Online Self-Repair of FIR Filters

Alfredo Benso, Stefano Di Carlo, Giorgio Di Natale, and Paolo Prinetto
Politecnico di Torino

Editor's note:
Chip-level failure detection has been a target of research for some time, but today's very deep-submicron technology is forcing such research to move beyond detection. Repair, especially self-repair, has become very important for containing the susceptibility of today's chips. This article introduces a self-repair solution for the digital FIR filter, one of the key blocks used in DSPs.
—Yervant Zorian, Virage Logic

The DSP market has achieved astonishing growth in the past few years, and no single vendor seems to profit at the expense of another. Driving this growth is the trend of increasingly more analog-based products to move to digital technologies. Examples include TV and audio ICs; serial- and optical-communication ICs; the Global Positioning System; cellular and Personal Communication Service (PCS) ICs; and multimedia ICs and modules.

In fact, more than 80 companies use digital signal processing in their chips but do not sell them as DSP chips. The market is even bigger than the revenue for traditional DSPs and embraces a group of relatively big DSP vendors, such as Siemens, Rockwell, and Zilog. Will Strauss, president of Forward Concepts, a market research firm in Tempe, Arizona,[1] predicts that the DSP market will grow by a compounded rate of 36% per year over the next five years.

Digital filters are perhaps the most widely implemented class of DSP applications because they are basic building blocks of many complex systems. To enable a filtering process at high bandwidth, designers use specialized hardware that can operate at much higher throughputs than are possible with general-purpose DSPs. This hardware includes ASICs, which allow hardware optimization of certain popular signal-processing algorithms or functions at the cost of flexibility. Comparing ASICs with DSP microprocessors, it's clear that DSPs offer slower speed but maximum flexibility (due to programmability), whereas ASICs provide higher speed with minimal flexibility.

With commercial products incorporating DSP functions and the market's increasing quality requirements, traditional architectures are becoming inadequate. Pressing issues among DSP designers include new design approaches to reduce time to market, as well as new architectures allowing high testability, reliability, and programmability without affecting performance.

Although researchers have proposed several architectures to reach optimal filter performance,[2-4] no one has adequately addressed the problem of designing self-repairing digital filters with programmability characteristics. Self-repairing technology could enrich commercial applications requiring high availability and serviceability. It could also benefit space or defense applications that must survive and perform at optimal functionality for long durations in unknown, harsh, and possibly changing environments.

In that light, we present a self-testable, self-repairable architecture for finite impulse response (FIR) digital filters. This architecture allows repair of permanent faults without interfering with the filter behavior or introducing performance overheads. Although we developed this approach to repair a single permanent fault, you can easily scale it to repair any number of faults occurring in the filter logic. A key feature of the architecture is its modularity, which allows automatic generation—that is, reduced design time. We've implemented a tool for automatic filter synthesis to generate VHDL descriptions of FIR filters, starting from functional parameters.

Filter architecture

FIR filters are perhaps the most widely implemented class of digital filters. Until recently, engineers designed digital filters using a recipe directly from signal theory. Starting from the transfer function, they could easily deploy filters by following the signal flow and adding glue logic and delay blocks. Although this time-consuming recipe can lead to various designs for the same parameters and function, it worked well when designs used only a few digital filters. Today, however, it is absolutely incompatible with the emerging market's constraints and reliability exigencies.

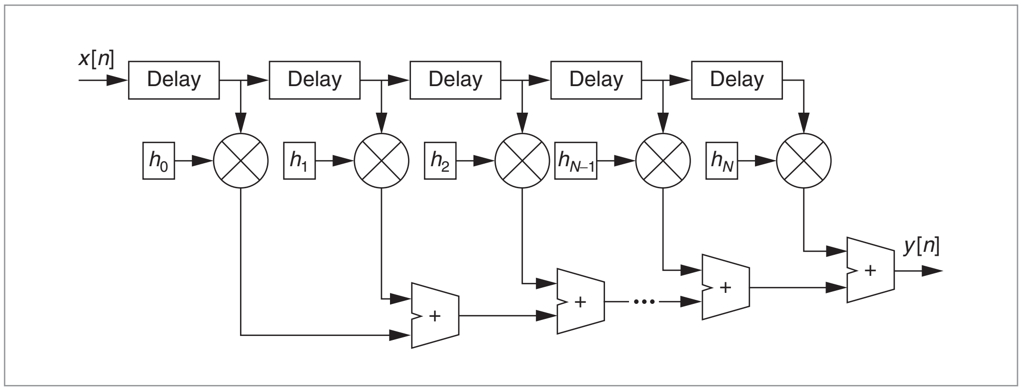

Figure 1. General FIR filter layout.

FIR filters essentially perform a moving, weighted average of a sequence of input samples, as the following equation indicates:

$$y[n] = \sum_{i=0}^{N} h_i \cdot x[n-i]$$

where N is the filter order; $y[n]$ is the output signal and $x[n]$ is the input signal, at time n; and h_i $(0 \le i \le N)$ is the set of filter coefficients, which also corresponds to the filter's impulse response. This representation is easy to implement with a modular circuit, including a weighted delay line, as Figure 1 shows.

The delay line is usually a chain of registers working in a pipeline. In an N-order filter, this sample-in-bus pipeline contains $N + 1$ delay elements. This means processing each output sample requires a filtering window of $N + 1$ input samples. The input samples shift every time a new sample is ready. A set of multipliers then multiplies these samples by the relative h_i coefficient, and a cascade of adders (see the lower part of Figure 1) adds the results to generate the output samples.

The downside of the architecture in Figure 1 is its use of multipliers, which are very costly blocks with massive footprints. However, multipliers are critical for reliability, which decreases quadratically with area. Many researchers are striving to reduce the hardware complexity of FIR filters to alleviate this problem. One example is signed-power-of-two algebra, which allows multiplication using only shift-and-add operations.[5-7] SPT algebra expresses numbers as sums and differences of negative powers of two, often called SPT terms. In general, many equivalent SPT representations, with a different number of SPT terms, exist for a single number. To minimize the number of SPT terms, we opted for the canonic signed digit (CSD) form, which is the only unique-minimum SPT representation.[8]

Despite its advantages, SPT algebra can introduce a loss of precision in the representation of coefficients and consequently in the filter's output. The first rule of thumb we could infer from physics is that the coefficients' precision must be at least equal to the desired output precision. But in filter design, this might not be enough.

Bellomo demonstrated that a simple coefficient truncation close to the desired precision does not produce the desired precision in the output.[9] Therefore, he coded an implementation of the Trellis search algorithm to choose the best approximating coefficients under precision constraints.[10] We use this algorithm to obtain the SPT terms in our experiments.

For example, we express coefficient $h_i = 0.0001$ with precision 10^6 as

$$h_i = 2^{13} - 2^{15} + 2^{17} + 2^{20}$$

The multiplication of h_i for generic input sample $x[n]$ becomes

$$\begin{aligned}(h_i)x[n] &= (2^{13} - 2^{15} + 2^{17} + 2^{20})x[n] \\ &= 2^{13}x[n] - 2^{15}x[n] + 2^{17}x[n] + 2^{20}x[n]\end{aligned}$$

Adders and shifters are necessary to implement this function. For each SPT term, a programmable shifter moves $x[n]$ to the right by the number of positions represented by the term's exponent, and then adds the shifted samples together. We call the basis block that performs these operations the multiply-accumulate

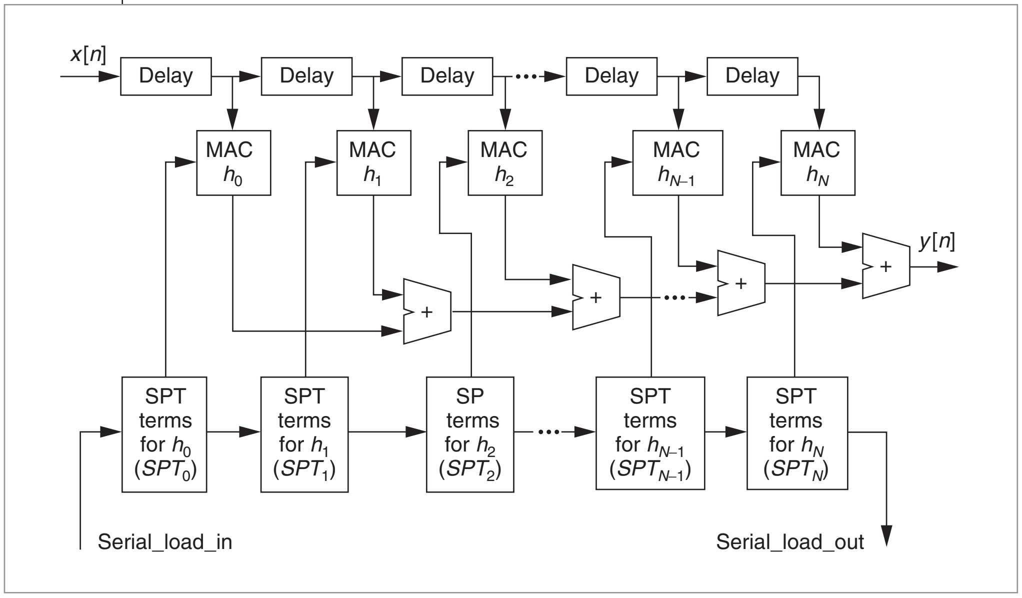

Figure 2. Multiply-accumulate (MAC) cell.

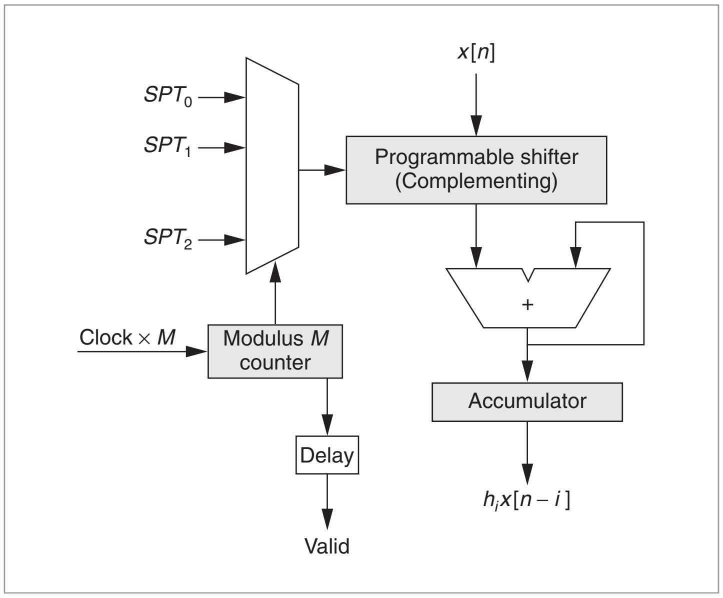

Figure 3. Modular filter architecture.

(MAC) cell, shown in Figure 2. The MAC cell is the basic building block of our modular architecture, which Figure 3 shows.

The MAC cell's behavior is easy to explain. Each MAC cell is fed by the input samples, and programmed by a set of SPT terms stored in external registers. Each set of SPT terms represents a given h_i coefficient. The MAC cell processes an input sample M times (where M is the number of SPT terms), and each time shifts it a number of positions equal to the related SPT value. The programmable shifter dynamically shifts the sample and, if necessary, takes the 2s complement of this term based on its sign. The adder adds it to the value stored in the accumulator. This operation requires M clock cycles; therefore, the entire block is overclocked by M cycles.

To allow flexibility and programmability, we can serially load the SPT terms from an outside source. This solution lets the user change the filter characteristics. Obviously, the SPT register size limits the precision of the SPT terms' representation.

A cascade of adders adds the MAC cell output values together to obtain the output signal. Because the number of SPT terms is not necessarily the same in each MAC cell, a delay network synchronizes the operation among the MAC cells.

Now, consider the cascade of adders in the lower part of Figure 3. Because of the long path between the first and last adder, the characteristics of these components heavily influence the filter's performance. The main constraints are low area and high speed.

Sklansky's topology seems to be the best choice in terms of complexity—it has complexity $(n/2)\log_2 n$ (where n is the adder parallelism)—and total delay, $\log_2 n$.[11]

Test strategies

Our work aims to obtain high testability and reliability toward hard and soft errors. We propose two solutions, each covering different faults at different stages of the filter's operating life:

- *Power-on self-test (POST).* This solution, which the system's power-on enables, mainly serves to detect permanent faults that affect the filter's logic.
- *Online self-repair.* The filter can execute self-tests concurrent with its normal behavior and when necessary to replace faulty blocks.

Power-on self-test

POST is an offline test strategy, usually enabled at the

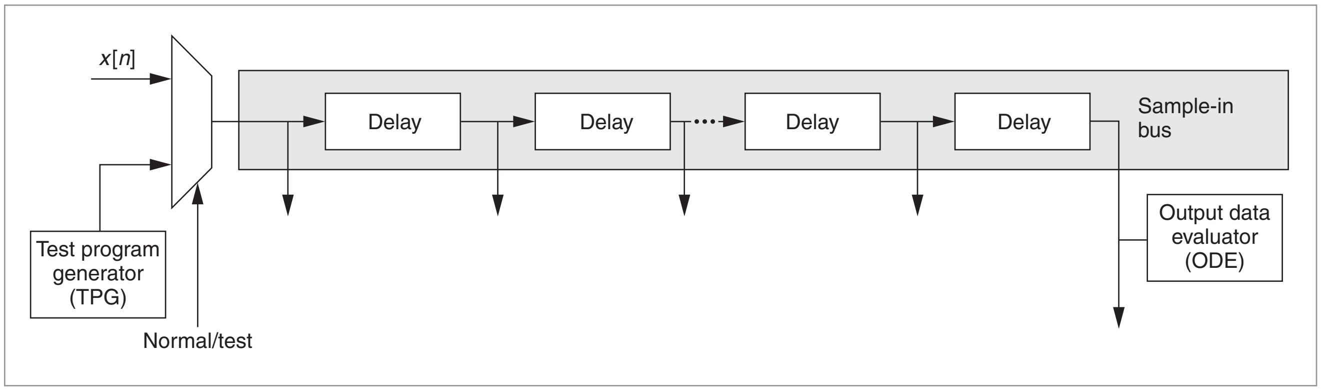

Figure 4. Sample-in bus test.

system's power-on to detect permanent faults. The idea is to give the filter a set of ad-hoc test samples, let the filter work on these samples, and compare the results with previously computed results.

Based on the general layout in Figure 3, we identify three categories of components to test:

- the sample-in bus,
- the MAC cells, and
- the cascade of adders.

Figure 4 shows the sample-in bus' structure. We must carefully test both registers and interconnections to avoid stuck-at or faulty connections, which could greatly influence system performance.

In normal mode, the sample-in bus streams the digital samples for processing. In test mode, a set of test patterns, called the mini bus test, feeds this bus. The test pattern generator applies a single sample to the filter and after N clock cycles (where N is the number of registers in the sample-in bus), the output data evaluator (ODE) observes the same pattern at the chain's output. If the ODE reads a different word, the bus is faulty.

The fault coverage depends on the test patterns applied to the sample-in bus. Detecting stuck-at faults requires only two test patterns (000 … 0 and 111 … 1), but detecting couplings or shorts between bus lines requires additional patterns. Background patterns, usually applied during memory testing, provide a good tradeoff between coverage and the number of test patterns.[12] The architecture can also accommodate custom test patterns if the designer has particular reliability requirements. If the block passes the mini bus test, the architecture considers both registers and interconnections in the sample-in bus as fault free.

Now consider the MAC cell in Figure 2. A set of registers program the MAC cell to store the SPT terms. The MAC cell receives input patterns from the sample-in bus, which the mini bus test feeds. During POST, the system modifies the SPT registers to work as a linear-feedback shift register (LFSR) to provide test patterns to the MAC cell. The architecture verifies the absence of faults by transforming the accumulator to work as a multiple-input signature register (MISR) and by checking the final signature.

Concerning timing, the sample-in bus produces test patterns according to the normal system clock, whereas the MAC cell is overclocked by M cycles and receives test patterns from the SPT inputs at this high frequency. The combination of high-speed patterns from the SPT inputs and low-speed patterns from the sample-in bus provides high fault coverage.

Now we address the problem of testing the chain of adders in the lower part of Figure 3. The idea is to exploit the entropy produced by the MAC cells during POST to produce test patterns for the adders. Every adder connects to two different registers, each coming from a different block. POST configures these registers as MISRs so that their content changes continually and pseudorandomly.

This approach has a drawback, however: If every MAC cell's POST begins simultaneously, the entropy on the adders will be insufficient to ensure high fault coverage, because the adders will see couples of similar operands. The solution is conceptually very simple, although its implementation requires more complexity overhead: In enabling each MAC block's POST, we place a 1-clock pulse delay between that block and the previous block.

The system finally signs the adder chain's output using a MISR to detect faults. A controller governs POST

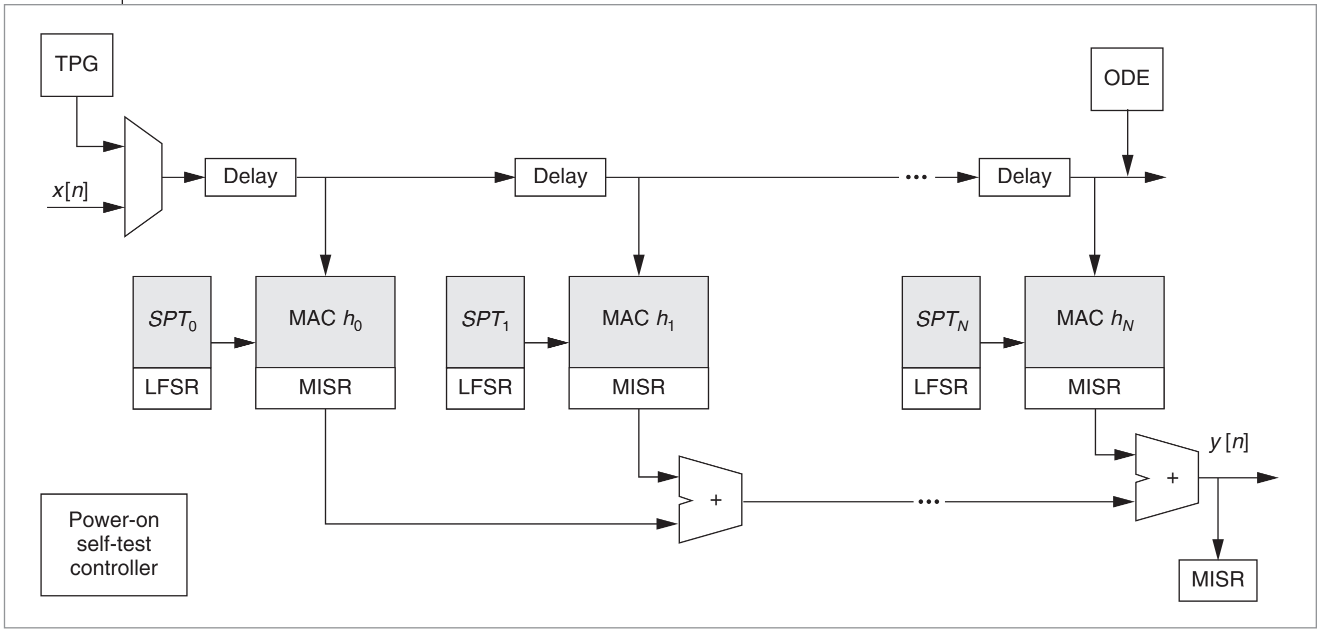

Figure 5. General power-on self-test (POST) architecture.

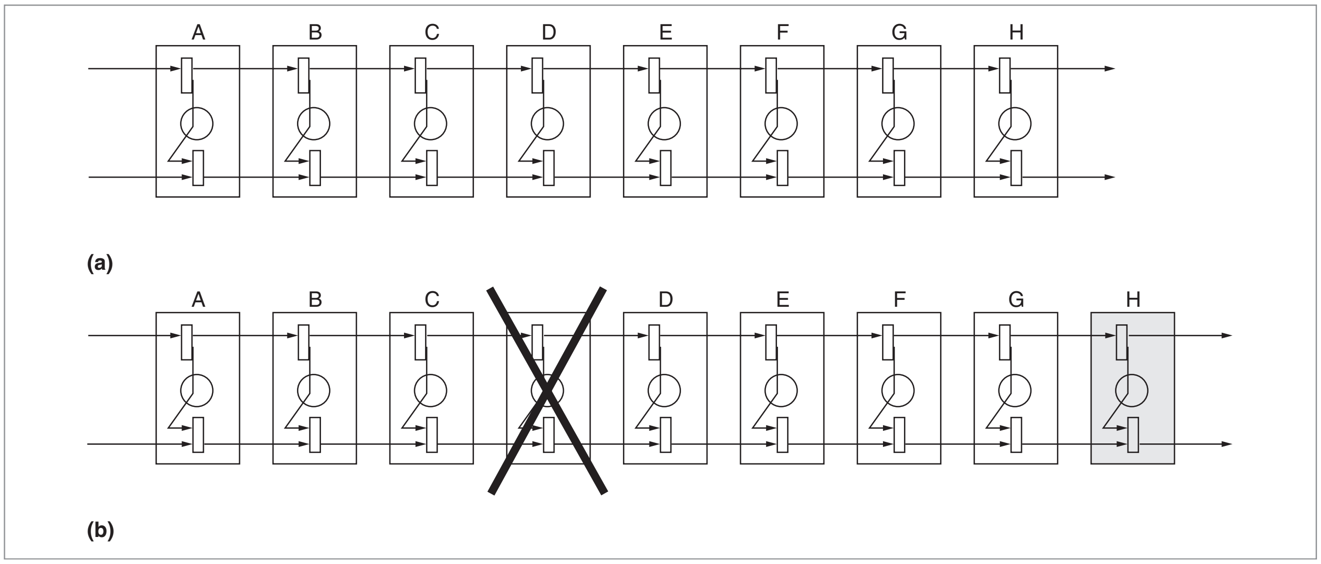

Figure 6. Filter before (a) and after (b) the repair process. The shaded area is the spare module.

procedures by synchronizing the test structures and checking the test results. Figure 5 shows the general architecture for the POST.

Online self-test and self-repair

The filter's modular structure is perfectly suited to implement efficient online BIST and self-repair strategies. A basic filter module (BFM) is the union of a sample register with its MAC cell and its final adder. Our approach is to introduce one or more spare BFMs into the architecture. During normal behavior, a spare BFM periodically replaces each module. The system tests each module using an approach similar to (and reusing the same structures as) the POST mechanism just described.

If the test detects no faults, the system reintroduces the module and selects the next one for test. If the test fails, the BFM is faulty, and the system interrupts the replacement mechanism. Thereafter, the system can no longer detect and correct the occurrence of any other fault online, but it can still work without degradation.

Figure 6 shows the fault-free filter structure and the repaired structure, in which the spare module shifts the pipeline's functionality and acts as a substitute for the faulty module.

Implementing this repair scheme needs the introduction of alternative routing paths to exclude the cell under test from the chain without introducing any delay in the filtering process. Switching devices manage the alternative routing-path mechanism.

Figure 7 shows the filter's new layout, where pairs of multiplexers ensure the correct input at every stage of the sample-in bus. After the repair operation, the system asserts an output signal to inform the user that the output values can be temporarily unreliable and that, if an error occurs again, the system will not be able to repair the chip.

Bypassing a faulty BFM is not enough to completely repair the filter. In fact, the blocks are all equal, but they receive different SPT terms from the SPT registers. The repair process must remap the SPT terms in the new configuration. A switching element distributes the SPT terms to different modules. The replacement mechanism requires programming each module with both its own SPT values and those of the previous module, thereby minimizing the area overhead introduced by the switching device.

Table 1 shows some routing scenarios. For the BFM_S test, the system programs each module with the related SPT_S terms (where S refers to the spare cell) and tests the additional module. For the BFM_0 test, routing is switched so that BFM_1 receives the SPT_0 terms, BFM_2 receives SPT_1 terms, and the additional BFM_S receives the SPT_N terms.

The downside of this strategy is that the switching elements and SPT registers are not repairable. However, these components represent only 2.4% of the total filter area, making the architecture's dependability level adequate for most of today's applications. If a higher reliability level is necessary and the area is available, it is possible to duplicate these elements to reach full repairability.

Experimental results

We analyzed our architecture's performance by implementing a fourth-order filter with 16-bit input samples. The filter's transfer function is

$$y_n = (0.1)x_n + (0.05)x_{n-1} + (0.07)x_{n-2} + (0.01)x_{n-3}$$

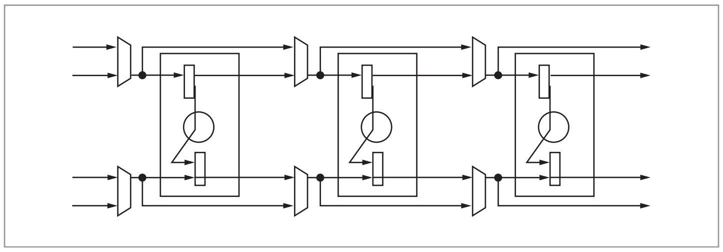

Figure 7. Alternative routing paths.

Table 1. Routing scenarios.

	Single-power-of-two terms received by				
Test of	***BFM_0***	***BFM_1***	***BFM_2***	***BFM_N***	***BFM_S***
BFM_S	SPT_0	SPT_1	SPT_2	SPT_N	Under test
BFM_0	Under test	SPT_0	SPT_1	SPT_{N-1}	SPT_N
BFM_1	SPT_0	Under test	SPT_1	SPT_{N-1}	SPT_N

The SPT terms representing the filter's coefficients are as follows:

$$0.1 = 2^{-3} - 2^{-5} + 2^{-7} - 2^{-9} + 2^{-11} - 2^{-14}$$

$$0.05 = 2^{-4} - 2^{-6} + 2^{-8} - 2^{-10} + 2^{12} - 2^{14}$$

$$0.07 = 2^{-4} + 2^{-7} - 2^{-12} - 2^{-14}$$

$$0.01 = 2^{-7} + 2^{-9} + 2^{-12}$$

This representation guarantees a precision of 10^{-4}.

To evaluate the area overhead, we described the filter in VHDL language, using Austriamicrosystems' csx_HDRLIB to synthesize it with Synopsys' design_compiler. Figure 8 shows the implementation. We implemented three different solutions in terms of dependability: no test, POST, and built-in self-repair (BISR). Table 2 lists the area (in Synopsys gate count values) for each of these solutions.

The resulting area overhead is 13% for POST and 33% for BISR. To evaluate the solutions' fault coverage, we used Synopsys' TetraMax for fault simulations. The BISR solution detected and repaired 97.2% of single stuck-at faults.

We tested the entire circuit in $(N + 6) \times M \times 6 \times 256$ fast clock pulses and $(N + 6) \times 6 \times 256$ clock pulses (where N is the filter order, and M is the number of SPT terms to represent a filter coefficient). For $N = 128$ coefficients with a 100-MHz clock, POST requires 205,824 clock pulses—that is, 2.05 ms.

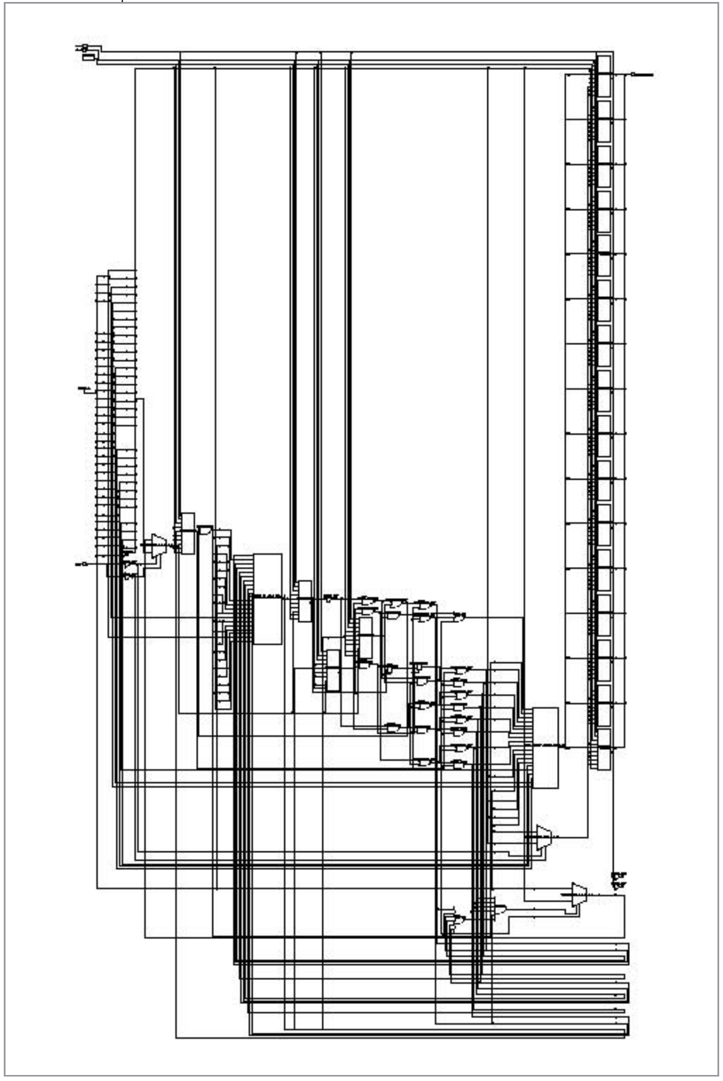

Figure 8. Fourth-order filter implementation used to analyze architecture performance.

Table 2. Area occupied by architecture for three different solutions.

Solution	Area (no. of gates)
No test	839,123
POST	713,698
BISR	900,908

FUTURE WORK will continue to apply and refine the design methodology presented here to achieve higher levels of testability and dependability. In particular, more work is necessary to identify hard-to-test areas in the circuit. Fault coverage higher than 99% is essential for mass production. In addition, new solutions should be exploited to solve the repairability shortcomings for switching elements and SPT registers, without strongly impacting the design area. ■

Acknowledgments

This work is partially supported by Instituto Superiore per le ICT Mario Boella under contract Test D.O.C.: Quality and Reliability of Complex SoC.

■ References

1. *Electronics Market Research*, Forward Concepts Co., Tempe, Ariz.; http://www.fwdconcepts.com.
2. L. Goodby and A. Orailoglu, "Redundancy and Testability in Digital Filter Datapaths," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 18, no. 5, May 1999, pp. 631-644.
3. C. Counil and G. Cambon, "A Functional BIST Approach for FIR Digital Filters," *Proc. IEEE VLSI Test Symp.* (VTS 92), IEEE CS Press, 1992, pp. 90-95.
4. C.-W. Wu and J.-C. Wang, "Testable Design of Bit-Level Systolic Block FIR Filters," *Proc. IEEE Int'l Symp. Circuits and Systems* (ISCAS 92), vol. 3, IEEE Press, 1992, pp. 1129-1132.
5. N. Benvenuto, L.E. Franks, and F.S. Hill Jr., "Dynamic Programming Methods for Designing FIR Filters Using Coefficients –1, 0 and +1," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 34, no. 4, Aug. 1986, pp. 785-792.
6. D. Li, J. Song, and Y.C. Lim, "A Polynomial Time Algorithm for Designing Digital Filters with Powers-of-Two Coefficients," *Proc. IEEE Int'l Symp. Circuits and Systems* (ISCAS 93), vol. 1, IEEE Press, 1993, pp. 84-87.
7. Y.C. Lim and S.R. Parker, "FIR Filter Design over a Discrete Powers-of-Two Coefficient Space," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 31, no. 3, June 1983, pp. 583-590.
8. R. Hartley, "Subexpression Sharing in Filters Using Canonic Signed Digit Multipliers," *IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing*, vol. 43, no. 10, Oct. 1996, pp. 677-688.
9. P. Bellomo, *Study of a Decimator for Sigma-Delta Conversion Suitable for Space Applications*, master's thesis, Dipartimento di Automatica e Informatica, Politecnico di Torino, Turin, Italy, 2000.
10. C.L. Chen and A.N. Wilson Jr., "A Trellis Search Algorithm for the Design of FIR Filters with Signed-Powers-of-Two Coefficients," *IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, no. 1, Jan. 1999, pp. 29-39.

11. J. Sklansky, "Conditional-Sum Addition Logic," *IRE Trans. Electronic Computers*, vol. 9, no. 2, June 1960, pp. 236-240.
12. A.J. Van de Goor, *Testing Semiconductor Memories: Theory and Practice*, Wiley, 1991.

Alfredo Benso is a researcher in the Department of Automation and Information Technology at Politecnico di Torino in Turin, Italy. His research interests include DFT techniques, BIST for complex digital systems, dependability analysis of computer-based systems, and software-implemented hardware fault tolerance. Benso has an MS in computer engineering and a PhD in information technologies, both from Politecnico di Torino. He chairs the IEEE Computer Society Test Technology Technical Council (TTTC) Web-Based Activities Group.

Stefano Di Carlo is a PhD candidate in the Department of Automation and Information Technology at Politecnico di Torino. His research interests include DFT techniques, SoC testing, BIST, and FPGA testing. Di Carlo has an MS in computer engineering. He is the chair of the TTTC's Electronic Submissions committee.

Giorgio Di Natale is a PhD candidate in the Department of Automation and Information Technology at Politecnico di Torino. His research interests include DFT techniques, BISR, and FPGA testing. Di Natale has an MS in computer engineering from Politecnico di Torino. He is an associate Webmaster of the TTTC.

Paolo Prinetto is a full professor of computer engineering at Politecnico di Torino and a joint professor at the University of Illinois at Chicago. His research interests include testing, test generation, BIST, and dependability. Prinetto has an MS in electronic engineering from Politecnico di Torino. He is a Golden Core Member of the IEEE Computer Society and the TTTC's chair-elect.

■ Direct questions and comments about this article to Stefano Di Carlo, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Turin, Italy; dicarlo@polito.it.

For further information on this or any other computing topic, visit our Digital Library at http://computer.org/publications/dlib.