

# BUILT-IN SELF-TEST FOR A MONOBIT FAST FOURIER TRANSFORM RECEIVER



Kripa Sankaranarayanan

Intel Corporation  
JF5-260, 2111 NE 25<sup>th</sup> Ave, Portland, OR 97124  
email: kripa.sankaranarayanan@intel.com

Charles Stroud

Dept. of Electrical and Computer Engineering  
200 Broun Hall, Auburn University, AL 36849-5201  
email: strouce@auburn.edu

**ABSTRACT:** The application of four different Built-In Self-Test (BIST) approaches to a monobit Fast Fourier Transform (FFT) receiver is discussed. The monobit FFT receiver is used in electronic warfare systems and presents a number of interesting obstacles to the application of BIST.

## 1. INTRODUCTION

The Monobit Receiver (MR) is a mixed-signal system to perform instantaneous frequency measurement in electronic warfare systems [1][2]. The design consists of a non-linear radio frequency front end, analog to digital converter and an application specific circuit (ASIC) for performing Fast Fourier Transform (FFT) and frequency selection [3]. The FFT and frequency selection circuitry is the main component of the MR which computes the FFT of the input data and finds the frequencies of the two highest amplitude signals. The focus of this paper is a discussion of the application of four different Built-In Self-Test (BIST) approaches to the MR and the various obstacles posed by the MR to each application. We begin in Section 2 with a detailed overview of the MR architecture, operation, and implementation. The application of four different BIST approaches to the MR is discussed in Section 3 where the BIST approaches include: Built-In Logic Block Observer (BILBO), Circular BIST, scan-based BIST, and a non-intrusive BIST approach. The paper concludes in Section 4 with the recommended BIST approaches for ASIC and Field Programmable Gate Array (FPGA) implementations of the MR.

## 2. MONOBIT FFT RECEIVER

There are five pipelined stages in the MR including the input, FFT, initial sorting, squaring and addition, and final sorting stages. Each stage is allocated a maximum delay of 102.4 ns. Sixteen 2-bit data windows are simultaneously input to the MR at a rate of 156.25 MHz. The input stage receives and stores input data until a total of sixteen such data sets are collected for a total 256 2-bit data samples. A complete data set is collected and fed to the FFT stage every 102.4 ns (or  $16 \times 6.4$  ns) where a 256-point FFT of the input data samples is performed with a frequency cell of 9.77 (1250/128) MHz. The FFT area and performance is optimized by eliminating multipliers in the design. The FFT requires only adders if either the kernel function of the discrete fourier transform (DFT) or the input data is 1-bit ( $\pm 1$ ). In the MR, the FFT kernel function is rounded to  $\pm 1$  or  $\pm j$  such that the kernel function is

mapped to a time decimated, radix-2 FFT algorithm. The frequency selection logic selects two input frequencies after further processing of the output from the FFT stage. Low and high threshold values each 7-bits long and set to the particular desired levels, are input to the frequency selection logic. The initial sorting stage considers a maximum of four values that are above either the high threshold value or the low threshold value, out of the 128 real and 128 imaginary values output by the FFT stage. If one or more FFT output values are above the high threshold, then those values are considered, otherwise, the low threshold value is used for signal selection. Next, the selected real/imaginary values are squared and added with their corresponding imaginary/real values. In the final sorting stage, the four selected signals are sorted to find the two signals with the highest amplitude values.

The ASIC implementation of the MR has limited flexibility with respect to trade-offs between hardware availability, efficient design, speed and cost of implementation. To improve the efficacy of the MR and its applications, design improvements were made and the MR was implemented in a FPGA since the high density of logic capabilities in FPGAs facilitates the construction of signal processing hardware that closely corresponds to the natural data flow of a desired algorithm. In the previous ASIC based MR implementation, the two threshold values used for frequency selection are fixed and the frequency selection logic does processing of the signals based on the fixed values. One improvement involves making the 7-bit threshold values programmable to lend flexibility to the design with respect to the amplitude of the signals detected by the MR, by protracting the range of the signals detected. Another design improvement eliminated the use of multiple clocks in the original design [1][2]. Figure 1 gives the block diagram of the MR design.

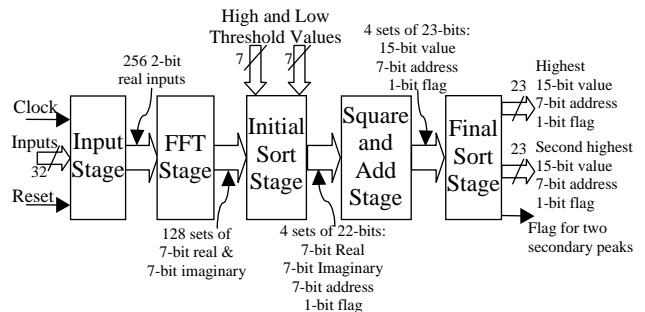


Figure 1. Monobit Receiver Block Diagram

## 2.1. Input Stage

The inputs to the input stage include clock, reset and 32-bit data. The input stage receives 32-bit parallel data every clock cycle, stores them and produces 256 sets of 2-bit binary numbers. The input data storage system consists of 16 units where each unit consists of sixteen 2-bit registers controlled by the clock enables. When the complete set of 512 bits of data has been loaded, the data is transferred to the output register on the next active edge of the clock. This output register is connected to the input to the FFT stage. The input stage also generates 16 clock enables used for this serial-to-parallel data conversion along with the synchronization of the subsequent stages. These clock enables are generated by a 16-bit shift register clocked by the input clock which has a frequency of 156.25 MHz such that the effective frequency of each clock enable is 9.77 MHz (156.25/16).

## 2.2. FFT Stage

The FFT stage computes the 256 point decimation in time (DIT) FFT of the 256 sets of 2-bit input data. The transform results in 128 sets of output data, each set consisting of a 7-bit real and 7-bit imaginary number [1][2]. Ten levels of transformation are done in the FFT stage using a Butterfly network cell (Figure 2a). There are no multiplications in the FFT computation since the MR uses a monobit kernel function. The 2-bit input to Level 0 is transformed to a 3-bit output when reordering the inputs as required by the DIT FFT algorithm. Levels 1, 2 and 3 consist of 4-bit, 5-bit and 6-bit operations, respectively. Level 4 consists of 7-bit operations. The 8-bit results obtained in this level is input to Level 5. Levels 6 and 7 consist of eight-bit operations. The results of operations in levels seven and eight are truncated to eight bits. Level 8 produces 8-bit results, which are in two's complement form. Another level is added to form Level 9 to truncate the 8-bit numbers to 7-bit positive numbers. The first eight Levels (0 through 7) have identical upper and lower halves. This permits pipelining to reuse the hardware required to implement the upper half of Levels 0 through 7. Multiplexers at the input and a bank of flip flops to store data are added to implement the pipelining as shown in Figure 2b.

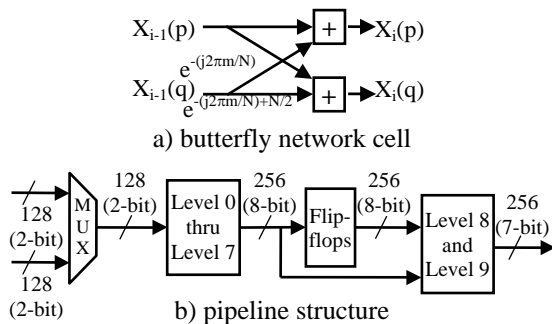


Figure 2. FFT Stage

Table 1. FFT Levels

Level	# Sets of Outputs	# Bits/Set
0	256	3
1	256	4
2	384	5
3	448	6
4	480	7
5	496	8
6	504	8
7	508	8
8	256	8
9	256	7

## 1.3. Initial Sorting Stage

The initial sorting stage (Figure 3) locates the addresses of a maximum of four signals from the 128 sets of output data from the FFT stage. The selected values are then squared and summed in the subsequent stage. Figure 3 gives a block level representation of the initial sorting stage. The inputs to this stage include the high and low threshold values and the 128 sets of 7-bit real and imaginary numbers from the FFT stage. The thresholds are programmable during operation through the input pins. The initial sort stage produces 4 sets of 22-bit outputs. The 22 bits consist of the 7-bit address, the 7-bit real part of the signal, 7-bit imaginary part, and a flag bit. The initial sorting stage consists of a threshold comparator and address decoder to locate the addresses of signals greater than or equal to the threshold.

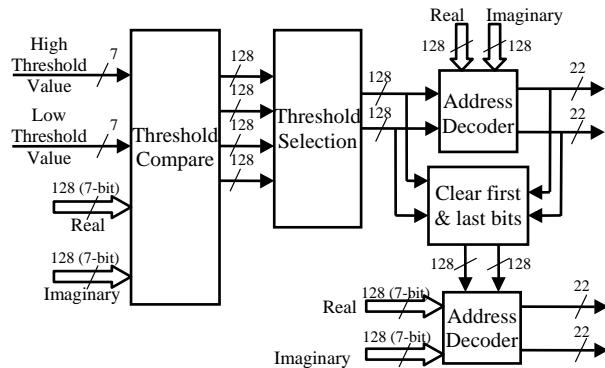


Figure 3. Initial Sorting Stage Block Diagram

Signal selection is based on the high and low threshold values. If any signal is above or equal to the high threshold, then the high threshold value is used for determining the strongest signals. If no value crosses the high threshold, the low threshold value is used for signal selection. The use of two programmable threshold values increases the probability of detection and reduces erroneous detection. This design assumes that a maximum of four values will cross the threshold value. In order to determine which threshold is to be used for signal detection, comparisons are made between the threshold values and the 128 real and 128 imaginary values. The comparison is done by inverting the value

of the number to be compared and adding it to the threshold. If the carry out bit equals zero, it implies that the number being compared is greater than or equal to the threshold, otherwise the threshold value is greater than the number being compared. Four sets of comparisons need to be made to determine which threshold is to be used to make signal selection. First, the high threshold is compared to the 128 sets of real and imaginary numbers. Next, the 128 sets of real and imaginary numbers need to be compared with the low threshold. The results of the four comparisons would be four vectors each 128 bits long, with a '1' in the bit positions which have inputs greater than or equal to the threshold. In order to select which threshold is to be used, the two 128-bit vectors which result from the comparison of the real and imaginary numbers with the high threshold are checked first. If any one of the bits is a 1, then the high threshold is used for signal selection. If none of the bits is a 1, then no signal is greater than or equal to the high threshold value and the low threshold value is used for signal selection.

Once the threshold to be used for signal selection has been determined, the addresses of the signals that are greater than or equal to that threshold need to be decoded. Since it is assumed that only a maximum of 4 signals cross the threshold, the two 128-bit arrays that give the results of comparison with the selected threshold can have a maximum of 4 bits set to 1. In order to decode the last and first of the 4 positions, the array is traversed from position 0 to position 127 and in the opposite direction. The last bit that is set to a 1 in each case gives the last and first addresses respectively. Once these addresses are known, the bits in those positions are cleared before the array is traversed again in the same manner. On this second pass, the remaining two addresses are decoded. Once the addresses are decoded, the corresponding real and imaginary parts are determined and the flag is set if the data is valid. In the event that there are less than 4 signals detected, the remaining addresses and values along with their flags are set to 0.

#### **1.4. Squaring and Addition Stage**

The inputs to the squaring and addition stage are the four sets of 22-bit values output by the initial sorting system. The 22 bits are the 7-bit address, 7-bit real and 7-bit imaginary values and the flag bit. The main function of the squaring and addition stage is to square the real and imaginary parts of each set and add them together in order to find the actual values of the signals. The output of this stage is four sets of 23-bit values, of which 7 bits are for address, 15 bits for the actual value of the signal and the one bit for flag. There are four identical subcircuits in this stage, each performing the squaring and addition operations on the 4 sets of inputs to produce the four sets of output data.

The squaring of the numbers is performed using a 7-bit wallace tree multiplier. Two multipliers are required for squaring the real and the imaginary parts of the signals and 14-bit adders for addition of the squared real and imaginary parts. Four such blocks are necessary for the squaring and addition of the four sets of numbers. No operation is done on the address and flag lines coming into the system. Eventually the computed values are combined along with the respective address and flag and output to the final sorting stage.

#### **1.5. Final Sorting Stage**

The function of the final sorting stage is to determine the two highest signal amplitudes, their addresses and flags from the four sets of 23-bit data input to the system. The outputs of the system are 2 sets of values, each 23 bits long and a flag indicating the presence of two secondary peaks. The 23 bits consist of the 7-bit address of the signal, the 15-bit value of the amplitude of the signal as given by the FFT operation, and the flag indicating the validity of the signal. The flag bits are set to zero if the addresses are invalid. The final sorting system sorts the values of the numbers in descending order and outputs the values, the corresponding addresses and flags of the two largest numbers. In the event that the second highest and third highest numbers are equal, the flag indicating the presence of two second highest peaks is set. The final sorting stage consists of five comparators. The first two comparators perform pairwise comparison of the four inputs. The results of comparison are put in the correct order. Then the two smaller numbers and the two bigger numbers are compared to determine the largest and smallest numbers. The final comparator circuit performs comparison of the two intermediate values to determine the second highest and the third highest values.

#### **1.6. Design Implementation**

The MR design was described in VHDL and synthesized onto a single Xilinx Virtex-E XCV2000E FPGA. Mapping the MR onto a FPGA, which is the ideal implementation media for FFT operations performed by the MR, ensures flexibility for redesign and the use of the most current technology in the electronic warfare systems. However, the MR can also be synthesized into a standard cell based ASIC which provides a higher performance design than an FPGA. In synthesis of the full design, the clock is constrained to 6.25 ns. Two clocks can be used to determine if any of the paths in the first stage that run at 156.25 MHz fail, and to verify if any paths in the other stages which run at 9.77 MHz fail by connecting the clock with a period of 6.25 ns to the input stage and a second clock with a period of 102.4 ns to all the other stages.

Two different timing constraints are used for the synthesis of the pipelined FFT stage. Since the FFT

stage has identical upper and lower halves, pipelining is implemented to reuse the hardware used for computation of the first half of the FFT operation. In the implementation of pipelining in the FFT stage, the results of computation of the first half of the FFT operation are stored in intermediate flip flops, while the other half of the FFT is computed. At the completion of computation of the entire FFT, results of the upper half of FFT operation are loaded from the intermediate flip flops and the other half from direct computation into the pipelined output flip-flops for that stage. Two different timing constraints are used to detect failure in either of these paths.

The address decoding stage of the initial sorting stage performs the function of a large encoder and typically produces the critical timing paths in the synthesized circuit. The circuit complexity is of the order of  $N$ , where  $N$  is the number of inputs but can be redesigned to reduce the levels of logic, in order to decrease the delay of the circuit. A tree structure can be used, consisting of  $\log_2 N$  levels where each level performs the OR operation of groups of two consecutive bits, resulting in an output that is half the length of the input. The final level has two inputs and one output, the flag bit. The tree is then traversed back to get the highest address. This is done by checking the branches of the tree on the side of the most significant bits. The most significant bit of the address is thus obtained from the right node of the tree at the second-last level. The next most significant bit is obtained from the right node in the direction of tree traversal. All the bits in the address are obtained in a similar manner, with the least significant bit obtained from the input bits, depending on the direction of tree traversal determined by the first level. This priority encoder design has a complexity of  $\log_2 N$  which can significantly reduce the propagation delay. For example, the worst case propagation delay was reduced from 182.56 ns to 93.06 ns while the number of levels of logic was reduced from 151 to 37. Furthermore, the percent of device utilization on the Virtex-E XCV2000E in terms of the number of slices required for implementation decreased from 95% to 51% with the improved encoder design.

### 3. APPLICATION OF BUILT-IN SELF-TEST

The MR presents a number of interesting challenges for implementing and executing Built-In Self-Test (BIST) due to its size and performance requirements. The implementation consisted of 5,062 flip-flops plus an additional 106,553 combinational logic gates. While the primary clock frequency for the circuit is 156 MHz, in the single-clock single-edge synchronous design, most of the delay paths through the device have 16 clock cycles for propagation prior to the next clock enable at the destination flip-flop giving an effective data rate of

9.77 MHz. However, the amount of combinational logic (number of levels of combinational logic gates) between these clock-enabled flip-flops is significant and constitutes the principal critical timing paths of the design. Therefore, it is helpful to discuss attributes that complicate application of BIST.

In the input stage of the MR, thirty-two 16-bit shift registers operate at the 156 MHz rate and perform a serial-to-parallel data conversion, after which the primary effective data rate is 9.77 MHz in the remaining four stages of the design. The exception to this lies in the pipelined FFT stage where portions of the FFT circuitry have an effective data rate of 19.54 MHz. While the remaining three stages operate at the 9.77 MHz effective data rate, the initial sorting stage contains the most critical timing paths of the complete design. This is due to a combinational logic feedback system used to determine the first two frequencies out of a total of 256 frequencies that exceed the high and/or low programmable thresholds. Therefore, any BIST approach to be considered for the MR cannot be implemented with respect to the 156 MHz system clock but instead must be implemented with respect to the lowest effective data rate, 9.77 MHz in this case. In other words, BIST must be implemented with respect to the clock enables for each stage or sub-stage of the design.

To further complicate the testing problem, there are about 2000 combinational logic cones with 512 dependencies in the FFT stage and about 20 combinational logic cones with over 2000 dependencies in the initial sorting stage. A combinational logic cone is defined as having a vertex at the input to its destination flip-flop and its dependencies are all of the flip-flops and/or primary inputs that drive that combinational logic. As a result, a given combinational logic cone is obtained by tracing back from the vertex to find all inputs (or dependencies) that affect the output logic value at that vertex. To illustrate the significance of these extremely large combinational logic cones, pseudo-exhaustive testing of the combinational logic requires a minimum of  $2^k$  unique test patterns where  $k$  represents the largest number of dependencies for any of the  $m$  logic cones in an  $n$ -input,  $m$ -output combinational logic circuit. Generally  $k$  is much less than  $n$  in most combinational logic circuits. However, in the MR design,  $k$  is usually equal to  $n$  in most of the stages. As a result, pseudo-exhaustive testing of the FFT stage would require  $2^{512}$  unique test patterns while the initial sorting stage would require  $2^{2000}$  unique test patterns – obviously both cases are impossible for practical application of BIST since pseudo-exhaustive testing of the FFT stage at a clock frequency of 9.77 MHz would require about  $4 \times 10^{139}$  years.

Four different BIST approaches were investigated as potential candidates for testing the MR from wafer-level

testing through system-level testing to also include field testing. These BIST approaches include: 1) the Built-In Logic Block Observer, 2) Circular BIST, 3) Scan Design based BIST, and 4) a non-intrusive BIST approach that is specifically intended for high-speed data-path circuitry. In the following subsections, each of these BIST approaches and a discussion of their implementation in the MR will be presented along with the advantages and disadvantages of each approach.

### 3.1. Built-In Logic Block Observer (BILBO)

The BILBO was the first BIST approach proposed circa 1980 [4][5]. It uses the existing flip-flops in the circuit with additional circuitry (Figure 4) to function in one of four modes of operation (Table 2): normal system mode, scan mode, test pattern generation (TPG) mode, and output response analyzer (ORA) mode which functions as a Multiple Input Signature Register (MISR). The TPG and ORA modes of operation are obtained by constructing a Linear Feedback Shift Register (LFSR) from the existing flip-flops in the circuit to perform pseudo-random pattern generation and multiple-input signature analysis, respectively. By alternating the operation of the BILBOs (one as a TPG, the next as an ORA, the next as a TPG, and so on) in a pipe-lined architecture such as that of the MR, alternating combinational logic circuits are tested during a given test session. Pseudo-exhaustive test patterns are supplied to a given combinational logic circuit by its driving BILBO operating in the TPG mode. The output responses produced by the combinational logic circuit are compacted via signature analysis by the BILBO that is operating in the ORA mode at its outputs. In a subsequent test session the roles of the TPG and ORA BILBOs are reversed such that the combination logic circuits not tested during the first test session are now tested.

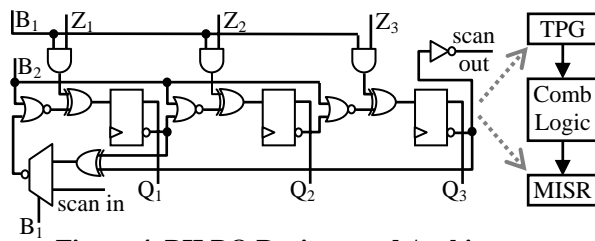


Figure 4. BILBO Register and Architecture

Table 2. BILBO Modes of Operation

B2	B1	Mode of Operation
0	0	Shift (scan) mode ( $Q_i \leftarrow Q_{i-1}$ )
0	1	MISR (BIST) mode ( $Q_i \leftarrow Z_i \oplus Q_{i-1}$ )
1	0	Initialization (reset) mode ( $Q_i \leftarrow 0$ )
1	1	System (normal operation) mode ( $Q_i \leftarrow Z_i$ )

For the BILBO approach to work properly (to apply pseudo-random test patterns), there can be no “self-feedback” in any given BILBO register. Fortunately, this is the case for the MR as long as the BILBO logic is

constructed outside of the clock-enables to the flip-flops; in other words test patterns are applied at a 9.77 MHz effective data rate. In order to guarantee high fault coverage (nearly 100% of all detectable stuck-at gate level faults), primitive polynomials must be used for the characteristic polynomial of the LFSRs to provide all  $2^n-1$  possible non-zero test patterns (excluding the all 0s test pattern) for an  $n$ -bit BILBO. However, primitive polynomials have only been derived for up to about 300 bits such that, given the size of the 512-input and 2000-input logic cones in the MR, primitive polynomials are not available. Regardless, the testing times for 512-bit and 2000-bit BILBOs for the FFT and initial sorting stages respectively, prevent the use of the BILBO approach. Area overhead for BILBO implementation in the MR is approximately 15% with a performance penalty of two gate delays in each path between flip-flops.

### 3.2. Circular BIST

Circular BIST resulted from the practical application of the BILBO approach in realistic sequential logic circuits where self-feedback is typically present [4][6]. Circular BIST uses the existing flip-flops in the circuit along with additional logic to create a large multiple-input signature analysis register similar to a MISR, but without a characteristic polynomial (Figure 5). During the BIST sequence, the signatures being compacted by the Circular BIST chain are used as the test patterns to be applied to the combinational logic under test during the subsequent clock cycle. As a result, the test patterns are not pseudo-random or pseudo-exhaustive but tend to be more random in nature with the possibility of repeating test patterns.

A smaller MISR is used to further compact test responses in the circular BIST chain, for more efficient system-level access as well as to supply test patterns on the primary inputs and to compact primary outputs. The additional logic added to existing flip-flops in the circuit (Figure 5) facilitates four modes of operation (Table 3) that include normal system mode, scan mode, initialization mode, and BIST mode. Similar BIST approaches include Circular Self-Test Path (CSTP) [7] and Simultaneous Self-Test (SST) [8]. However, it should be noted that CSTP has only two modes of operation (system mode and BIST mode) while SST has three modes of operation (system mode, scan mode, and BIST mode)

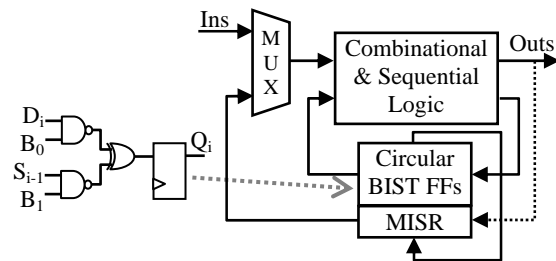


Figure 5. Circular BIST Flip-Flop and Architecture

and are best suited for manufacturing level testing from wafer to board level testing.

**Table 3. Circular BIST Modes of Operation**

B1	B0	Mode of Operation
0	0	Initialization (reset) mode ( $Q_i \leftarrow 0$ )
0	1	System (normal operation) mode ( $Q_i \leftarrow D_i$ )
1	0	Shift (scan) mode ( $Q_i \leftarrow Q_{i-1}$ )
1	1	BIST mode ( $Q_i \leftarrow Z_i \oplus Q_{i-1}$ )

Most practical applications of Circular BIST in production VLSI devices have obtained fault coverage ranging from 90% to 97% of all single stuck-at gate level faults. However, low fault coverage has been observed in applications where the number of dependencies of the largest logic cone is greater than twice the number of flip-flops in the Circular BIST chain [4]. The low fault coverage is due to limit cycling where only a small subset of the test patterns needed for fault detection is generated by the Circular BIST chain. Fortunately, this is not of concern in the MR since the largest logic cone is about 2000 dependencies while the total number of flip-flops in the Circular BIST chain would be greater than 4,500. Hence, limit cycling should not occur.

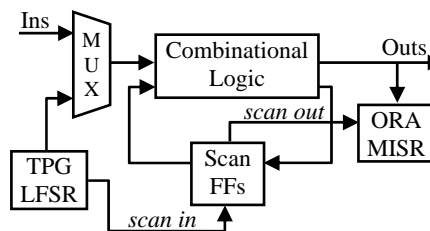
In Circular BIST, there is often a trade-off between test time and fault coverage. However, since the Circular BIST approach creates a closed system, then at some point the test patterns will begin to repeat and no additional fault coverage can be obtained. By changing the ordering of the Circular BIST chain, the point at which test patterns begin to repeat can be altered such that different fault coverage can be obtained for a different ordering of the Circular BIST chain, but this requires synthesis. Alternatively, the scan mode can be used to initialize the Circular BIST chain to a new state and the BIST sequence can be repeated for additional fault coverage. However, this brings out the first problem with Circular BIST in that fault simulation is necessary to determine the fault coverage that can be obtained for a given application. In contrast, the pseudo-exhaustive test patterns provided by the BILBO can ensure 100% fault coverage without the need for fault simulation.

Another potential problem with the Circular BIST approach is the area penalty (three gates per flip-flop: an exclusive-OR gate and two 2-input NAND gates) and performance penalty (two gate delays per critical path, an XOR delay and a 2-input NAND delay). This area overhead is slightly less than that of the BILBO while the performance penalty is about the same as the BILBO. The area overhead for the MR is approximately 14%. The performance penalty is possibly the more critical factor in the MR since the performance tends to be the more difficult design criteria to satisfy for the FPGA-base implementation. The additional logic for each Circular BIST flip-flops would require a minimum of one complete look-up table (LUT), adding a mini-

mum 5,062 LUTs to the FPGA implementation, which could be expected to add significant area and performance penalties to the extent that the FPGA would not operate at the required clock frequency. While selective replacement of existing flip-flops with Circular BIST flip-flops has been shown to greatly reduce the area overhead and performance penalties, there are too many critical paths in the FPGA-based MR implementation to employ selective replacement. In an ASIC implementation on the other hand, Circular BIST provides a good solution to BIST for the MR since performance penalty of the Circular BIST flip-flop logic can be controlled to a greater extent than in a FPGA.

### 3.3. Scan Based BIST

Scan based BIST uses a traditional scan chain similar to Level Sensitive Scan Design for the principal points of controllability and observability by placing a 2-to-1 multiplexer at the input to each existing flip-flop [4][8]. This creates two modes of operation: 1) normal system mode and 2) scan mode. In the scan mode of operation the existing flip-flops form a serial shift register used to shift in test patterns from the TPG (Figure 6). Once a test pattern has been shifted in, one clock cycle in the normal system mode of operation is used to apply the test patterns to the combinational logic of the circuit under test and to clock the output responses back into the flip-flops. The scan mode of operation is then used to shift out the output responses to the ORA while the next test pattern is shifted into the scan chain. The TPG function is performed by an LFSR located at the input to the scan chain while the ORA function is performed by a signature analysis register (SAR) located at the output of the scan chain. This BIST approach has shown to provide high fault coverage in most practical applications and the automatic insertion of traditional scan chains in design for testability provides the basis for automatic implementation for this BIST approach.



**Figure 6. Scan-Based BIST Architecture**

The main problem with the Scan BIST approach lies in being able to test the circuit at the normal system operating frequency, referred to as at-speed testing. This is due to the scan mode control signal that must drive the select input on every multiplexer in the scan chain such that the loading on this signal is about the same as the loading on the system clock. Therefore, to be able to test at-speed with scan BIST, the scan mode control signal must undergo the same design considerations and effort

as that of the system clock itself, which can be done in ASIC-based implementations. In a FPGA-based implementation, dedicated interconnect resources are available for clock routing but these resources are not typically accessible by control signals. This makes the routing of the scan mode control signal more difficult than the clock signals to ensure operation at the desired clock frequency.

In some FPGAs (like the Optimized Reconfigurable Cell Array from Lattice), a 2-to-1 multiplexer is incorporated into the flip-flops in the logic blocks of the FPGA such that scan BIST can be implemented with very little area overhead and performance penalties. Unfortunately for the MR, the Xilinx Virtex FPGA does not provide this feature such that the 2-to-1 multiplexer must be implemented in the LUTs of the logic blocks in the FPGA. The area and performance penalties in scan BIST approach are primarily due to this 2-to-1 multiplexer at the input to each flip-flop. This results in an area overhead of approximately 5% for the MR, not counting the test controller and the LFSRs for the TPG and ORA, which adds about 1% to the total area overhead. The performance penalty is less than that of Circular BIST which is potentially attractive for an FPGA-based implementation of the MR.

Like Circular BIST, test time can be traded-off with fault coverage. To ensure high fault coverage, the characteristic polynomial of the LFSR used for the TPG must be primitive and must be of degree greater than or equal to the number of dependencies of the largest combinational logic cone in the circuit under test. Here again, like the BILBO, the MR would require a primitive polynomial of degree 2000 and only primitive polynomials up to degree 300 have been established. Regardless, the total test time associated with such a polynomial is impractical.

For an actual application of scan BIST, fault simulation will be required to determine the fault coverage that can be obtained with this approach as well as the number of clock cycles (test pattern applications) required to obtain that fault coverage. However, test patterns of a more random type nature can be obtained by switching the scan mode control signal back and forth between system mode and scan mode before a complete test pattern has been shifted into the scan chain from the TPG. This results in the next test pattern to be applied being a function of the output response of the previous test pattern as opposed to a pseudo-random pattern produced directly from the LFSR in the TPG. This technique has been shown to increase fault coverage by several percent while reducing testing time by as much as 50%. The test controller for implementing this technique of switching between scan mode and system mode on an irregular basis would add 1% to the area overhead.

### 3.4. Non-Intrusive BIST

The non-intrusive BIST approach originally developed for high-speed, pipelined data-path circuitry [4][9] was also considered for the MR (Figure 7). This BIST approach uses a MISR as both the ORA and TPG function with two modes of operation: 1) normal system mode and 2) BIST mode. Therefore, like Circular BIST, the output responses being compacted as signatures by the MISR are simultaneously used as the test patterns applied to the circuit under test. The BIST circuitry is located outside the circuit under test such that there is no performance penalty incurred by the circuit under test. This BIST approach has been shown to work well in high-speed, pipelined data-path structures like self-routing switching systems and those similar to the MR, obtaining fault coverage ranging from 90% to 95% in most reported applications [9]. The size of the MISR is typically chosen to be either the number of primary inputs or the number of primary outputs of the circuit under test, whichever is greater. This would correspond to 47-bits in the case of the MR due to the number of primary outputs. As a result, the area overhead is typically very small. The area overhead is approximately 0.2% in the case of the MR.

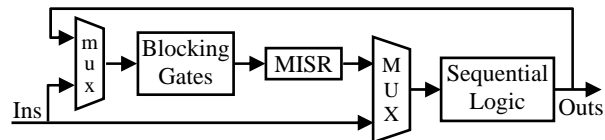


Figure 7. Non-Intrusive BIST Architecture

While the test patterns produced by this approach are more random-like, as in the case of Circular BIST, the system is closed and test patterns will begin to repeat at some point with no additional fault coverage obtained. This problem can be overcome to a certain extent by re-seeding the MISR to put the system into a different starting state once limit cycling has occurred. However, faults that are resistant to random patterns are even more difficult to detect with this BIST approach than with the other three BIST approaches. For example, consider the time stamp counters at the final sorting stage of the MR. These 16-bit counters require 65,536 consecutive addresses from the FFT and programmable threshold sorter to be the same in order to count through a complete cycle of the counter. Based on logic simulations of this BIST approach it was observed that, with the random-type test patterns produced by this BIST approach, the time stamp count rarely gets above a count of eight before resetting to zero. While this BIST approach overcomes the area and performance penalties of the other BIST approaches, it is very dependent on fault simulations to determine the fault coverage that can be obtained, the length of the test sequence needed to obtain that fault coverage, and any re-seeding of the MISR needed to obtain that fault coverage.

#### 4. SUMMARY AND CONCLUSIONS

The Monobit FFT Receiver poses a number of interesting obstacles to the implementation of BIST even though the architecture is nicely pipelined. The primary problem is that the size of the circuit (5,062 flip-flops plus an additional 106,553 combinational logic gates) coupled with the application of a new test pattern only once every 16 clock cycles results in extremely long fault simulation times. In addition, the number of collapsed single stuck-at gate-level faults is also over 100,000. Simulation of the complete MR for approximately 160 clock cycles (corresponding to the application of only 10 test patterns) required about 20 hours on a Sun Sparc Workstation to simulate 500 faults in the circuit. Using statistical random sampling of the complete collapsed fault list, a set of 1000 randomly sampled faults was obtained for a more practical fault simulation time. However, tens of thousands of random or pseudo-random test patterns are typically required to obtain near 90% fault coverage with most of these BIST approaches; for the MR, this corresponds to hundreds of thousands of clock cycles to obtain this number of test patterns (due to the clock enables).

After three months of almost continuous fault simulation with a randomly sampled set of 1000 faults, we had obtained only about 60% fault coverage for the MR with the data-path BIST approach. As in the case of most BIST approaches, the final 30% of fault coverage generally takes about four times as many test patterns to obtain as compared to the number of test patterns needed to obtain the first 60% of fault coverage. Therefore, for an accurate evaluation of these BIST implementations in the MR in terms of fault coverage, a fault simulator hardware accelerator would be required. Alternatively, a smaller version of the MR could be simulated such as a 32-point FFT based design instead of the current 256-point FFT based design. The size of the circuit under test would be about 1/8 that of the current version such the much faster fault simulation times could be obtained. However, in order to extrapolate the fault coverage that would be obtained in the full Monobit FFT Receiver, additional sizes should be considered, such as a 64-point FFT (which would be only 1/4 that of the current version) and a 128-point FFT (which would be only 1/2 that of the current version). If these other sizes are not considered then the effect of the increased logic cone size due to the larger FFT function will be ignored and inaccurate fault coverage results would be obtained. While each of these fault simulations will be faster than the full 256-point FFT based design, collectively they still exceed practical fault simulation time.

The recommended implementation of BIST for an FPGA-based MR is the non-intrusive BIST approach originally developed for high-speed data-path circuitry.

This approach provides the lowest area overhead with no performance penalty; the latter being the more critical design consideration for the FPGA-based implementation. While this BIST approach may not obtain the level of fault coverage produced by the other three approaches in most applications, this is the best BIST approach of the four that can realistically be implemented in a Xilinx Virtex-E FPGA and still meet the area and performance requirements. For an ASIC implementation, on the other hand, Circular BIST is the recommended BIST approach for implementation in the Monobit FFT Receiver, since area and performance penalties can be more effectively controlled in the ASIC implementation while the approach can be expected to produce high fault coverage without the need for large primitive polynomials for LFSR construction as is the case in the BILBO and without the need for long test times associated with scan BIST approaches.

#### REFERENCES

- [1] D. Pok, H. Chen, J. Schamus, C. Montgomery, J. Tsui, "Chip Design for Monobit Receiver", *IEEE Trans. on Microwave Theory & Techniques*, vol. 45, no. 12, pp. 2283-2295, 1997.
- [2] D. Pok, H. Chen, C. Montgomery, J. Tsui, "ASIC for 1-GHz Wide Band Monobit Receiver", *Proc. IEEE International Symp. on Circuits & Systems*, Vol. 2, pp. 77-80, 1998.
- [3] L. Concha, B. Read, D. Bawcom, P. Jarusewic, B. Kadrovach, K. Pedersen, "Simulation Based Design of the Monobit Receiver," *Proc. IEEE National Aerospace and Electronics Conf.*, pp. 451-453, 1998.
- [4] C. Stroud, *A Designer's Guide to Built-In Self-Test*, Kluwer Academic Publishers, 2002.
- [5] B. Koenemann, J. Mucha, G. Zwiehoff, "Built-In Logic Block Observer Techniques," *Proc. IEEE International Test Conf.*, pp. 37-41, 1979.
- [6] C. Stroud, "Automated Built-In Self-Test for Sequential Logic Synthesis," *IEEE Design & Test of Computers*, Vol. 5, no. 6, pp. 22-32, 1988.
- [7] A. Krasniewski, S. Pilarski, "Circular Self-Test Path: A Low Cost BIST Technique for VLSI Circuits," *IEEE Trans. on Computer-Aided Design*, Vol. 8, No. 1, pp. 46-55, 1989.
- [8] P. Bardell, W. McAnney, "Self-Testing of Multi-chip Logic Modules," *Proc. IEEE International Test Conf.*, pp. 200-204, 1982.
- [9] C. Stroud, "Built-In Self-Test for High-Speed Data-Path Circuitry," *Proc. IEEE International Test Conf.*, pp. 47-56, 1991.