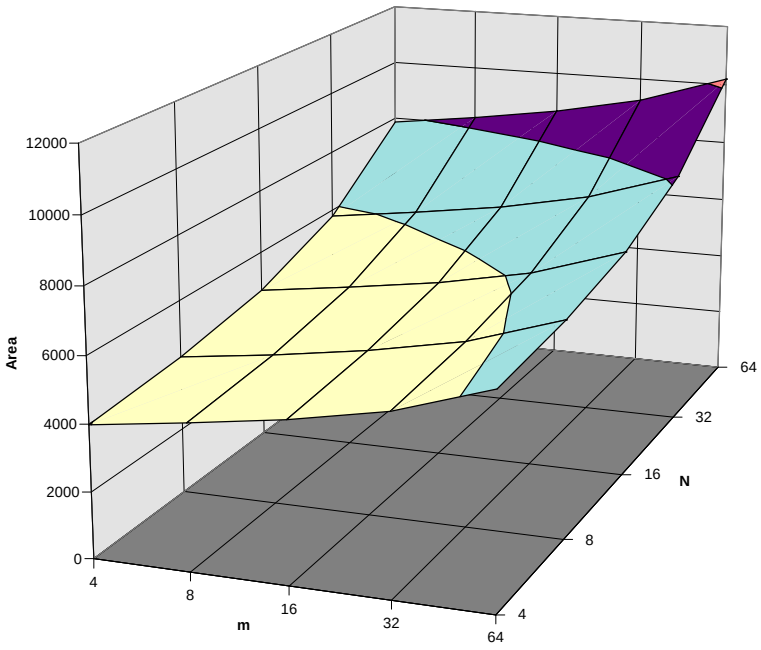

Figure 4: *area overhead*

5. Conclusions

This paper showed the design of a FIFO component with BIST capabilities. A careful analysis of the possible test architectures shows that, in order to guarantee sufficient fault coverage in complex BIST schemes, different test strategies must be applied to different parts of the circuit.

In particular, critical parts of the design, amounting to the normal mode control unit and to the BIST controller, were particularly difficult to test with traditional BIST solution. Only the adoption of more advenced testing strategies, such as on-line self-cheching and fault-tolerant circuits, allowed us to improve the testability of the design.

6. Acknowledgments

The authors wish to thank Dr. Stefano Barbagallo, Dr. Andrea Burri, and Dr. Davide Medina of the Italtel Design Center for their useful discussions and suggestions and Alfredo Benso and Laura Biason for participating in the design of the component.

7. References

[ABFr90] M. Abramovici, M.A. Breuer, A.D. Friedman: “Digital Systems Testing and Testable Design”, Computer Science Press, 1990

[AbRe83] M.S. Abadir, H.K. Reghbati: “Functional Testing of Semiconductor Random Access Memories”, Computing Surveys, Vol.15, No.3, September 1983

[CPSB95] P. Camurati, P. Prinetto, M. Sonza Reorda, S. Barbagallo, A. Burri, D. Medina: “Industrial BIST of Embedded RAMs”, IEEE Design and Test of Computers, Fall 1995, pp. 86-95

[Lala85] P.K. Lala: “Fault Tolerant & Fault Testable Hardware Design”, Prentice Hall, 1985

[ShRo85] M. Shephard, D. Rodgers: “Asynchronous FIFOs Require Special Attention”, ITC’85: IEEE International Test Conference 1985, pp. 445-450

[vdGo93] J. van de Goor: “Using March Tests to test SRAMs”, IEEE Design and Test of Computers, March 1993, pp. 8-14

[VGSZ94] J. van de Goor, I. Schanstra, Y. Zorian: “Fault Models and Test for Ring Address Type FIFOs”, VTS’94: IEEE VLSI Test Symposium, 1994

[VdGZ93] J. van de Goor, Y. Zorian: “Effective March Algorithms for Testing Single-Order Addressed Memories”, EDAC’93: IEEE European Design Automation Conference, 1993, pp.499-505

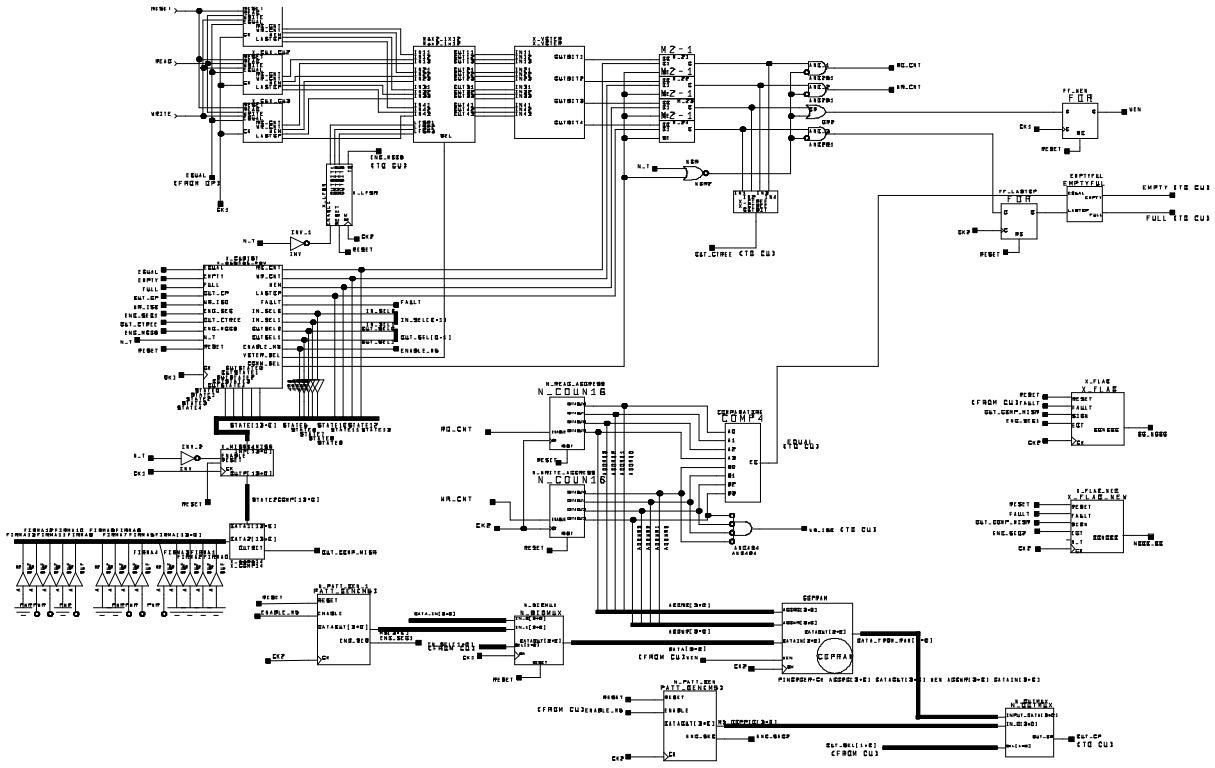

Figure 5: schematic of the FIFO BIST