1997 IEEE Asian Test Symposium, pp. 242-247, November 1997.

Testing for the programming circuit of LUT-based FPGAs

H. Michinishi, T. Yokohira, T. Okamoto

Department of Information Technology
Faculty of Engineering, Okayama University
Tsushima-naka, Okayama, 700 Japan

T. Inoue, H. Fujiwara

Graduate School of Information Science
Nara Institute of Science and Technology
Takayama, Ikoma, Nara, 630-01 Japan

Abstract

The programming circuit of look-up table based FPGAs consists of two shift registers, a control circuit and a configuration memory (SRAM) cell array. Because the configuration memory cell array can be easily tested by conventional test methods for RAMs, we focus on testing for the shift registers. We show that the testing can be done by using only the faculties of the programming circuit, without using additional hardware.

1 Introduction

Field programmable gate arrays (FPGAs) are modern logic devices to implement logic circuits in various fields[1-3]. Various FPGAs with different architectures driven by different programming technologies exist. The most popular one is a class of FPGAs with SRAM-based architecture, also called *look-up table based* FPGAs[1-4]. In this paper, We consider such a class of FPGAs and call it FPGA for short. The hardware of FPGAs consists of a programmable logic part and a programming circuit part.

Some researchers[5,6] have proposed testing for programmed FPGAs, on which logic circuits are implemented. But, the testing is not applicable to unprogrammed FPGAs at manufacturing time. In order to solve such a problem, we developed testing for unprogrammed FPGAs, and proposed testing for the logic part [7,8]. However, we have not yet discussed test for the programming circuit.

This paper considers testing for the programming circuit which consists of four components: a configuration memory cell array, a data shift register (DSR), an address shift register (ASR), and a control circuit. We can test the configuration memory cell array by means of conventional test methods for random access memories when the other components of the programming circuit have no fault. We will therefore focus our attention on testing for the DSR and the ASR under the assumption that both the control circuit and the configuration memory cell array are fault free.

When we use a FPGA, we first program it by loading a configuration bit-stream into the configuration memory cell array, and check the correctness of the loading by reading out its contents[1-3]. The former and the latter faculties are referred to simply as configuration and readback, respectively[4]. If we can chose the configuration bit-streams so that test sequences for the DSR and the ASR can be produced, and that the responses for them can be observed at the output of FPGA, we can test the DSR and the ASR efficiently by using configuration and readback. On such a strategy, we consider testing for DSRs and ASRs.

In this paper, we first describe the architecture of the programming circuit and functional fault models of the DSR and the ASR. We next describe the test procedures with the configuration bit-streams derived on the strategy mentioned above. Also, we show the validness of the test procedures. Each of the test procedures requires only one loading and one reading. Finally, we show an application of the test procedures.

2 Programming circuit and fault models

2.1 Programming circuit

The programming circuit of the FPGA considered in this paper is illustrated in Fig. 1. It consists of a configuration memory cell array, a data shift register (DSR), an address shift register (ASR) and a control circuit (not shown in Fig. 1). The size of the configuration memory cell array is $F \times W$ (M_i^j is the ij-th cell of the array). The DSR and the ASR are constructed by cascading W pieces of modules DSR_i $(1 \leq i \leq W)$ and F pieces of modules ASR_j $(1 \leq j \leq F)$, respectively. D_{in} is an input of the DSR to which a configuration bit-stream is applied in configuration process and D_{out} is an output of the DSR from which the contents of configuration memory cells are read out in readback process. Fig. 2 and Fig. 3 show the structures of DSR_i and ASR_j, respectively. G, S, and

P are control signals for configuration and readback. $DCLK$ and $ACLK$ are clock signals to the DSR and the ASR, respectively.

In configuration process, a configuration bit-stream of length $F \times W$ bits is applied to D_{in} continuously and shifted bit by bit through multiplexers MU_1, MU_2, $\cdots$, MU_W and D-type latchs DD_1, DD_2, $\cdots$, DD_W (See Fig. 2). Every W-th shift of the bit-stream, W bits of the bit-stream (sub-stream) are stored in DD_1, DD_2, $\cdots$, DD_W. Then, the content of DD_i is transmitted instantaneously through b_i called *bitline* driven by 3-state buffer TB_i to configuration memory cell M_i^j specified by an activated w_j called *wordline* (See Fig. 3). In this way, W bits of the bit-stream are loaded into W configuration memory cells, M_1^j, M_2^j, $\cdots$, M_W^j. Hereafter, the W configuration memory cells specified by w_j are referred to as j-th frame. Activations of *wordline*s are performed exclusively in the ascending order of j by the ASR operating as a modulo W shift counter, consisting of F ASR_js (WD_j shows an AND-gate). In order to realize the activations mentioned above, the seed sequence $(100 \cdots 0)$ is applied to the input of AD_1, after all the D-type latches AD_js in the ASR is reset by the control circuit. The repetition rate of the shift clock $ACLK$ is $1/W$ of that of the shift clock $DCLK$. Thus, the $F \times W$ memory cell array is programmed by the configuration bit-stream.

In readback process, the contents of all the configuration memory cells are read out frame by frame through the DSR under the control of the ASR in the same order as that in configuration process. In particular, the precharge gate PC_is precharge b_is before every activation of w_j, so that the contents of M_i^js are surely loaded to DD_is in parallel.

2.2 Fault models

The objective of the test is to detect faults that exist in the DSR or the ASR. We introduce the following assumptions:

(A1) Every function block of the programming circuit except the DSR and the ASR is correct.

(A2) In at most one module of the DSR or the ASR (DSR_i or ASR_j), multiple faults may exist.

In the succeeding discussion, we consider the following fault models:

[Functional faults in D-type latch (DD_i, AD_j)]

Any of the faults transforms the D-type latch into one of other sequential circuits where the number of states is less than or equal to that of D-type latch. It may cause loss of the reset function used in the beginnings of configuration and readback processes.

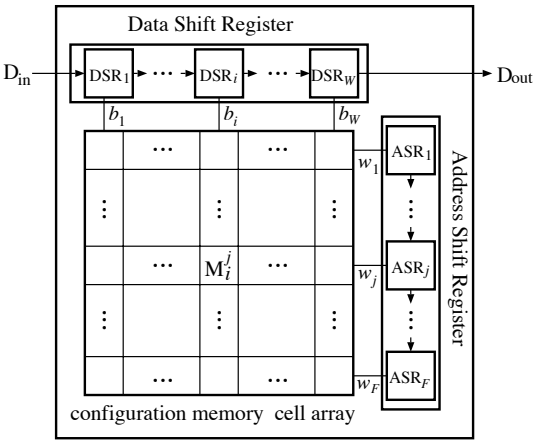

Figure 1: Construction of programming circuit of FPGA

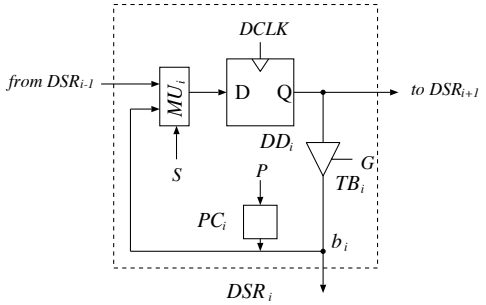

Figure 2: Structure of a DSR module (DSR_i)

[Functional faults in other components (MU_i, TB_i, PC_i, WD_j)]

Any of the faults transforms the corresponding component into one of other combinational circuits.

In order to simplify the testing under the assumptions and the fault models mentioned above, we further introduce two assumptions about logical values on *bitline*s as follows: in at most one module in the DSR and the ASR (DSR_i or ASR_j).

(A3) If there exists a fault which causes PC_i not to charge b_i in readback process, then the value 0 is latched in DD_i every $DCLK$, independently of the content of the configuration memory cell M_i^j specified by w_j.

[Validness of (A3)] First, suppose that the content of M_i^j is 0. When w_j =1 in readback process, there exists a path from b_i to the grand (GND) (See Fig. 4). Next, suppose that the content of M_i^j is 1. When w_j =1 in readback process, M_i^j tries to charge b_i, but can not charge it enough to set DD_i by the following reason[4]:

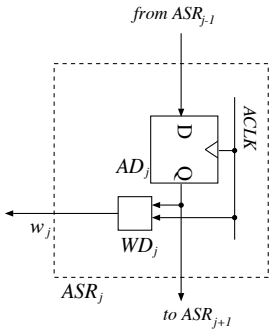

Figure 3: Structure of an ASR module (ASR_j)

(1) The capacitance C shown in Fig. 4 is not precharged before $w_j = 1$ due to the fault described in the assumption (A3).
(2) If the content of M_i^j is 1, the conductance of the pass-transistor Tr_i^j shown in Fig. 4 is too small to charge C from M_i^j during the period that w_j =1.

Thus, the assumption (A3) is valid.
(A4) If the function of PC_i is correct and the content of M_i^j is 1, the charge on b_i which has been precharged by PC_i for the purpose of reading out the content of M_i^j is maintained for at least one cycle of $DCLK$ after it sets DD_i.
[Validness of (A4)] As soon as w_j changes 1 to 0 after setting DD_i, b_i is floating and the leakage conductance between b_i and GND becomes very small. Thus, the assumption (A4) is valid.

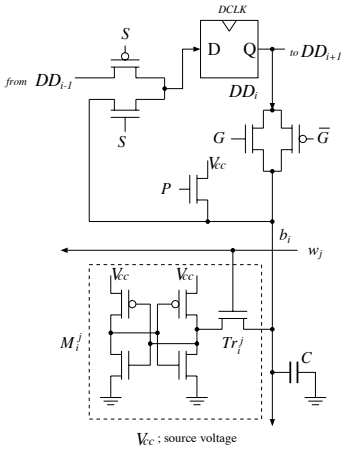

Figure 4: Circuit structure of a configuration memory cell

3 Test procedure for data shift register

The data input and the outputs of the DSR in configuration process are D_{in} and b_is as seen from Fig. 1. While a configuration bit-stream applied to D_{in} from the external environment is directly controllable, the values on b_is are not observable directly. On the other hand, the data inputs and the output in readback process are b_is and D_{out}, respectively. The inputs are not controllable directly, while the output is observable directly. Taking into such situations, we will test DSRs under the assumptions described in 2.2 by the following procedure.

[TP-D:Test Procedure for DSRs]

(1) Configure FPGA by loading the configuration bit-stream shown in Fig. 5.
The 1st sub-stream of W bits ($11\cdots1$) shown in Fig. 5 is the contents to be loaded into the 1st frame of the configuration memory cell array. The 2nd sub-stream ($00\cdots0$) is those into the 2nd frame. In the same way, the j-th sub-stream is those into the j-th frame. Each of I_1 and I_2 included in the 5th sub-stream is one of the 6 length characteristic (input) sequences [9] for D-type latch, and A is an arbitrary bit sequence of $w-12$ length.

(2) Read out the contents of the configuration memory cell array and observe a bit-stream appearing on D_{out}. □

In the test of the DSR, we can consider from the assumptions (A1) and (A2) that every function block of the programming circuit except the DSR is correct and there exist some faults in at most one module of the DSR. It is apparent that if there exists no faulty module in the DSR, the observed bit-stream agrees with that of Fig. 5. On the contrary, if the same bit-stream as that of Fig. 5 is observed in the procedure (2), it is assured that there exists no fault except a special case that the correct outputs are observed in spite of contrary faults existing both in DD_i and MU_i or both in TB_i and MU_i, where a contrary fault in each component is such a fault that causes the the component to produce the complements of the correct output values. This can be proved as follows.

[Lemma 1] If the outcome of TP-D is correct, then DSR_i for $^{\forall}i$ operates correctly during any frame data shifting in both configuration and readback processes.
[Proof] We assume that there exists some fault in DSR_i for $^{\exists}i$. Then, the input can be applied correctly from DSR_{i-1} in the procedures (1) and (2), and the response of DSR_i can be observed correctly at D_{out} in the procedure (2), because DSR_h for $^{\forall}h$ ($h \neq i$) has no fault from the assumption (A2).

i) The case of $1 \leq i \leq w-6$.
Suppose that the outcome of TP-D is correct. At the beginning of the procedure (1), the first W 1s of the configuration bit-stream appear on the output of DD_i.

In this while, the value of b_i never change, because any w_j is not activated, and inputs of both TB_i and PC_i are kept invariable independently of the presence of faults in DSR_i. The output of MU_i is therefore kept at a constant logic value v_1. In the same way, it is kept at a constant logic value v_0 during the next frame data shifting in which W 0s appear on the output of DD_i. In addition, the output of MU_i during the 3rd (4th) frame data shifting in the procedure (1) becomes logic values v_2 and v_3 by turns corresponding to the input values 0 and 1 from DSR_{i-1}, respectively. Thus, it is assured that the input sequence v_0 v_0 v_3 v_1 v_1 v_2 which corresponds to the characteristic input sequence (I_1) can be applied to DD_i during the 5th frame data shiftings. The state transition diagram of DD_i, therefore, can be written by the use of v_0, v_1, v_2 and v_3 as shown in Fig. 6, where $s_0(s_1)$ is a label of the internal state in which DD_i produces the logical value 0 (1). v_0, v_1, v_2 and v_3 correspond to values 0,1,0 and 1 of the data input of DSR_i from their definitions, respectively. This means that if the data input of DSR_i is 0 (1), the next value of the output of DSR_i become 0 (1) independently of the values, v_0, v_1, v_2 and v_4 during frame data shifting in configuration process. In the same way, we can prove that DSR_i operates correctly during any frame data shifting in also readback process. Hence, Lemma 1 holds.

ii) The case of $w-5 \leq i \leq w$.
The proof for this case can be easily given by replacing the procedure (1) and I_1 with the procedure (2) and I_2, respectively, on the way of the process of the proof for the case i). □

Hereafter, we assume that DSR_i for $^{\forall}i$ operates correctly during any frame data shifting in both configuration and readback processes.

[Lemma 2] Assume that the outcome of TP-D is correct. TB_i for $^{\forall}i$ operates correctly, if $G=0$ and the output value of DD_i is 0.

[Proof] Assume that the outcome of TP-D is correct. The hardware of the programming circuit is designed so that the logical value 0 is forcibly supplied to the input of the DSR on the way of shifting out the contents of frame data in the procedure (2). The input of TB_i maintains 0 during the period from the end of a frame reading to the start of the next frame reading, because the shifting function of the DSR is valid from Lemma 1. And the control signals G and P are held 0 on the way of the shifting mentioned above. Thus, if the output of TB_i is not at high-impedance state by any fault, the value on b_i is fixed to a certain constant voltage independently of the presence of fault in PC_i and regardless of the value stored in M_i^j. This means the same values are read out from M_i^1, M_i^2, $\cdots$, M_i^F in the procedure (2). This is contradiction. □

Hereafter, we assume that TB_i for $^{\forall}i$ operates correctly, if $G=0$ and the output value of DD_i is 0.

[Lemma 3] If the outcome of TP-D is correct, then PC_i for $^{\forall}i$ has no fault.

[Proof] If there exists a fault which causes PC_i to charge b_i when $P=0$, the value on b_i is fixed to 1 regardless of the content of M_i^j. Thus, the fault can be detected by TP-D. So, we assume that when $P=0$, PC_i never charges b_i. If there exists a fault which causes PC_i to never charge b_i when $P=1$, the value 1 stored in M_i^j can never been transmitted to DD_i from the assumption (A3). Hence, Lemma 3 holds. □

Hereafter, we further assume that PC_i for $^{\forall}i$ has no fault.

[Lemma 4] If the outcome of TP-D is correct, then DD_i for $^{\forall}i$ either has no fault or a contrary fault.

[Proof] Suppose that the output of TB_i is at high-impedance state when $G=0$ and the output value of DD_i is 1. It is apparent from the argument for the proof of Lemma 1 that $v_0=v_2$. Even if TB_i has some fault, we can also derive from the assumption (A4) and the argument for the proof of Lemma 1 that $v_0=v_2$ or $v_1=v_3$. Thus, we can obtain that $v_0 \neq v_1$ in either of the cases, because it is apparent that $v_0 \neq v_3$ and $v_1 \neq v_2$. If v_0 equals to 0 and 1, the state transition diagram in Fig. 6 is identical with that of D-type latch with no fault and a contrary fault, respectively. Hence, Lemma 4 holds. □

Hereafter, we assume that DD_i for $^{\forall}i$ either has no fault or a contrary fault.

[Lemma 5] If the outcome of TP-D is correct, then MU_i for $^{\forall}i$ either has no fault or a contrary fault.

[Proof] Suppose that there exists some fault in MU_i. It is assured from the assumption (A4) and Lemmas 2 $\sim$ 4 that all the possible input patterns can be applied to MU_i on the way of the execution of TP-D. From Lemma 4, the responses of MU_i for such input patterns are transmitted to the output of DD_i by the next $DCLK$, as true values or their complements. Hence, the lemma holds. □

[Lemma 6] Assume that the outcome of TP-D is correct. If $G=1$, then TB_i for $^{\forall}i$ operates correctly.

[Proof] This can be easily proved from Lemmas 2 $\sim$ 5. □

It is clear from Lemmas 2 $\sim$ 6 that TP-D can detect all the faults except contrary ones which exist in DSR modules under the conditions described in 2.2. Then, the following theorem holds.

[Theorem 1] TP-D detects all the faults in the DSR except contrary ones which are redundant. □

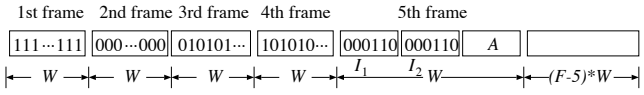

Figure 5: Configuration bit-stream of TP-D

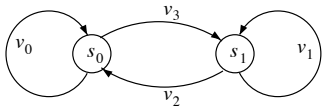

Figure 6: State transition diagram of DD_i

4 Test procedure for address shift register

It is apparent from Fig. 1 and Fig. 3 that the inputs of the ASR are not controllable directly and the outputs of the ASR, *wordline*s, are not observable directly. For these restrictions, we will apply all the possible input patterns to ASR through the control circuit in configuration and readback processes, and test the outputs of the ASR by observing the bit sequence appearing on D_{out}. We should select a configuration bit-stream so that it never mask wrong outputs of the ASR. A test procedure for ASRs is shown as follows.

[TP-A:Test Procedure for ASRs]

(1) Configure FPGA by loading a configuration bit-stream which satisfies the following condition.

[Condition] Let fd_j $(1 \leq j \leq F)$ be the substream to be loaded into j-th frame. For $^{\forall}j_1$, $^{\forall}j_2$ $(1 \leq j_1, j_2 \leq F, j_1 \neq j_2)$, fd_{j_1} and fd_{j_2} never have any ordered relation[10],[11], that is, neither $fd_{j_1} \leq fd_{j_2}$ nor $fd_{j_1} \geq fd_{j_2}$ holds.

(2) Read out the contents of the configuration memory cell array and observe a bit-stream appearing on D_{out}. □

In the test of the ASR, we can consider from the assumptions (A1) and (A2) that every function block of the programming circuit except the ASR is correct and there exist some faults in at most one module of the ASR. If we observe the correct bit-stream in the procedure (2), then there exist no fault in the ASR except redundant ones. This can be proved as follows.

[Lemma 7] If the outcome of TP-A is correct, then AD_j for $^{\forall}j$ $(1 \leq j \leq F-1)$ either has no fault or a redundant one.

[Proof] Suppose that there exists some fault in AD_j for $^{\exists}j$ $(1 \leq j \leq F-1)$. From the assumption (A2), ASR_h for $^{\forall}h$ $(1 \leq h \leq F, h \neq j)$ has no fault. Thus, the input sequence($\underbrace{00\cdots0}_{j-1}$ 1 $\underbrace{00\cdots0}_{F-j}$) can be applied to AD_j on the way of the executions of TP-A. If the fault is not redundant, the output sequence of AD_j is one of the following five cases. Note that the output sequence of AD_{F-1} is one of the first four cases.

(A) The j_1-th bit is 1 for $^{\exists}j_1$ $(1 \leq j_1 \leq j-2)$.

(B) The j_1-th bit is 0 for $^{\forall}j_1$ $(1 \leq j_1 \leq j-2)$ and the $j-1$-th bit is 1.

(C) The j_1-th bit is 0 for $^{\forall}j_1$ $(1 \leq j_1 \leq j-1)$ and the j-th bit is 0.

(D) The j_1-th bit is 0 for $^{\forall}j_1$ $(1 \leq j_1 \leq j-1)$, the j-th bit is 1 and the j_2-th bit is 1 for $^{\exists}j_2$ $(j+1 \leq j_2 \leq F-1)$.

(E) The j_1-th bit is 0 for $^{\forall}j_1$ $(1 \leq j_1 \leq j-1)$, the j-th bit is 1, the j_2-th bit is 0 for $^{\forall}j_2$ $(j+1 \leq j_2 \leq F-1)$ and the F-th bit is 1.

In the case (A), the outputs of both AD_{j_1} and AD_j are 1s at the j_1-th cycle of $ACLK$ on the way of the executions of the procedures (1) and (2). So, at least two outputs w_{j_1+1} and w_{j+1} which are produced from ASR_{j_1+1} and ASR_{j+1} are activated simultaneously at the next clock cycle . Thus, plural frames are selected simultaneously. In the same way, the output sequence of AD_j in the case (D) has at least two 1s, so that plural frames are selected simultaneously.

In the cases (B) and (E), if WD_j has such a fault that causes it to produce 0 no matter when its input from AD_j is 1, no frames are selected at j-th $ACLK$. If otherwise, however, plural frames are selected at that time.

In the case (C), if WD_j has such a fault that causes it to produce 1 no matter when its input from AD_j is 0, plural frames are selected at j-th $ACLK$. If otherwise, however, no frames are selected at that time in the case (C).

Next, we will show that both plural frame selection and no frame selection can be detected by TP-A.

If the ASR selects no frames at j-th $ACLK$ in the procedure (2), a wrong frame data $11\cdots1$ $(\neq fd_j)$ is read out, because the charges on b_is are not lost. If it selects plural frames at j-th $ACLK$ in the procedure (2), bitwise-AND of their contents is read out to D_{in} because of the structure of configuration memory cell array[4]. If at least one of the contents differs from the others, the bitwise-AND becomes a wrong frame data $(\neq fd_j)$, and if otherwise, there exists at least one frame data which is never read out [12]. Thus, we can detect all the faults mentioned above. □

Thus, we assume that AD_j for for $^{\forall}j$ $(1 \leq j \leq F-1)$ has no fault, hereafter.

[Lemma 8] If the outcome of TP-A is correct, then WD_j for $^{\forall}j$ $(1 \leq j \leq F-1)$ has no fault.

[Proof] It is assured from Lemma 7 that all the possible input patterns can be applied correctly to WD_j

for $^{\forall}j$ $(1 \leq j \leq F-1)$ on the way of the execution of TP-A. Suppose that there exists some fault in WD_j for $^{\exists}j$ $(1 \leq j \leq F-1)$. Thus, the fault makes some WD_j's responses corresponding to the input patterns wrong. We will therefore show that the fault can be detected by TP-A as follows.

If the input pattern to activate w_j fails due to the fault, f_{d_j} is nonexistent in j-th frame at the procedure (2). Thus, it can be considered that the input pattern never fail to activate w_j, because the fault can be detected by TP-A. If the input pattern to activate *wordline* other than w_j activates w_j due to the fault, plural frames are selected simultaneously in the procedure (2). Thus, the fault can be detected by TP-A. It can be therefore considered that the input pattern never activate w_j. If one of the input patterns not to activate any *wordline* activates w_j due to the fault (The input patterns occur only the period that w_j for $^{\forall}j$ $(1 \leq j \leq F)$ should be 0), the content of j-th frame become $11\cdots1$ in the procedure (2), because w_j is activated while PC_is precharge b_is. It is clear that the fault can be detected by TP-A. Hence, Lemma 8 holds. □

[**Lemma 9**] If the outcome of TP-A is correct, then AD_F and WD_F either have no fault or redundant ones.

[Proof] This can be easily proved from Lemmas 7 and 8. □

From Lemmas 7 $\sim$ 9, the following theorem holds.

[**Theorem 2**] TP-A detects all the faults except redundant ones in the ASR. □

5 Case study

In this section, we try to apply the test procedures presented in the previous sections to XC4025 of the Xilinx XC4000 family. In this family, the repetition rate of $DCLK$ is 1 MHz. The time required to execute one time of loading (reading) is 1 μs $\times$ the number of the configuration memory cells[4]. Since XC4025 has 346 $\times$ 1220 configuration memory cells ($F = 346$, $W = 1220$), it takes about 0.8 seconds to test the DSR (ASR) of XC4025 by TP-D (TP-A). In a word, even if the test procedures are applied to such FPGAs with high logic density as XC4025, the time required to execute them is short.

6 Conclusion

In this paper, we considered testing for the DSR and the ASR in the programming circuit of FPGAs, under the assumption that at most one module included the DSR or the ASR may have fault. We also derived the test procedures for DSRs and ASRs. Each of them requires only one loading and one reading.

One of our future works is to consider more efficient testing for FPGAs, by combining the test procedures for all components each other.

References

[1] S. D. Brown, R. J. Francis, J. Rose and Z. G. Vranesic, "Field-programmable gate arrays," Kluwer Academic Publishers, 1992.

[2] S. M. Trimberger, "Field-programmable gate array technology," Kluwer Academic Publishers, 1994.

[3] J. V. Oldfield and R. C. Dorf, "Field- programmable gate array technology," John Wiley & Sons, Inc., 1995.

[4] *The Programmable Logic Data Book*, Xilinx Inc., 1994.

[5] I. Pomeranz and S. M .Reddy, "Testability considerations in technology mapping," *Proc. ATS '94*, pp. 151–156, Nov. 1994.

[6] H. Tsuboi, H.Nakada and T. Miyazaki, "Testing for circuits realized as FPGAs using register insertion method," Technical report of IEICE, *FTS94-53*, pp. 55–60, Oct. 1994.

[7] T. Inoue, H. Fujiwara, H. Michinishi, T. Yokohira and T. Okamoto, "Universal Test Complexity of Field-Programmable Gate Arrays," *Proc. ATS '95*, pp. 259–265, Nov. 1995.

[8] H. Michinishi, T. Yokohira, T. Okamoto, T. Inoue and H. Fujiwara, "A Test Methodology for Interconnect Structures of LUT-based FPGAs," *Proc. ATS '96*, pp. 68–74, Nov. 1996.

[9] D. E. Farmer, "Algorithms for Designing Fault-detection Experiments for Sequential Machines," IEEE Trans. on Comput., *Vol. C-22*, No. 2, Feb. 1973.

[10] J. M. Berger, "A Note on Error Detecting Codes for Asymmetric Channels," Inf. Control, *Vol. 4*, pp. 68–73, Mar. 1961.

[11] C. V. Frieman, "Optimal Error Detection Codes for Completely Asymmetric Binary Channel," Inf. Control, *Vol. 5*, pp. 64–71, Mar. 1962.

[12] W. K. Fuchs and J. A. Abraham, "A Unified Approach to Concurrent Error Detection in Highly Structured Logic Arrays," *Proc. FTCS-14*, pp. 4–9, June 1984.