Spectral Analysis for Statistical Response Compaction During Built-In Self-Testing

Omar Khan Michael L. Bushnell
okhan@caip.rutgers.edu bushnell@caip.rutgers.edu
ECE Dept. and CAIP Research Center, Rutgers University
96 Frelinghuysen Rd., Piscataway, NJ 08854-8088

Abstract

Spectral generation of patterns, to excite the natural frequencies of a digital circuit, is highly effective in testing sequential circuits. We have created a hardware embodiment of the spectral test-pattern generator for built-in self-test *(BIST). We present five new spectral response compactors SRC1-5 for BIST. Each analyzes the spectral content of circuit output responses, and accumulates their spectrum in one or more counters. The method has astonishing results. SRC1 never aliased for any faults in the ISCAS '89 benchmarks. SRC2, a low-overhead version of SRC1, aliased slightly more than the* multiple-input signature register *(MISR), but used less hardware than the MISR. This new spectral BIST system has a 91.26% shorter test sequence than for a conventional LFSR pattern generator and MISR system, with at least 8.42% higher fault coverage. The benefits of this are drastically shorter test sequences, the elimination of scan-shifting sequences, much lower test power dissipation, and higher fault coverage.*

1 Introduction

BIST is an on-chip testing system that generates test vectors to detect faults and verifies whether the hardware is performing correctly. The test control unit, test pattern generator, and response analyzer hardware are part of the chip being tested. BIST requires extra hardware, but it reduces testing cost compared to an external *automatic test equipment* (ATE), because:

1. There is an extremely high logic-to-pin ratio on the chip, making it very hard to observe the device signals through the pins.
2. The circuits are denser and operating at clock rates above 2.5 *GHz*. At-speed testing is difficult because no ATE operates above 1.6 GHz and external ATE is very expensive due to high tester pin costs.
3. Dense chips require longer pattern sets for high fault coverage, which increases testing time and cost.

We propose a BIST system with spectral pattern generation, where we treat the *circuit-under-test* (CUT) as a digital signal processor and use an orthogonal basis set of *digital signal processing* (DSP) vectors to stimulate and analyze the CUT's test responses. An arbitrary bit stream can be analyzed as the sum of digital spectral tones, each a row in an orthogonal Hadamard matrix. This procedure is called spectral analysis. We use a spectral sequential automatic test-pattern generator (described below) to generate test patterns. Then, we approximate these spectral test patterns using a BIST hardware pattern generator. We propose five novel response compactors that use DSP with the Hadamard matrix $H(1)$ to perform spectral analysis on the CUT's response to the test patterns. They calculate either the *auto-correlation* of testing responses at *primary outputs* (POs) with the two spectral tones, each a row in $H(1)$, or the *cross-correlation* between different POs. The response compactors store the correlation coefficients, i.e., the spectral content in terms of the tones in $H(1)$, in two counters, which represent the BIST *signature*.

The main contributions of this paper are the discovery of no aliasing in the SRC1 response compactor, and a theory for creating hardware efficient spectral response compactors. SRC2 has a slightly higher aliasing rate than the MISR, but uses less hardware. Spectral analysis based on *cross-correlation* between POs appears to have less aliasing and requires less hardware than response analysis based on *auto-correlation*. This is because the cross-correlator does not need to store the circuit output values during the immediately prior clock period. The reason why SRC2 uses less hardware than the MISR, particularly as circuits become large, is that SRC2 does not require a flip-flop on every PO, and the number of flip-flops required for SRC2 is proportional to $\log_2 \#POs$, rather than being equal to the $\# \ POs$, as with the MISR.

Section 2 discusses prior work, Section 3 presents the spectral model of response compaction, Section 4 presents five response compactors, Section 5 analyzes aliasing, Section 6 presents results, and Section 7 concludes.

0-7803-8580-2/04 $20.00 Copyright 2004 IEEE

2 Prior Work

2.1 Hadamard Transform and Properties

We define the fundamental properties of a *Hadamard matrix* and how it is used in BIST. A Hadamard matrix is a symmetric square matrix for DSP containing only -1's and 1's, such that when any two columns or rows are placed side by side, half of the adjacent cells are of the same sign, and the others have the opposite sign. Hadamard matrices are generated with this recurrence relation:

$$H(k) = \begin{bmatrix} H(k-1) & H(k-1) \\ H(k-1) & -H(k-1) \end{bmatrix}, \quad k = 1, 2, ..., n. \quad (1)$$

where $H(0) = 1$ and $n = \log_2 N$, where N is the total number of elements in the matrix. Each row or column of the Hadamard matrix represents a frequency component that spans the basis of the vector space. Figure 1 shows the columns of the Hadamard matrix as digital waves, each having a distinct frequency and phase shift.

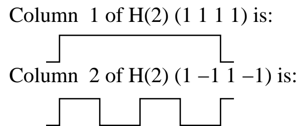

Figure 1: Wave Representation of Columns of $H(2)$ [6]

2.2 Compaction-Based ATPG

One testing technique is a combination of *linear reverse order restoration* (LROR) compaction and a sequence extending technique [14]. LROR is a *static compaction* scheme that eliminates useless subsequences in a sequential circuit test set without losing fault coverage. Vectors are removed from the test set, and the faults are simulated with the remaining tests. Only if the fault coverage drops, are the vectors replaced [14]. Sequences are extended by repeating perturbed input vectors or subsequences from the original compacted sequence. In *holding*, a randomly-selected vector is repeated a random number of clock cycles. Holding has greatly improved the fault coverage obtained by generating pseudo-random sequences [13].

Giani *et al.* [6] used the Hadamard transform to extend test sequences. Iterative processes of filtering via compaction and spectral analysis of the filtered test set are performed for each *primary input* (PI) bit stream. The extracted spectral characteristics of the compacted test-vector set [6, 7] are used to generate more vectors to extend the test set. Very high fault coverages are achieved with small vector sets for the *ISCAS '89* benchmark circuits.

They showed how the following 8-bit stream is extended [6]: $[1, 0, 1, 1, 1, 0, 1, 0]^T$. They replaced each 0 with a -1 and got: $[1, -1, 1, 1, 1, -1, 1, -1]^T$. Next, the bit stream was left multiplied by an 8×8 Hadamard matrix, $H(3)$, to obtain the spectral correlation coefficients:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 1 \\ 1 \\ 1 \\ -1 \\ 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 2 \\ 6 \\ -2 \\ 2 \\ 2 \\ -2 \\ -2 \\ 2 \end{bmatrix}$$

Every coefficient whose absolute value was less than 4 (the *cutoff*) was set to 0. After filtering, they got: $[0, 1, 0, 0, 0, 0, 0, 0]^T$. Left multiplying the new vector by $H(3)$ yielded: $[1, 0, 1, 0, 1, 0, 1, 0]^T$. In this extension, which is appended to the test sequence, the 4th bit in the original vector changed from 1 to 0.

Zhang *et al.* developed a sequential pattern generator [21] using a *selfish gene* algorithm with random bit perturbation. They randomly but periodically perturbed bits in the compacted test sequence and appended them to it. The results were comparable to Giani *et al.*'s [6], but with shorter test sets having the same fault coverage.

2.3 Pattern Generation Hardware

The *linear feedback shift register* (LFSR) is the most commonly used pseudo-random pattern generation method [5]. The LFSR is a modulo-2 autonomous linear sequential network, with unidirectional interconnections of D flip-flops and XOR gates. An n-stage maximum-length LFSR traverses $2^n - 1$ non-zero distinct states. A simple addition of a NOR gate enables it to generate the all-zero pattern, as well. Pseudo-random pattern generation requires more patterns than deterministic ATPG, but fewer than exhaustive test generation.

Weighted pseudo-random patterns are used to test hard-to-detect faults. In a pseudo-random test, each input has a probability of 0.5 of being 1. With hardware weights, we vary the input probabilities of generating $1's$, to improve the fault coverage and shorten the tests.

Hellebran *et al.* [10] proposed BIST using an embedded microprocessor. Giani *et al.* proposed testing sequential cores in a *system-on-a-chip* (SoC) [7], where they generate test patterns using a real-time program running on an embedded processor. This program uses Hadamard spectral coefficients, extracted from high fault-coverage compacted tests. Unfortunately, the method requires the addition of a microprocessor to the CUT.

Kasturirangan and Hsiao observed that some embedded cores in an SoC may not directly connect to the embedded test pattern generator, so they must be tested via other cores [11]. They computed spectral coefficients for each core. They analyzed the relationship between the output characteristics of one core and the input characteristics

Table 1: Test Length and Fault Coverage for Spectral BIST and LFSR Pattern Generators

Circuit	# PIs p	Hold Time	Spectral BIST [19] Fault Coverage	Spectral BIST [19] Test Length	LFSR Fault Coverage	LFSR Test Length 2^p-1
s208	11	32	7.31%	512	6.27%	2047
s298	3	32	81.5%	128	20.77%	10000
s344	9	8	84.2%	48	73.1%	511
s349	9	8	76.9%	160	73.3%	511
s382	3	64	84.5%	544	11.78%	10000
s386	7	8	43.5%	512	30.46%	127
s400	4	16	18.3%	128	14.1%	10000
s420	19	16	37.5%	1024	31.63%	10000[b]
s444	3	64	77.3%	640	10.34%	10000
s510	19	0	0%	0	0%	0
s526	3	64	69.2%	640	8.65%	10000
s641[a]	35	32	72.4%	2288	82.83%	10000[b]
s713[a]	35	16	60.1%	256	81.07%	10000[b]
s820[a]	18	16	16.8%	160	40.12%	10000[b]
s832	18	32	47.6%	516	39.15%	10000[b]
s838	35	32	4.40%	256	2.9%	10000[b]
s953	16	32	42.4%	320	6.34%	10000[b]
s1196[a]	14	16	37.8%	266	86.6%	10000[b]
s1238[a]	14	16	34.8%	290	85.5%	10000[b]
s1423[a]	17	32	44.8%	2560	37.23%	10000[b]
s1488	8	16	83.4%	256	45.3%	255
s1494	8	32	74.3%	144	48.03%	255
s5378	35	64	55.4%	576	44.76%	10000[b]
s9234	36	64	8.4%	2304	3.19%	10000[b]
s13207	62	64	10.7%	2048	1.06%	10000[b]
s15850	77	64	15.1%	304	3.68%	10000[b]
s35932	35	64	77.9%	512	54.71%	10000[b]
s38417	28	64	15.4%	4832	3.52%	10000[b]
s38584	12	128	31.1%	2048	7.16%	10000[b]
Average	20	39.14	46.7%	867	34.06%	7990

[a]Fault coverage of LFSR better than Spectral BIST.
[b]Test Length limited to only 10000 clock cycles.

of its successor. Similar characteristics indicate that the successor may be tested through the predecessor.

Upadhyayula implemented a new BIST method that generated test patterns for the CUT using the circuit spectral properties [19] as shown in Table 1. The Hadamard spectra required for testing the circuit were generated on-chip. They held vectors at the PIs of circuits for multiple clocks, to increase the fault coverage. The test length for spectral BIST was much less than for LFSRs, but the fault coverage was higher. On average, spectral BIST added only 446 transistors for the pattern generator, but the LFSR added 461 transistors.

2.4 Response Compaction Hardware

Signature analysis compacts the circuit response into a *signature*, of very few bits, representing a statistical circuit property, for on-chip comparison of good and bad chips.

The *transition count* $C(R)$ is the number of times the signals in the circuit output response R change during BIST. An advantage of transition count compaction is that $|C(R)|$, the number of bits to represent $C(R)$, is $|C(R)| \leq \lceil \log_2 |R| \rceil$. $|R| = \#$ *bits in* R [8, 9].

Savir [18] proposed exhaustive pattern generation, and response compaction using ones counting. He designed combinational circuits so that the signature was the *syndrome*, or the number of switching function minterms.

In LFSR methods, the circuit output data stream is treated as a descending order coefficient Boolean polynomial. The output response compactor LFSR divides this data stream polynomial by the characteristic polynomial of the LFSR [5]. The final state of the modular LFSR is the polynomial remainder of this division. A circuit error changes the output and, hence, the remainder or signature. Figure 2 shows the MISR, with much less hardware than if separate compactors were used at each CUT output [12]. *Aliasing* occurs when the compacted responses of the faulty

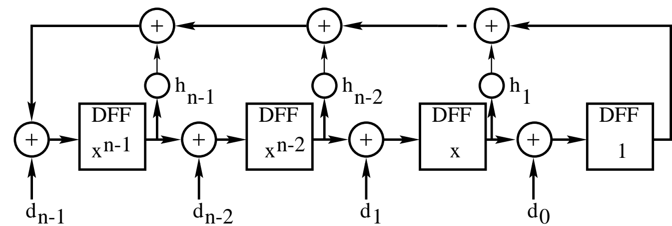

Figure 2: Multiple Input Signature Register [3, 4]

and fault-free circuit match. The aliasing probability is 2^{-k}, where k is the number of LFSR flip-flops [2, 20].

Bakalis *et al.* [1] used the addition of all PO values from the circuit (syndromes) as a signature.

3 BIST Signature Analysis Using the Hadamard Transform

The Hadamard matrix is used to identify the vectors with predominant digital spectra in the existing set of compacted test patterns [6]. The test set is extended using these identified spectra. One uses non-linear signal processing to filter out the undesired spectra, which may have prevented the detection of certain hard-to-detect faults. Iterations of filtering via compaction and spectral analysis of the filtered test set are performed for each PI, extracting inherent spectral information embedded within the test sequence. This information, when viewed in the frequency domain, reveals the characteristics of the input spectrum.

3.1 BIST System Description

We use Zhang *et al.*'s [21] software *spectral test generation* method of *bit flipping* to generate test sequences for sequential circuits. This tool tells us the spectra (in order of importance) for testing each PI, and the holding time for the test patterns that maximizes the fault coverage. We use Upadhyayula's [19] method to design a spectral test pattern generator for the CUT in hardware that approximately replicates Zhang *et al.* 's test patterns. Finally, we use the CUT with Upadhyayula's pattern generator and evaluate the system performance using different spectral response compactors. Results are presented for these response compactors, the MISR and the transition counter.

3.2 Signature Analysis with Hadamard Transforms

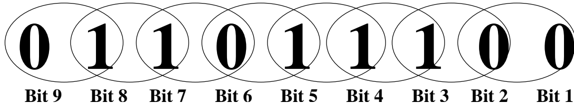

Figure 3: Overlapping in a Bit Stream at a PO

We use the Hadamard matrix for compacting the response, caused by the test pattern set, of the CUT by counting the spectra in a bit stream at each circuit PO using a 2×2 Hadamard matrix ($H(1)$) for response compaction. We take 2-bit "chunks" of the bit stream at a PO and pre-multiply them with $H(1)$. This gives us a set of two correlation coefficients, one for multiplication with each row of the Hadamard matrix. Figure 3 shows us how a bit stream is partitioned into 2-bit chunks where the second bit of the preceding chunk and the first bit of the succeeding chunk will *overlap*. Notice how the first row/column of the matrix just contains 1's ([1 1]) and the second row/column contains [1 -1]. Pre-multiplying with the first row is equivalent to adding the two bits in each 2-bit "chunk" and pre-multiplying with the second row is equivalent to the difference of the two bits. We send the spectral content of each two-bit chunk to the counters (one for each column of the Hadamard matrix). We implement this BIST system using the two's-complement binary number system, because it has one unique value for 0 (unlike the one's-complement system). Also, subtraction can be implemented with very minor adjustments to the addition hardware. Subtraction hardware would be more complex for sign-magnitude or one's-complement arithmetic. In the following hardware implementations for SRC2, 3, 4, and 5 (but not SRC1), all carries and borrows are preserved in the computation, including those carrying or borrowing out of the most significant bit of the response compaction counter. For SRC1, most significant carries and borrows are discarded.

Example. In SRC1, we operate on the bit stream at a particular PO and then repeat the process for the rest of the POs in the circuit. Here we take a circuit with four POs and each PO has a stream of six bits, as shown below. We take each bit pair from each PO and premultiply it by $H(1)$ as shown below. Table 2 shows the results of these multiplications for each bit chunk. The first row of $H(1)$ gives the *add* counter and the second gives the *subtract* counter.

$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

	$PO1$	$PO2$	$PO3$	$PO4$	*Special Condition*
	0	0	0	0	*Initialization*
	1	0	0	1	
	0	1	1	0	
	0	1	1	1	
	1	1	0	0	
	0	0	1	1	
	0	0	0	1	

The spectral content for each 2-bit chunk is added and the total spectral content for $PO1$ extracted by row 1 of the Hadamard transform is 4 $(1+1+0+1+1+0)$, and that extracted by row 2 is 0 $(-1+1+0-1+1+0)$ (see Table 2). Spectral content for other POs is given below.

$PO1\ (+,-)$	$PO2\ (+,-)$	$PO3\ (+,-)$	$PO4\ (+,-)$
$4, 0$	$6, 0$	$6, 0$	$7, -1$

Note: $+$ is the value on the *Add* Counter, $-$ is the value on the *Subtract* Counter

The results at each PO for addition and subtraction are fed into two counters at each PO. After the test set is applied, the counters give the circuit signature (fault-free or faulty). So, the signatures in the 4-bit counters at $PO1$ produced by the first and second columns of the matrix will be [0 1 0 0] and [0 0 0 0], respectively. The signatures at $PO2$ will be [0 1 1 0] and [0 0 0 0]. The signatures are compared with good machine signatures to detect faults.

Table 2: Spectral Content in $PO1$ Extracted by Rows 1 and 2 (with One Preceding 0)

Bit Chunk	Spectral Content (Add)	COUT	Spectral Content (Sub)	BOUT
0 1	1	0	1	1
1 0	1	0	1	0
0 0	0	0	0	0
0 1	1	0	1	1
1 0	1	0	1	0
0 0	0	0	0	0

3.3 Hardware Implementation of Spectral Response Compactor

The first column of the Hadamard transform is implemented with a half adder with a carry generator, and the second with a half subtracter with borrow generator. In

Figure 4: Hardware Implementation for SRC1 with Shared Hardware

SRC1, we will be extracting spectra in the bit stream at each PO; therefore, we will require an adder, a subtracter and two counters at each PO. We reduced the hardware by almost half by using hardware sharing, by performing addition and subtraction operations in SRC1 with one half-adder with a carry generator. We repeat testing twice, the first time to calculate the first spectral tone, and the second time to calculate the second tone. Figure 4 shows the use of signal *sub* to achieve this. When *sub* is 0, the adder performs addition to calculate the first tone. When *sub* is 1, the adder performs subtraction to obtain the second. The result of addition/subtraction is added into a spectral counter. This technique doubles the test time to generate a signature, but the spectral test pattern length is so vastly shorter than the length for an LFSR that this extra time is inconsequential. Thus, only one counter is required at each PO. The hardware implementation of both columns of the Hadamard matrix for SRC1 is shown in Figure 4. The *full adder* (FA) and subtracter are used here.

We need to add (or subtract) 2 bits produced at a PO. First, all of the flip-flops in SRC1 are initialized to 0. The half adder/half subtracter calculates the spectral content in the bit stream and sends the result to the counters. In the first cycle, 0 from DFF1 is latched out and becomes operand A of the adder/subtracter. The first response bit at the PO becomes operand B of the adder/subtracter and is latched into DFF1 in the same clock cycle. Addition/subtraction is performed on these bits and the result is fed to the *add/subtract* counters. In the second clock cycle, the first bit produced at the PO in the prior clock cycle is latching out of DFF1 and becomes operand A, whereas the second bit produced at the PO becomes operand B. The second bit is also latched into DFF1 for the next clock cycle at the same time, and so on. The process is repeated for the entire response of the circuit, and then the value in the tone counters is the signature.

4 Dual-Tone Spectral Model

We discuss some favorable aspects of SRC1 and propose SRC2, SRC3, SRC4, and SRC5, which have much lower hardware overhead than SRC1.

4.1 Spectral Response Compactor 1 (SRC1)

The method has a higher area overhead compared to a MISR (see Table 6) but is completely free of aliasing (see Table 5) for the circuit examples in this paper, whereas the MISR tends to alias. Holding test vectors for a number of cycles can improve fault coverage in sequential circuits [15, 16, 17], but can potentially cause aliasing in a MISR for sequential circuit testing. When two test vectors cause an identical faulty response at the POs feeding into the MISR, the XORing operation (which should give a 1 in the presence of a fault) often masks the two faulty responses. Due to masking, the faulty response and the fault-free response are identical, since $1 \oplus 1 = 0$. The hardware overhead in SRC1 is higher than any of the other compaction techniques we explore, because using a separate spectral tone counter at each PO requires much hardware.

4.2 Spectral Response Compactor 2 (SRC2)

Instead of adding 2-bits at a PO, one bit produced in the previous cycle and the second bit produced in the current cycle to calculate the *auto-correlation*, we pair up adjacent POs and add/subtract the 2-bits in each pair to calculate the *cross-correlation*. The hardware implementation is illustrated in Figure 5. In the case of an odd number of POs, we connect a fifth PO directly into the unused B port of the most significant full adder FA (see Figure 5). We can save hardware with this approach by eliminating the need for a D flip-flop to store the bit at a PO produced in the previous clock cycle so it can be added to the current bit or subtracted from the current bit. The results of this addition/subtraction are then added to the tone counter. Similarly to SRC1 (described above), we will use the *sub* signal to perform addition ($sub = 0$) or subtraction ($sub = 1$) using the same hardware. The results generated by these structures are then fed to 4-bit binary counters. This technique has a much lower hardware overhead than SRC1, since there is one counter for the entire circuit, rather than one counter for each PO. As the number of POs increases, the hardware savings are

more noticeable. We also use end-around carries in the counter, so that when it overflows, we generate a carry into the least significant counter bit.

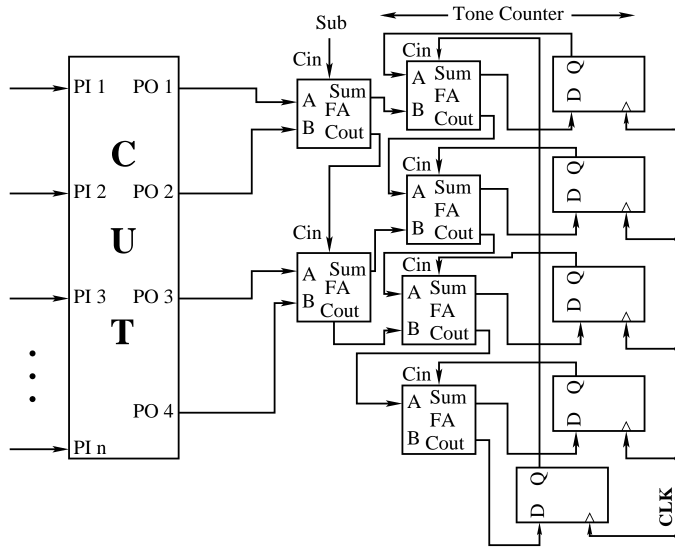

Figure 5: Hardware Implementation for SRC2

The test procedure will have two runs of the test set. First, we will apply the entire test set with the adder tree ($sub = 0$) and compute the signature for the adder tree. Then we will apply the test set again with the subtract tree ($sub = 1$) to compute the signature generated by the subtract tree. We eliminate almost half of the hardware. Although it will take longer to generate a signature since we will have to run the test pattern set two times for response compaction, the spectral test pattern set is so much shorter than the LFSR pattern set, that this is inconsequential.

4.3 Spectral Response Compactors 3 and 4 (SRC3 and SRC4)

To reduce hardware overhead, we eliminated the *subtract* counter and the hardware associated with it from SRC1, and only use the first spectral tone in SRC3. Results show that the aliasing probability for SRC3 is higher than for SRC1. The hardware overhead for SRC3 is slightly less than that for SRC1. SRC4 is identical to SRC3 except that we implement the second spectral tone from SRC1 instead of the first. The overhead is slightly higher than for SRC3 because a subtracter requires an extra inverter.

4.4 Spectral Response Compactor 5 (SRC5)

The implementation of SRC5 is almost the same as SRC2. In SRC2 we extracted the spectral content in the bits at each PO using cross-correlation, which is accumulated in the counter. In SRC5, instead of cross-correlating the POs, we auto-correlate them. This means that we extract the spectral content in the bit streams at each PO

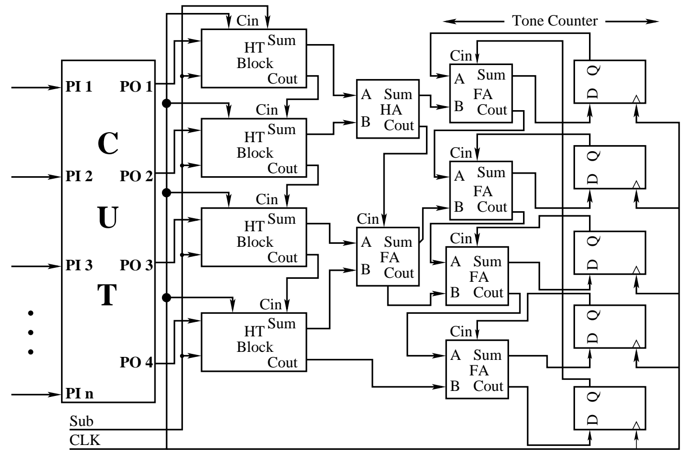

Figure 6: Block Diagram of SRC5

(as in SRC1). Then we add the spectral content of each PO and store the results in one counter. We apply the idea from SRC1 of sharing the hardware in SRC5, as well. A counter collects the spectral content extracted by the first column of the Hadamard matrix at all POs in the first run of the test set. In the second run, the same counter collects the spectral content extracted by the second column of the matrix. This method is a hybrid of SRC1 and SRC2. The hardware implementation is shown in Figure 6. This scheme uses more hardware than SRC2 and loses more faults than SRC2. The *HT Block* in Figure 6 is the hardware implementation of the *Hadamard Transform* (HT) shown in Figure 7.

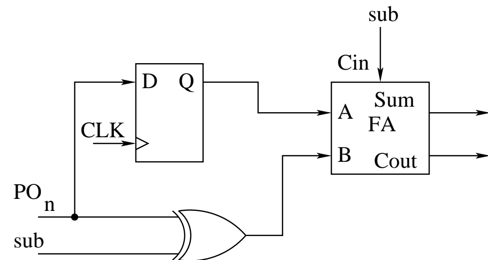

Figure 7: Hadamard Transform (HT) Block

5 Aliasing Analysis

We presented designs for five new spectral response compactors. The simulation results showed that SRC1 is free of aliasing for the circuits given here whereas the MISR aliased in certain cases. The only disadvantage of the SRC compactors was higher hardware overhead compared to the MISR, except for SRC2, which had lower overhead. We discuss the causes of aliasing in the SRC's.

5.1 Counter Overflow

Aliasing can be caused due to a counter overflowing. When the counter changes from the all-one state to the all-zero state, it has *overflowed*. For instance, if we use a 4-bit counter and we apply the system clock at the input of the counter, the counter will count from 0 ([0000]) up to 15 ([1111]). After counting up to 15, the counter will restart from 0. The recommended counter length is:

$$n = \lceil \log_2 (Length\ of\ Test\ Set) \rceil \quad (2)$$

where n is the number if bits in a ripple counter. For example, if the length of the test set is ten, we should use a 4-bit counter ($\lceil \log_2 (10) \rceil \approx 4$) to avoid the possibility of aliasing by overflowing.

We use 3-bit counters for response compaction. There are ten vectors in the test vector set. But, according to Equation 2 we should use a 4-bit counter. Assume that the fault-free machine signature in the *add* counter is [001]. The faulty circuit produces many more 1's (a 1 for each vector) than the good circuit, causing the counter to overflow after the 8th vector. Thus, the *add* counter gives us the same signature as the fault-free machine, causing the response compactor to alias. This type of aliasing is eliminated with end-around carries to generate a carry into the *least significant bit* (LSB) when the counter overflows.

5.2 Bit Flipping

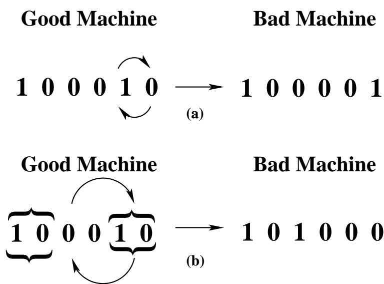

Figure 8: (a) Bit Flipping and, (b) Subsequence Flipping

Aliasing can occur if two contiguous bits or contiguous subsequences are flipped as shown in Figure 8. Even though the bits or subsequences are in different orders, meaning that the fault has been actually excited, the spectra in the bit stream will remain the same and thus aliasing arises. Assume that we have a circuit with one PO and the bit stream produced at the PO is [1 0 0 0 1 0]. We are using SRC1 for response compaction in this example. We apply the Hadamard transform to overlapping chunks of 2 bits in this bit sequence. We get six chunks in this case (the rightmost bit in the sequence is the first bit produced at the PO after an initial 0 bit is generated), which are (00), (01), (10), (00), (00), (01). After the Hadamard transform's first column is applied, we get 3, so the 4-bit *add* counter should carry the bits [0 0 1 1] and after the second column has been applied, we get -1, so the *subtract* counter should be [1 1 1 1] (2's complement). Table 3 shows the step by step calculation. But if the bits are flipped in the output sequence due to a fault so that the sequence becomes [1 0 0 0 0 1], the spectra in the *add* and *subtract* counters will be [0 0 1 1] and [1 1 1 1], the same as for the good machine. Table 4 shows detailed computations for the example where a fault has flipped the bits.

Table 3: *Add* and *Subtract* Counter Values for the Original Bit Stream

Input	Add	COUT	Counter	Sub	BOUT	Counter
00	0	0	0000	0	0	0000
01	1	0	0001	1	1	1111
10	1	0	0010	1	0	0000
00	0	0	0010	0	0	0000
00	0	0	0010	0	0	0000
01	1	0	0011	1	1	1111

Table 4: *Add* and *Subtract* Counter Values for the Bit Stream with Bits Flipped

Input	Add	COUT	Counter	Sub	BOUT	Counter
01	1	0	0001	1	1	1111
10	1	0	0010	1	0	0000
00	0	0	0010	0	0	0000
00	0	0	0010	0	0	0000
00	0	0	0010	0	0	0000
01	1	0	0011	1	1	1111

If a subsequence of bits in the sequence flips, aliasing can occur. For example, in our output in the above example ([1 0 0 0 1 0]) a subsequence of bits is switched due to a fault and the output sequence becomes [1 0 1 0 0 0]. The *add* counter will have the signature [0 0 1 1] and the *subtract* counter will have the signature [1 1 1 1] in both the fault-free circuit and the faulty circuit. So, flipping of subsequences can cause aliasing as well.

6 Results

We present results for the response compaction techniques we implemented. We discuss a few advantages of the *spectral response compactors* (SRCs). First, the spectral test pattern generator generates far fewer test vectors than an LFSR to achieve a higher fault coverage. Shorter test pattern lengths mean that less power will be consumed during test. The hardware overhead of the SRCs is higher compared to the MISR, except for SRC2, which uses less hardware than the MISR. In other BIST systems hardware

is required for *design-for-testability* (DFT), LFSR reseeding and test point insertion. The SRC's require less DFT, i.e., reseeding and test point insertion hardware, than other BIST systems, because spectral BIST has inherently much higher fault coverage than other BIST systems and a much shorter test length.

6.1 Experimental Conditions

We wrote a program that inserted Upadhyayula's [19] spectral pattern generator hardware (given in Table 1) and our spectral response compactors (SRC1-5) into the CUT. All flip-flops in the circuit, the spectral pattern generator, and the spectral response compactor were initialized to 0. Then, we simulated the entire system for the test pattern length of Upadhyayula's spectral pattern generator given in Table 1 using *rsfsim*, a sequential differential fault simulator in the SEST test-pattern generator program. We noted all fault effects that were sensitized to POs in the circuit, and then noted which faults were not detected by each response compactor.

6.2 Fault Coverage

The results indicate that the fault loss due to aliasing is within a tolerable range because the spectral test generator manages to detect 8.42% more faults (compared to an LFSR). This greatly exceeds the number of faults that the response compactor loses due to aliasing. Therefore, there is an overall increase in the number of faults being detected. Table 5 shows the faults that are lost in the SRC's compared to the MISR and the transition counter on *ISCAS '89* benchmark circuits using the patterns produced by Upadhyayula's method as shown in Table 1. SRC1 never aliases. The compactors in increasing aliasing order are SRC1, the MISR, SRC2, SRC5, SRC3, SRC4 and the TC. SRC2 essentially has the same aliasing probability as the MISR, SRC5 is intermediate, and SRC3, SRC4 and the TC have the highest levels of aliasing. SRC2 always aliases less than the other response compactors (except for SRC1 and the MISR), except that in two cases, SRC3 and SRC4 have less aliasing, and in 11 cases, SRC5 has less aliasing. SRC5 does better than the MISR in a few cases. SRC1, SRC2 and SRC5 always alias less than the TC. The MISR loses fewer faults than SRC3 and SRC4 in almost all of the benchmark circuits. SRC5 aliases less than the MISR for seven circuits. The probability of aliasing is given by:

$$P_{aliasing} = \frac{\text{Number of Faults Lost}}{\text{Number of Total Faults}} \quad (3)$$

Implementing end-around carries in SRC2 and SRC5 was critical to reducing their aliasing. Without the end-around carry hardware, SRC2 would have aliased for 822 faults instead of for 135, and SRC5 would have aliased for 634 faults instead of for 369.

6.3 Test Lengths

Table 1 shows the test pattern lengths and fault coverages achieved with patterns generated by the LFSR and by the *spectral test pattern generator* [19]. In most cases the spectral test pattern generator gets much higher fault coverage with a shorter test length than the LFSR. The exceptions to this are benchmark circuits s641, s713, s820, s1196, s1238, and s1423, which require all-random spectra to be tested effectively. They have no frequency preference, so the LFSR is a better test generator for these circuits.

6.4 Hardware Overhead

The hardware overhead, in number of gates, for SRC1, SRC2, SRC3, SRC4, SRC5, MISR, and TC is given in Table 6. SRC2's hardware overhead is the lowest of all the SRCs, the MISR and the TC. SRC1 has the most hardware. As described earlier, we use hardware recycling in SRC1, SRC2 and SRC5 to reduce hardware overhead.

7 Conclusions and Future Work

We presented the hardware overhead of the response compaction techniques SRC1-5 in comparison with a MISR and with transition count testing. The hardware overhead of SRC2 is the lowest as shown in Table 6, but at the cost of slightly higher aliasing than SRC1. SRC1 does not alias for any faults in the ISCAS '89 circuits. The area overhead of SRC1 is much higher than for the MISR, the TC and SRC2-5. The optimal choice, keeping in mind area overhead and probability of aliasing, will be SRC2, due to its minor aliasing. But, if the least amount of aliasing is the primary issue, SRC1 should be used. Notice that the MISR has not aliased as much as MISR theory would have predicted, and that the MISR is a reliable response compactor for spectral test.

Future Work. We have yet to model the probability of aliasing in the SRC response compactors. We can model the counter states as Markov chains and formulate a robust aliasing probability formula. Then the probability of bit flips in the output bit stream due to a fault can be calculated. The two probabilities combined will give us the probability of aliasing.

Even more variations of the spectral response compaction can be tried to reduce hardware overhead and aliasing. We can apply columns of a higher-order Hadamard matrix to the bit stream at a PO. Also, we can devise a technique to group the POs to give us the least aliasing. Each group will have a pair of counters (one for each column of the Hadamard matrix). The POs being grouped can be contiguous or they can be far apart. POs reachable from a common gate should not be grouped.

The hardware for the response compactors SRC1-5 needs to be made self-testing, to avoid the situation where

Table 5: Simulation Results – Faults Lost Due to Aliasing

Circuit	Faults	POs	Vectors	Detected[a] Faults	Fault Cov. %	Faults Aliased in SRC 1	2	3	4	5	MISR Aliasing	TC Aliasing
s298	308	6	512	251	81.5	0	4	10	16	0	1	16
s344	342	11	64	288	84.2	0	1	0	10	0	1	11
s349	350	11	128	269	76.9	0	1	5	8	1	0	11
s382	399	6	512	337	84.5	0	3	13	46	2	1	47
s386	384	7	512	168	43.5	0	1	20	23	3	0	20
s420	429	2	1024	161	37.5	0	9	17	14	10	36	12
s444	472	6	512	365	77.3	0	2	7	28	0	3	28
s526	555	6	512	384	69.2	0	1	19	27	0	8	27
s641	467	24	1024	338	72.4	0	3	54	32	1	10	29
s713	581	23	1024	349	60.1	0	3	54	55	8	4	51
s820	850	19	512	143	16.8	0	5	8	3	1	0	5
s832	870	19	512	414	47.6	0	1	55	69	15	0	74
s953	1079	23	256	457	42.4	0	0	49	29	6	0	38
s1196	1242	14	256	470	37.8	0	6	63	43	3	0	42
s1238	1355	14	256	471	34.8	0	7	107	41	4	4	37
s1423	1515	5	2048	679	44.8	0	2	69	30	1	28	29
s1488	1486	19	512	1240	83.4	0	18	43	81	38	0	72
s1494	1506	19	128	1119	74.3	0	7	38	68	25	0	52
s5378	4603	49	1024	2544	55.3	0	56	95	303	86	1	398
s9234	6927	39	1024	582	8.4	0	1	66	121	0	0	98
s13207	9244	152	2048	993	10.7	0	12	50	131	40	0	218
s15850	9676	150	1024	1460	15.1	0	14	24	40	17	0	59
s35932	39094	320	1024	30451	77.9	0	110	101	118	140	14	130
s38417	31180	106	2048	4815	15.4	0	34	62	294	54	0	375
s38584	36101	278	1024	11238	31.1	0	108	289	297	96	9	2018
Total				46204		0	409	1318	1927	551	120	3897
$P_{aliasing}$						0.000	0.0089	0.028	0.04	0.012	0.0026	0.094

[a]Fault effects propagating to POs.

the circuit is correct, but faults in the response compactor cause the circuit to fail testing. An even worse situation is where the circuit is faulty, but faults in the response compactor hardware cause the circuit to pass testing.

Acknowledgement

This material is based on work supported by the National Science Foundation under Grant No. 0098304.

References

[1] D. Bakalis, D. Nikolos, and X. Kavousianos, "Test Response Compaction by an Accumulator Behaving as a Multiple Input Non-Linear Feedback Shift Register," in *Proc. of the International Test Conf.*, Oct. 2000, pp. 804–811.

[2] P. H. Bardell, W. H. McAnney, and J. Savir, *Built-In Test for VLSI: Pseudorandom Techniques.* John Wiley & Sons, Inc.. New York, 1987.

[3] M. L. Bushnell and V. D. Agrawal, *Essentials of Electronic Testing: For Digital, Memory & Mixed-Signal VLSI Circuits.* Kluwer Academic Publishers. Boston, Massachusetts, 2000.

[4] R. David, "Signature Analysis of Multi-Output Circuits," in *Proc. of the International Fault-Tolerant Computing Symp.*, June 1984, pp. 366–371.

[5] A. Frohwerk, "Signature Analysis: A New Digital Field Service Method," *Hewlett-Packard Journal*, vol. 28, pp. 2–8, May 1977.

[6] A. Giani, M. S. Hsiao, S. Sheng, and V. D. Agrawal, "Efficient Spectral Techniques for Sequential ATPG," in *Proc. of the Design-Automation and Test in Europe Conf.*, 2001, pp. 204–208.

[7] A. Giani, M. S. Hsiao, S. Sheng, and V. D. Agrawal, "Novel Spectral Techniques for Built-In Self-Test in a System-on-a-Chip Environment," in *Proc. of the* 19th *IEEE VLSI Test Symposium*, 2001, pp. 163–168.

[8] J. P. Hayes, "Transition Count Testing of Combinational Logic Circuits," *IEEE Trans. on Computers*, vol. C-25, no. 6, pp. 613–620, June 1976.

[9] J. P. Hayes, "Generation of Optimal Transition Count Tests," *IEEE Trans. on Computers*, vol. C-27, no. 1, pp. 36–41, Jan. 1978.

[10] S. Hellebrand, H.-J. Wunderlich, and A. Hertwig, "Mixed-mode BIST Using Embedded Processors," in *Proc. of the International Test Conf.*, Oct. 1996, pp. 195–204.

Table 6: Hardware Overhead (Number of Logic Gates) for All SRC's

Circuit	POs	Number of Logic Gates							
		Ckt.	1	2	3	4	5	MISR	TC
s208	2	176	262	73	136	138	107	35	108
s298	6	259	766	86	388	394	188	95	324
s344	11	310	1396	118	703	714	305	170	594
s349	11	196	1396	118	703	714	305	170	594
s382	6	368	766	86	388	394	188	95	324
s386	7	219	892	94	451	458	213	110	378
s400	6	374	766	86	388	394	188	95	324
s420	2	356	262	73	136	138	107	35	108
s444	6	391	766	86	388	394	188	95	324
s526	6	406	766	86	388	394	188	95	324
s641	24	457	3034	196	1522	1546	604	375	1296
s713	23	583	2908	186	1459	1482	577	350	1242
s820	19	339	2404	162	1207	1226	485	300	1026
s832	19	337	2404	162	1207	1226	485	300	1026
s953	23	463	2908	186	1459	1482	577	350	1242
s1196	14	709	1774	132	892	906	370	225	756
s1238	14	688	1774	132	892	906	370	225	756
s1423	5	1397	640	80	325	330	165	80	270
s1488	19	713	2404	162	1207	1226	485	300	1026
s1494	19	707	2404	162	1207	1226	485	300	1026
s5378	49	4569	6184	334	3097	3146	1167	540	2646
s9234	39	7387	4944	502	2467	2506	941	335	2106
s13207	152	8803	19162	948	9586	9738	3532	2295	8208
s15850	150	15742	18910	936	9460	9610	3486	1310	4698
s35932	320	33345	40330	1952	20170	20490	7392	3526	17280
s38417	106	38539	13366	684	6688	6794	2486	1167	5724
s38584	278	20995	34038	1814	17524	17802	6540	3064	15012
Average		5141	6208	357	3126	3177	1189	594	2546

[11] G. Kasturirangan and M. Hsiao, "Spectrum-Based BIST in Complex SOCs," in *Proc. of the 20th VLSI Test Symp.*, Apr. 2002, pp. 163–168.

[12] B. Köenemann and J. Mucha, "Built-In Test for Complex Digital Integrated Circuits," *IEEE Journal of Solid-State Circuits*, vol. SC-15(3), pp. 315–319, June 1980.

[13] L. Nechman, K. K. Saluja, S. Upadhyaya, and R. Reuse, "Random Pattern Testing for Sequential Circuits Revisited," in *Proc. of Annual Symposium on Fault Tolerant Computing Symp.*, June 1996, pp. 45–52.

[14] I. Pomeranz, S. M. Reddy, and R. Guo, "A Fault Simulation Based Test Pattern Generator for Synchronous Sequential Circuits," in *Proc. of the* 17th *VLSI Test Symp.*, 1999, pp. 260–267.

[15] I. Pomeranz, S. M. Reddy, and R. Guo, "The Effects of Test Compaction on Fault Diagnosis," in *Proc. of the Interantional Test Conference*, Sept. 1999, pp. 1083–1089.

[16] I. Pomeranz, S. M. Reddy, and R. Guo, "On Improving Static Test Compaction for Sequential Circuits," in *Proc. of the* 14th *International Conference on VLSI Design*, 2001, pp. 111–116.

[17] I. Pomeranz, S. M. Reddy, and R. Guo, "Reverse-Order-Restoration-Based Static Test Compaction for Synchrounous Sequential Circuits," *IEEE Trans. on Computer-Aided Design of Integrated Circuits & Systems*, vol. 22, no. 3, pp. 293–304, March 2003.

[18] J. Savir, "Syndrome Testable Design of Combinational Circuits," *IEEE Trans. on Computers*, vol. C-32, pp. 442–451, June 1980.

[19] K. Upadhyayula, *Test Pattern Generation Using Spectral BIST*. M.S. Thesis, ECE Dept., Rutgers University, Oct. 2003.

[20] T. W. Williams, W. Daehn, M. Gruetnzer, and C. W. Starke, "Bounds and Analysis of Aliasing Errors in Linear-Feedback Shift-Registers," *IEEE Trans. on Computer-Aided Design*, vol. CAD-7, no. 1, pp. 75–83, Jan. 1988.

[21] J. Zhang, M. L. Bushnell, and V. D. Agrawal, "On Random Pattern Generation with the Selifsh Gene Algorithm for Testing Sequential Circuits," in *Proc. of the International Test Conf.*, Oct. 2004. Paper 21.3.