Universal Fault Diagnosis for Lookup Table FPGAs

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 FPGAs5 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 FPGA2 is \(O(N^2n \log n)\), 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 \(6N\) I/O blocks; in other words, the FPGA contains \(4N\) I/O blocks.

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

connected 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 $n = 2^k$. 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. 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 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:

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

where $C_i$ is the $i$th 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 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 another-
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 of CLBs connected to a primary output—that is, the first test procedure’s diagnostic resolution (the number of candidates for faulty CLBs)—is $N_o^2/N_o$. 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 $d_i = N^2/N_o^2$.

When $N_o$ primary inputs are applied to feed the same input sequences to all CLBs concurrently, the number $N_o$ of primary inputs that can be implemented is $4tN - N_o$. Since the number of input lines of a CLB is $k + 1$, $N_o = k + 1$. 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 $N^2 \leq [4tN - (k + 1)]^2$.

When $i = 2$, any FPGA satisfies this condition even if its array size $N$ is large. Thus, we present a universal diagnosis procedure, $D_P$, 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 $1 \leq i \leq 2k + 4$. As shown in Table 1, $C_i$ and $S_i$ are the configuration and input sequence in $TP_{CLB}$. We can also express $D_P$ as $[(C_1, S_1), (C_2, S_2), \ldots, (C_{2k+4}, S_{2k+4})]$ where

$$n_c = 2(2k + 4) = 4k + 8$$

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 $CLB_i(m)$ be the CLB at the $ith$ column in the $mth$ row. At all $C_i$, the output line of $CLB_i(m)$ is connected with an appropriate input line of $CLB_i(l + 1, m)$ for $1 \leq l \leq N - 1$ 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).
In DP1, steps 1 and 2 identify the row and column respectively that include a faulty CLB with \( N \) primary outputs. As a result, DP1 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 \( G \):

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

Let \( c(i) \) be the number of CMCs loaded to implement \( i \)th 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

\[
T^c_G(DP) = \sum_{i=1}^{nc} t_i c(i)
\]

where \( t_i \) 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

\[
T^s_G(DP) = \sum_{i=1}^{nc} t_i s(i)
\]

where \( t_i \) is the clock cycle time of a configuration implemented in \( G \).

Thus, the universal diagnosis complexity of DP for \( G \) is

\[
T_G(DP) = T^c_G(DP) + T^s_G(DP) = \sum_{i=1}^{nc} [t_i c(i) + t_i s(i)]
\]

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

\[
n, = 2(2k + 4) = O(\log n)
\]

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, \( N_m = O(N^2 n) \), where \( n = 2^k \). Hence, the time required to implement all the configurations in DP for an SL-FPGA is

\[
T^c_{SL}(DP) = t_i N_m O(\log n) = O(N^2 n \log n)
\]

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 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 is

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

Thus, the total time required to apply the input sequences in DP for an SL-FPGA is

\[
T^s_{SL}(DP) = [4kn + (2N + 10) + 8 + (4N + 12)] t_i = O(N^2 n \log n)
\]

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

\[
T_{SL}(DP) = O(N^2 n \log n)
\]

This equation shows that the universal diagnosis complexity of DP 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 is a term that describes an important class of testable iterative systems. An earlier work 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, depends on the array size, and hence with DP, 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, 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. As mentioned earlier, there are configurations that implement shift registers in DP, 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, we obtain another universal diagnosis procedure, DP. Figure 4 illustrates the modification. We substitute the following two configurations for configuration Ci in DP, for $i = 2^m + 1, 2^m + 2, 2^m + 4, 4k + 5, 4k + 6, 4k + 8$.

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,.
- 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 $C'_i$.

DP 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 exchanges these groups’ roles by means of the doubled configurations. As a result, configurations in DP, 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 is

$$T_{\text{sl}}(\text{DP}) = (4kn + 24 + 8 + 32)t_x = O(n \log n)$$

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 alternatingly in each row or in each column. Hence, if we regard $2 \times 2$ CLBs—CLB($l, m$), CLB($l, m + 1$), CLB($l + 1, m$), CLB($l + 1, m + 1$) for $l = 1, 3, 5, \ldots$—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. Therefore, we can see that DP always implements iterative systems in which each logic block consists of $2 \times 2$ CLBs. To this diagnosis procedure, we can adapt the programming scheme called block-sliced loading. An FPGA with block-sliced loading can load the same program into each block concurrently.

Let $t_S$ 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 $N_b = 2^l(n) = O(n)$, where $n$ is the size of a lookup table. On the other hand, as mentioned earlier, when we derive DP from DP, we double six configurations in DP. Hence, from Equation 3, the number of configurations in DP is $n_c = 2(2k + 4) + 6 = O(\log n)$. Therefore, the
time required to implement configurations in $D_P_c$ on a block-sliced SL-FPGA is

$$T_{res}(D_P_c) = t_{b}N_{c}O(\log n) = O(n \log n)$$ (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 $D_P_c$ for block-sliced SL-FPGAs is $T_{res}(D_P_c) = O(n \log n)$. 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—indeed 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.

**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**


**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.