Level-testability of multi-operand adders

Nobutaka Kito and Naofumi Takagi
Department of Information Engineering, Nagoya University
Nagoya, 464-8603, Japan
{nkito, ntakagi}@takagi.i.is.nagoya-u.ac.jp

Abstract

Level-testability of multi-operand adders consisting of carry save adders is shown by showing test design for them. A multi-operand adder is a main part of a multiplier. $6L+2$ patterns are sufficient to test a multi-operand adder under cell fault model, where L denotes the depth of the multi-operand adder. A test method of the multi-operand adder used as a partial product compressor in a multiplier is also shown. This result gives an upper bound of the number of required test patterns for a multi-operand adder in any multiplier.

1. Introduction

Growth of the number of logic gates integrated in a VLSI chip has made a test of a chip harder. In order to reduce the cost of a test of a VLSI chip, it is important to study test of its component circuits, such as arithmetic circuits.

In this paper, we consider test of multi-operand adders consisting of carry save adders. A multi-operand adder is a main part of a multiplier. We will show that any multi-operand adder consisting of carry save adders is level-testable, by showing test design for it. A circuit is said to be level-testable if it can be tested by a test set whose number of patterns grows linearly with its depth. We will also show a test method of a multi-operand adder in a multiplier.

We can define the level of each carry save adder in a multi-operand adder. We let L denote the largest level of carry save adders in the multi-operand adder. We will show that $6L+2$ patterns are sufficient to test the multi-operand adder under the cell fault model[1] with treating FAs in carry save adders as cells. We will also show that we can feed the $6L+2$ patterns to a multi-operand adder in any multiplier through the partial product generators.

In [2, 3, 4], and [5], C-testable designs of array multipliers were shown. An array multiplier contains a multi-operand adder consisting of serial connection of carry save adders. An arithmetic circuit is said to be C-testable if it can be tested by a test set whose number of patterns is constant independent of its operand size. In [6] and [7], level-testable designs of 4-2 adder trees (also known as 4-2 reduction trees) for high-speed multipliers were shown. A 4-2 adder tree is a multi-operand adder which has regular tree-type structure. In [8] and [9], C-testable designs of 4-2 adder trees for high-speed multipliers were shown. However, testability of multipliers of the other types such as a Wallace multilier[10], which has irregular tree-type structure with reconvergence, has not been shown. This paper gives an upper bound of the number of required test patterns for a multi-operand adder in any multiplier. This upper bound will be useful as a criterion in test design of a multiplier.

This paper is organized as follows. In the next section, we will describe multi-operand adders and the cell fault model. In Section 3, we will show level-testability of multi-operand adders. In Section 4, we will show test design for a multi-operand adder in a multiplier. In Section 5, we will conclude this paper.

2. Preliminaries

2.1. Multi-operand Adders

We consider multi-operand adders consisting of carry save adders(CSAs). As shown in Fig. 1(a), a CSA consists of full adders(FAs). It sums up three input numbers and generates a sum and a carry. We represent the CSA simply as shown in Fig. 1(b). Fig. 2 shows an example of a multi-operand adder consisting of CSAs.

For each FA in a CSA, we name the input terminals as u, v and w, and the output terminals as c and s as shown in Fig. 1 (a). We name the input terminals of a CSA as U, V and W, and the output terminals as C and S as shown in Fig. 1 (b). The terminals U, V, W, C and S consist of the terminals u, v, w, c and s of FAs in the CSA, respectively. For a multi-operand adder, we name inputs as $I_0, \ldots, I_{M-1}$, where M is the number of operands.

1081-7735/08 $25.00 © 2008 IEEE
DOI 10.1109/ATS.2008.40

IEEE computer society

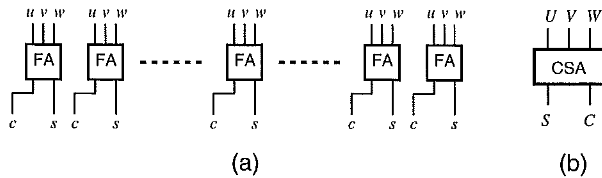

Figure 1. A carry save adder

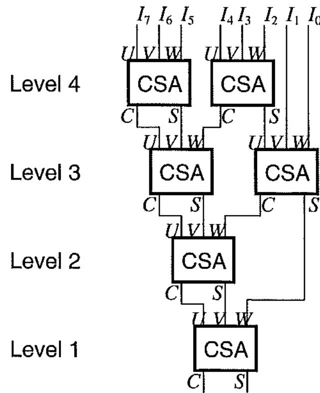

Figure 2. An example of a multi-operand adder

We define a level of each CSA in a multi-operand adder. We let the level of the CSA at the output of the multi-operand adder be 1. For each of the other CSAs, we let its level be one larger than the maximum level of the CSAs connected with its output terminals. We represent the largest level of a multi-operand adder by L. In Fig. 2, the largest level L is four.

2.2. The Cell Fault Model

We adopt the 'cell fault model' [1] as the fault model. In the model, it is assumed that the considered circuit consists of cells. We treat full adders as cells.

In the model, the followings hold.

- At most one cell can be faulty in the circuit.
- The faulty cell is memoryless. Namely, the faulty cell works as a combinational circuit, and only its present input determines its output.
- There is at least one input pattern of the faulty cell that makes the output of the cell incorrect.

A test set with respect to this model must satisfy the following two conditions. Note that the test set is independent of gate-level implementations of cells.

- Each cell in the circuit must receive exhaustive input patterns when we apply all test patterns in the test set to the circuit.
- The effect of a faulty cell must propagate to at least one of the primary outputs of the circuit.

3. Level-testability of Multi-operand Adders

We show level-testability of multi-operand adders by showing test design for them. For a test of a multi-operand adder, we need a test set by which we can apply all input patterns $000, \ldots, 111$ to input terminals u, v and w of each FA.

We use test patterns by which we can feed the same patterns to all FAs in each CSA. We represent an input pattern of a CSA by input bits of its FAs in bold face. For example, we represent the input pattern of a CSA as **011** when any FA in it receives 0,1, and 1 at input terminals u, v, and w, respectively. We also represent an input to a terminal of a CSA by **0** and **1**. For example, when the input pattern to a CSA is **100**, we represent the input of terminal U in the CSA as **1**, the input of V as **0** and the input of W as **0**. We represent each input pattern of inputs $I_0, \ldots, I_{M-1}$ by **0** and **1** similar to an input to a terminal of a CSA.

Applying **0** and **1** to all inputs of a multi-operand adder is sufficient to feed **000** and **111** to any CSA, respectively. Therefore, we can feed 000 and 111 to any FA in the multi-operand adder by the two patterns.

We must feed the other patterns, i.e., 6 patterns **001**, **010**, **011**, **100**, **101** and **110**, to any CSA in the multi-operand adder. To show how to construct a pattern, we construct a pattern to feed **001** to any CSA in level l as an example. There is at least one input pattern for each CSA in level $l+1$ feeding **001** to any CSA in level l, because the function of an FA is surjective. By using this property recursively, we can obtain an input pattern of the multi-operand adder which feeds **001** to any CSA in level l.

Fig. 3 shows an example of pattern construction. In this figure, we construct a test pattern to feed **001** to the CSA in level 2(CSA2). To construct a pattern for feeding **001** to CSA2, we obtain input patterns **000** and **101** for CSA3 and CSA4, respectively. Then, we obtain input patterns **000** and **001** for CSA5 and CSA6, respectively. Thus, we obtain the test pattern **00000101** for feeding input pattern **001** to the CSA in level 2.

As a result, L test patterns are sufficient for feeding **001** to any CSA in the multi-operand adder. The same as the application of input pattern **001**, L test patterns are sufficient for feeding each of **010**, **011**, **100**, **101** and **110** to any CSA. Therefore, $6L + 2$ test patterns suffice for test of the multi-operand adder.

Effects of an faulty cell at any bit position propagates the

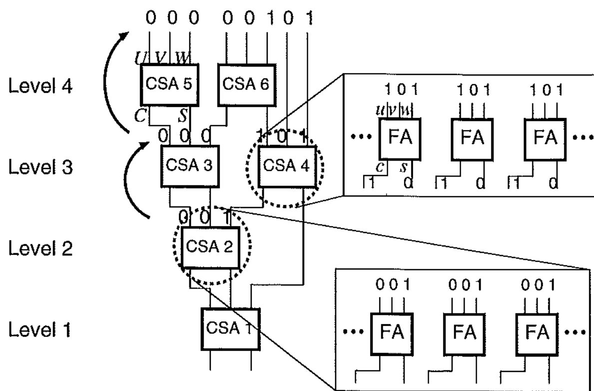

Figure 3. An example of test pattern construction for a multi-operand adder

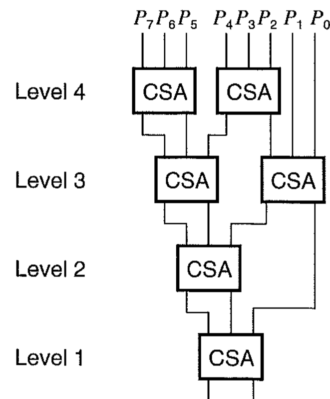

Figure 5. Examples of a partial product compressor in an 8-bit multiplier

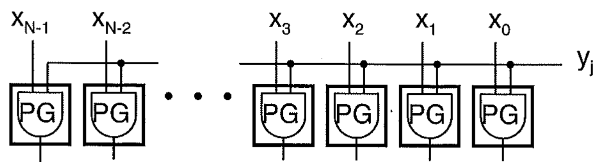

Figure 4. A partial product generator

output of the multi-operand adder. When the faulty cell locates at bit position with weight 2^k, we will observe output value which differs $\pm 2^k$, $\pm 2 \cdot 2^k$, or $\pm 3 \cdot 2^k$ from the correct output value.

4. Test of the partial product compressor in a multiplier

In a multiplier, a multi-operand adder is used as the partial product compressor which is a main part of a multiplier. We show a test method of the multi-operand adder in a multiplier.

We consider an N-bit unsigned multiplier. We let the multiplicand and the multiplier be $X = [x_{N-1}x_{N-2} \cdots x_0]$ and $Y = [y_{N-1}y_{N-2} \cdots y_0]$, respectively.

In general, a parallel multiplier consists of three parts: partial product generators (PPGs), a partial product compressor, and a final adder. The PPGs generate partial products $P_j = X \cdot y_j \cdot 2^j$ for $j = 0, 1, \cdots, N-1$. We consider each PPG consists of N PG cells as shown in Fig. 4. We can compose each PG cell with an AND gate.

The partial product compressor adds up the partial products by carry-save additions. It is carried out by a multi-operand adder. Fig. 5 shows an example of a partial product compressor in an 8-bit multiplier. Note that the input terminals of the partial product compressor are connected with the partial product generators.

The PPG generates an input value for an input terminal of the partial product compressor. The output value of the PPG is all 0 or all 1 if X is all 1. In this case, we can represent the output value of the PPG by **0** or **1**. By a test pattern, X being all 1 and y_i being 0/1, the PPG generates **0/1** as the partial product P_i. Therefore, $6L+2$ test patterns are sufficient to test the partial product compressor.

As an example, we show an input pattern for feeding **001** to the CSA in level 2 of Fig. 5. From Fig. 3, we need to obtain a pattern which generates partial products $(P_7, \ldots, P_0) = (\mathbf{0,0,0,0,0,1,0,1})$. The pattern, $X = 11111111$ and $Y = 00000101$, generates these partial products.

Next, we consider test of PPGs. We need test patterns to feed four input patterns to each PG cell. The PG cell, which we connect with x_i and y_j, must receive input bits $(x_i, y_j) = (0,0), (0,1), (1,0), (1,1)$. In the test of the partial product compressor shown above, any PG cell receives the latter two patterns. We use two more patterns to feed the former two patterns. One is X being all 0 and Y being all 0, and the other is X being all 0 and Y being all 1.

Thus, the partial product compressor with PPGs is testable with $6L+4$ test patterns. We will observe effects of a faulty cell at the outputs of the final adder because the result of multiplication is corrupted by the faulty cell.

By this result, we can show that a partial product compressor with PPGs of a Wallace multiplier is testable with the number of test patterns growing linearly with logarithm of bit width N, because the number of levels of a multi-operand adder used as the partial product compressor is proportional to the logarithm of bit width N.

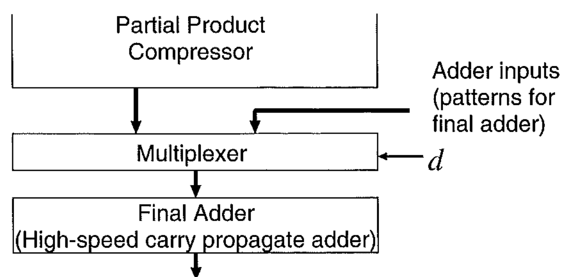

Figure 6. A design with a high-speed carry propagate adder

We have shown level-testability of the partial product compressor with PPGs in any multiplier. If we use a ripple carry adder in [7] as the final adder, we can test the final adder with constant patterns. We can use one of the previously proposed high-speed easily testable carry propagate adders such as [11, 12, 13, 14] as the final adder when we separate the final adder from the multiplier as shown in Fig. 6. In Fig. 6, we use a multiplexer and a selection signal d (controlling externally) to separate the final adder.

5. Conclusion

We have shown level-testability of any multi-operand adder consisting of CSAs by showing test design for it. We have shown $6L + 2$ patterns are sufficient to test any multi-operand adder, where L denotes the depth of it. We have also shown a test method of a multi-operand adder in a multiplier. The result of this paper gives an upper bound of the number of required test patterns for a multi-operand adder in any multiplier. This upper bound will be useful as a criterion in test design of a multiplier.

Acknowledgment

The authors thank Associate Professor Kazuyoshi Takagi of Nagoya University for his comments and discussions.

This work was supported by Graint-in-Aid for Scientific Research(20300016) from Japan Society for the Promotion of Science(JSPS).

References

[1] W. Kautz, "Testing for faults in cellular logic arrays," *Proc. Eighth Ann. Symp. Switching and Automata Theory*, pp. 161–174, 1967.

[2] J. Shen and F. Ferguson, "The design of easily testable vlsi array multipliers," *IEEE Trans. on Computers*, vol. 33, pp. 554–560, June 1984.

[3] A. Chatterjee and J. Abraham, "Test generation, design-for-testability and built-in self-test for arithmetic units based on graph labeling," *Journal of Electronic Testing*, vol. 2, pp. 351–372, Nov. 1991.

[4] D. Gizopoulos, D. Nikolos, A. Paschalis, and C. Halatsis, "C-testable modified-booth multipliers," *Journal of Electronic Testing*, vol. 8, pp. 241–260, June 1996.

[5] K. O. Boateng, H. Takahashi, and Y. Takamatsu, "Design of c-testable modified-booth multipliers," *IEICE Trans. on Inf. and Syst.*, vol. E83-D, no. 10, pp. 1868–1878, 2000.

[6] B. Becker, "An easily testable optimal-time vlsi-multiplier," *Acta Informatica*, vol. 24, pp. 363–380, 1987.

[7] P. Zeng, Z. Mao, Y. Ye, and Y. Deng, "Test pattern generation for column compression multiplier," *Proc. Seventh Asian Test Symposium*, pp. 500–503, 1998.

[8] K. Hanai and N. Takagi, "Testing a 4-2 adder tree with a constant number of patterns," *Proc. of the IEICE General Conference*, p. 70, 2003.

[9] N. Kito, K. Hanai, and N. Takagi, "Easily testable multiplier with 4-2 adder tree," *Technical report of IEICE*, vol. 106, no. 549, pp. 1–6, Mar. 2007.

[10] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Trans. Elec. Comput.*, vol. EC-13, no. 1, pp. 14–17, Feb. 1964.

[11] B. Becker, R. Drechsler, and P. Molitor, "On the generation of area-time optimal testable adders," *IEEE Trans. on CAD*, vol. 14, pp. 1049–1066, Sep. 1995.

[12] R. Blanton and J. Hayes, "Testability of convergent tree circuits," *IEEE Trans. on Computers*, vol. 45, pp. 950–963, Aug. 1996.

[13] ——, "On the design of fast, easily testable alu's," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 8, pp. 220–223, Apr. 2000.

[14] D. Gizopoulos, M. Psarakis, A. Paschalis, and Y. Zorian, "Easily testable cellular carry lookahead adders," *Journal of Electronic Testing*, vol. 19, pp. 285–298, June 2003.