Test Quality Analysis and Improvement for an Embedded Asynchronous FIFO

Tobias Dubois[3] Mohamed Azimane[1] Erik Larsson[3]
Erik Jan Marinissen[1] Paul Wielage[1*] Clemens Wouters[2]

[1] NXP Semiconductors
Research
High Tech Campus 48, M/S-02
5656AE Eindhoven
The Netherlands
mohamed.azimane@nxp.com
erik.jan.marinissen@nxp.com
paul.wielage@nxp.com

[2] NXP Semiconductors
Digital Library Technology
High Tech Campus 46, M/S-11
5656AE Eindhoven
The Netherlands
clemens.wouters@nxp.com

[3] Linköpings Universitet
Dept. of Computer Science
Embedded Systems Laboratory
SE-581 83 Linköping
Sweden
tobdu865@student.liu.se
erila@ida.liu.se

Abstract

Embedded First-In First-Out (FIFO) memories are increasingly used in many IC designs. We have created a new full-custom embedded FIFO module with asynchronous read and write clocks, which is at least a factor two smaller and also faster than SRAM-based and standard-cell-based counterparts. The detection qualities of the FIFO test for both hard and weak resistive shorts and opens have been analyzed by an IFA-like method based on analog simulation. The defect coverage of the initial FIFO test for shorts in the bit-cell matrix has been improved by inclusion of an additional data background and low-voltage testing; for low-resistant shorts, 100% defect coverage is obtained. The defect coverage for opens has been improved by a new test procedure which includes waiting periods.

1 Introduction

An increasing number of ICs is utilizing large numbers of First-In First-Out (FIFO) memories in their design. These embedded FIFOs are used for intermediate storage, data rate conversion, and clock domain crossing. Also new design paradigms like Globally-Asynchronous Locally-Synchronous (GALS) [1] and Network-on-Chip (NOC) [2] make extensive use of embedded FIFOs. Conventional FIFO designs are typically based on either an embedded SRAM, or made up entirely from standard-cell logic. Despite the fact that the individual FIFOs are typically small in area size, due to the large numbers of FIFOs per IC design, their overall impact on silicon area is significant. Consequently, NXP Semiconductors has decided to add full-custom FIFO modules to their library, which are both smaller and faster than their conventional counterparts. The new full-custom FIFO module is a micropipeline [3], consisting of a series of asynchronously communicating stages, each implemented as a register of latches and a control cell. Design and design-for-test (DfT) details of the FIFO are described in a companion paper at this conference [4].

As all on-chip circuitry, this new FIFO module needs to be tested for manufacturing defects. The size of an individual FIFO is small, and hence its impact on the IC-level yield and quality is small. However, the typical use scenario is that hundreds of these FIFOs are used in a single IC design. This implies that the collective impact of all on-chip FIFOs on IC-level yield and quality is significant. Consequently, effective, yet efficient testing is important.

At NXP Semiconductors, we are accustomed to check the defect detection qualities of our tests for embedded memories by means of a defect-based analysis method based on analog simulation at transistor level [5, 6, 7]. For a newly developed module and its test, such as our new FIFO, this is even more important, as during the conceptual development of the tests of the module, some defects might otherwise easily be overlooked. This paper describes the defect-based analysis procedure for embedded memories and how it was applied to our new FIFO module. The analysis led to an improvement of our initial FIFO test suite, by inclusion of additional data backgrounds and low-voltage testing.

The remainder of this paper is organized as follows. The FIFO design is briefly described in Section 2. The defect-based analysis method used to evaluate and optimize the test set is described in Section 3, while the original FIFO test procedure is described in Section 4. Analysis results and test improvements are discussed in Section 5. The paper is concluded in Section 6.

* Paul Wielage is currently with NXP Semiconductors' IC Laboratory in Eindhoven, The Netherlands.

978-3-9810801-2-4/DATE07 © 2007 EDAA

2 FIFO Design

The FIFO design and DfT is described in more detail in a companion paper [4]. In order to make this paper self-contained, this section gives a brief overview of the most prominent FIFO design features.

The architecture of the FIFO design is depicted in Figure 1. The FIFO consists of a write interface, a read interface, and a kernel. The kernel is built according to the principle of an asynchronous micropipeline [3]. It consists of stages that utilize local handshake signals to propagate data from one stage to the next. Each stage can hold one word of data. Since there is no global clock in the design, a word that is written into the FIFO ripples automatically through the pipeline till it arrives at the last empty stage. Similarly, when a word is read, the last stage becomes empty, which creates an empty word slot (*'hole'*) that ripples towards the input side of the pipeline.

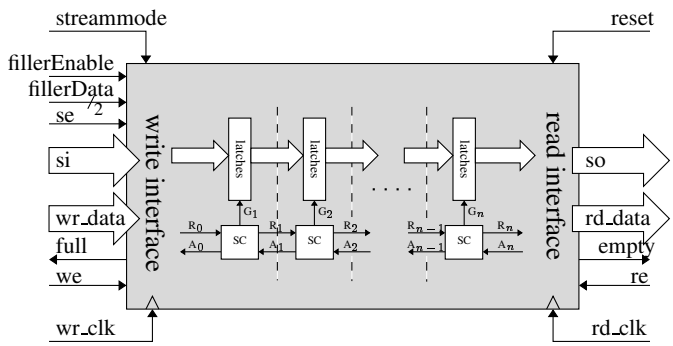

Figure 1: The architecture of the FIFO module.

The schematic of the data-path latch is given in Figure 2. The bits in the data path are dual-rail encoded, i.e., both the bit value and its inverse are stored. This allows to employ an SRAM-like write technique in the transparent mode of the latch. Two cross-coupled inverters for keeping state are implemented by transistors $P1$, $P2$, $N1$, and $N2$. Transistors $N3$, $N4$, and $N5$ allow to change state. To write, transistor $N5$ must be turned on by a high level on `G` such that the true or the complement signal of the cell will be forced to ground, depending on the data input `D` (and its complement `DN`).

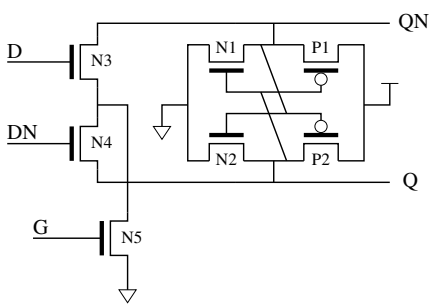

Figure 2: Static latch used in the data path.

The FIFO supports the basic operations (1) *write*, (2) *read*, and (3) *shift*. While the write and read operations are truly asynchronous, the shift operation, which actually is a combined write and read operation, only works if `wr_clk` and `rd_clk` are synchronous to each other. The shift operation can be used when the FIFO is embedded in a synchronous context, or when the two clock signals are explicitly made synchronous.

Figure 3: The layout of the 16×19-bit FIFO instance.

Three FIFO instances have been developed in a 90 nm CMOS technology: 16×19 (= 16 words × 19 bits per word), 32×37, and 64×37. The layouts of these instances were hand-crafted to obtain the smallest possible area. The layout of the 16×19-bit instance (with a total silicon area of 2,582 μm^2) is shown in Figure 3. The largest layout block is formed by the bit-cell matrix. Its 19-bit words with alternating data (`D`) and data-not (`DN`) cells are clearly distinguishable as horizontal rows. Power lines are routed left and right of the bit-cell matrix, as well as through the middle. The asynchronous control cells are left of the bit-cell matrix. The height of a control cell corresponds to the height of two rows of bit-cells, and hence the control cells could be laid out in eight horizontal rows of two control cells each. Other layout blocks implement the input and output buffers and write and read interfaces.

3 Defect-Based Analysis Method

At NXP Semiconductors, a defect-based analysis method is used to assess the defect detection qualities of a test suite for embedded full-custom modules, such as embedded memories [5, 6]. The method, which is based on Inductive Fault Analysis (IFA) [8] uses analog simulations on a transistor-level netlist of the module to determine if realistic resistive shorts or opens are detected by a given test suite. For embedded memories, the method has been largely automated in the form of an in-house tool called MEMSIM. Investigation of undetected defects typically leads to modifications and additions in the tests and/or their stress conditions [7], in order to improve the corresponding defect coverage.

An outline of our method is given by the flow chart shown in Figure 4. The top part of the flow with dark-gray boxes with white lettering indicates the part of the flow which is executed only once. The bottom part of the flow consists of two nested loops. The medium-gray boxes depict what is part of only the outer loop, while the light-gray boxes indicate which operations are performed in the inner loop.

The flow starts with a layout of the module under consideration. From this layout, a transistor-level netlist is extracted, which may include parasitic elements such as resistances, capacitances, and diodes. With the help of an in-house tool called FAULTGEN, a list of potential defect locations for realistic shorts and opens is extracted from the layout on the basis of critical area analysis [9]. Both operations are only performed once.

Iteratively, the following procedure is executed. A simulation testbench is defined for the netlist, with as parameters the test stimuli, supply voltage, temperature, and clock frequency. A (Spice-like) analog simulation is performed on the defect-free (*golden*) transistor-level netlist, and the corresponding responses are stored in a database for later reference. Subsequently, for each defect location in the defect list, the defect injection operation creates a dedicated defective netlist, containing the original defect-free netlist augmented with exactly that defect with a certain specified resistive value. Hence, for a defect list of size d, we will create d different defective netlists. Analog simulations are performed on all defective netlists. The responses of these simulations are compared against the response database for the golden netlist. The defects of which the corresponding netlist simulation leads to a mismatch with the simulation of the golden netlist are considered as 'detected', while others are marked as 'undetected'.

In the inner loop, this procedure is repeated for a set of representative defect resistance values. The outer loop is entered only if the resulting defect coverage is unacceptably low. In that case, we try to improve the test suite by modifying the testbench parameters. Note that the outer loop requires re-simulation of the golden netlist.

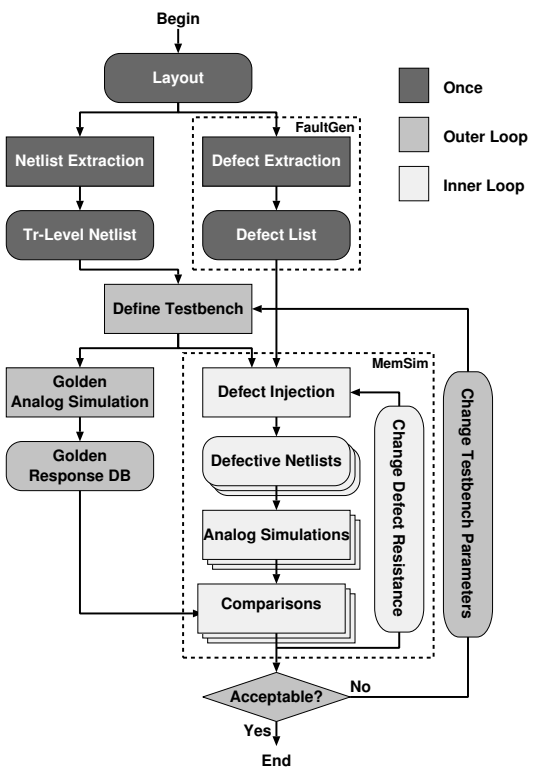

Figure 4: The defect-based analysis flow chart.

Due to its many analog simulations, this flow can be very expensive with respect to compute time. For a circuit with d potential defect locations, we run $d + 1$ analog simulations per iteration; on the golden fault-free netlist, and on the set of d netlists each containing exactly one defect. This inner loop is repeated for the number of distinct defect resistances r. A crucial aspect of our experiments is to obtain realistic and relevant results, while keeping the corresponding compute time within tractable limits. To this end, we have applied multiple techniques.

- *Selection of analog simulator*. We estimated that an accurate in-house analog simulator would need several hours of compute time for one golden simulation. As this was considered too long, we switched to Synopsys' simulator HSIM. HSIM is a hierarchical simulator, capable of exploiting regularities in the circuit and avoiding duplication of simulations. With HSIM we obtained a compute time of less than 10 minutes per complete simulation run.
- *Parallel computation*. Our tool MEMSIM supports the distribution of simulations over multiple compute servers. The number of simultaneous simulations was mostly limited by the number of HSIM licenses available at any given time.
- *Abort-on-first-fail*. The simulations of the defective netlists are compared on a pattern-by-pattern basis with the 'golden' simulation, and aborted as soon as a mismatch is detected. As many defects are detected somewhere halfway the simulation run, apply abort-on-first-fail reduces the average simulation time quite effectively.
- *Selection of simulation instance*. We used the smallest FIFO instance of 16 words $\times$ 19 bits for our analog simulations. This transistor-level netlist contains around 3,500 transistors and 8,000 parasitic capacitances.
- *Windowing*. In order to reduce the number of defect locations d, we exploited the regular character of the FIFO layout by considering only defects in a small window. For the bit-cell matrix of the FIFO, we considered all possible shorts and opens within a single static latch, as well as all possible shorts between this latch and its direct neighbor latches. For the asynchronous control of the FIFO, we considered all possible shorts and opens within a single control slice and all possible shorts between this control slice and its direct neighbors.
- *Small set of resistances*. In order to reduce the number of distinct defect resistances r, only a small number of defect resistances were considered, while still covering the entire range from 'hard' to 'weak' shorts and opens. For shorts, we considered $r = 5$ with defect resistance values of 100 Ω ('hard' short), 1 kΩ, 10 kΩ, 50 kΩ, and 100 kΩ ('weak' short). While focusing on low-ohmic shorts, we included 50 kΩ and 100 kΩ, as Montañés et al. showed in [10] that high-ohmic shorts of 20 kΩ and more do occur in deep sub-micron designs. In [11], Montañés et al. showed that deep sub-micron designs can contain opens and that the resistive value of these opens shows a wide spread. Therefore, we considered for opens $r = 3$ with 1 GΩ ('hard' open), 1 MΩ, and 10 kΩ ('weak' open).
- *Implied detection*. If for a defect location a weak defect is detected, it is implied that all harder defects will be detected as well. This can be exploited by simulating weak defects first, and treating all detected weak defects also as 'detected' for harder resistive values, i.e., without actually simulating them. For shorts, this implies that we iterate from high to low resistive values, while for opens, we go from low to high resistive values.

4 Initial FIFO Test

For an $n \times m$-bit FIFO (with n even), our initial INTEST procedure [4] consisted of three steps as listed in Table 1, to be executed at nominal frequency, temperature, and supply voltage (= 1.2 Volts).

Step	Operation	#ops
1	*Reset*	1
2	*Write* $(00\ldots0;11\ldots1)^{n/2};00\ldots0$	$n+1$
3	*Shift-Out* $(00\ldots0;11\ldots1)^{n/2}$	n

Table 1: Initial FIFO INTEST procedure.

Step 1 resets the FIFO by applying `reset`$=1$ for one clock cycle, thereby effectively flushing the FIFO's contents.

Step 2 writes an alternating sequence of $00\ldots0$ and $11\ldots1$ words into the FIFO. The net result is that the FIFO's memory matrix is filled with a physical *checkerboard* pattern, as in the dual-rail encoded FIFO layout every word consists of an alternating sequence of bit and bit-not lines. This test checks correct writing at different fill levels of the FIFO, and aims to detect single-cell stuck-at, transition, and delay fauls, as well as coupling faults between neighboring bit-cells. For an n-word FIFO, we try to write $n+1$ words, in order to check whether the last write operation is refused by the then full FIFO.

All response observation of this test takes place in Step 3. The FIFO contents is read out by shifting. This exercises some of the worst-case timing paths in the FIFO, as for every individual shift operation on a full FIFO a hole needs to be propagated from the read interface to the write interface through the entire FIFO [4].

5 Defect-Based Analysis Results

In this section, we describe the analysis results for intra- and inter-cell defects within the largest layout block, i.e., the bit-cell matrix. The same approach has been used on the other layout blocks, but due to lack of space, detailed results are omitted here.

5.1 Resistive Shorts in the Bit-Cell Matrix

Resistive short defects, in which two or more signal lines are shorted, are the most common defect type. In the bit-cell matrix, they may cause among other stuck-at, transition, delay, or coupling faults. Column 2 of Table 3 shows the results of the defect-based analysis of the initial INTEST procedure. Even in the case of hard (= 100 Ω) shorts, one out of 18 potential defects was not detected, leading to an unacceptably low defect coverage of 94.4%. For weaker shorts, the defect coverage was even lower.

Simple faults localized to one bit-cell, such as stuck-at and transition faults, are easily detected by our INTEST procedure. The alternating bit values that pass through the bit-cells during Steps 2 and 3 will be disturbed by the stuck-at or transition fault, and hence detected upon read-out. Analysis showed that indeed all stuck-at and transition faults were detected by the initial INTEST procedure.

The detection of coupling faults is more complex. The values ('data background') in physically adjacent neighbor cells determine whether or not a coupling fault is sensitized and detected. The *'all-zero'* and *'all-one'* words written in Step 2 of the initial INTEST procedure lead to a physical *checkerboard* pattern, given the fact that in the bit-cell matrix layout words are laid out as an alternating sequence of bit and bit-not wires. Our analysis showed that this was insufficient to detect all coupling faults, as one hard short defect led to a coupling fault that escaped the initial INTEST procedure.

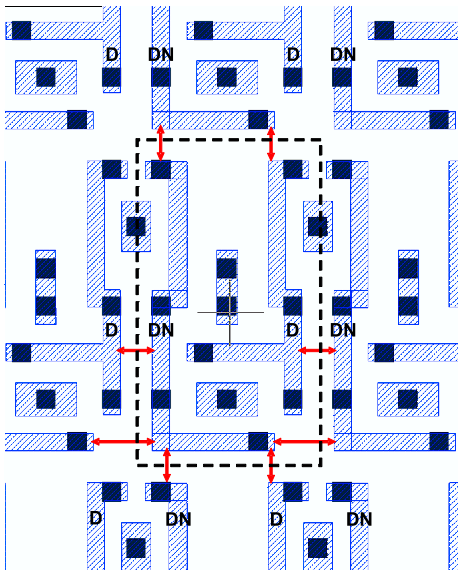

Figure 5: All possible shorts between a bit-cell and its neighbours in the Metal 1 layer (words are horizontal).

The missed coupling fault is caused by a short between two neighboring bit-not wires within one word. Opposite from what is the case in the poly-silicon layer, neighboring bit-not wires are adjacent in the Metal 1 layer. Figure 5 shows all potential shorts between a cell and its neighbors in the Metal 1 layer. At the bottom right-hand of the dashed box that indicates the bit-cell, the potential coupling between two bit-not wires of neighboring cells is shown.

In order to detect this missing fault, a second test was devised, equal to the initial INTEST procedure, but with a different data background. With this new data background, neighboring bits within one word get different logic values, due to which also neighboring bit-not wires get different values, and hence the missing coupling fault can be detected. The new test consisted of a repetition of the initial test, with two different data backgrounds, and is listed in Table 2.

Step	Operation	#ops
1	*Reset*	1
2	*Write* $(00\ldots0;11\ldots1)^{n/2};00\ldots0$	$n+1$
3	*Shift-Out* $(00\ldots0;11\ldots1)^{n/2}$	n
4	*Reset*	1
5	*Write* $(01\ldots0;10\ldots1)^{n/2};01\ldots0$	$n+1$
6	*Shift-Out* $(01\ldots0;10\ldots1)^{n/2}$	n

Table 2: Modified FIFO INTEST procedure.

Column 3 of Table 3 shows the resulting, improved defect coverage for the modified INTEST procedure which applies two different data backgrounds. Good news is that we obtained complete detection of all hard shorts of 100 Ω and 1 kΩ , and that also detection of weaker shorts of 10 kΩ and 50 kΩ has improved. However, the detection of weak shorts still requires improvement, especially since weak shorts

of 100 kΩ are not detected at all yet.

To improve the defect coverage of highly resistive shorts, we applied our defect-based analysis with different stress conditions for supply voltage, temperature, and test frequency. We found out that especially low-voltage testing increases the defect coverage of highly resistive shorts. Lowering the supply voltage from (nominal) 1.2 Volts down to 0.9 Volts increased the detection of weak shorts of 50 kΩ from 50% to 83% and for 100 kΩ even from 0% to 62%. Hence, the final test advised for this FIFO for detecting resistive shorts consists of the modified INTEST procedure with two data backgrounds, run at 0.9 Volts. Column 4 of Table 3 shows the defect coverage improvements between the three tests.

Short Resistance	Relative Defect Coverage		
	1.2 Volts		0.9 Volts
	One DBG	Two DBGs	Two DBGs
100 Ω	94.4%	100.0%	100%
1 kΩ	94.4%	100.0%	100%
10 kΩ	83.3%	88.9%	88.9%
50 kΩ	44.4%	50.0%	83.0%
100 kΩ	0.0%	0.0%	62.0%

Table 3: Relative defect coverage for resistive shorts with one (00-11) or two (00-11 + 01-10) data-backgrounds at nominal and low voltage.

5.2 Resistive Opens in the Bit-Cell Matrix

Our defect extraction procedure identified 27 possible open defects in each bit-cell. We considered open defects caused by line breaks, broken contacts, and broken vias.

Most of the 1 GΩ and 1 MΩ opens are detected by the initial INTEST. The relative defect coverage achieved for the investigated resistances is shown in Column 2 of Table 5.

Resistive opens cause an increased delay in the affected nets. Since the asynchronous FIFO pipeline is self-timed, there are time constraints on data transfers between bit-cells that may be violated by an increased wire delay. Because of the asynchronous nature of the FIFO, changing the external write frequency does not influence these time constraints. To detect violations of the time constraints, it is sufficient to use an alternating data background. If a data transfer from one cell to another is delayed long enough, old data will be latched. Because of the alternating sequence of words the old data will be different from the data that should have been latched.

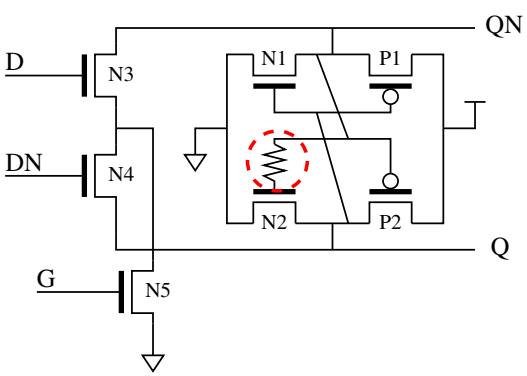

Figure 6: Schematic of a bit-cell with a resistive open at one of the NMOS transistor gates of the cross-coupled inverters.

Two hard 1 GΩ opens in the gates of the NMOS transistors in the cross-coupled inverters of the bit-cell were not detected by the initial INTEST. The defect location for one of these opens is shown in Figure 6. A hard open at the gate of a transistor substantially limits the current that can flow to and from the gate, but it is still possible to charge the gate capacitance through the open. The gate will also be very sensitive to capacitive coupling.

The voltage level on the gate in the inverter NMOS can be changed by writing the cell and then waiting until the gate capacitance has been charged, through the open, to the voltage level of the data line it is connected to. This takes a long time compared to the regular write and read speed. If the gate behind the open is initialized to a logic one it will stay at this level even when the data in the cell changes, if the changes are fast. This situation can be seen in the first part of Figure 7. The top graph shows the voltage on a bit-cell data net when the FIFO is filled with alternating data. The switching of the data comes from the alternating words that ripple through the bit-cell. The bottom graph shows the voltage on the gate behind the open resistance. As shown in the figure, the gate voltage level is affected by capacitive coupling and the gate is slowly charged when the data line is written to one. But neither of these effects are strong enough to significantly change the gate voltage. At the moment when all writes have been performed and the bit-cell latches the data, the gate voltage is still so low that the NMOS transistor is turned off although it should be on. This is not detected at this point because data is always written to a bit-cell by forcing a data net to ground through the write transistors. Hence, the NMOS transistors in the bit-cell inverters are never used to change the data in the cell, they are only used to keep the data stable. Even if the inverter NMOS transistor is turned off, the cell will still keep its data. If instead the NMOS gate voltage is close to the supply voltage, the transistor will be turned on. If the data net connected to the gate is then written to a logic zero the gate voltage will still be high because of the delay effect of the open. In this situation both the NMOS transistor and the PMOS transistor of the bit-cell will be conducting at the same time. Since the NMOS transistor is stronger than the PMOS transistor, the NMOS transistor will force the inverter output down to a logic zero although it should be a logic one. This creates a detectable fault.

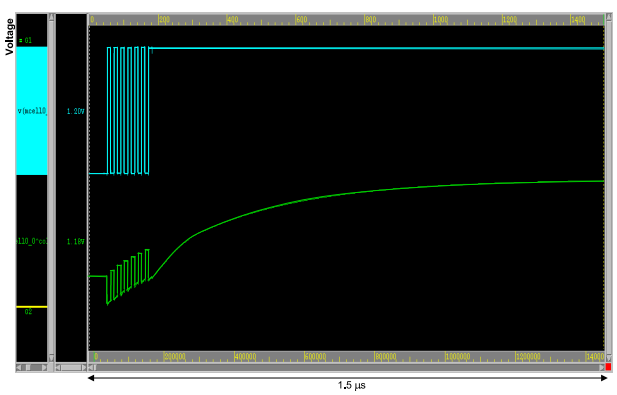

Figure 7: Voltage on the data line (upper curve) and on the broken gate connected to the same data line by a 1 GΩ resistance (lower curve).

One possible way of detecting the NMOS gate opens in the cross-coupled inverters of the cell is to add two waiting periods to the initial INTEST before the data is read out. The proposed test is presented in

Table 4. First, the FIFO is filled with the $(00\ldots0;\ 11\ldots1)^{n/2}$ data background. Then there is the first waiting period. After this waiting period half of the NMOS inverter gates will be written to a logic one. According to simulations, the waiting time should be at least 2 μs. The next step is to scan out one word of data. When one word is scanned out, all words in the FIFO move one stage forward in the pipeline. Because of the alternating data background this means that the opposite data will be written into the cells. This enables the detection of the open defect for half of the NMOS transistors in a cell. To detect them for the other half, there needs to be a second wait and a second scan of data. The drawback of this test procedure is that some defect coverage may be lost for the first words since the data that is scanned into the FIFO cannot be controlled by the proposed DfT hardware in the general case. It is possible to detect these opens as well, but that would require an extension of the current DfT-hardware.

Step	Operation	#ops
1	Reset	1
2	Write $(00\ldots0;11\ldots1)^{n/2}$	n
3	Wait $2\mu s$	1
4	Shift-Out $(00\ldots0)$;	1
5	Wait $2\mu s$	1
6	Shift-Out $(11\ldots1)$;	1
7	Shift-Out $(00\ldots0;11\ldots1)^{n/2}$	n

Table 4: FIFO bit-cell inverter gate open test procedure.

Table 5 shows the improvement achieved by the improved INTEST.

Open Resistance	Relative Defect Coverage	
	Initial	Improved
1 GΩ	92.0%	100.0%
1 MΩ	92.0%	92.0%
10 kΩ	18.0%	18.0%

Table 5: Relative defect coverage for resistive opens with the original and improved INTEST.

6 Conclusion

Many IC designs use numerous embedded FIFO memories for intermediate storage, data rate conversion, and clock domain crossing. The usage of embedded FIFOs is expected to grow, as new design paradigms such as Network-on-Chip (NOC) and Globally-Asynchronous Locally-Synchronous (GALS) use FIFOs extensively. NXP Semiconductors has developed a new embedded asynchronous FIFO module, based on a micropipeline architecture. Due to its full-custom design, the new FIFO is substantially smaller and faster than SRAM-based and standard-cell-based counterparts.

In this paper, we described how we have investigated and improved the defect detection qualities of our initial test procedure for resistive shorts and opens in the bit-cell matrix of the new FIFO.

By including an additional data background we managed to reach 100% defect coverage for 100 Ω and 1 kΩ shorts. The defect coverage of higher resistance shorts was increased substantially by including low-voltage testing. The probability of occurence of high-resistant shorts is a lot smaller than it is for low-resistant shorts, so achieving 100% defect coverage here is not as important.

We found 1 GΩ opens in the gates of the NMOS transistors of the cross-coupled inverters in the bit-cell that were not detected by the initial test and we were able to create a test procedure with two waiting periods to detect them.

Acknowledgements

The authors thank Michel Altheimer of NXP Semiconductors in Sophia-Antipolis, France for working with us on the design, layout, and test strategy for the embedded FIFO module. We thank Erik van Geest of NXP Semiconductors and Ananta Majhi and Bart Vermeulen of NXP Research in Eindhoven, The Netherlands for constructive criticism on an early draft of this paper.

References

[1] Jens Muttersbach, Thomas Villiger, and Wolfgang Fichtner. Practical design of globally-asynchronous locally-synchronous systems. In *Proceedings Intnl. Symposium on Advanced Research in Asynchronous Circuits and Systems (ASYNC)*, pages 52–59, April 2000.

[2] Kees Goossens, John Dielissen, and Andrei Rădulescu. The Æthereal network on chip: Concepts, architectures, and implementations. *IEEE Design & Test of Computers*, 22(5):21–31, September/October 2005.

[3] Ivan E. Sutherland. Micropipelines. *Communications of the ACM*, 32(6):720–738, June 1989.

[4] Paul Wielage, Erik Jan Marinissen, Michel Altheimer, and Clemens Wouters. Design and DfT of a High-Speed Area-Efficient Embedded Asynchronous FIFO. In *Proceedings Design, Automation, and Test in Europe (DATE)*, Nice, France, April 2007.

[5] Mohamed Azimane and Ananta K. Majhi. New Test Methodology for Resistive Open Defect Detection in Memory Address Decoders. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 123–128, April 2004.

[6] Mohamed Azimane et al. A New Algorithm for Dynamic Fault Detection in RAMs. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 177–182, May 2005.

[7] Ananta K. Majhi et al. Memory Testing Under Different Stress Conditions: An Industrial Evaluation. In *Proceedings Design, Automation, and Test in Europe (DATE)*, pages 438–443, March 2005.

[8] J.P. Shen, W. Maly, and F.J. Ferguson. Inductive Fault Analysis of MOS Integrated Circuits. *IEEE Design & Test of Computers*, 2(6):13–26, December 1985.

[9] Jose Pineda de Gyvez. IC Defect Sensitivity for Footprint-Type Spot Defects. *IEEE Transactions on Computer-Aided Design*, 11(5):638–658, May 1992.

[10] Rosa Rodrígues Montañés, Eric Bruls, and Joan Figueras. Bridging Defects Resistance Measurements in a CMOS Process. In *Proceedings IEEE International Test Conference (ITC)*, pages 892–899, Baltimore, MD, USA, September 1992.

[11] Rosa Rodrígues Montañés, José Pineda de Gyvez, and Paul Volf. Resistance Characterization for Weak Open Defects. *IEEE Design & Test of Computers*, 19(5):18–26, September/October 2002.