Diagnosis for Interconnect Faults in Memory-based Reconfigurable Logic Device

Xihong Zhou, Senling Wang, Yoshinobu Higami and Hiroshi Takahashi
Dept. of Computer Science, Ehime University
Matsuyama, Japan
g863003a@mails.cc.ehime-u.ac.jp

***Abstract*—The memory-based reconfigurable logic device (MRLD) is a new type of logic reconfigurable device constructed by general SRAM array in a special internal connection structure that offers many advantages including the small delay, low production cost and energy efficiency (low power), is thus an alternative to edge computing for AI and IoT applications. In order to guarantee the reliability of MRLD, this paper proposes a diagnostic process to locate a single stuck-at fault on the interconnect network of SRAM array of MRLD. The proposed method creates fault propagation paths in row-direction and column-direction through a pre-generated test set, and determines the coordinate of the fault by observing the location of the faulty value mapped on the outputs. Experimental results with fault injection confirmed the effectiveness of the proposed diagnostic method.**

Keywords—reliability, reconfigurable device, interconnect faults, fault diagnosis, memory-based reconfigurable logic device

I. INTRODUCTION

Nowadays, programmable logic device such as FPGAs (Field Programmable Gate Arrays) is gaining increasing attention for implementing the applications of AI (Artificial Intelligence) and IoT (Internet of Things) systems. A programmable logic device allows the developer/user to edit/modify the hardware logic of functions in the field that offers a flexible platform to implement the hardware accelerator for the algorithm consists of large amount of arithmetic operations such as deep neural networks (DNN) in a rapid development cycle including the designing, implementing and debugging [1][2].

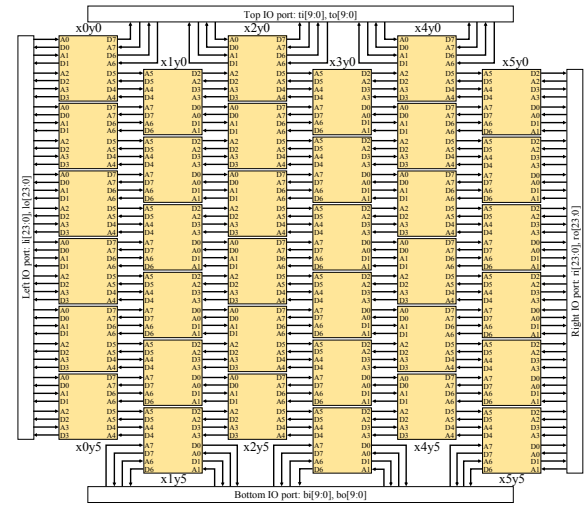
MRLD [3] (Memory-based Reconfigurable Logic Device) is a new type of programmable logic device which is under development as an alternative edge computing device for AI/IoT applications [4]. In contrast to FPGA which usually consists of a very large programmable interconnect network (switch matrix, programmable switch blocks) to realize the programmability and a small array of configurable logic blocks (CLBs), MRLD is constructed by an array of MLUTs (Multiple Look-Up Tables) in a mesh connection structure w/o any programmable interconnect resources as shown in Figure 1 (a). In a MRLD device, MLUT (Multiple Look-Up Table) is the basic reconfigurable element which consists of four general SRAMs (see Figure 1 (b)) and are connected to each other in mesh connection through m-bit address-inputs and m-bit data-outputs called AD-pair interconnects. In such a structure, each MLUT can work in either memory mode or logic configuration mode. The logic function and wire function can be directly configured into the MLUT by writing the corresponding truth tables into the SRAMs. Hence, the large amount of interconnect resources like in FPGA are not needed anymore, it thus makes a highly density of reconfigurable device with small delay and low power possible.

In order to improve the yield and guarantee the reliability of MRLD device, extensive production tests with high quality are required to detect the possible defects exist in the SRAMs and in the AD-pair interconnects between MLUTs. The former defects can be tested by conducting the existing test technologies of SRAM memory [5][6]. For the latter, we have analyzed the interconnect fault models including the stuck-at faults and bridge faults at the AD-pair interconnects between the MLUTs, and proposed the test approaches for detecting the stuck-at fault and bridge fault, respectively in [7] and [8].

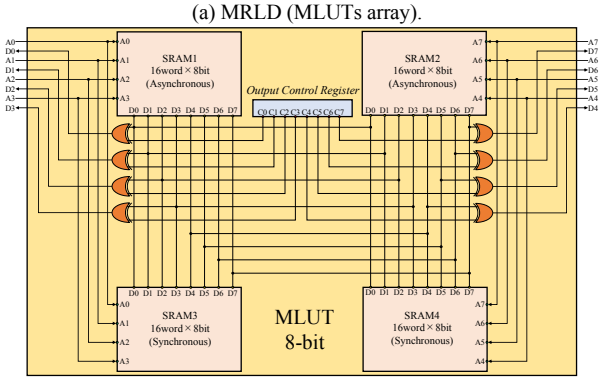
Beside the fault detection, fault diagnosis is also known to play an important role in improving the yield and reliability of products. In manufacturing, to diagnose the location of interconnect fault in the MLUTs array is beneficial to improving the process. When the MLRD is put in actual use in the field, to locate the interconnect fault is helpful to avoid configuring the logic into a faulty MLUT block for high reliability. The fault diagnosis for locating the interconnect fault in FPGA has been investigated deeply [9][10]. In [11] an universal fault diagnosis technique is presented for locating the interconnect fault in the Lookup Table array of a FPGA device. The method can diagnose all faulty points in LUT array through two steps: the horizontal diagnosis and the vertical diagnosis. For MRLD constructed by a MLUTs array, the basic idea presented in [11] is also available. However, implementing the horizontal and vertical diagnosis in MRLD must be considered carefully, because the interconnects between MLUTs are un-reconfigurable.

In this paper, we proposed a diagnostic method to identify the location of a stuck-at fault at the interconnect between MLUTs. The proposed method is consisted with two phases. The first phase creates fault propagation paths in row-direction and column-direction through a pre-generated test set including test cubes configured in MLUTs and patterns applied in the external inputs. The second phase determines the coordinate of the target interconnect fault by observing the location of the faulty value in the external outputs of MRLD. Main contribution of this paper is to address not only the fault detection but also the fault diagnosis of MRLD. To evaluate the method, we design an MRLD with 6×6 MLUTs array and perform the logic simulation experiments by injecting the stuck-at fault node to the netlist of the MRLD. The results confirmed the effectiveness of the proposed diagnostic method which can diagnose the location of the injected stuck-at fault.

This paper is organized as follows. Section II introduce the architecture of the MRLD and then describe the stuck-at faults model in MRLD. Section III propose a method for diagnosing the location of the stuck-at interconnect faults of MLUTs in MRLD. Section IV shows the experimental results for evaluating the proposed diagnostic method of the stuck-at interconnect faults. Section V concludes the paper.



(a) MRLD (MLUTs array).



(b) Structure of a single MLUT.

Fig. 1. MRLD structure.

II. Stack-at Interconnect Faults in MRLD

In this section, we first introduce the MRLD architecture and then describe the stuck-at interconnect faults in MRLD.

A. Structure of MRLD

MRLD consists of multiple general-purpose memory cells (MLUTs: Multiple Look-Up Tables) arranged in an array. Figure 1 (a) shows the structure of an MRLD composed of 6×6 8-bit (with 8 pairs of AD interconnects) MLUTs. Between the MLUTs, address input lines and data output lines are bidirectionally interconnected in pairs (called AD pairs). The address input lines of each MLUT are connected to the data output lines of its adjacent MLUTs. The address input line and data output line of the outermost MLUT are connected to the IO (Input/Output) ports of the MRLD device.

Figure 1 (b) shows the structure of a single 8-bit MLUT. The MLUT consists of two asynchronous SRAMs (SRAM1, SRAM2) and two synchronous SRAMs (SRAM3, SRAM4). For the address input of the asynchronous SRAM and the synchronous SRAM, the upper 4 bits and lower 4 bits of the address input line of the MLUT are shared and used in order. The address input of the synchronous SRAM is controlled by the clock. Asynchronous SRAM address input executes asynchronous operation by detecting the address change via ATD (Address Transition Detector). The data output line of the SRAM is connected to the OR gate. In addition, a 8-bit ORC (Output-Control-Register) controls the data output line of the MLUT through an XOR gate.

In such architecture, each SRAM works as a single look-up table (LUT), users can configure logics or wires in the LUT by writing the corresponding truth tables in the SRAM of the reconfigurable element MLUT. The Figure 2 shows an example to configure a logic circuit in two MLUTs. The circuit has two

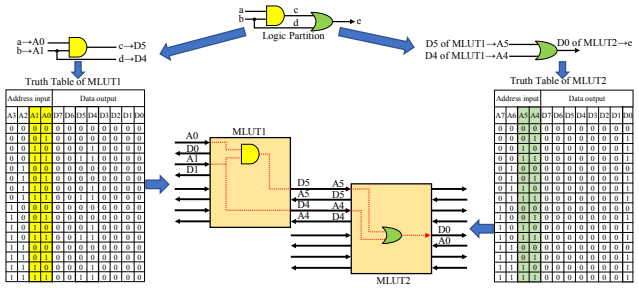
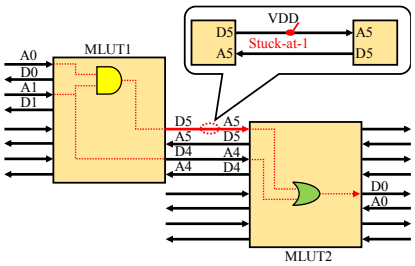


Fig. 2. Configure a logic circuit in two MLUTs.



Logic behavior of Stuck-at Fault

MLUT1_D5	MLUT2_A5	MLUT2_D0
0	0/1	MLUT2_D0/1
0	0/1	MLUT2_D0/1
1	1	1
1	1	1

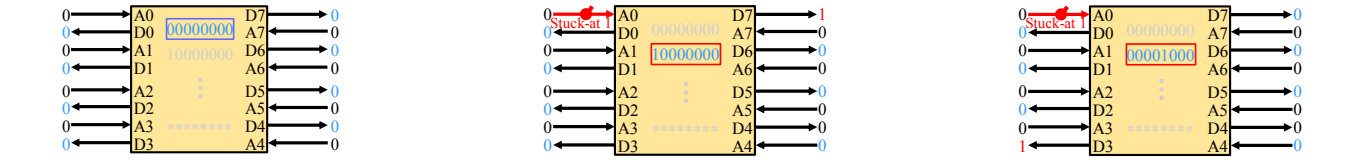
Fig. 3. Stuck-at-1 interconnect fault of MLUT.

inputs a and b, two internal signal lines c and d, and an output e. First, a logic partition is performed to divide the circuit into two sub-logics. Then, determining the address input and data output lines of the MLUTs in according to each sub-logic (e.g.: a→A0, b→A1, c→D5 and d→D4), and computing the truth table of the sub-logics. Finally, writing the truth tables in the SRAMs within the MLUTs. It is worth to note that wires can be configured in MLUTs as logic interconnects which can provide smaller delay and lower power consuming than FPGA.

B. Stuck-at interconnect faults in MRLD

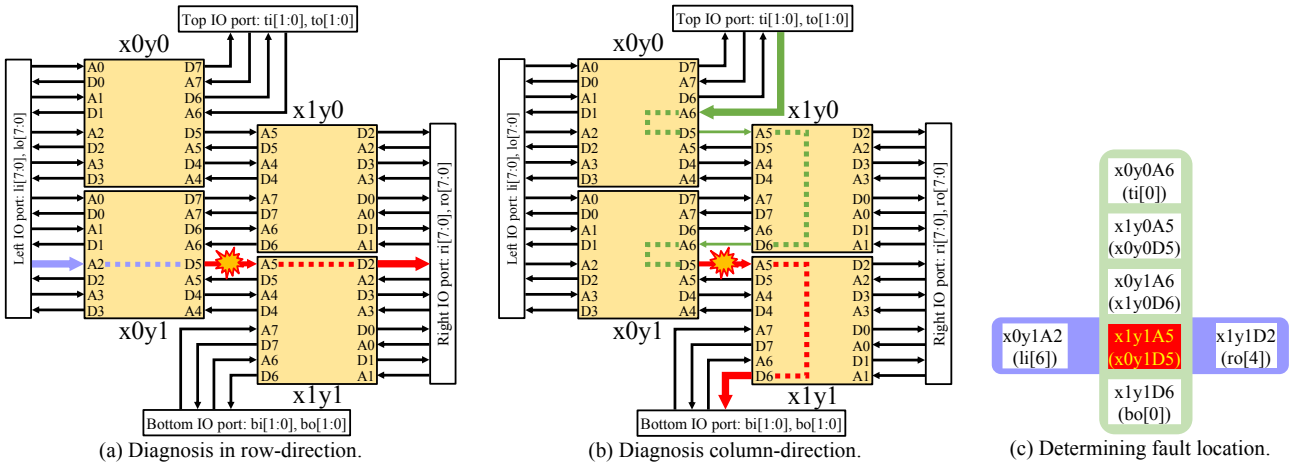
As described in subsection A, MLUTs are connected with each other by AD-pair interconnects including the address inputs and data outputs. The address inputs of a target MLUT come from the data outputs of its neighbor MLUTs and will access the look-up table stored in the target MLUT to generate logic outputs. A defect at the AD-pair interconnect can cause a change of the address data that would result in logical faults in the configured circuit. Stuck-at faults are typical fault models for wiring interconnects. If there is a short between the ground (supply) and AD pair interconnect (address line or data line), the value of address input (data output) of the MLUT will be fixed at logic 0 (logic 1). Figure 3 shows an example of the behavior of stuck-at fault of the AD-pair interconnects. We call a stuck-at fault at an interconnect **a stuck-at interconnect fault.** Suppose that we configure the circuit shown in Figure 2 in MLUT1 and MLUT2, and a stuck-at-1 (sa-1) fault occurs at the AD-pair interconnect MLUT1_D5 → MLUT2_A5. The address input A5 of MLUT2 will be fixed at 1 that dominates the output of OR logic to 1 and blocks the propagation of logic value generated by D4 of MLUT1 to access the address A4 of MLUT2 to generate a logic OR output at D0 of MLUT2.

In [7], we have proposed the testing method for detecting stuck-at interconnect faults of MRLD. The testing strategy of the method is to store the test data (call them Test Cubes) in the SRAMs of MLUTs, and observe the change of logic outputs of MRLD by applying the external logic inputs of MRLD (call them External Patterns) and performing the logic operation. It is an effective method for detecting whether a



(a) Data outputs come from the look-up table. (b) Stuck-at fault affect data outputs. (c) Data outputs vary with the look-up table.

Fig. 4. Stuck-at Fault change of the address to affect data outputs.



(a) Diagnosis in row-direction. (b) Diagnosis column-direction. (c) Determining fault location.

Fig. 5. Diagnostic strategy

fault exists in MRLD, but it cannot diagnose the location of the fault. In this paper, we proposed the diagnostic method to diagnose the location of the stuck-at interconnect faults of MRLD.

III. Fault Dignosis for MRLD

In this section, we introduce the method to diagnose the location of stuck-at interconnect faults of MLUTs in MRLD. First, we introduce the diagnostic strategy. Then, we describe the diagnostic process for stuck-at interconnect fault.

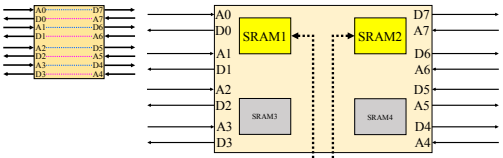
A. Diagnostic strategy

In MRLD, since the data output of an MLUT comes from the contents of look-up tables stored in the SRAMs accessed by the address inputs [Figure 4 (a)], a stuck-at (sa) fault at AD pair interconnect can cause a change of the address that would access the different contents of the look-up tables [Figure 4 (b)]. On the other hand, for the same of address input, the data outputs vary according to the contents of look-up tables accessed [Figure 4 (c)]. Therefore, for diagnosing the location of a stuck-at fault at AD pair interconnect, a simple strategy is to design the look-up tables which can propagate the fault to the different output ports of MRLD in different paths. Then to view the output ports and find the ports with fault output. Finally to determine the fault location via computing the intersection of propagation paths corresponding to different ports with fault output.

Figure 5 shows an example of the diagnostic strategy proposed. Here for ease of explanation, we use an MRLD with 2×2 MLUTs array, and suppose a stuck-at fault at the address input A5 of MLUT x1y1 (or the data output D5 of MLUT x0y1). To diagnose the location of stuck-at fault at the x1y1A5 (x0y1D5), we do diagnosis in row-direction and column-direction, respectively. For row-direction diagnosis, to configure the look-up tables into SRAMs of MLUTs, making the stuck-at fault propagate to the output port of MRLD in the row-direction. As shown in Figure 5 (a), when an external pattern is applied to input ports of MRLD, the li[6] (x0y1A2) will be faulted at x1y1A5 (x0y1D5) and then the fault is propagated to the output port ro[4] (x1y1D2) of MRLD. The stuck-at Fault Path in row-direction (rFPsa) can be determined as rFPsa = {x0y1A2 (li[6]), x1y1A5 (x0y1D5), x1y1D2 (ro[4])}. For column-direction diagnosis, to configure the look-up tables into SRAMs of MLUTs, making the stuck-at fault propagate to the output port of MRLD in the column-direction. As shown in Figure 5 (b), when an external pattern is applied to the input ports of MRLD, the ti[0] (x0y0A6) also will be faulted at x1y1A5 (x0y1D5) and then the stuck-at fault is propagated to the output port bo[0] (x1y1D6) of MRLD. The stuck-at Fault Path in column-direction (cFPsa) can be determined as cFPsa = {x0y0A6 (ti[0]), x1y0A5 (x0y0D5), x0y1A6 (x1y0D6), x1y1A5 (x0y1D5), x1y1D6 (bo[0])}. After diagnosing in the row-direction and column-direction, respectively, as shown in Figure 5 (c), the fault location can be determined out through compute the intersection of rFPsa and cFPsa. i.e. x1y1A5 (x0y1D5) = rFPsa ∩ cFPsa.

The diagnostic strategy is as follows.

1) *Row-direction diagnosis:* Creating the stuack-at faults propagation path in row-direction (rFPsa) as follws.
   a) *Writing Diagnostic Test Cubes:* Configuring the diagnostic test look-up tables in MLUTs for exciting the fault propagation in row-direction.
   b) *Appliying External Patterns:* Appliying external pattern to input ports of MRLD.
   c) *Obtaining Fault Path:* Determining the fault path in row-direction via observing the output port with fault port of MRLD.
2) *Col-direction diagnosis:* Creating the stuack-at faults propagation path in col-direction (cFPsa) as follws.
   a) *Writing Diagnostic Test Cubes:* Configuring the diagnostic test look-up tables in MLUTs for exciting the fault propagation in column-direction.
   b) *Appliying External Patterns:* Appliying external pattern to input ports of MRLD.
   c) *Obtaining Fault Path:* Determining the fault path in column-direction via observing the output port with fault port of MRLD.
3) *Determining fault location:* Finding out stuak-at faults location (LOCsa) through computing the intersection of rFPsa and cFPsa: rFPsa ∩ cFPsa.



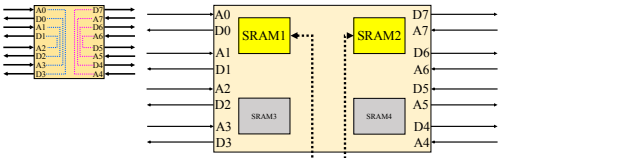
Cube1

Address				Data							
A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0	0	0	0	0
*	*	*	*	*	*	*	*	0	0	0	0
1	1	1	1	1	1	1	1	0	0	0	0

Cube2

Address				Data							
A7	A6	A5	A4	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	1	0	0	0
0	0	1	0	0	0	0	0	0	1	0	0
*	*	*	*	0	0	0	0	*	*	*	*
1	1	1	1	0	0	0	0	1	1	1	1

Fig. 6. Diagnostic Test Cubes for row-direction.



Cube1

Address				Data							
A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	1	0	0	0
0	0	1	0	0	0	0	0	0	1	0	0
*	*	*	*	0	0	0	0	*	*	*	*
1	1	1	1	0	0	0	0	1	1	1	1

Cube2

Address				Data							
A7	A6	A5	A4	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0	0	0	0	0
*	*	*	*	*	*	*	*	0	0	0	0
1	1	1	1	1	1	1	1	0	0	0	0

Fig. 7. Diagnostic Test Cubes for column-direction.

B. *Diagnostic test generation for stuck-at interconnect faults*

In this subsection, we describe concretely the diagnostic test generation for stuck-at interconnect faults on the basis of the diagnostic strategy proposed in subsection A.

For row-direction diagnosis, we propose the Diagnostic Test Cubes in row-direction (rDTC). The rDTC consists of two Cubes (Cube1 and Cube 2). The Cube 1 and the Cube 2 are written respectively into SRAM1 (or SRAM3) and SRAM2 (or SRAM4) for creating the path of fault propagation in row-direction. When applying External Pattern all-zero to input ports of MRLD, the stuck-at-1 (sa-1) fault will be propagated to output port of MRLD in row-direction, thus the path for stuck-at-1 fault in the row-direction (rFPsa1) will can be obtained by observe the output ports of MRLD. When applying External Pattern all-one to input ports of MRLD, the stuck-at-0 (sa-0) fault will be propagated to output port of MRLD in row-direction, thereby the path for stuck-at-0 fault in the row-direction (rFPsa0) will can be obtained by observe the output ports of MRLD. For an MLUT with m pairs of AD interconnects A[m-1:0] and D[m-1:0], and constructed by four $2^{m/2}$word×m-bit SRAMs including two asynchronous SRAMs (SRAM1, SRAM2) and two synchronous SRAMs (SRAM3, SRAM4). The diagnostic test generation in row-direction for stuck-at (sa) fault is as follows.

Process 1: *Diagnostic Test Generation in row-direction for sa*

[rDTC]

Cube 1: *For the SRAMs share the low-order address inputs (A[m/2-1:0]) of MLUT, set contents of the address lines A[m/2-1:0] to D[m-1:m/2]=A[0:m/2-1], D[m/2-1:0]=all-zero.*

Cube 2: *For the SRAMs share the high-order address inputs (A[m-1:m/2]) of MLUT, set contents of the address lines A[m-1:m/2] to D[m-1:m/2]=all-zero, D[m/2-1:0]=A[m/2:m-1]*

[Test for sa-1 fault]

External Pattern*: Apply pattern all-zero to the input ports.*

Fault Path*: The path (rFPsa1) that the output port is one.*

[Test for sa-0 fault]

External Pattern*: Apply pattern all-one to the input ports.*

Fault Path*: The path (rFPsa0) that the output port is zero.*

Figure 6 shows an example of rDTC configured in an MLUT with 8 pairs of AD interconnects. For each MLUT, we write Cube1 and Cube2 in the asynchronous SRAM1 and SRAM2, respectively. In the Cube1, the low 4-bit data outputs [D3:D0] of all address are all-zero, the high 4-bit data outputs [D7:D4] of each address are [A0:A3]. In the Cube2, the high 4-bit outputs [D7:D4] of all address are all-zero, the low 4-bit outputs [D4:D0] of each address are [A4:A7]. Because that the data outputs [D7:D0] of SRAM1 and SRAM2 are connected by OR function as illustrated in Figure 1 (b), the data outputs [D7:D0] of each MLUT are the values of address inputs [A0:A7]. i.e. the address line A_k is connected to the data output line D_{7-k} for each MLUT, thus the construction of the propagation path of the fault from the row direction is realized.

Similarly, for column-direction diagnosis, we designed the Diagnostic Test Cubes in column-direction (cDTC). The diagnostic test generation in column-direction for stuck-at fault is as follows.

Process 2: *Diagnostic Test Generation in col-direction for sa*

[cDTC]

Cube 1: *For the SRAMs share the low-order address inputs (A[m/2-1:0]) of MLUT, set contents of the address lines A[m/2-1:0] to D[m-1:m/2]=all-zero, D[m/2-1:0]=A[0:m/2-1].*

Cube 2: *For the SRAMs share the high-order address inputs (A[m-1:m/2]) of MLUT, set contents of the address lines A[m-1:m/2] to D[m-1:m/2]=A[m/2:m-1], D[m/2-1:0]=all-zero.*

[Test for sa-1 fault]

External Pattern*: Apply pattern all-zero to the input ports.*

Fault Path*: The path (cFPsa1) that the output port is one.*

[Test for sa-0 fault]

External Pattern*: Apply pattern all-one to the input ports.*

Fault Path*: The path (cFPsa0) that the output port is zero.*

Figure 7 shows as an example of cDTC configured in an MLUT with 8 pairs of AD interconnects. Where, the data outputs [D7:D0] of all address are $[0000A_0A_1A_2A_3]$ and $[A_4A_5A_6A_7 0000]$ for SRAM1 and SRAM2, respectively. i.e. for each MLUT, the address line A_k in low 4-bit address (k=0,1,2,3) is connected to the data output line $D_{7-(k+4)}$ and the address line A_k in the high 4-bit address (k=4,5,6,7) is connected to the data output line $D_{7-(k-4)}$, thus the propagation path of the fault from the column direction is created.

Executing the process 1 and process 2, the paths (rFPsa and cFPsa) of stuck-at fault in row-direction and column-direction can be obtained. The location of stuck-at fault can be determined by computing the intersection of rFPsa and cFPsa. The whole diagnostic process for stuck-at fault is as follows.

Process 3: *Determining fault location of stuck-at fault.*

[Row Diagnosis]

Executing the ***process 1****, obtaining the path of sa-1 fault and sa-0 fault in row-direction (rFPsa1 and rFPsa0).*

[Col Diagnosis]

Executing the ***process 2****, obtaining the path of sa-1 fault and sa-0 fault in column-direction (cFPsa1 and cFPsa0).*

[Fault Location]

sa-1 fault location: *LOCsa1 = rFPsa1 ∩ cFPsa1*

sa-0 fault location: *LOCsa0 = rFPsa0 ∩ cFPsa0*

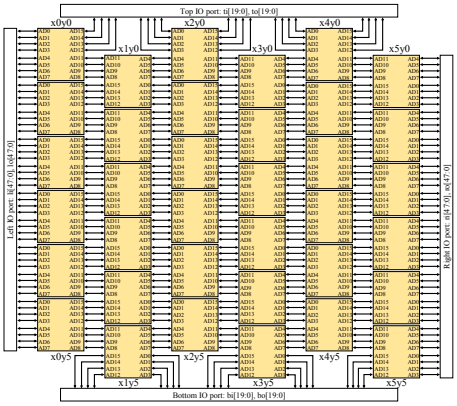


Fig. 8. MRLD with 6×6 16-bit MLUTs array.

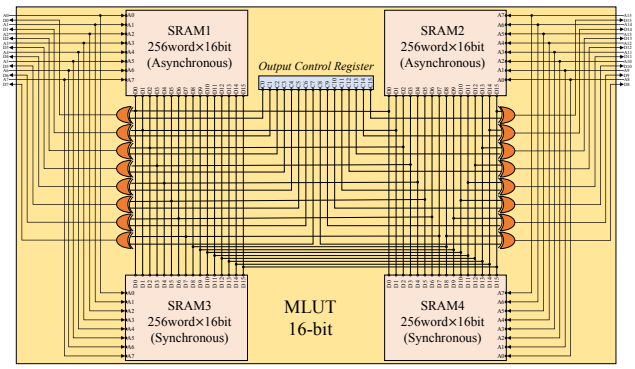


Fig. 9. The 16-bit MLUT.

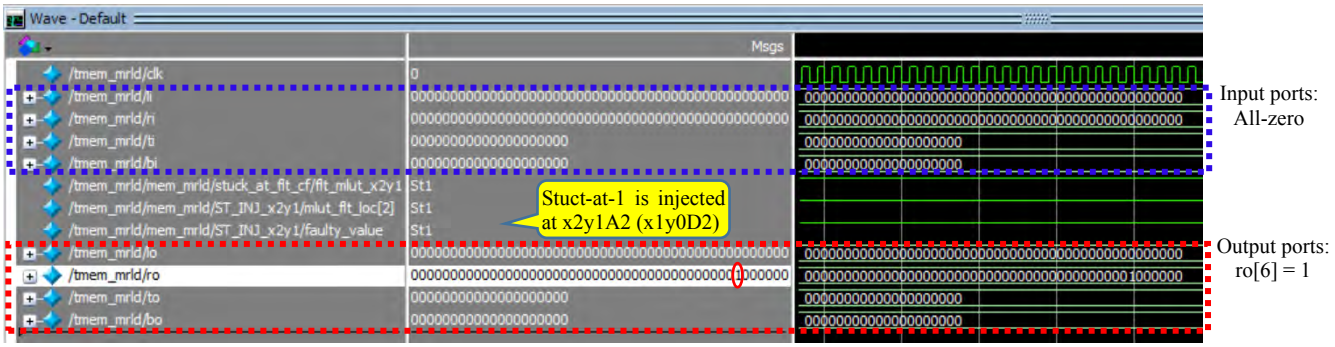


Fig. 10. Simulation results of stuck-at-1 fault injection in row-direction diagnosis

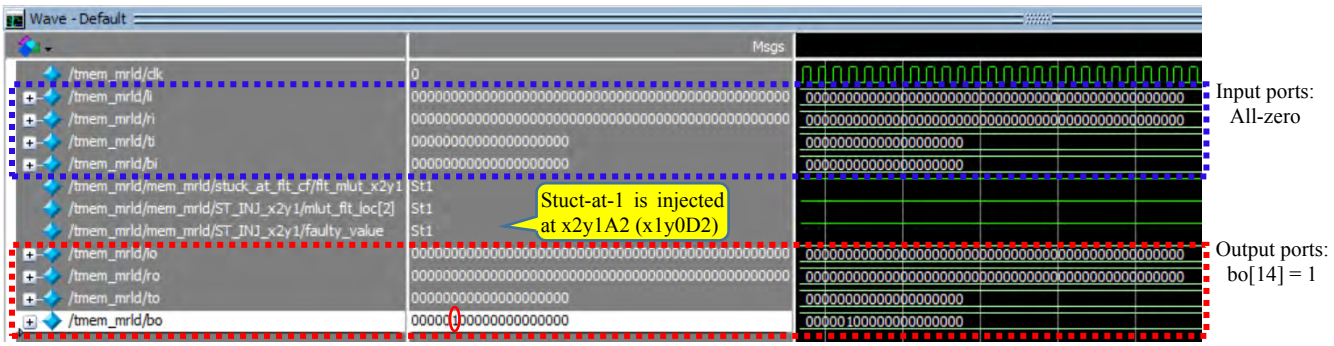


Fig. 11. Simulation results of stuck-at-1 fault injection in column-direction diagnosis

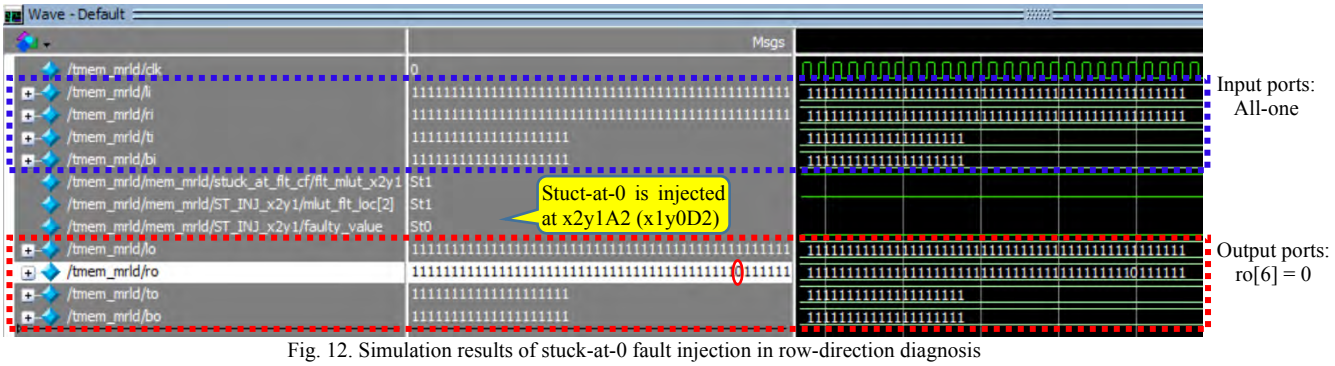


Fig. 12. Simulation results of stuck-at-0 fault injection in row-direction diagnosis

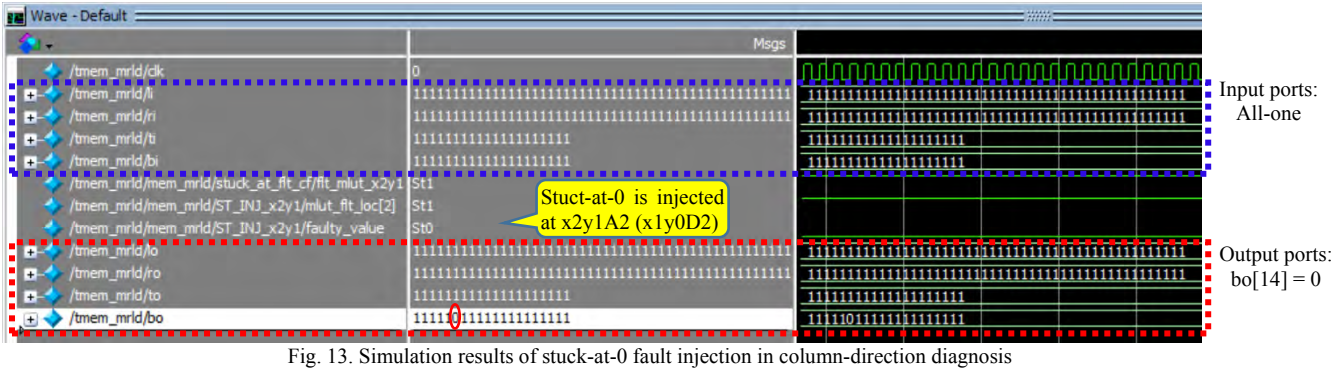


Fig. 13. Simulation results of stuck-at-0 fault injection in column-direction diagnosis

IV. Experimental Results

To evaluate the proposed diagnostic method, we designed a MRLD with 36 MLUTs arranged in a 6×6 array as shown in Figure 8. It consists of 48-bits IO ports at the left and right side, 20-bits IO ports at the top and bottom side of the array, respectively. As shown in Figure 9, each MLUT has 16 pairs of AD interconnects [A15:A0] and [D15:D0], and consists of four 256word×16bit SRAMs including two asynchronous SRAMs and two synchronous SRAMs, respectively.

To verify the capability of the fault diagnosis method, we performed the logic simulation using ModelSim by injecting the stuck-at fault (0 and 1) nodes to the netlist of MRLD, and observe the logic value at the output ports (lo, ro, to, bo) of MRLD. As process 3 described in section III, we performed the logic simulation for the row-direction diagnosis and the column-direction diagnosis, respectively.

Figure 10 shows the simulation results of stuck-at-1 fault diagnosis in row-direction. We write the diagnostic test cubes (rDTC) to in all MLUTs, and apply the external patterns all-zero to the input ports (li, ri, ti, bi) of MRLD. When a stuck-at-1 is injected into the address input A2 of MLUT x2y1 (data output D2 of MULT x1y0), the output value of ro[6] is changed to 1. The fault path in row-direction can be determined as rPFsa1 = {x0y1A2 (li[10]), x1y0A13 (x0y1D13), x2y1A2 (x1y0D2), x3y0A13 (x2y1D13), x4y1A2 (x3y0D2), x5y0A13(x4y1D13), x5y0D2 (ro[6])}. Figure 11 shows the simulation results of stuck-at-1 fault diagnosis in column-direction. We write the diagnostic test cubes (cDTC) to in all MLUTs, and apply the external patterns all-zero to the input ports, When a stuck-at-1 is injected into the address input A2 of MLUT x2y1 (data output D2 of MULT x1y0), the output value of bo[14] is changed to 1. The fault path in column-direction can be determined as cPFsa1 = {x2y0A2(ti[14]), x1y0A5 (x2y0D5), x2y1A2 (x1y0D2), x1y1A5 (x2y1D5), x2y2A2 (x1y1D2), x1y2A5 (x2y2D5), x2y3A2 (x1y2D2), x1y3A5 (x2y3D5), x2y4A2 (x1y3D2), x1y4A5 (x2y4D5), x2y5A2 (x1y4D2), x1y5A5 (x2y5D5), x1y5D2 (bo[14])}. The location of stuck-at-1 fault can be determined by computing the intersection of rFPsa1 and cFPsa1: LOCsa1 = rPFsa1 ∩ cPFsa1 = x2y1A2 (x1y0D2).

Figure 12 and Figure 13 shows the simulation results of stuck-at-0 fault diagnosis in row-direction and column-direction, respectively. we inject the stuck-at-0 fault at the same location x2y1A2 (x1y0D2) as the stuck-at-1 fault. When applying the external pattern all-one to the input ports of MRLD, the output value of ro[6] and bo[14] is changed to 0 in the row-direction diagnosis and in the column-direction diagnosis, respectively. The same fault path rPFsa0 and cPFsa0 as the fault path of stuck-at-1 is determined. The location of stuck-at-0 fault can be determined by computing the intersection of rFPsa0 and cFPsa0: LOCsa0 = rPFsa0 ∩ cPFsa0 = x2y1A2 (x1y0D2).

V. Conclutions

In this paper, we introduced the MRLD which is a promising alternative reconfigurable device for the next-generation IoT/AI edge devices. To guarantee the reliability of MRLD, we proposed a diagnostic method that can determine the fault location in the MLUT array by computing the intersection of the fault propagation paths in row-direction and column-direction. Logic simulation with fault injection fault confirmed the effectiveness of the proposed diagnostic method that can diagnose all injected stuck-at fault.

In our future work, we will evaluate the effectivity of the proposed diagnosis method for multiple stuck-at faults and explore the diagnostic generation method for locating others interconnect faults (such as open fault, bridge fault, etc.) in the MRLD device.

Acknowledgment

This work was supported in part by KAKENHI (19K20234) and the part of this study is supported by the TAIYO YUDEN CO., LTD., and the TRL Co., Ltd.

References

[1] C. Wang, L. Gong, Q. Yu, X. Li, Y. Xie, and X. Zhou, "DLAU: A Scalable Deep Learning Accelerator Unit on FPGA," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 36, no. 3, pp. 513–517, 2017, doi: 10.1109/TCAD.2016.2587683.

[2] K. Guo, S. Zeng, J. Yu, Y. Wang, and H. Yang, "[DL] A survey of FPGA-based neural network inference accelerators," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 12, no. 1, Apr. 2019, doi: 10.1145/3289185.

[3] TAIYO YUDEN CO., LTD., "MRLD," Trademark 85735093, 2014.

[4] M. Sato, K. Sato, M. Katsu, and I. Shimizu, "Reconfigurable logic device," WO Patent WO2014163099A3, 2014.

[5] T. Q. Bui, L. D. Pham, H. M. Nguyen, V. T. Nguyen, T. C. Le, and T. Hoang, "An Effective Architecture of Memory Built-In Self-Test for Wide Range of SRAM," in *2016 Int. Conf. Adv. Comput. Appl.*, pp. 121–124, 2016, doi: 10.1109/ACOMP.2016.026.

[6] A. Sharma and V. Ravi, "Built in self-test scheme for SRAM memories," in *2016 Int. Conf. Adv. Comput. Commun. Informatics*, pp. 1266–1270, 2016, doi: 10.1109/ICACCI.2016.7732220.

[7] S. Wang, Y. Higami, H. Takahashi, M. Sato, M. Katsu, and S. Sekiguchi, "Testing of Interconnect Defects in Memory Based Reconfigurable Logic Device (MRLD)," in *2017 IEEE 26th Asian Test Symp.*, pp. 17–22, 2017, doi: 10.1109/ATS.2017.16.

[8] S. Wang *et al.*, "Test Method for the Bridge Interconnect Faults in Memory Based Reconfigurable-Logic-Device(MRLD) Considering the Place-and-Route," in *2018 33th Int. Tech. Conf. Circuits/Systems, Comput. Commun.*, 2018.

[9] W. K. Huang, X. T. Chen, and F. Lombardi, "On the diagnosis of programmable interconnect systems: Theory and application," in *Proc. 14th VLSI Test Symp.*, pp. 204–209, 1996, doi: 10.1109/VTEST.1996.510859.

[10] D. Das and N. A. Touba, "A low cost approach for detecting, locating, and avoiding interconnect faults in FPGA-based reconfigurable systems," in *Proc. Twelfth Int. Conf. VLSI Des. (Cat. No.PR00013)*, pp. 266–269, 1999, doi: 10.1109/ICVD.1999.745159.

[11] T. Inoue, S. Miyazaki, and H. Fujiwara, "Universal fault diagnosis for lookup table FPGAs," *IEEE Des. Test Comput.*, vol. 15, no. 1, pp. 39–44, 1998, doi: 10.1109/54.655181.