

On Built-In Self-Test for Adders

Mary D. Pulukuri and Charles E. Stroud

Dept. of Electrical and Computer Engineering, Auburn University, Alabama

Abstract - We evaluate some previously proposed test approaches for various types of adders in an attempt to find an architecture-independent algorithm for testing adders in embedded Digital Signal Processors (DSPs) in Field Programmable Gate Arrays (FPGAs). We find that a minor modification to a previously proposed Built-In Self-Test (BIST) approach provides the highest fault coverage for most types of adders and, equally important, it is simple to implement.

Keywords: built-in self-test, carry look ahead adders, ripple carry adders

Introduction

While developing Built-In Self-Test (BIST) approaches for Xilinx Virtex-4 Field Programmable Gate Arrays (FPGAs), we investigated various test approaches for adders. The target adder has three input ports of 48-bits each and was incorporated in the embedded Digital Signal Processor (DSP) cores in Virtex-4 FPGAs. Unfortunately, the architecture and implementation of the adder are not explicitly defined in the data sheet [1]. Based on the timing specifications, we rule out sequential logic implementations leaving ripple carry, carry select, and carry look ahead (CLA) adders. Since the adder is used to sum the final partial products from the multiplier portion of the DSP, the CLA adder seems the most likely candidate based on data sheet timing specifications and the fact that CLA adders are typically used for summing the final partial products in modified-Booth/Wallace-tree multipliers [2].

Carry Look Ahead Adders

The basic structure of a CLA adder is summarized in Figure 1 where 4-bit implementations are typically used due to fanin limitations. Each adder cell takes two inputs (A_i and B_i) and a carry-in (C_i) to produce sum (S_i), propagate (P_i), and generate (G_i) signals. The P_i and G_i signals from the adder cells are used in conjunction with the carry-in signal in the look ahead carry unit (LCU) to produce carry signals to subsequent adders as summarized by the LCU logic equations in Figure 1. There are two approaches to implementing the adder cells as summarized by the adder logic equations in Figure 1; these include generating the P_i signal via an OR gate (denoted in Figure 1 as the POR implementation) and generating the P_i signal via the exclusive-OR (XOR) gate (denoted in Figure 1 as PXOR implementation) used for the sum.

In order to construct larger CLA adders from the basic 4-bit CLA adder illustrated in Figure 1, there are three basic approaches. In the ripple CLA, the carry-out from one LCU is connected to the carry-in of the next LCU [3]. Alternatively, the LCU itself can also produce a propagate signal (PG) and a generate signal (GG) which can be fed to a second stage of LCU such that a 16-bit CLA adder can be constructed. For even larger adders, such as the 48-bit adder in the Virtex-4 DSP cores, one can either ripple the carry outputs of the second stage LCU (which we refer to as a ripple LCU) or include additional stages of LCUs (which we refer to as multi-stage LCU). Therefore, our goal is to find an architecture-independent test algorithm that provides high fault coverage regardless of the adder implementation.

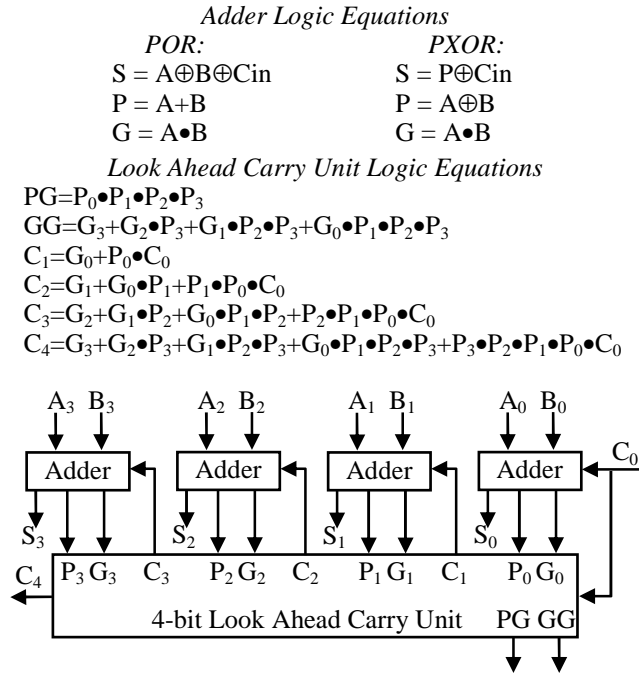


Figure 1: Carry Look Ahead Adder

Prior Testing Approaches

A minimum set of 10 test vectors was proposed to detect all single stuck-at faults in a 4-bit CLA adder in which the P_i signal in the adder is produced by an OR gate (POR implementation) [3]. This was extended to a set of 11 test vectors to test any size ripple CLA adder [3]. Another test algorithm was proposed for BIST implementation which produces a $2 \times (N+1)$ test vector sequence to test an N -bit adder [4]. The test pattern generator (TPG) implementation for this algorithm requires an $N+1$ bit shift register and an inverter to form a twisted ring counter in conjunction with N XOR gates and N XNOR gates as shown in Figure 2. This BIST circuit is easy to implement and the test vector sequence produced by this circuit is illustrated in Figure 2 for $N=4$.

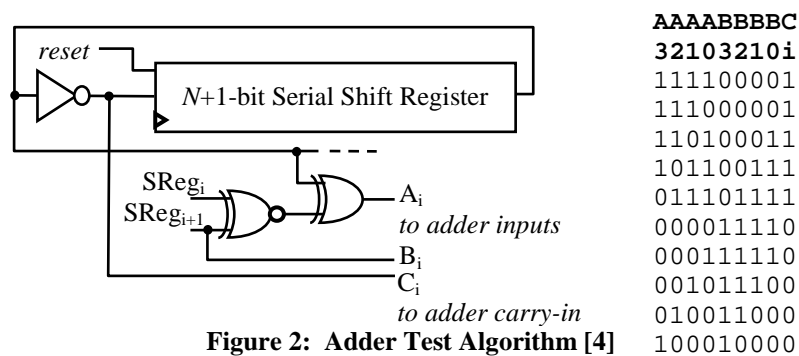


Figure 2: Adder Test Algorithm [4]

When applying these two test algorithms to various implementations of CLA adders (as well as a simple ripple carry adder), we find that both the test algorithms lack the ability to provide 100% single stuck-at fault coverage, as summarized in Table 1 for 48-bit adders implementations. Note that all CLA implementations use the OR implementation for the propagate signal. While the minimum set of test vectors described in [3] provides 100% fault coverage for its target ripple CLA implementation as well as for the simple ripple carry adder, it fails to provide complete fault

coverage for other CLA implementations. The BIST algorithm presented in [4] provides the best overall fault coverage for all CLA adder implementations but fails to provide 100% fault coverage.

Table 1: Stuck-at Fault Simulation Results for 48-bit Adders

Adder Implementation	Gate Delays	Number of Faults	Test Algorithm Vector Set		
			vector set [3]	BIST [4]	Modified BIST
Ripple Carry Adder	96	1296	100%	99.9%	100%
Ripple CLA	28	1392	100%	99.9%	100%
Ripple LCU	12	1542	95.7%	99.9%	100%
Multi-stage LCU	10	1506	95.9%	99.9%	100%

Modification to BIST Approach

Upon investigation of the undetected faults from the BIST-based test vectors, we observed that two additional vectors were needed to detect the remaining faults and provide 100% fault coverage. These two missing vectors can be produced with a simple modification to the TPG implementation by replacing the inverter in the twisted ring counter with a flip-flop, as illustrated in Figure 3, and using the Q-bar output of the flip-flop to provide the inversion for the twisted ring counter. This minor modification to the TPG produces a $2 \times (N+2)$ test vector sequence, a sample of which is illustrated in Figure 3 for $N=4$ where the new test vectors are noted. With this simple modification, 100% stuck-at fault coverage is obtained for all adder implementations, as summarized in the “Modified BIST” column of Table 1.

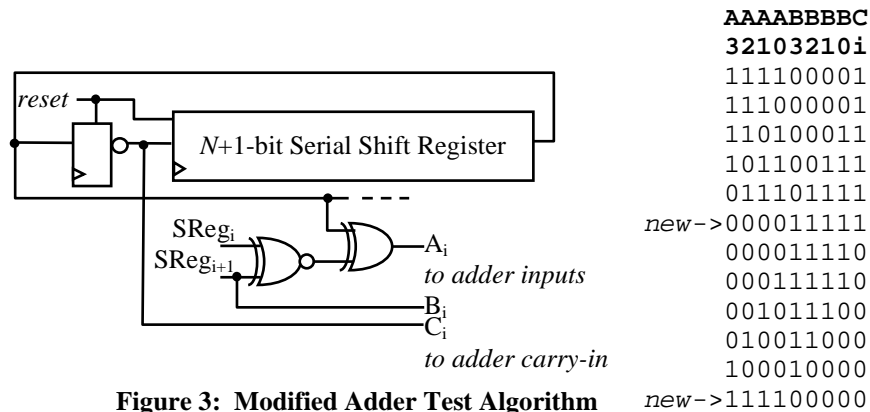


Figure 3: Modified Adder Test Algorithm

Application to Embedded DSPs in FPGAs

Virtex-4 FPGAs incorporate from 32 to 512 embedded DSPs depending on the family and size of the device. Each DSP has an 18×18 -bit 2's complement multiplier followed by three multiplexers (denoted X, Y and Z) and a 3-port adder/subtractor, as illustrated in Figure 4a. The select inputs to the X, Y and Z multiplexers are dynamically controlled by seven OPMODE input signals. The 3-port adder/subtractor performs $P=Z \pm (X+Y+Cin)$ [1] where X, Y, and Z are 48-bit busses. We assume the implementation is a 2-stage CLA adder as illustrated in Figure 4b. Only the C port input to the DSP provides 48-bit access to the adder via the Y and Z multiplexers. Unfortunately, the concatenation of input ports A and B (denoted A:B) provides only 36-bits since the A and B ports are 18-bits each. The only other 48-bit access to the adder/subtractor is through the accumulator register (denoted P) which provides feedback access to both the X and Z multiplexers [1]. Therefore, application of a single test vector requires two clock cycles: one to load a portion of the test vector in the P register and the second to apply the complete test vector to the adder.

With this limited access, each stage of the adder can be tested in turn, as summarized in Table 3. During the first clock cycle of each test vector application, 48 bits of the 97-bit test vector (including CIN input for stage 1 adder and SUBTRACT input for stage 2 adder) can be loaded into the P register via the Z or the Y multiplexers (depending on the stage of the adder that is being tested) while 0s are applied to the other two multiplexers under the control of the OPMODE signals. Logic 0s are also applied to the CIN and SUBTRACT inputs to facilitate passing the 48-bit portion of the test vector to the P register; note that when testing the second stage adder, however, the 48-bit vector to the P register must be inverted for those cases where the overall test vector will apply a logic 1 to the SUBTRACT input. During the second clock cycle, the 48-bit vector in the P register is applied to the X multiplexer while the remaining 48 bits of the test pattern are applied via the C port to either the Y multiplexer or the Z multiplexer (depending on the stage of the adder that is being tested). During this second clock cycle, logic 0s are applied through the third multiplexer and the appropriate test pattern values are applied to the CIN or SUBTRACT inputs (depending on the stage of the adder that is being tested). These 2-clock cycle test patterns can be generated by the TPG by simply incorporating a clock enable on the shift register such that the complete test pattern is held for two clock cycles while the appropriate OPMODE values control the X, Y, and Z multiplexers to transmit the complete test vector to the adder stage under test.

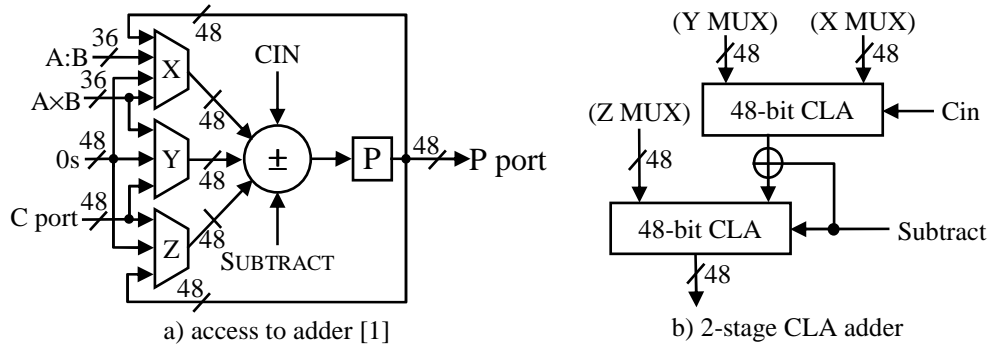


Figure 4: Adder in Virtex-4 DSP

Table 2: Test Pattern Application to 2-Stage Adder

Adder Under Test	Clock Cycle	X MUX	Y MUX	Z MUX
Top Adder	1	0s	0s	C port
	2	P register	C port	0s
Bottom Adder	1	0s	C port	0s
	2	P register	0s	C port

Conclusion

The BIST approach proposed by Al-Asaad, Hayes, and Murray in [4] has proven to be an excellent approach for testing adders; it provides complete stuck-at fault coverage for many different types of adder implementations. Furthermore, the BIST approach is easy to implement as well as easy to modify, as we have shown here in our application to the 3-port, 48-bit adder/subtractor in the embedded DSPs in Virtex-4 FPGAs. Furthermore, a similar 3-port, 48-bit adder/subtractor is incorporated in DSP cores in Virtex-5 FPGAs [5], such that this BIST approach can again be used. It should be noted, however, that this BIST approach is not architecture independent; for example, it does not detect all faults in a CLA implementation that uses the XOR implementation for the propagate signal. On the other hand, the basic method described in [4] can be used to test other structures such as multipliers and arithmetic logic units (ALUs).

References

- [1] Xilinx, "XtremeDSP for Virtex-4 FPGAs," User Guide UG073 (v2.7), Xilinx Inc., 2008.
- [2] A. Paschalis, N. Kranitis, M. Psarakis, D. Gizopoulos and Y. Zorian, "An Effective BIST Architecture for Fast Multiplier Cores", *Proc. Design, Automation and Test in Europe Conf.*, pp. 117-121, 1999.
- [3] S. Kajihara and T. Sasao, "On the Adders with Minimum Tests," *Proc. IEEE VLSI Test Symp.*, pp. 10-15, 1997.
- [4] H. Al-Asaad, J. Hayes, and B. Murray, "Scalable Test Generators for High-Speed Datapath Circuits," *J. Electronic Testing: Theory and Applications*, vol 12, pp. 111-125, 1998.
- [5] Xilinx, "Virtex-5 XtremeDSP Design Considerations," User Guide UG193 (v3.1), Xilinx Inc., 2008.