Universal Fault Diagnosis for Lookup Table FPGAs

TOMOO INOUE
SATOSHI MIYAZAKI
HIDEO FUJIWARA
Nara Institute of Science and Technology

Focusing on configurable logic blocks in a lookup table FPGA, the authors present universal fault diagnosis procedures that can locate a fault to just one CLB. The complexity of the proposed procedure for FPGAs using block-sliced loading is independent of FPGA array size.

FIELD-PROGRAMMABLE GATE ARRAYS (FPGAs) are digital devices that implement logic circuits required by users in the field. Because of their short turnaround time, low manufacturing cost, and field programmability, interest in system prototyping and reconfiguration using FPGAs has steadily increased. There are many different FPGA architectures, driven by different programming technologies. The type we consider here is SRAM-based, or lookup table, FPGAs, which users can reprogram any number of times.

FPGA testing, like the testing of conventional digital integrated circuits, is indispensable to ensuring proper operation. Testing can be applied to either unprogrammed or programmed FPGAs. Here, we focus on unprogrammed FPGAs, for which a number of researchers have proposed testing methodologies.[1-4]

FPGA fault diagnosis is also an important problem. Like testing, fault diagnosis also applies to either unprogrammed or programmed FPGAs. Fault diagnosis for unprogrammed FPGAs[5] is particularly important. If engineers can identify and isolate a faulty part in an FPGA prior to programming it, they can implement a required logic function using only fault-free parts.

In this article, we introduce a universal fault diagnosis approach for unprogrammed FPGAs. Based on a test procedure for configurable logic blocks (CLBs) developed by Michinishi et al.,[4] our procedure's diagnostic resolution is one CLB; that is, it can locate a fault to just one CLB. The procedure's complexity—the time required to perform a universal fault diagnosis—for a sequentially loadable FPGA[2] is $O(N^2 n \log n)$, which depends on the FPGA's array size. N is the FPGA's array size, and n is the size of the lookup table.

If we can make universal diagnosis complexity independent of array size, we can reduce complexity considerably. To that end, we propose a class of FPGAs with a universal fault diagnosis procedure whose complexity is independent of array size. We call these C-diagnosable FPGAs. We will present another universal fault diagnosis procedure whose complexity is independent of array size N, and we will show its application to block-sliced, sequentially loadable FPGAs.

FPGA architecture

Figure 1 illustrates the FPGA architecture we consider in this article. The FPGA consists of an $N \times N$ array of programmable CLBs, programmable I/O blocks, and a programmable interconnect structure. On each side of the FPGA are tN I/O blocks; in other words, the FPGA contains $4tN$ I/O blocks.

Each CLB consists of one lookup table, two multiplexers, and one D flip-flop (DFF), con-

0740-7475/98/$10.00 © 1998 IEEE

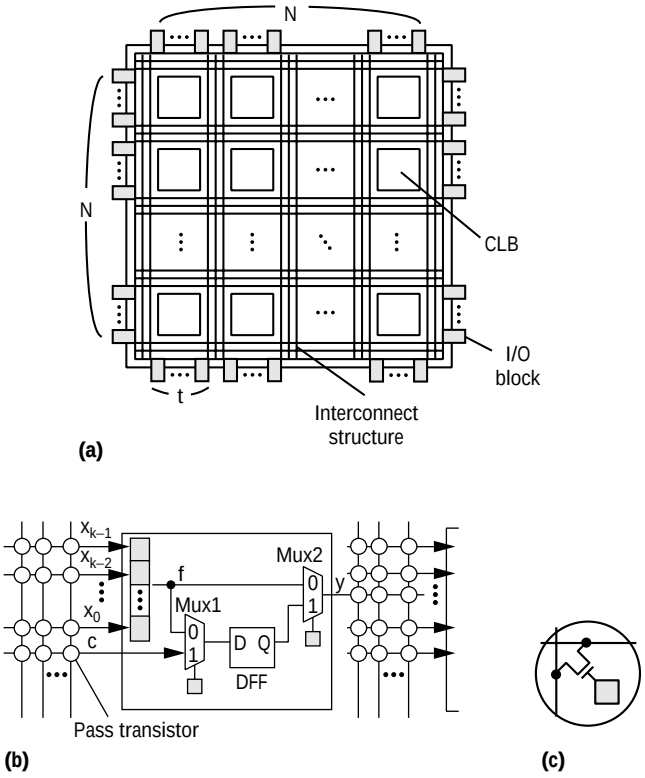


Figure 1. *FPGA architecture: N × N FPGA* ***(a)****; CLB and interconnect structure* ***(b)****; pass transistor* ***(c)****. Shaded cubes in* ***(b)*** *and* ***(c)*** *are CMCs (mux: multiplexer).*

nected as shown in Figure 1b. A lookup table implements combinational logic as a $2^k \times 1$ memory composed of configuration memory cells (CMCs), where k is the number of input lines of the lookup table. When one applies an input pattern to a lookup table, the table selects a CMC addressed by the input pattern, and the cell's output provides the function's value. A lookup table therefore can implement any of 2^n functions of its inputs, where n equals 2^k. In programming the FPGA, one loads the memory with the bit pattern corresponding to the function's truth table. In the CLB, the connections among the input and output lines, the lookup table, and the D flip-flop are configured as multiplexers controlled by CMCs.

An interconnect structure surrounds the CLBs, connecting them and the I/O blocks. A connection in the interconnect structure is configured as a pass transistor, also controlled by a CMC, as shown in Figures 1b and 1c.

One programs a lookup table FPGA by loading a program composed of a bit sequence into the FPGA's CMCs. Storing each bit of the program in the corresponding CMC configures all the CLBs and interconnections, thus implementing a logic function in the FPGA. Such a logic function is called a configuration. The FPGA must contain circuitry that allows program loading. The FPGA-programming scheme we consider here is sequential loading. This scheme shifts the program into the FPGA, storing each bit of the program in the corresponding CMC. An FPGA with this type of loading is called a sequentially loadable FPGA (SL-FPGA). When an SL-FPGA implements configurations, it loads all CMCs.

Universal fault diagnosis

First, we introduce a procedure that locates a fault in any faulty programmed FPGA corresponding to an unprogrammed FPGA. That is, it locates a fault in any faulty configuration implemented on the FPGA.

Fault model. Our approach applies to stuck-at, incorrect-access, nonaccess, and multiple-access faults of lookup tables in CLBs, and functional faults of multiplexers and D flip-flops in CLBs.[6] We assume that several of these faults may occur in a CLB simultaneously and that the number of CLBs including these faults in an FPGA is at most one.

Universal test procedure. We denote the procedure[4] for testing CLBs in SL-FPGAs as TP_{CLB}. We perform this procedure by repeatedly implementing a configuration and alternately applying an input sequence to the configuration. That is, we represent TP_{CLB} by a sequence of pairs consisting of a configuration and an input sequence applied to the configuration as follows:

$$TP_{CLB} = [(C_1, S_1), (C_2, S_2), \ldots, (C_{2k+4}, S_{2k+4})]$$

where C_i is the ith configuration, S_i is the input sequence applied to C_i, and k is the number of input lines of a lookup table. Tables 1 and 2 list the configurations and input sequences in TP_{CLB}. This test procedure can detect any fault defined in our fault model.

By applying TP_{CLB} to each CLB in an FPGA implemented with a connection between the CLB and I/O blocks, we can identify faulty CLBs. However, this will consume much time. If we can configure connections so that we can apply the test procedure to all the CLBs simultaneously, we can locate the faulty ones immediately. However, the number of CLBs in an FPGA is $N^2 = N \times N$, whereas the number of I/O blocks in an FPGA is $O(N) = 4tN$. Hence, if array size N becomes large, it will be impossible to configure such connections.

Universal fault diagnosis procedure. Let's assume input sequence S_i at configuration C_i in TP_{CLB}. Let R_i be the output sequence obtained by applying S_i to a fault-free CLB that implements C_i. Earlier work[4] has shown that R_i can be used as a certain bit sequence of S_i for other CLBs implementing the same configuration C_i. Therefore, if the output line of a CLB is connected with the appropriate input line of anoth-

Table 1. *CLB test procedure TPCLB: configurations C_i.*

Configuration C_i	Lookup table	Mux1	Mux2
1 to k	$f = x_{i-1}$	1	0
$k+1$ to $2k$	$f = \overline{x_{i-1}}$	1	0
$2k+1$	$f = x_0$	1	1
$2k+2$	$f = x_0$	1	1
$2k+3$	$f = x_0$	1	0
$2k+4$	$f = x_0$	0	1

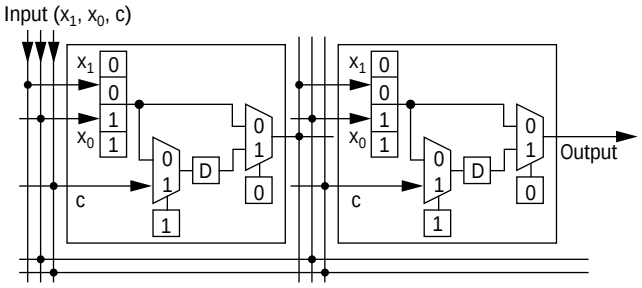


Figure 2. *Testing two CLBs at configuration C_2 in test procedure TP_{CLB}.*

Table 2. *CLB test procedure TP_{CLB}: input sequences S_i.*

Input sequence S_i	Input lines $x_{k-1} \ldots x_1 x_0$	Input line c
$1 \le i \le k$	0…0 1, 0...1 0, …, 1…1 1, 0…0 0 (applied in this order)	0 (constant)
$k+1 \le i \le 2k$	0…0 1, 0…1 0, …, 1…1 1, 0…0 0 (applied in arbitrary order)	0 (constant)
$2k+1$	0…0 0 (constant)	0, 0, 0, 1, 1, 0 (in this order)
$2k+2 \le i \le 2k+4$	0…0 0, 0…0 1, …, 1…1 1, 1…1 1 (applied in this order)	0, 1, 0, 1 (in this order)

er CLB, we can simultaneously test multiple CLBs with a small number of I/O blocks. Figure 2 shows an example of such testing. Based on this idea, we previously presented a test procedure that connected several CLBs in a cascade to test all the CLBs in an FPGA concurrently.

However, our diagnostic resolution goal is one CLB. As long as we observe output responses from several CLBs via a single primary output implemented by an I/O block, as in Figure 2, we cannot see which CLB is faulty. Hence, to identify just one faulty CLB by means of a limited number of I/O blocks, we must apply the test procedure several times as the configurations in the interconnect structure change.

Now, let's consider the number of test procedures required to locate a faulty CLB. Suppose that N_O primary outputs can be implemented at each configuration in the underlying test procedure. Then, the average number d of CLBs connected to a primary output—that is, the first test procedure's diagnostic resolution (the number of candidates for faulty CLBs)—is N^2/N_O. By applying the test procedure with different connections of CLBs again, we can further reduce the diagnostic resolution to N^2/N_O^2. We express the diagnostic resolution after applying the test procedure i times as $d_i = N^2/N_O^i$.

When N_I primary inputs are implemented to feed the same input sequences to all CLBs concurrently, the number N_O of primary inputs that can be implemented is $4tN - N_I$. Since the number of input lines of a CLB is $k+1$, $N_I = k+1$. Hence, $N_O = 4tN - (k+1)$. Based on these equations, we can express the condition of the number i of test procedures needed for a diagnostic resolution of one CLB as $N^2 \le [4tN - (k+1)]^i$.

When $i = 2$, any FPGA satisfies this condition even if its array size N is large. Thus, we present a universal diagnosis procedure, DP_1, that consists of two test steps. Each step tests all the CLBs concurrently in the same way as test procedure TP_{CLB}. That is, we implement C_i on each CLB and apply S_i to all the CLBs for $1 \le i \le 2k+4$. As shown in Table 1, C_i and S_i are the configuration and input sequence in TP_{CLB}. We can also express DP_1 as $[(C_1, S_1), (C_2, S_2), \ldots, (C_{n_c}, S_{n_c})]$, where

$$n_c = 2(2k+4) = 4k+8 \quad (1)$$

and n_c is the number of configurations.

The difference between the two steps is the configuration of connections implemented on the interconnect structure (Figure 3, next page):

- *Step 1 (horizontal diagnosis).* Let CLB(l, m) be the CLB at the lth column in the mth row. At all C_i, the output line of CLB(l, m) is connected with an appropriate input line of CLB($l+1, m$) for $1 \le l \le N-1$ for all m. The output line of the rightmost CLB(N, m) is connected with an I/O block as a primary output for all m. The other input lines are connected with remaining I/O blocks as primary outputs.
- *Step 2 (vertical diagnosis).* Exchanging column number l for row number m configures the interconnect structure the same way as in step 1.

Note that a cascade consisting of N CLBs is implemented on each row at step 1 (or on each column at step 2).

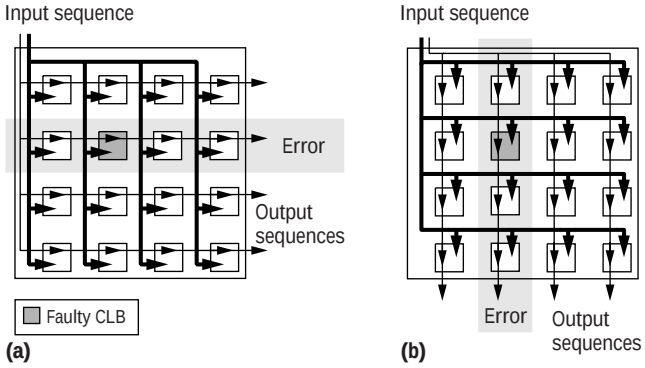


Figure 3. *Universal diagnosis procedure DP_1: step 1* ***(a)*** *and step 2* ***(b)****.*

In DP_1, steps 1 and 2 identify the row and column respectively that include a faulty CLB with N primary outputs. As a result, DP_1 can identify just one CLB. This universal diagnosis procedure is preset; that is, we execute step 2 irrespective of step 1's result.

Universal fault diagnosis complexity. We refer to the time required to perform a universal diagnosis procedure for an FPGA as the FPGA's universal diagnosis complexity. As an example, let's assume the following universal diagnosis procedure for an FPGA G:

$$\mathrm{DP} = [(C_1, S_1), (C_2, S_2), \ldots, (C_{n_c}, S_{n_c})]$$

Let $c(i)$ be the number of CMCs loaded to implement ith configuration C_i. Let $s(i)$ be the length of input sequence S_i for C_i.

The time required to implement all the configurations in DP for G is

$$T_G^C(\mathrm{DP}) = \sum_{i=1}^{n_c} t_c c(i)$$

where t_c is the time required to load one bit of a program into a CMC in G. The time required to apply all the input sequences in DP for G is

$$T_G^S(\mathrm{DP}) = \sum_{i=1}^{n_c} t_c s(i)$$

where t_s is the clock cycle time of a configuration implemented in G.

Thus, the universal diagnosis complexity of DP for G is

$$T_G(\mathrm{DP}) = T_G^C(\mathrm{DP}) + T_G^S(\mathrm{DP}) = \sum_{i=1}^{n_c} [t_c c(i) + t_s s(i)] \qquad (2)$$

Now, let's consider DP's universal diagnosis complexity for SL-FPGAs. When we change from one configuration to another on an SL-FPGA, we must load all the program bits. Hence, the time required to implement the configuration for DP for an SL-FPGA is $c(i) = N_m$ for all i, where N_m is the total number of CMCs in G. As Equation 1 showed, the number of configurations in DP is

$$n_c = 2(2k + 4) = \mathrm{O}(\log n) \qquad (3)$$

where k is the number of input lines of a lookup table and n is the size of a lookup table. That is, $n = 2^k$.

Without loss of generality, we assume the total number of CMCs in an FPGA is proportional to both the number of CLBs, N^2, and the size of a lookup table (or the number of CMCs in a lookup table), 2^k. That is, $N_m = \mathrm{O}(N^2 n)$, where $n = 2^k$. Hence, the time required to implement all the configurations in DP_1 is

$$T_{SL}^C(\mathrm{DP}_1) = t_c N_m \mathrm{O}(\log n) = \mathrm{O}(N^2 n \log n) \qquad (4)$$

Recall the configurations in test procedure TP_{CLB} (Table 1). Note that each of the configurations C_{2k+1}, C_{2k+2}, and C_{2k+4} implements a path including a D flip-flop. Accordingly, each CLB cascade at the corresponding configurations in DP_1 includes a shift register composed of N D flip-flops. Consequently, we can observe the output response for an input pattern N clock cycles later after applying the input pattern. From Table 1, the length $s(i)$ of S_i applied to C_i in DP_1 is

$$s(i) = \begin{cases} n & (1 \le i \le 2k,\ 2k+5 \le i \le 4k+4) \\ 6 + N - 1 & (i = 2k+1,\ 4k+5) \\ 4 & (i = 2k+3,\ 4k+7) \\ 4 + N - 1 & (i = 2k+2,\ 2k+4,\ 4k+6,\ 4k+8) \end{cases}$$

Thus, the total time required to apply the input sequences in DP_1 is

$$\begin{aligned} T_{SL}^S(\mathrm{DP}_1) &= [4kn + (2N + 10) + 8 + (4N + 12)]t_s \\ &= \mathrm{O}(N + n \log n) \end{aligned} \qquad (5)$$

From Equations 4 and 5, the complexity of DP_1 for SL-FPGAs is

$$T_{SL}(\mathrm{DP}_1) = \mathrm{O}(N^2 n \log n) \qquad (6)$$

This equation shows that the universal diagnosis complexity of DP_1 for SL-FPGAs depends on the FPGA's array size N. Making universal diagnosis complexity independent of array size would considerably reduce complexity. Next, we present another universal diagnosis procedure and a class of

FPGAs, called C-diagnosable FPGAs, for which universal diagnosis complexity is independent of array size.

C-diagnosable FPGAs

Since an FPGA consists of an array of CLBs, we can consider it an iterative system. C-testable[7] is a term that describes an important class of testable iterative systems. An earlier work[2] extended the concept of C-testability for general LSI circuits to FPGAs. Here, we further extend the concept to FPGA fault diagnosis.

We define an FPGA as C-diagnosable if there exists a universal fault diagnosis procedure whose complexity is independent of the FPGA's array size. As we saw in Equation 6, the universal diagnosis complexity of DP_1 depends on the array size, and hence with DP_1 an FPGA is not C-diagnosable. As shown by Equation 2, a universal diagnosis procedure's complexity is the sum of the total configuration implementation time and the total input sequence application time. From Equations 4 and 5, we see that both implementation time and application time of DP_1 for SL-FPGAs depend on array size. Here, we describe methods for making both times independent of array size.

First, let's consider the application time of input sequences in DP_1. As mentioned earlier, there are configurations that implement shift registers in DP_1, and the application time for such configurations depends on array size. A CLB lookup table can implement any k-input logic function. Therefore, we connect the output lines of multiple CLBs to the input lines of another CLB that implements an exclusive-OR (XOR). Then we can observe the multiple CLBs' output responses from the XOR's output without cascades of D flip-flops.

Thus, by modifying part of the configurations in DP_1, we obtain another universal diagnosis procedure, DP_C. Figure 4 illustrates the modification. We substitute the following two configurations for configuration C_i in DP_1 for $i = 2k + 1$, $2k + 2$, $2k + 4$, $4k + 5$, $4k + 6$, $4k + 8$.

1) C_i' for all rows m (respectively all columns l):

- For $h = 1, 2, \ldots, N/2$, the CLB at the odd column (row), CLB($2h - 1, m$) (CLB($l, 2h - 1$)), as well as its input lines, implements the same configuration as C_i in DP_1.
- For $h = 1, 2, \ldots, N/2$, the CLB at the even column (row), CLB($2h, m$) (CLB($l, 2h$)), implements a 2-input XOR.
- For $h = 1, 2, \ldots, N/2 - 1$, the output line of CLB($2h, m$) (CLB($l, 2h$)) connects to an input line of the XOR on CLB($2h + 2, m$) (CLB($l, 2h + 2$)).
- The output line of the rightmost (bottommost) CLB, CLB(N, m) (CLB(l, N)), connects to an I/O block.

2) C_i'': By exchanging the even CLBs for the odd ones, we implement these configurations the same way as we implement C_i'.

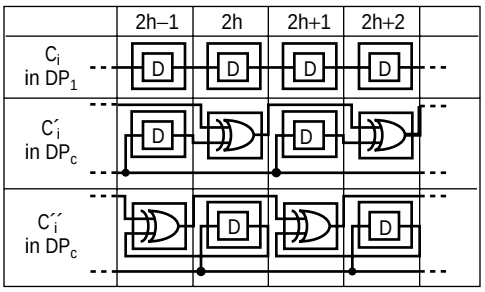


Figure 4. *Modification of DP_1 to DP_C.*

DP_C partitions CLBs into two groups: $N/2$ CLBs that are diagnosed and $N/2$ CLBs that configure $N/2$-input XORs to compress the output sequences from diagnosed FPGAs. DP_C exchanges these groups' roles by means of the doubled configurations. As a result, configurations in DP_C include no cascades of D flip-flops; accordingly, we can observe the output responses just one cycle later after applying the corresponding input pattern.

Consequently, although the number of configurations increases, the application time becomes independent of array size N. After substituting $2(6 \times 2)$ and $2(4 \times 4)$ for $(2N + 10)$ and $(4N + 12)$ in Equation 5, the total time required to apply the input sequences in DP_C is

$$T_{SL}^{S}(\mathrm{DP_C}) = (4kn + 24 + 8 + 32)t_s = \mathrm{O}(n \log n) \qquad (7)$$

which is independent of N.

Next, let's consider the configuration time of this universal fault diagnosis procedure. At the 12 substituted configurations just described, we implement a configuration to be diagnosed and an XOR alternately in each row or in each column. Hence, if we regard 2×2 CLBs—CLB(l, m), CLB($l, m + 1$), CLB($l + 1, m$), CLB($l + 1, m + 1$) for $l, m = 1, 3, 5, \ldots$—as one block, we implement an iterative system at any of the substituted configurations. Also, each CLB implements the same logic function at any other configuration in DP_C. Therefore, we can see that DP_C always implements iterative systems in which each logic block consists of 2×2 CLBs. To this diagnosis procedure, we can adapt the programming scheme called block-sliced loading.[2] An FPGA with block-sliced loading can load the same program into each block concurrently.

Let t_b be the time required to load the same program bit into the corresponding CMC in each block. The number of CMCs in a block is $N_b = 2^2\mathrm{O}(n) = \mathrm{O}(n)$, where n is the size of a lookup table. On the other hand, as mentioned earlier, when we derive DP_C from DP_1, we double six configurations in DP_1. Hence, from Equation 3, the number of configurations in DP_C is $n_c = 2(2k + 4) + 6 = \mathrm{O}(\log n)$. Therefore, the

time required to implement configurations in DP_C on a block-sliced SL-FPGA is

$$T^{C}_{BSSL}(DP_C) = t_b N_b O(\log n) = O(n \log n) \quad (8)$$

Note that the application time of input sequences does not depend on the programming scheme. Therefore, from Equations 7 and 8, the universal fault diagnosis complexity of DP_C for block-sliced SL-FPGAs is $T_{BSSL}(DP_C) = O(n \log n)$. Thus, we can obtain C-diagnosable FPGAs.

WE HAVE INTRODUCED two important concepts: *universal* and *C-diagnosable*. Since our proposed fault diagnosis procedure is universal—independent of the logic functions to be realized—we need not provide a different diagnosis procedure for each logic function. Further, our proposed FPGA is C-diagnosable—the time required to diagnose the FPGA is independent of its array size. This means that diagnosis complexity does not increase even if FPGA array size becomes large. Therefore, our approach can be applied to large and complex FPGA systems. Although this article considered only faults in CLBs, diagnosis should also address faults in other components (interconnect structure, I/O blocks). In future work, we plan to investigate diagnosis procedures for all FPGA faults. D&T

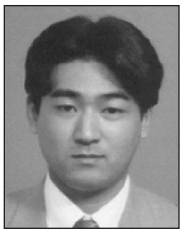
Acknowledgments

We thank Takuji Okamoto, Tokumi Yokohira, and Hiroyuki Michinishi of Okayama University, Japan, and Toshimitsu Masuzawa and Michiko Inoue of Nara Institute of Science and Technology for their helpful comments and discussions.

References

1. W.K. Huang and F. Lombardi, "An Approach for Testing Programmable/Configurable Field Programmable Gate Arrays," *Proc. 14th IEEE VLSI Test Symp.*, IEEE Computer Society Press, Los Alamitos, Calif., 1996, pp. 450-455.
2. T. Inoue et al., "Universal Test Complexity of Field-Programmable Gate Arrays," *Proc. Fourth IEEE Asian Test Symp.*, IEEE CS Press, 1995, pp. 259-265.
3. M. Renovell, J. Figueras, and Y. Zorian, "Test of RAM-Based FPGAs: Methodology and Application to the Interconnect," *Proc. 15th IEEE VLSI Test Symp.*, IEEE CS Press, 1997, pp. 230-237.
4. H. Michinishi et al., "A Test Methodology for Configurable Logic Blocks of Lookup Table Based FPGAs," *IEICE Trans. D-I*, Vol. J79-D-I, No. 12, Dec. 1996, pp. 1141-1150 (in Japanese).
5. X.T. Chen, W.K. Huang, and F. Lombardi, "On the Diagnosis of Programmable Interconnect Systems: Theory and Application," *Proc. 14th IEEE VLSI Test Symp.*, IEEE CS Press, 1996, pp. 204-209.
6. T. Inoue, S. Miyazaki, and H. Fujiwara, "On the Complexity of Universal Fault Diagnosis for Lookup Table FPGAs," Tech. Report NAIST-IS-TR97020, Graduate School of Information Science, Nara Institute of Science and Technology, 1997 (http://isw3.aist-nara.ac.jp/IS/TechReport2/report/97020.ps).
7. A.D. Friendman, "Easily Testable Iterative Systems," *IEEE Trans. Computers*, Vol. C-22, No. 12, Dec. 1973, pp. 1061-1064.

Tomoo Inoue is an instructor at the Graduate School of Information Science, Nara Institute of Science and Technology, Japan. His research interests include test generation, synthesis for testability, and parallel processing. Inoue received the BE, ME, and PhD from Meiji University, Kawasaki, Japan. He is a member of the IEEE, the Institute of Electronics, Information, and Communication Engineers of Japan, and the Information Processing Society of Japan.



Satoshi Miyazaki is with the Tenri Integrated Circuits Group, Sharp Corporation, Nara, Japan, where he is engaged in microprocessor development. He received the BE from Okayama University and the ME from the Nara Institute of Science and Technology, where he participated in the work described in this article.

Hideo Fujiwara is a professor at the Graduate School of Information Science, Nara Institute of Science and Technology, Japan. His research interests are logic design, digital systems design and test, VLSI CAD, and fault-tolerant computing. Previously, he held positions at Osaka University and Meiji University. He is the author of *Logic Testing and Design for Testability* (MIT Press, 1985). Fujiwara received the BE, ME, and PhD from Osaka University. He is a fellow of the IEEE, a member of the Institute of Electronics, Information, and Communication Engineers of Japan, and a member of the Information Processing Society of Japan.

Send questions and comments about this article to Tomoo Inoue, Graduate School of Information Science, Nara Institute of Science and Technology, Ikoma, Nara 630-01, Japan; inoue@is.aist-nara.ac.jp.