Built-In Self-Test of Digital Signal Processors in Virtex-4 FPGAs

Mary D. Pulukuri and Charles E. Stroud
Dept. of Electrical and Computer Engineering
Auburn University
Auburn, Alabama 36849
mdp0005@auburn.edu strouce@auburn.edu

***Abstract*— We present a Built-In Self-Test (BIST) approach for testing and diagnosing the embedded digital signal processors (DSPs) in Xilinx Virtex-4 series Field Programmable Gate Arrays (FPGAs). The BIST architecture and configurations needed to test these programmable DSPs in all of their modes of operation are presented along with fault injection and timing analysis of the BIST configurations.**

Keywords - Built-In Self-Test, multiplier test, adder test, digital signal processor, Field Programmable Gate Array, Virtex-4

I. INTRODUCTION AND BACKGROUND

Built-In Self-Test (BIST) approaches have been developed for Field Programmable Gate Arrays (FPGAs) by programming some of the configurable logic blocks (CLBs) to function as Test Pattern Generators (TPGs) and Output Response Analyzers (ORAs) to test the remaining programmable logic and interconnect resources [1]. In addition, diagnostic procedures have been developed to identify the faulty logic resources in FPGAs [2]. However, there has been little work in the area of testing embedded digital signal processor (DSP) cores in FPGAs. Our goal was to develop a BIST approach for DSP cores in Xilinx Virtex-4 FPGAs.

The historical roots of DSPs in FPGAs began the early 1990s with the simple $N\times N$ arrays of unit cells. In Xilinx 4000 and Spartan series FPGAs, the unit cell consisted of a CLB and routing, with dedicated carry logic in the CLB for fast adders. In Virtex and Spartan-2 FPGAs with an $M\times N$ array of unit cells, the CLBs incorporated AND gates along with fast carry logic to implement array multipliers. Virtex-II and Spartan-3 FPGAs incorporated an 18×18-bit multiplier located adjacent to each 18K-bit block random access memory (RAM). Virtex-4 and Virtex-5 FPGAs have embedded DSP cores which include multipliers, adders, and logic units for performing complex computations. In addition to these specific units, the DSPs are primarily composed of multiplexers and flip-flops that are relatively straightforward to test. Hence, the multiplier and adder components pose the biggest testing challenges.

The data sheets for most FPGAs are vague in terms of the architectures used to implement these multipliers and adders, with only a mention that the multiplier is based on a modified Booth architecture in one application note [3]. However, from the data sheets, it is obvious that sequential logic architectures are not used since there are propagation delay specifications but no mention of clock cycle latency. Choices for combinational logic multipliers include array, Booth, modified Booth, Wallace tree, and modified Booth/Wallace tree multipliers. Based on area and performance analysis, we conclude that the latter is most likely used for DSPs in Virtex-4. The adder for combining the final partial products of the multiplication process has been separated from the multiplier to provide independent user access for addition and accumulator functions. The choices for combinational logic adders include ripple carry, carry select, carry save, and carry look ahead (CLA) adders [4]. Based on area and performance analysis, we conclude that CLA is the most likely adder used in the DSP. This is supported by the fact that a CLA adder is typically used in modified Booth, Wallace tree, and modified Booth/Wallace tree multipliers to sum the final partial products.

In this paper, we present a BIST approach for the DSPs in Virtex-4 FPGAs. We begin with a brief overview of the architectural features of Virtex-4 series FPGAs and their DSPs in Section II. This is followed by an overview of prior work related to our BIST for DSPs in Section III. Section IV presents the BIST architecture and details of the BIST configurations for the DSPs in all of their modes of operation. Experimental results, including analysis of timing and fault detection capabilities, are presented in Section V. Finally, the paper is summarized and concludes in Section VI.

II. OVERVIEW OF VIRTEX-4 ARCHITECTURE

The basic architecture of Virtex-4 FPGAs is illustrated in Fig. 1 and consists of columns of CLBs, programmable input/output blocks (IOBs), block RAMs, and DSPs [5]. The number of rows and columns of CLBs, IOBs, block RAMs, and DSPs varies with the size and family (LX, SX, or FX) of Virtex-4 FPGA. Ranges in the total number of elements are included in the legend of Fig. 1.

Two DSP slices form a DSP tile. A simplified version of the architecture of a DSP slice is shown in Fig. 2. Each DSP slice has a 2-port, 18×18-bit, two's-complement multiplier and a 3-port, 48-bit adder/subtractor [6]. The A and B ports of the DSP slice provide the 18-bit inputs to the multiplier [6]. The partial products from the multiplication are sign-extended to 48-bits and fed to the adder/subtractor through the X and Y

This work was sponsored by the National Security Agency under contract H98230-04-C-1177 and supported in part by the National Science Foundation Grant CNS-0708962.

multiplexers. The C port is shared by both DSP slices in the DSP tile and provides 48-bit access to the adder/subtractor through the Y and Z multiplexers. The DSP slice has an optional accumulator register, denoted by P, which also provides 48-bit feedback access to the adder/subtractor through the X and Z multiplexers, as shown in Fig. 2. The select inputs to the X, Y and Z multiplexers are dynamically controlled by the DSP OPMODE signals [6]. The adder/subtractor performs P=Z±(X+Y+Cin) producing a 48-bit two's complement result, where P is the output port, Cin is the carry-in, and X, Y and Z are 48-bit multiplexer busses [6]. We assume the architecture to be a 2-stage CLA adder as illustrated in Fig. 3.

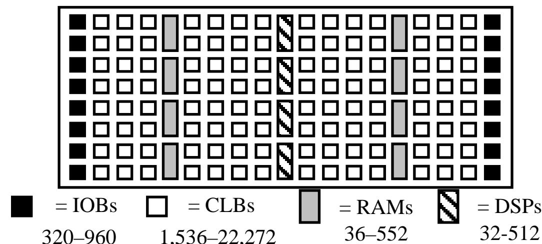


Figure 1. Basic Virtex-4 Architecture

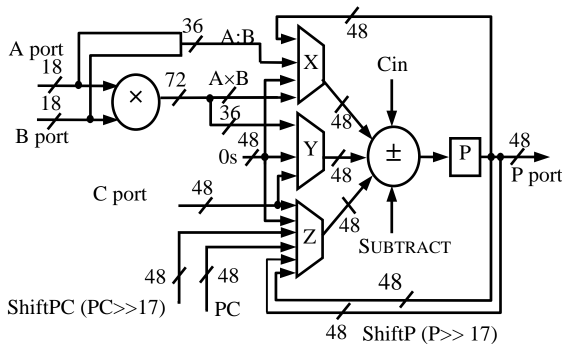


Figure 2. Virtex-4 DSP Slice Architecture [5]

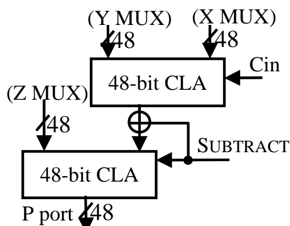


Figure 4. 2-stage CLA adder

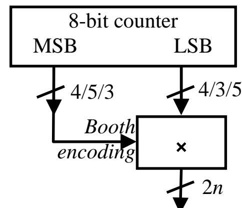


Figure 3. Multiplier BIST Approaches

Input and output ports have optional pipeline registers. Each pipeline register introduces a delay of one clock cycle. Additional multiplexers controlled by configuration memory bits select the number of pipeline registers in the I/O paths of the DSP slice. The control signals to the DSP (including the carry-in, clock enables, and resets on various pipeline registers) can be configured to be either active high or active low. The B and P ports of the DSP slices in a column of DSPs can be cascaded (the cascaded P port is indicated by PC in Fig. 2) to form larger DSPs. The P port can be shifted right by 17 bits (indicated by ShiftP in Fig. 2) before being fed to the adder through the Z multiplexer. The cascaded P port from the previous slice can likewise be shifted by 17 bits (indicated by ShiftPC in Fig. 2) [6].

III. OVERVIEW OF PRIOR WORK

The authors in [7] give a diagnostic procedure for detecting faulty blocks under test and mention that DSP cores in Virtex-4 FPGAs may be tested by applying pseudo-random patterns generated by linear feedback shift registers (LFSRs) but they do not provide specific test algorithms or test sequences to test the adder and the multiplier in the DSPs and there is no mention of the fault coverage obtained. Two approaches for testing modified Booth and modified Booth/Wallace tree multipliers have been proposed [8][9]. Both use an 8-bit binary counter, as illustrated in Fig. 4, to generate the test patterns needed to test any size multiplier and obtain fault coverage in the 99% range [8]. One approach proposes what we will refer to as the 4×4 algorithm where the four most significant bits (MSBs) of the counter are applied to one multiplier input port while the four least significant bits (LSBs) are applied to the other input port [9]. Starting at the LSB of the multiplier port, the four counter bits are applied and repeated for each group of four inputs. This approach was previously used for BIST of the multipliers in Virtex-II FPGAs [10]. The other approach proposes what we will refer to as the 5×3 algorithm where the five MSBs are applied to the multiplier port containing Booth encoding and the three LSBs are applied to the other port [9].

Unfortunately, we do not know which of the two multiplier input ports is associated with Booth encoding. Therefore, since the 5×3 algorithm was claimed to be the better testing approach, we run both 5×3 and 3×5 test algorithms to ensure that we have applied the five MSBs to the Booth encoding port during one of the two test sequences. This doubles the test time, but we only increase the number of test vectors from 256 to 512. We found via single stuck-at gate-level fault simulation that the combination of these two tests provides the highest fault coverage for 8-bit modified Booth, Wallace tree, and modified Booth/Wallace tree multipliers, as can be seen in Table I, which should also be the case of any size multiplier.

TABLE I. MULTIPLIER TEST FAULT COVERAGE

Algorithm	%FC
4×4	98.7%
5×3	99.0%
3×5	99.2%
4×4 & 5×3	99.2%
4×4 & 3×5	99.4%
5×3 & 3×5	99.9%

A BIST approach for carry look-ahead adders (CLAs) was proposed in [11] which produces a $2\times(N+1)$ test vector sequence to test an N-bit adder. The TPG implementation requires an N+1-bit shift register, N XOR gates, N XNOR gates and an inverter. However, we found via fault simulation that two additional patterns are needed to obtain 100% fault coverage. These missing test patterns can be obtained by replacing the inverter by a flip flop with Qbar output driving the input to the shift register as shown in Fig. 5. This modification produces a $2\times(N+2)$ test vector sequence giving 100% single stuck-at gate-level fault coverage.

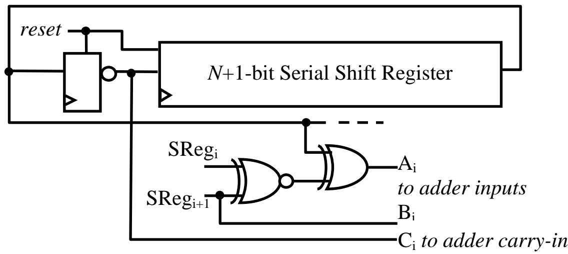


Figure 5. Modified Adder Test Algorithm

IV. BIST FOR VIRTEX-4 DSPS

The DSP BIST architecture is shown in Fig. 6. Two TPGs drive identical test pattern sequences to alternate rows of identically configured DSP tiles under test. Each TPG drives both DSP slices in a tile in order to facilitate testing the cascade modes of operation. The bottom DSP slice in any given DSP tile is denoted *s0* and the top DSP slice in the tile is denoted *s1*. The outputs of each DSP slice are compared by two sets of ORAs, as shown in Fig. 6, forming two circular comparison chains – one chain compares *s0* DSPs and the other compares *s1* DSPs. As a result, the outputs of each DSP slice are monitored by two ORAs and compared with the outputs of two different DSPs of the same slice type. This overcomes the problem of using circular comparison when testing cascade modes of operation pointed out in [7]. The use of multiple TPGs overcomes the problem of faults in the TPG escaping detection [1]. The ORAs observing DSPs at the bottom of the column include a clock enable controlled by the TPGs to disable these ORAs for test vectors where the unconnected cascade inputs to those DSPs will cause a mismatch. The CLBs used to construct the ORAs are assumed to have been tested and found to be fault-free using BIST configurations specific to CLBs, similar to those described in [12].

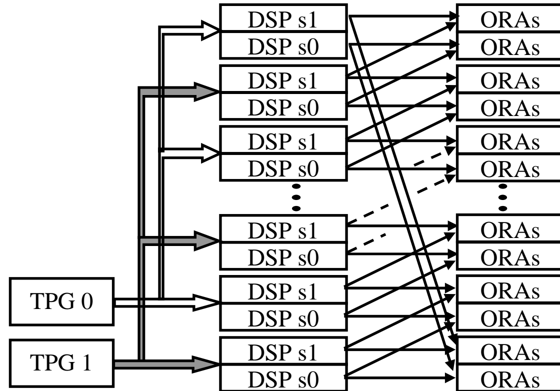


Figure 6. DSP BIST Architecture

At the end of the BIST sequence, the contents of the ORA flip-flops can be obtained via partial configuration memory readback. However, we are only interested in obtaining ORA contents for diagnosis when faults have been detected. Therefore, we exploit the dedicated carry logic in CLBs, as illustrated in Fig. 7, to create an iterative OR chain that provides a single bit pass/fail indication at the end of the BIST sequence such that partial configuration memory readback is not required to determine pass/fail status. This reduces the total test time for fault-free BIST sequences.

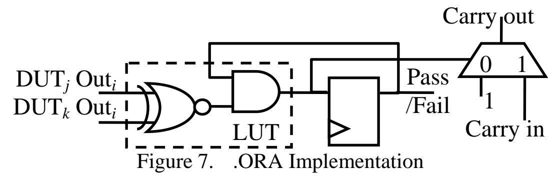


Figure 7. .ORA Implementation

All DSPs are tested concurrently such that the length of the BIST sequence is independent of the array size. The DSPs are repeatedly reconfigured, and retested, until they have been tested in all modes of operation. Since download time is a function of the size of the array, an important goal is to reduce the number of times the DSPs must be reconfigured without sacrificing fault detection. By maintaining constant placement and routing of TPGs, ORAs, and DSPs under test, partial reconfiguration can write only the portions of configuration memory that contain configuration bits that control the DSPs.

The Virtex-4 DSPs are tested in five BIST configurations summarized in Table II. Column 1 gives the BIST download configuration number, column 2 shows the number of pipeline registers in the I/O paths of the DSP slices, column 3 indicates the programmable active level of control signals described in Section II, column 4 denotes whether the B port is in direct or cascade mode of operation, and column 5 describes the test sequences applied for each BIST configuration. The multiplier is tested during the first, second, and fourth applications of BIST sequence. The adder is tested during the third and fifth applications of the BIST sequence. The cascade modes for slice 1 and slice 0 DSPs are tested during the sixth and seventh applications of the BIST sequence, respectively. BIST configurations 2 and 3 are run twice each by changing the TPG control signals to run the multiplier and adder tests during the same BIST configuration download. As a result, each Virtex-4 DSP slice is tested in seven applications of the BIST sequence (indicated by the number in the "Test Modes Applied" columns). This reduces total test time as seven tests are run in only five downloads to the FPGA.

TABLE II BIST CONFIGURATIONS

BIST Config #	Pipeline Registers	Signals Active Level	B Input Source		Test Modes Applied		
			Slice 0	*Slice 1*	*Mult*	*Add*	*Casc*
1	All Regs=0	High	Direct	Direct	#1		
2	All Regs=1	High	Direct	Direct	#2	#3	
3	A/Breg=2, Others=1	Low	Direct	Direct	#4	#5	
4	Preg=1, Others=0	High	Direct	Cascade			#6
5	Preg=1, Others=0	Low	Cascade	Direct			#7

The BIST sequences for the three test modes (multiply, adder and cascade) are summarized in Table III. Each BIST sequence is 1,024 clock cycles and is divided into four groups of 256 clock cycles (denoted 'ccs' in the table) each. During each group of 256 clock cycles, test patterns are applied via all possible inputs to the DSP slices. The multiplier 5×3 and 3×5 test algorithms are applied alternately through the four groups of 256 clock cycles during the multiplier test sequence.

The 2-stage CLA adder is tested one stage at a time with the first stage adder (see Fig. 3) tested during the first and third groups of 256 clock cycles and the second stage adder is tested during the second group 256 clock cycles. Since the C port is

the only direct 48-bit access to the adder and the accumulator is the only other 48-bit access to the adder, two clock cycles are required to apply each test vector during the adder test sequence. During the first clock cycle, the 48 LSBs of the 97-bit test vector are loaded into the accumulator register (denoted by the top P equation in each entry of the adder test column of Table III) via the Y or Z multiplexers. During the second clock cycle, the 48 MSBs of the test vector are applied via the C port through the Y or Z multiplexers while the 48-bit vector in the P register is applied to the X or Z multiplexers. Also during this second clock cycle, the test vector value for the Cin or Subtract signals (depending on the stage of the adder under test) is applied to complete the test vector to the adder. Weighted pseudo-random patterns are applied to test the various clock enables and resets inputs to the DSP slices during the third and fourth groups of 256 clock cycles in the multiplier and adder test sequences. The two DSP slices in each tile are controlled independently during the cascade test sequence to test the dedicated cascade multiplexers and interconnect between adjacent DSPs. This is indicated by different equations for the P registers in Table III where P0 denotes slice 0 and P1 denotes slice 1.

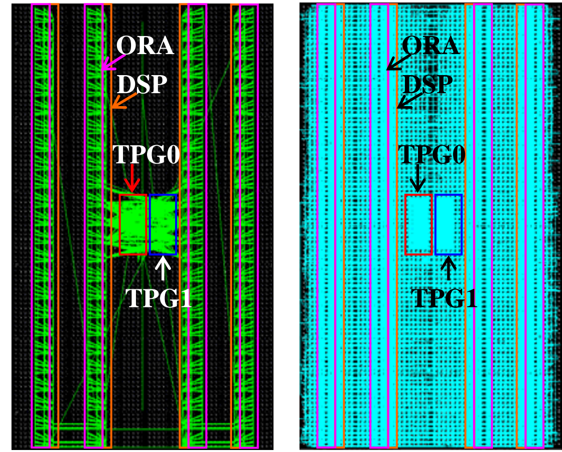
TABLE III BIST SEQUENCES

Test	Multiply	Adder	Cascade
First 256 ccs	P = A×B	P = Z(C) P = X(P)+Y(C)	P1 = A:B+Z(PC) P0 = Z(C)
Second 256 ccs	P = A×B	P = Y(C) P = X(P)+Z(C)	P1 = A:B+Z(ShiftPC) P0 = Z(C)
Third 256 ccs	P = A×B+C	P = Z(C) P = Y(C)+Z(P)	P1 = Z(C) P0 = A:B+Z(PC)
Fourth 256 ccs	P = A:B+C	P = Y(C) P = Y(C)+Z(ShiftP)	P1 = Z(C) P0 = A:B+Z(ShiftPC)

V. EXPERIMENTAL RESULTS

We developed two C programs to automatically generate the DSP BIST configurations for any size and family of Virtex-4 FPGAs. *V4DSPBIST.exe* produces a DSP BIST template in Xilinx Design Language (XDL) format which is then converted to NCD format for routing with Xilinx place and route software. The routed NCD file is converted back to XDL for modification by *VDSPMOD.exe* to generate the five DSP BIST configurations. The final XDL files are converted to NCD format from which the actual download configuration bit files are generated. The NCD files of the DSP BIST template can be viewed in FPGA Editor, a Xilinx tool that gives a graphical representation of the FPGA. For example, Fig. 8a shows the unrouted DSP BIST template for a Virtex-4 SX-35 and Fig. 8b shows the routed DSP BIST template.

Each TPG drives both DSPs in the tile (see Fig. 9) and consists of a 10-bit counter for the multiplier test and overall control of the BIST sequence, a 50-bit shift register for the adder test, two linear feedback shift registers (LFSRs) for producing weighted pseudo-random control signals, and a finite state machine for controlling OPMODE inputs. The TPG is written in VHDL (266 lines of code) and synthesizes into 44 CLBs. The 48-bit output of each DSP slice is observed by 48 ORAs in two rows and three columns of CLBs. Each ORA is implemented in one look-up table (LUT) and flip-flop of a CLB. For DSP columns to the left of the center column, the ORAs are located in three consecutive columns of CLBs to the left of the DSP. For DSP columns to the right of the center column, the ORAs are located in three consecutive columns of CLBs to the right of the DSP. To avoid the PowerPC modules in FX family devices larger than FX40, the ORAs are located to the right of the DSPs located left of the center column and to the left of the DSPs located right of the center column.



a) unrouted b) routed
Figure 8 BIST Template as Seen in FPGA Editor

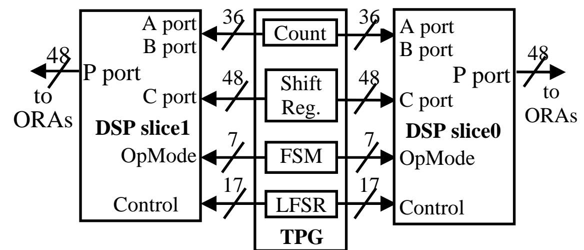


Figure 9 DSP TPG Architecture

Fig. 10 shows the individual fault coverage for each of the seven DSP BIST sequences and the cumulative fault coverage for all the seven tests. Fault detection capabilities were verified by injecting faults in the configuration memory bits controlling DSP cores of Virtex-4 FX12 devices. Cumulatively, 97.4% of injected faults were detected by the seven BIST sequences. Of the 154 faults injected, four faults were not detected but can be detected by adding two more BIST configurations. The BIST was able to detect faulty DSP cores in engineering samples of Virtex-4 SX35 and LX60 devices. A total of ten SX35 FPGAs were tested along with a total of 18 LX60 FPGAs. DSP BIST of these devices detected as many as five different faulty DSPs in four SX35 engineering sample parts and one faulty DSP each in two of the LX60 engineering sample parts tested.

To determine the maximum clock frequency of DSP BIST, timing analysis was performed on all five BIST configurations for all Virtex-4 FPGAs using the Xilinx timing analysis tool *TRCE.exe*. From the timing analysis results, we observed that the position of the TPG in the array has a significant impact on the BIST clock frequency. Fig. 11 shows the maximum clock frequency of the slowest BIST configuration for all Virtex-4 FPGAs as a function of the TPG position. From the figure we see that the clock frequency is higher when the TPG is placed at the middle of the array when compared to the bottom of the array. Hence, the TPG is placed at the middle of the array for

all devices except the FX12 and FX20 devices that have a PowerPC near the middle of the array.

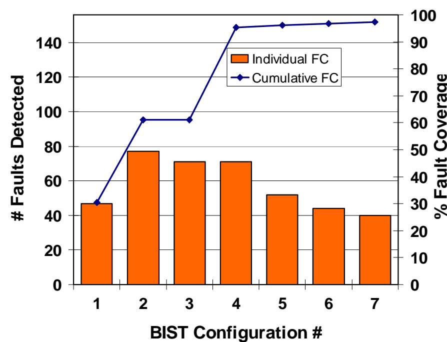


Figure 10 Individual and Cummulative Fault Coverage

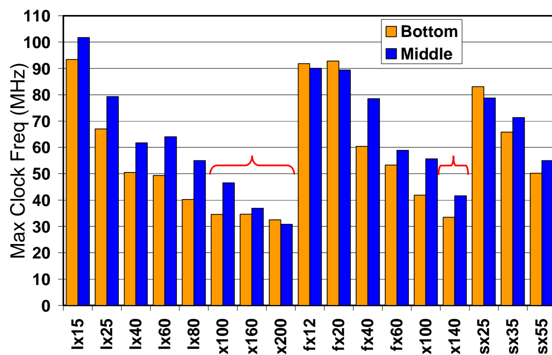


Figure 11 Maximum BIST Clock Frequency

For some of the larger Virtex-4 FPGAs, like LX100, LX160, LX200 and FX140, the maximum clock frequency for BIST is less than 50MHz. For these devices, sub-array testing is done where each half of the array is tested separately. Fig. 12 shows the maximum clock frequency for the sub-arrays based on the position of the TPG in the array. From the figure we see that the maximum clock frequency for the bottom half of the array is less than for the top half of the array when the TPG is placed at the middle of the array. This is because for the top half of the array, the TPG routes across to the DSP tiles and then routes up to the DSP tiles above. But for the bottom half of the array, the TPG routes down to bottom and then routes across to the DSP tiles and up to the DSP tiles above; Hence, the routing path is longer. Therefore, the TPG is placed at the bottom when testing the bottom half of the array achieving similar clock frequencies for both halves of the array.

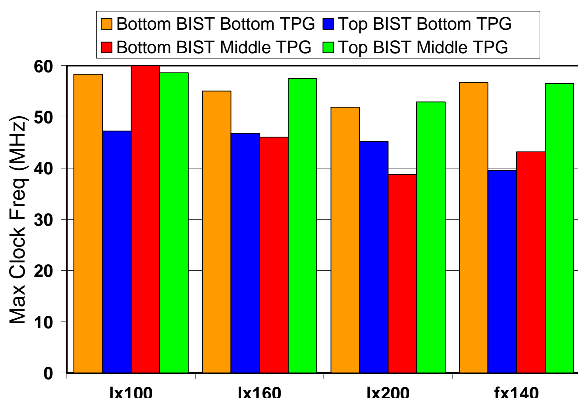


Figure 12 Maximum Clock Frequency for Sub-Arrays

Fig. 13 summarizes total test time for various size devices when using a 32-bit parallel configuration interface. As can be seen, the BIST execution time is negligible compared to the download time. Readback time is eliminated in the fault-free case or when diagnostics is not required or desired.

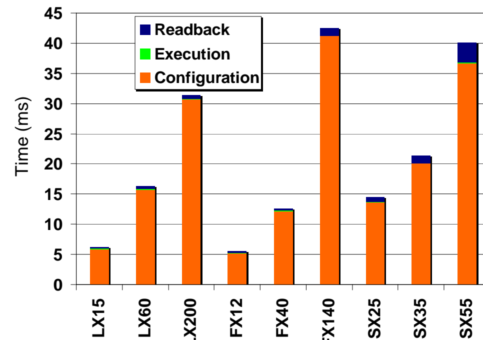


Figure 13 32-bit Parallel Interface Test Time

VI. SUMMARY AND CONCLUSIONS

A BIST approach for DSP cores in Virtex-4 FPGAs was presented where a minimum set of five BIST configurations was developed to test the logic in DSP cores. Fault detection was evaluated by injecting faults in the DSP cores of several FPGAs where 97.4% of the faults injected were detected. In addition, the DSP BIST was able to detect faulty DSP cores in some Virtex-4 SX35 and LX60 engineering sample devices. Timing analysis was performed for all Virtex-4 FPGAs to verify that the maximum clock frequency of DSP BIST is at least 50MHz. Sub-array testing is done on larger devices where the maximum clock frequency is less than 50MHz and where the power dissipation due to testing over 500 DSPs simultaneously could cause problems in the system.

REFERENCES

[1] L-T Wang, C. Stroud and N. Touba, *System-on-Chip Test Architectures*, Elsevier, 2007

[2] M. Abramovici and C. Stroud, "BIST-Based Test and Diagnosis of FPGA Logic Blocks," *IEEE Trans. on VLSI Systems*, vol. 9, no. 1, pp. 159-172, 2001

[3] "Using Embedded Multipliers in Spartan-3 FPGAs," Application note XAPP467 (v1.1), Xilinx, Inc., 2003

[4] C. Nagendra, M-J Irwin and R-M Owens, "Area-Time-Power Tradeoffs in Parallel Adders," *IEEE Trans. on Circuits and Systems II*, vol 43, no. 10, pp 689-702, 1996

[5] "Virtex-4 User Guide," UG070 (v1.5), Xilinx, Inc., 2006

[6] "XtremeDSP for Virtex-4 User Guide," UG073 (v2.1), Xilinx, Inc., 2005

[7] A. Sarvi and J. Fan, "Automated BIST-based diagnostic solution for SOPC," *Proc. Int. Conf. on Design & Test of Integrated Systems in Nanoscale Technology*, pp. 263 – 267, 2006

[8] D. Gizopoulos, A. Paschalis and Y. Zorian, "Effective Built-In Self-Test for Booth Multipliers," *IEEE Design & Test of Computers*, vol. 15, no. 3, pp. 105-111, 1998

[9] A. Paschalis, N. Kranitis, M. Psarakis, D. Gizopoulus and Y. Zorian, "An Effective BIST Architecture for Fast Multiplier Cores," *Proc. Design, Automation and Test in Europe Conf.*, pp. 117-121, 1999

[10] C. Stroud and S. Garimella, "BIST and Diagnosis of Multiple Embedded Cores in SoCs," *Proc. Int. Conf. on Embedded Systems & Applications*, pp. 130-136, 2005

[11] H. Al-Asaad, J. Hayes, and B. Murray, "Scalable Test Generators for High-Speed Datapath Circuits," *J. Electronic Testing: Theory & Applications*, vol 12, pp. 111-125, 1998

[12] B. Dutton and C. Stroud, "Built-In Self-Test of Configurable Logic Blocks in Virtex-5 FPGAs," *Proc. Southeastern Symp. on System Theory*, 2009