BUILT-IN SELF-TEST OF GLOBAL ROUTING RESOURCES IN VIRTEX-4 FPGAS

Except where reference is made to the work of others, the work described in this thesis is my own or was done in collaboration with my advisory committee. This thesis does not include proprietary or classified information.

Jia Yao

Certificate of Approval:

Vishwani D. Agrawal James J. Danaher Professor Electrical and Computer Engineering	Charles E. Stroud, Chair Professor Electrical and Computer Engineering
Victor P. Nelson Professor Electrical and Computer Engineering	George T. Flowers Dean Graduate School

BUILT-IN SELF-TEST OF GLOBAL ROUTING RESOURCES IN VIRTEX-4 FPGAS

Jia Yao

A thesis

Submitted to

the Great Faculty of

Auburn University

in Partial Fulfillment of the

Requirements for the

Degree of

Master of Science

Auburn, Alabama
August 10, 2009

BUILT-IN SELF-TEST OF GLOBAL ROUTING RESOURCES IN VIRTEX-4 FPGAS

Jia Yao

Signature of Author

Date of Graduation

VITA

Jia Yao, daughter of Jun Yao and Mingqin Wang, was born on February 23, 1984 in JiNan, ShanDong, People's Republic of China. She graduated from ShanDong University with a Bachelor of Science degree in July, 2006. She entered Auburn University majoring in Electrical and Computer Engineering to pursue her Master of Science degree in August, 2007.

THESIS ABSTRACT

BUILT-IN SELF-TEST OF GLOBAL ROUTING RESOURCES IN VIRTEX-4 FPGAS

Jia Yao

Master of Science, August 10, 2009
(B.S. ShanDong University, 2006)

102 Types Pages

Directed by Charles E. Stroud

It is important to test programmable routing resources in Field Programmable Gate Arrays (FPGAs) because they take up the largest portion of configuration memory bits. In Virtex-4 FPGAs, routing resources account for over 80% of the configuration memory. Built-In Self-Test (BIST) is adopted to test the routing resources in FPGAs and overcomes issues residing in previously developed test approaches.

The cross-coupled parity BIST approach has proven to be the most effective method for testing FPGA routing resource architectures with high fault coverage. BIST configurations are developed in this thesis to test global routing resources using cross-coupled parity approach in Virtex-4 FPGAs, focusing on hex lines and long lines. The total number of BIST configurations for LX devices is 34. This number increases to 42 for SX25 and SX35 devices and to 66 for SX55 devices. Analysis and evaluations of developed BIST configurations are provided as well. All BIST configurations are downloaded and verified on LX60 and SX35 devices.

ACKNOWLEDGMENTS

First I would love to give my deepest thankfulness to my mother. It is much harder than what I thought to live and study in a foreign country. Without her endless love and encouragements, I would not complete my master's study. I would love to thank Dr. Stroud for his great guidance on my study and research. I would love to thank Brad for his helps and patience all the time. Also I would like to thank Brooks, Joey, Jie, Bobby, Lee, and Mary for their assistance on research work. To Dr. Hodel, I am grateful for his kindly advices and care since I came to Auburn. And also thanks for my sweet friends I met in Auburn—Julia Evans, Elizabeth Wills, Elizabeth Williams, Jena Robison, Man Zhang, Yueqin Lin and Binying Tan. All the encouragements give me strength not to give up. And I am so glad that I did make it. Thanks for the tough time I survived and also the happy time I enjoyed. Life is tough, always. But I will keep on moving.

Style manual or journal used IEEE Editorial Style Manual.

Computer software used The entire document, including text, figures, and tables, was prepared using *Microsoft Word*.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Field Programmable Gate Arrays (FPGAs) are commonly used reconfigurable devices that are becoming increasingly popular. FPGAs represent a significant proportion of the IC market [1]. Applications of FPGAs include a large range of areas such as digital signal processing, aerospace systems, speech recognition and so on [1]. As the size and speed of FPGAs increase, along with their advantages including short time to market, ability to reprogram and low cost prototypes, they have become more competitive and extensively used in many applications.

Testing FPGAs has been brought into the spotlight as their internal complexity increases while the feature size decreases, which may lead to decreasing reliability of FPGAs. In order to achieve high probability of proper operation, FPGAs need to be completely tested and determined fault-free before the intended system function is configured [2]. However, there can be defects during the manufacturing process, and new faults can appear during operation of FPGAs. These possible internal faults make the testing of FPGAs meaningful and important.

1.1 Overview of Field Programmable Gate Arrays (FPGAs)

The ability to reprogram FPGAs makes them useful for application to many digital logic systems. FPGA architectural components are divided into two constituents: logic resources and routing resources [3]. As illustrated in Fig. 1.1, a typical FPGA architecture consists of an array of configurable logic blocks (CLBs), programmable Input/Output blocks (I/OBs), special cores such as random access memories (RAMs), digital signal processors (DSPs), and programmable routing resources which interconnect all the internal elements in FPGAs [2]. The logic functions and the interconnections among the logic resources are determined by the configuration bit stream that must be downloaded into the FPGA. This programming data are stored in the configuration memory and each configuration bit controls the state of a logic or routing element such as a transmission gate or multiplexer [2].

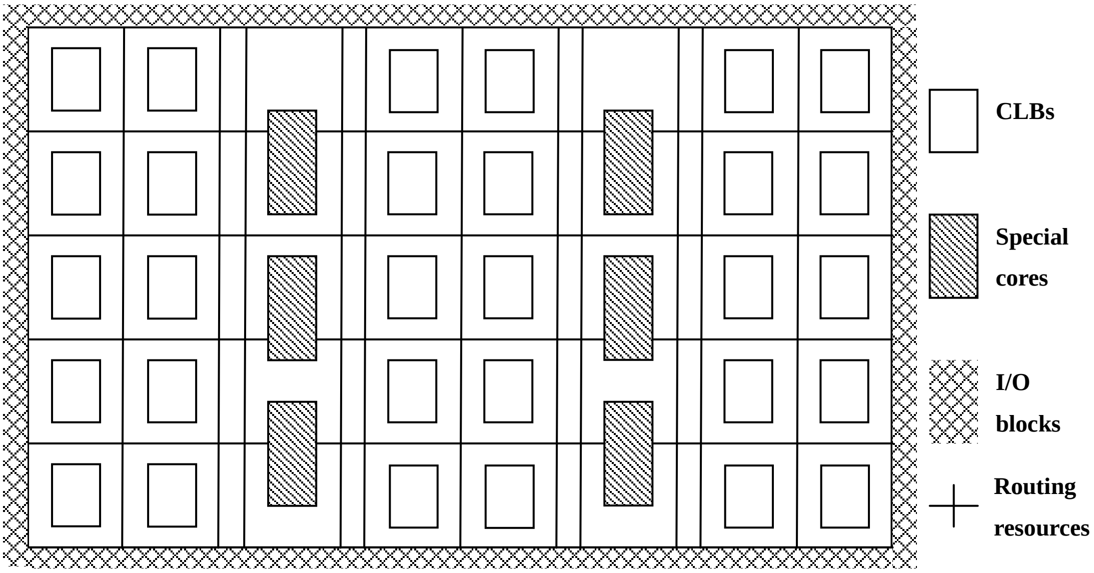


Figure 1.1 FPGA architecture

1.2 Overview of Programmable Routing Resources

Programmable routing resources take up the largest area in FPGAs and account for 80% of the total configuration memory bits [2]. Programmable routing resources can be classified as local and global routing resources. Local routing resources are specific to a given CLB and used to connect the CLB to global routing resources or to adjacent CLBs. Global routing resources interconnect all logic resources in FPGAs [2][4][5]. Fig. 1.2 illustrates a simplified view of programmable routing resources.

Global routing resources consist of horizontal and vertical wire segments of varying length along with programmable interconnect points (PIPs) which are used to connect or disconnect wire segments to create signal paths through the FPGA [4][5]. The PIPs are controlled by configuration memory bits, as illustrated in Fig. 1.3(a). The PIPs can be classified as three types: break-point PIP, cross-point PIP, and multiplexer PIP [4][5]. A break-point PIP, illustrated in Fig. 1.3 (b), can be activated to form longer wire segments, either vertical or horizontal. A cross-point PIP, illustrated in Fig. 1.3 (c), makes it is possible to connect horizontal wire segments with vertical wire segments. The multiplexer PIP, illustrated in Fig. 1.3 (d), is directional and buffered. One of multiple inputs is connected to a single output wire by activating the appropriate configuration memory bit. Most recent FPGA interconnect resources are primarily constructed from buffered multiplexer PIPs [2].

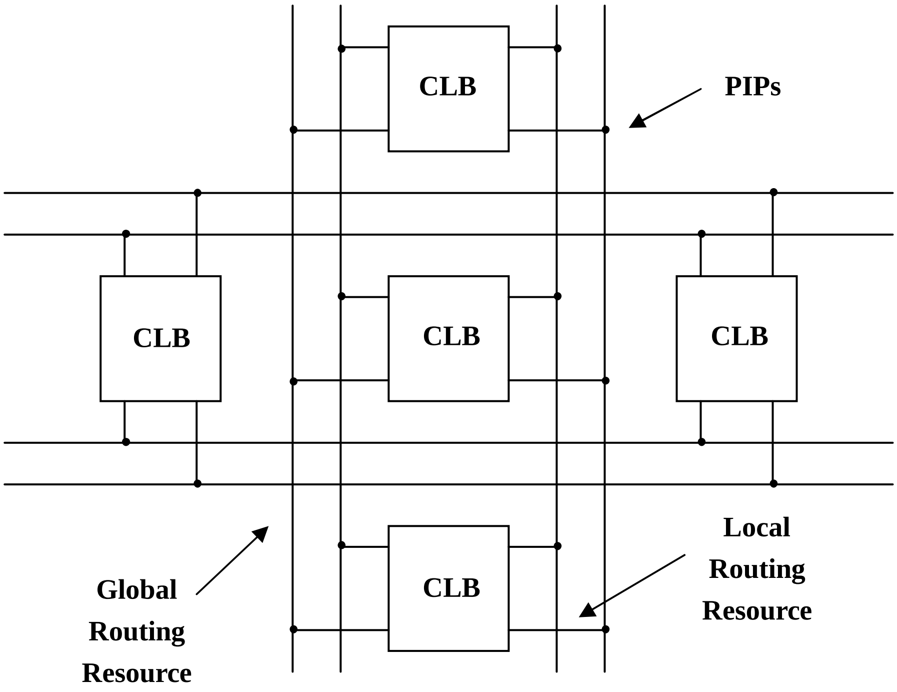
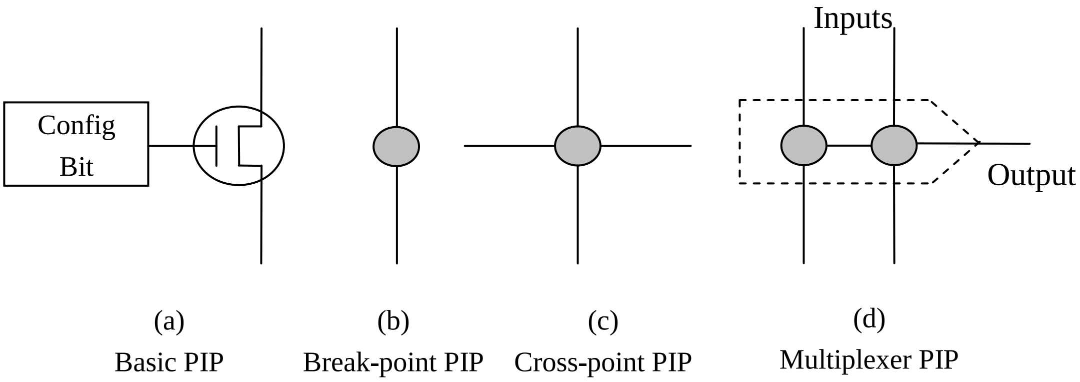


Figure 1.2 Programmable routing resources



(a) Basic PIP
(b) Break-point PIP
(c) Cross-point PIP
(d) Multiplexer PIP

Figure 1.3 Basic PIP structures

In Xilinx Virtex-4 FPGAs, all wire segments are connected to switch boxes, each of which is associated with a programmable logic element. There are many multiplexer PIPs internal to the switch box which connect the paths between logic elements and global routing resources [6].

1.3 Overview of Built-In Self-Test

There have been many approaches developed and applied to test FPGAs including external testing with automatic test equipment (ATE), design for test (DFT) techniques and Built-In Self-Test (BIST) [2]. BIST has been proved to be a more effective approach than other approaches. The main principle of BIST is to build test circuitry internal to a chip to test itself. BIST has the potential of being not only fast and efficient but also economical of cost [7].

A basic BIST architecture requires three parts: test pattern generator (TPG), circuit under test (CUT) and output response analyzer (ORA) [8], as illustrated in Fig. 1.4. The TPG generates test patterns which will test the CUT. The test patterns run through the CUT into the ORA which analyzes the output response signals from the CUT [8]. A pass/fall indication is given by the ORA which indicates if the CUT is fault-free or faulty.

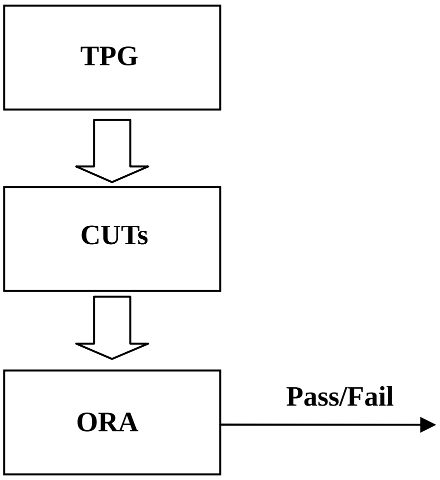


Figure 1.4 Basic BIST architecture

1.4 BIST of FPGAs

In BIST of FPGAs, TPGs and ORAs are configured internal to FPGAs using existing FPGA logic and routing resources. Because every BIST element is configured internal to the FPGAs and there are no additional resources needed for implementing BIST, the area overhead for BIST of FPGAs can be regarded as 0% [4].

The BIST process consists of a series of test configurations, where each configuration follows the following steps: (1) configuring the FPGA to create BIST structures for target resource, (2) applying BIST test pattern sequences, and (3) analyzing output responses [2][4][9]. The complete test of target CUTs generally requires a set of multiple test configurations, which is collectively referred to as a test session. The BIST of the FPGAs requires multiple test sessions to completely test all logic and routing resources [2].

1.5 Overview of Prior Work In Testing FPGA Routing Resources

Testing the FPGA routing resources is an important and potentially difficult task. Wire segments and PIPs take up the largest part of the total configuration bits and faults are most likely to appear in routing resources [2]. In addition to stuck-open and stuck-closed PIPs faults, the fault models of FPGA routing resources also include wires stuck-at 0 or 1, shorted wires, and open wires [4]. Test patterns need to be able to test all of these fault models. For example, in order to test shorted wires, the applied test patterns should not only check if every wire segment and PIP are able to transmit both 0 and 1, but should also ensure that both (0,1) and (1,0) values can be passed along every pair of adjacent wire segments [4][5].

Several approaches have been proposed to test routing resources in the FPGAs. Before BIST was applied for testing routing resources, testing was mainly dependent on externally applied vectors [10]-[13]. The first BIST-based approach for FPGA interconnects was developed in [4], and it provided complete test of interconnect faults in both global and local routing resources. The strategy was to construct a set of CLBs into TPGs and ORAs and two groups of wires under test (WUTs) which receive identical test pattern signals, as exemplified in Fig. 1.5. The outputs of WUTs were then compared by the ORA. The implementation in ORCA 2C series FPGAs was described and the experiment results were offered in [4]. Using a similar comparison-based BIST structure, a BIST-based diagnosis approach for programmable interconnect resources in FPGAs for either on-line or off-line was latter proposed [5]. This diagnostic approach is able to detect, locate and indentify single and multiple faults on the routing resources.

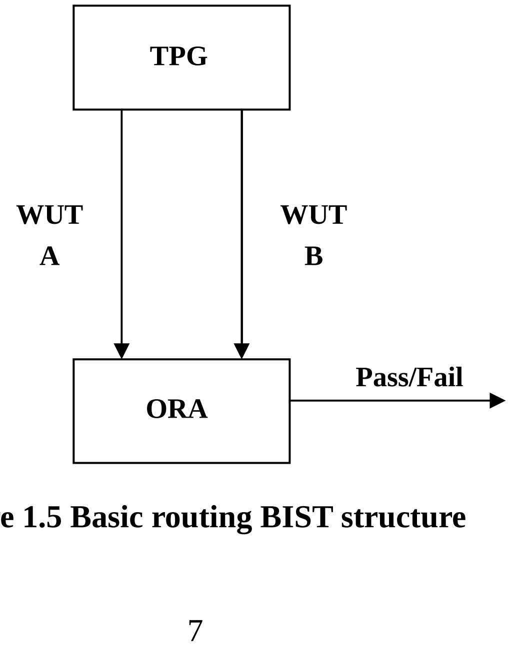


Figure 1.5 Basic routing BIST structure

The first parity-based routing BIST approach was proposed in [14]. The TPG consists of an *N*-bit binary counter and an *N*-bit parity generator. The ORA consists of an *N*-bit parity generator and check circuit. The parity-based approach overcomes the potential problem of the comparison-based approach in which equivalent faults in two sets of WUTs which feed into the same ORA would escape detection.

A modified parity-based approach was proposed in [15] to test the routing resources in Atmel AT94K FPGAs. The TPG is configured as a 2-bit up-counter with even parity or a 2-bit down-counter with odd parity. This BIST approach is able to detect stuck-at faults, bridging faults and opens in wire segments, along with stuck-open faults in PIPs [15].

The parity-based BIST approach was further developed in [16] where a cross-coupled parity approach was proposed for Xilinx Virtex-4 FPGAs. A 2-bit up-counter and a 2-bit down-counter are configured as TPGs. The next state of the most significant bit of each counter is used as a parity bit. The parity bits are cross-coupled to the ORAs receiving count values from the other counter. This cross-coupled parity approach ensures that any faults affecting one of the counters will be detected. Both cellular automata register (CAR) approach and cross-coupled parity approach were examined and compared in [16] and [17]. However, the cross-coupled parity approach was determined to be the best for Virtex-4 FPGAs [17].

1.6 Thesis Statement

This thesis presents the implementation of the cross-coupled parity BIST approach for testing global routing resources in Xilinx Virtex-4 FPGAs. The structures, actual implementations as well as specific BIST configurations are presented. The remainder of this thesis is organized as follows: Chapter 2 presents details about Virtex-4 FPGA architectures, programmable routing resources, and prior work in routing BIST. In Chapter 3, the cross-coupled parity approach is described in detail as well as its implementation in the BIST configurations developed to test specific routing resources in Virtex-4 FPGAs. Experimental results obtained from implementing routing BIST configurations and evaluations on routing BIST configurations are provided in Chapter 4. Finally, Chapter 5 presents a summary and conclusion of the thesis, along with suggestions for future work.

CHAPTER 2

BACKGROUND

This chapter provides background information on the Virtex-4 FPGA architecture, primarily focusing on the programmable routing resources. A brief overview of the architecture of some logic resources, including configurable logic blocks (CLBs), block random access memories (RAMs) and digital signal processors (DSPs) will also be given. More background knowledge about BIST is presented in this chapter, as well as an overview of prior work in routing BIST.

2.1 Virtex-4 FPGA Architectures

Xilinx Virtex-4 FPGAs comprise various configurable logic elements and embedded cores. The basic logic elements for Virtex-4 FPGAs are CLBs which provide combinatorial and synchronous logic capability. The embedded cores include: block RAMs, DSPs and Input/Output blocks (I/OBs) which provide the interface between internal configurable logic and external resources [18].

Virtex-4 FPGAs contain three families: LX, SX and FX. Among the three families, Virtex-4 LX FPGAs offer largest number of CLBs for logic applications, Virtex-4 SX FPGAs are optimized for digital signal processing applications and FX FPGAs include embedded PowerPC cores (PPCs) which can support embedded system functionality and embedded platform applications [18][19]. All elements are arranged in a column-based architecture and the numbers of all types of elements are summarized in Table 2.1.

Table 2.1 Virtex-4 family devices

V4 Devices	CLBs		Block RAMs		DSPs		I/OBs		PPCs	Total Rows
	Total	Cols	Total	Cols	Total	Cols	Total	Cols		
LX15	1536	24	48	3	32	1	320	2	N/A	64
LX25	2688	28	72	3	48	1	448	2	N/A	96
LX40	4608	36	96	3	64	1	640	2	N/A	128
LX60	6656	52	160	5	64	1	640	2	N/A	128
LX80	8960	56	200	5	80	1	768	2	N/A	160
LX100	12288	64	240	5	96	1	960	2	N/A	192
LX160	16896	88	288	6	96	1	960	2	N/A	192
LX200	22272	116	336	6	96	1	960	2	N/A	192
SX25	2560	40	128	8	128	4	320	2	N/A	64
SX35	3840	40	192	8	192	4	448	2	N/A	96
SX35	6144	48	320	10	512	8	640	2	N/A	128
FX12	1536	24	36	3	32	1	320	2	1	64
FX20	2304	36	68	5	32	1	320	2	1	64
FX40	4224	44	144	7	48	1	448	2	2	96
FX60	6656	52	232	8	128	2	576	2	2	128
FX100	10880	68	376	10	160	2	768	2	2	160
FX140	16128	84	552	12	192	2	896	2	2	192

CLBs are the basic logic components for implementing both combinatorial and sequential logic. Each Virtex-4 FPGA CLB consists of four slices, as illustrated in Fig. 2.1. Each slice contains two flip-flops, two 4-input look-up tables (LUTs), multiplexers and fast carry logic [18]. The Virtex-4 CLB has two separate carry chains, as shown in Fig. 2.1. The carry chains run upward and can be used to cascade the whole column of CLBs. A simplified structure of half of a slice is illustrated in Fig. 2.2.

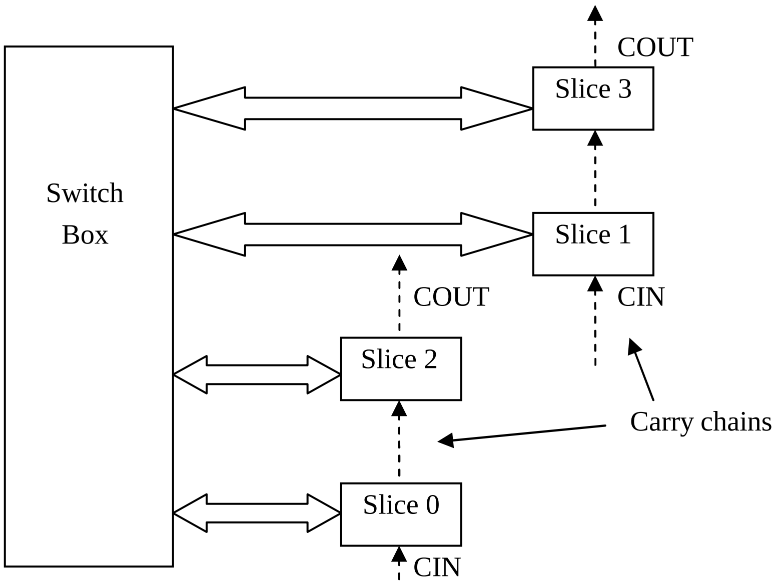


Figure 2.1 CLB structure

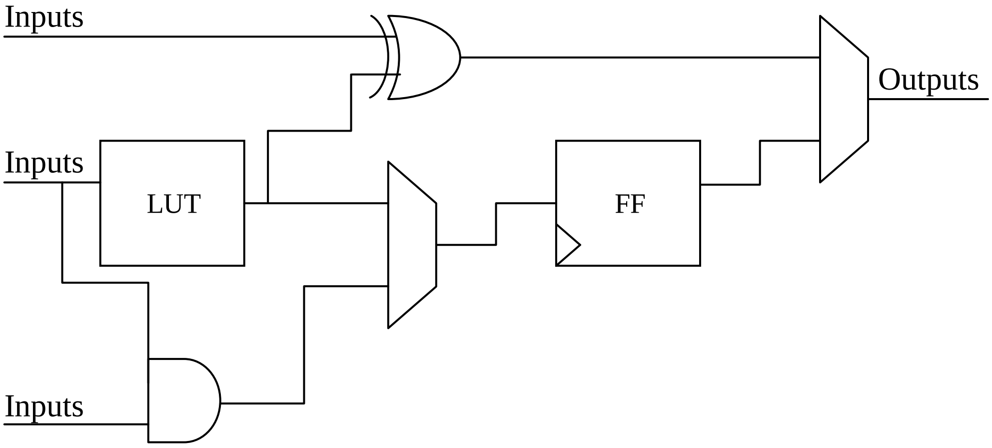


Figure 2.2 Simplified half slice structure

Virtex-4 FPGAs include a large number of 18Kb block RAMs. Each block RAM consists of two independent access ports where data can be written or read. Virtex-4 block RAMs can function as either dual-port or single-port RAMs [18]. Block RAMs are arranged in columns in Virtex-4 FPGAs and the total number of block RAMs is dependent on the family and size of different devices, as listed in Table 2.1.

DSPs are included for implementing digital signal processing algorithms [18]. There are two DSP slices per tile and all tiles are arranged into vertical columns. The number of DSPs as well as the number of DSP columns vary according to different families and sizes, as listed in Table 2.1.

Virtex-4 FPGA FX devices contain one or two PowerPC processor cores, depending on the size of the chip [18]. For Virtex-4 FX12 device, there is one PowerPC core which is located in the left part of the chip and takes up an area of 9 cols × 24 rows, including 7 columns of CLBs and 2 columns of block RAMs. PowerPC cores block the normal propagation of both vertical and horizontal routing resources.

2.2Virtex-4 Programmable Routing Resources

Programmable routing resources account for over 80% of the total configuration memory bits in Virtex-4 FPGAs [2]. Programmable routing resources, along with the switch box which resides with every logic resource, make up the programmable interconnect network. Routing resources consist of wire segments and programmable interconnect points (PIPs).

Programmable routing resources are classified as local routing resources or global routing resources. Local routing resources refer to the wire segments and PIPs which bring the signals associated with logic resources into and out of the switch box and those dedicated routing resources which connect adjacent logic resources. For example, the carry chains in Fig. 2.1 existing in every CLB are considered as local routing resources. Global routing resources interconnect switch boxes associated with the logic resources [2][4][5].

Global routing resources in Virtex-4 FPGAs basically include three types of wire segments: double lines, hex lines and long lines [17], as illustrated in Fig. 2.3. Double and hex lines are referred to by the number of switch boxes they span. Double lines span two switch boxes and hex lines span six switch boxes. Double and hex lines are directional, where a given wire segment propagates in one of four directions: north, south, east or west. For each double and hex wire segment, there are BEG, MID and END terminals [17]. The BEG terminal is located in the switch box where the wire segment begins, the MID terminal resides in the middle switch box along the wire segment, and END terminal refers to the terminal where the wire segment ends, as shown in Fig. 2.3. The BEG terminals only provide paths out of switch boxes while MID and END terminals only provide paths that feed into switch boxes. For each switch box, there are 10 double wire segments as well as 10 hex wire segments associated with each terminal type (BEG, MID and END) for each direction. In this way, there are 40 BEG terminals, 40 MID terminals and 40 END terminals for double lines and hex lines in each switchbox for a total of 240 double and hex line terminals [17]. Terminal names follow certain rules: the first letter (N/S/W/E) indicates the signal propagation direction of the wire segment; the following number (2/6) represents double line or hex line; next comes three characters referring to BEG/MID/END terminal, and the last number ranges from 0 through 9 with respect to the wire segment number. For example, the BEG terminal on a north double line wire segment 9 is named as N2BEG9; the END terminal along a west hex line wire segment 2 is referred to as W6END2.

Long lines have terminals every six switch boxes and pass by a total of 24 switch boxes [17], as shown in Fig. 2.3. Among the five terminals along its span, the two end terminals can source signals onto long lines since they provide paths into or out of switch boxes. The other three terminals can only be used as inputs since they only offer connections fed into switch boxes. Long lines are bi-directional since signals can be sourced from either end of the wire segment. There are only 10 long line terminals associated with each switch box. Horizontal long line terminals are named as LH0, LH6, LH12, LH18 and LH24, where LH0 and LH24 are the bi-directional terminals. Similarly, LV0, LV6, LV12, LV18 and LV24 are the vertical long line terminals, where LV0 and LV24 are the bi-directional terminals.

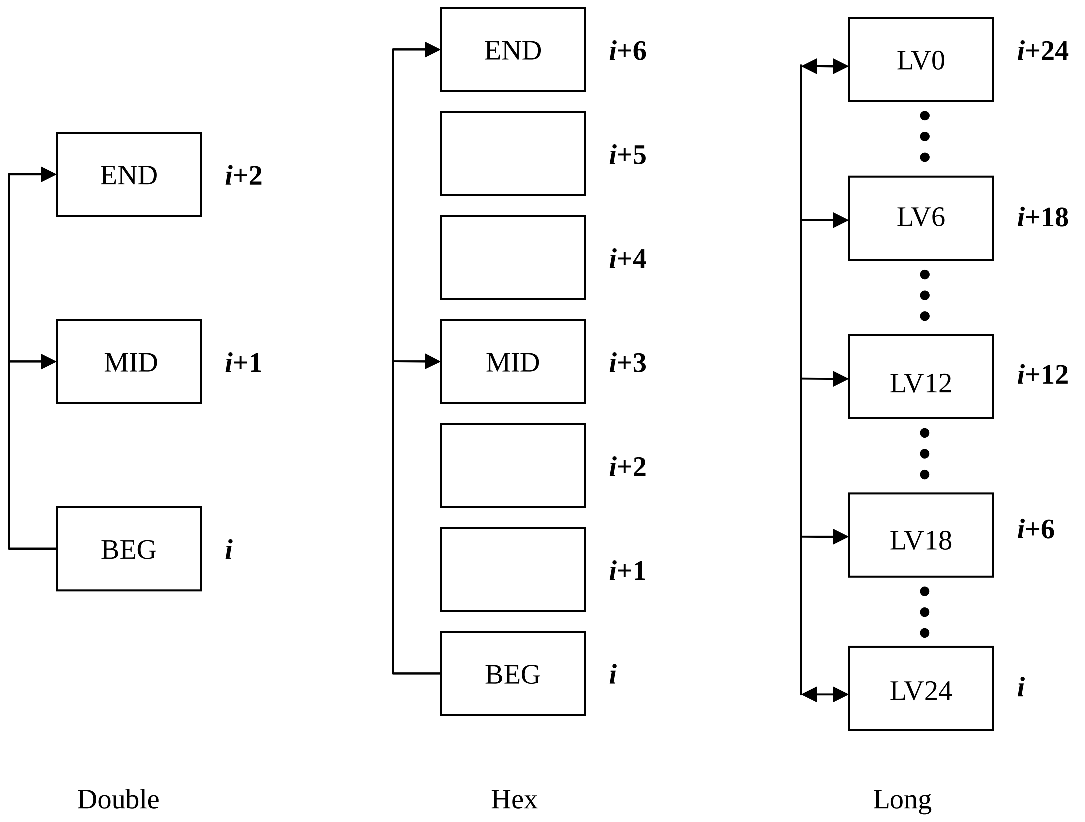


Figure 2.3 Virtex-4 global routing resources: double, hex and long lines

Each switch box consists of 3,312 PIPs. A PIP is a transistor switch which can be programmed to be activated or deactivated [20]. When a PIP is activated, the wire segments that are controlled by the PIP are connected. The switch box acts as a connection matrix between local routing resources and global routing resources. The switch box offers thousands of possible paths and each path consists of internal multiplexer PIPs. In addition to CLBs, there is a switch box residing in non-CLB components such as I/OBs, DSPs and block RAMs. For each CLB and non-CLB component, all terminals of global and local routing resources are tied to a switch box. Therefore, any signal sourced from or fed into these embedded components must be passed through the switch box.

A screen image from FPGA Editor, shown in Fig. 2.4, illustrates detailed CLB switch box connections in Virtex-4 FPGAs. The four right-hand rectangles represent the four slices and the left-hand rectangle represents the switch box. The horizontal wire segments between the switch box and four slices, as well as their associated wire segments internal to the switch box are referred to as local routing resources, while all other wire segments are global routing resources. The terminals associated with double, hex, and long lines are labeled in Fig. 2.4.

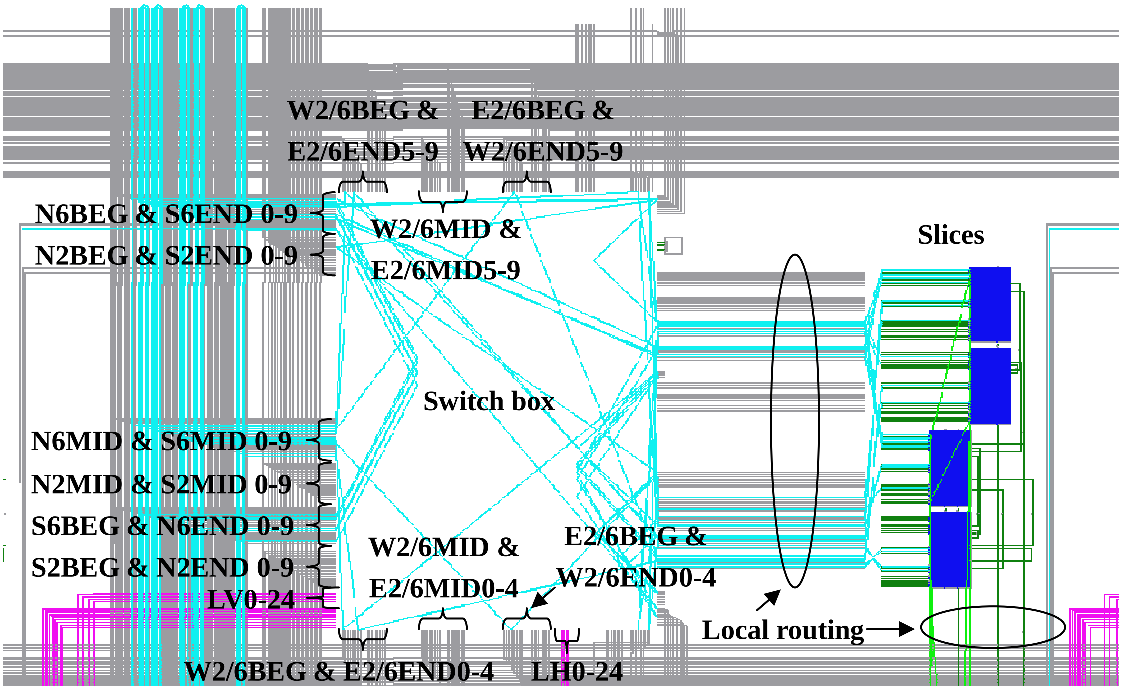


Figure 2.4 CLB switch box structure in FPGA Editor

When the routing resources arrive at the edges of the array, internal loopback connections can be activated to turn the routing onto the routing resources of the opposite direction. Take north double line wire segment 0 for instance, at the top edge of the array, a loopback connection is activated to turn the routing onto south double line wire segment 0. Loopback connections are available at the four edges of FPGA: top, bottom, left and right.

2.3 Built-In Self-Test

BIST is a design for test (DFT) technique that does not require external devices [8]. Test generation and test application are accomplished through built-in hardware features. BIST has the potential of being fast and efficient since all of its elements are built into the hardware [7].

In general, BIST includes a test pattern generator (TPG), circuit under test (CUT) and an output response analyzer (ORA). BIST circuitry can be classified into centralized and distributed architectures, as illustrated in Fig. 2.5 and Fig. 2.6 [8]. In centralized BIST architectures, all CUTs are driven by a shared TPG and CUT outputs are routed through a multiplexer into the ORA. This structure leads to more test sessions since one test session is needed for each CUT. In distributed BIST architectures, each CUT is driven by its own TPG and has its own ORA. In this way, all CUT outputs can be tested in the same test session [8].

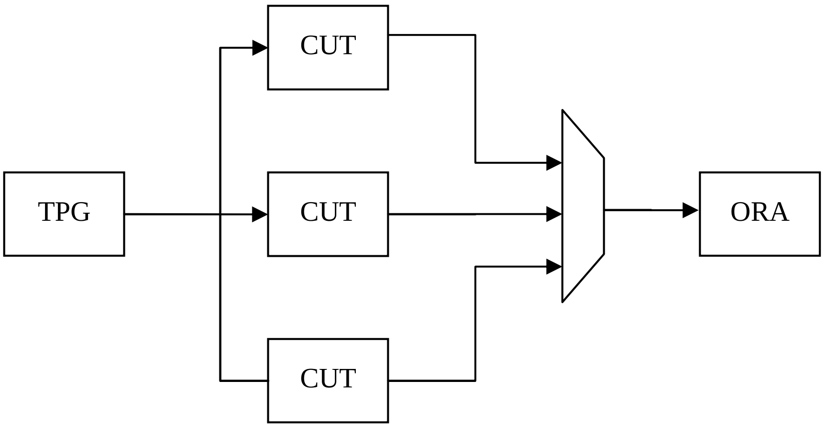


Figure 2.5 Centralized BIST architecture

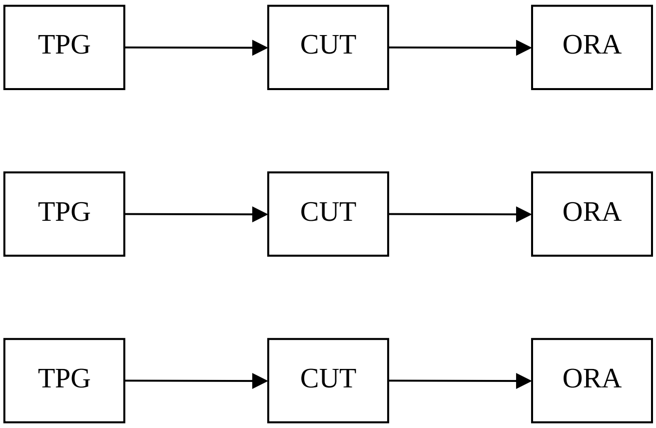


Figure 2.6 Distributed BIST architecture

In general, test patterns can be classified into the following types: deterministic test patterns, algorithmic test patterns, exhaustive test patterns, pseudo-exhaustive test patterns, random test patterns, pseudo-random test patterns, and weighted pseudo-random test patterns [7][8]. Test patterns generated by TPGs pass through CUTs and converge at ORAs, where CUT outputs will be examined. TPG implementations could be counters, FSMs, ROM with stored test patterns, cellular automata registers (CARs) and linear feedback shift registers (LFSRs) [7] [8]. Different types of TPGs may result in varying fault coverage in a certain amount of test time, depending on the CUT and the ORA used.

CUT output responses are examined in ORAs in several ways. Comparison-based ORAs use comparators to detect any mismatches between identical CUTs. The basic comparison-based ORA structure is illustrated in Fig. 2.7. Other ORA implementations also include concentrators, counting techniques, signature analysis, accumulators and parity check circuitry [8]. Concentrators are valuable in reducing the total number of outputs of the CUT that are monitored during test. Counting techniques can be implemented as counting the number of 1s or 0s or counting toggles in output responses. Signature analysis uses an LFSR to obtain a signature to determine if the CUT is fault or fault-free. Accumulators are often used for checksum circuits where the final sum provides the pass/fail indication [8].

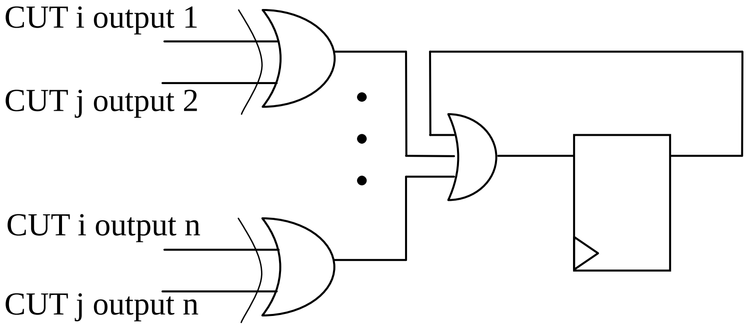


Figure 2.7 Comparison-based ORA structure

2.4 Routing BIST

It’s important to make sure that routing resources are fault-free because they are often assumed to be fault-free when testing other resources in FPGAs [2][4][5]. Only a small portion of the routing resources can be under test in a given test configuration. Due to the large number of wire segments and PIPs, the total number of test configurations required to completely test all routing resources is large [2]. Routing BIST generally consists of groups of logic resources which are configured as TPGs and ORAs as well as targeted routing resources under test, including wire segments and PIPs, which form the wires under test (WUTs).

2.4.1 Routing BIST Fault Models

Fault models are used to emulate faults or defects to evaluate the effectiveness of a set of test patterns or a TPG. They are required to accurately reflect any possible behavior of manufacturing defects or faults in a system during operation. In general, the fault models used in routing BIST include wires stuck-at 0 or 1, shorted wires and open wires, as well as stuck-open and stuck-closed PIPs [4]. PIP fault models are also included since any faults affecting PIPs can make wire segments connected or disconnected. Detection of stuck-at PIP faults also detects the stuck-at faults in configuration memory bits which control PIPs.

Open wires and shorted wires can result from over-etching or under-etching problems during the fabrication process [8]. Open wires prevent signal propagation beyond the open point. Faults on shorted wires are referred to as bridging faults. The probability of bridging faults between any two wire segments depends on actual physical layout information [8]. Bridging faults are more likely to occur between two wire segments which have larger area of adjacency and shorter distance between each other [8].

In order to detect shorted wires, test vectors (0,1) and (1,0) both should be applied at the inputs while monitoring both WUTs at the outputs. Open wire faults can be detected via applying both 0 and 1 to one end of a wire and monitoring the other end of the wire segment. This method also detects stuck-open faults for any activated PIPs along the WUTs. Stuck-closed PIP faults can be detected by applying opposite logic values at the ends of wire segments connected by a deactivated PIP while monitoring both sides of the PIP since a stuck-closed PIP will produce the same effect as a bridging fault [4].

2.4.2 Previous Work In Routing BIST

Several approaches have been proposed to test routing resources in FPGAs. Routing tests were mainly dependent on externally applied test patterns and monitored devices before BIST was introduced to FPGA routing resources [10]-[13], [21]. Externally applied test vectors are only possible at wafer and device level test. On the other hand, routing BIST approaches facilitate all levels of testing since the whole test circuitry is implemented using FPGA internal programmable resources.

2.4.2.1 Comparison-Based BIST Approach

The first BIST approach was developed for FPGA interconnect in [4], as shown in Fig. 2.8. This approach was proposed and implemented in ORCA 2C series FPGAs. The methodology is to configure two groups of CLBs into TPGs and ORAs, respectively, and configure two sets of wire segments and PIPs as WUTs. The two sets of WUTs are driven by a single counter-based TPG and the WUT output responses are compared by the ORA. As shown in Fig. 2.8, the solid lines represent WUTs, while the dashed lines represent some logic elements along the WUTs since WUTs may consist of multiple global and local wire segments. The black solid circles represent PIPs along the WUTs which are activated to connect wire segments.

Comparison-based ORAs, as shown in Fig. 2.7, are used in this strategy. An ORA compares the output responses from a pair of WUTs. Their output responses should be exactly the same if no faults exist since they are driven by a single TPG. Otherwise, any faults affecting any set of WUTs will be detected by the ORA. Fault escape will not happen unless identical faults occur to two sets of WUTs at the same time. The possibility of fault escape can be reduced by comparing one set of WUTs with two other sets of WUTs [4]. In this BIST structure, the CLBs which are configured as TPGs and ORAs are assumed to be fault-free. It was assumed that faults only happen along the WUTs.

The same off-line routing BIST approach was later used in [22] and first development for interconnect faults diagnosis was proposed in [23], by defining interconnect fault equivalence.

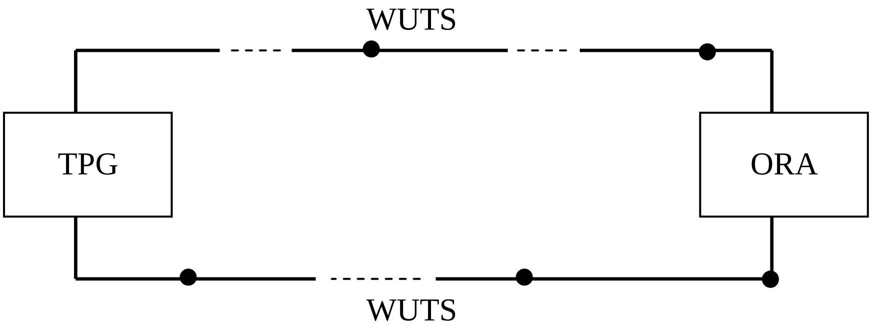


Figure 2.8 Basic BIST structure for FPGA interconnects

2.4.2.2 Roving STARs Approach

Roving STARs (Self-Testing Areas) was first proposed for on-line FPGA testing and diagnosis in [24]. It was later introduced to both on-line and off-line testing of FPGA routing resources in [5]. The roving STARs structure is illustrated in

Fig. 2.9. The grey area represents the logic resources which are configured into a horizontal STAR and a vertical STAR. The other square areas represent the logic resources for system applications. Roving STARs are implemented using spare logic resources. After the current STARs complete the tests, the STARs rove to a new location. By periodically roving STARs, every portion of FPGA resources will be eventually tested [5].

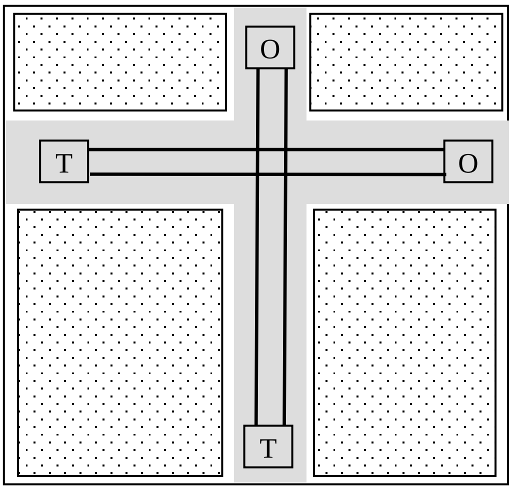


Figure 2.9 Roving STARs structure

Roving STARs was extended to off-line testing in [5]. Because there is no system operation executed during off-line testing, the entire FPGA can be configured into horizontal or vertical STARs to test all horizontal or vertical routing resources, as shown in Fig. 2.10. Multiple horizontal STARs run in parallel to reduce test time and test sessions [5]. After a horizontal STARs test session is completed, the entire FPGA will be configured into vertical STARs to test vertical routing resources.

Compared to the off-line BIST approach in [4], the parallel STARs approach improved diagnostic resolution [5]. The BIST approach proposed in [4] could only deduce information on possible types of interconnect faults but could not determine the location of the faults. On the other hand, the BIST approach introduced in [5] could provide higher diagnostic resolution because the possible fault location can be narrowed down to the area of a STAR [5].

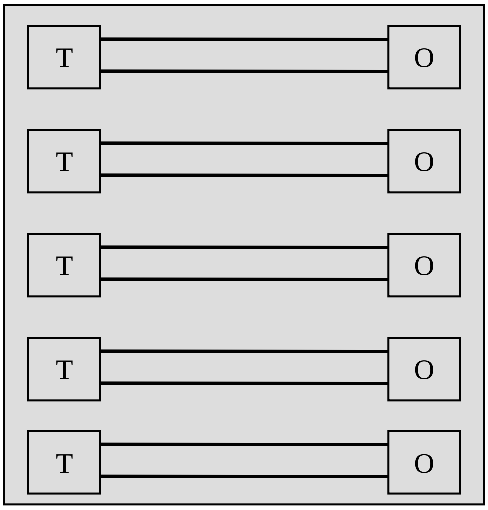


Figure 2.10 Off-line H-STARs testing

2.4.2.3 Parity-Based BIST Approach

A BIST scheme using error coding control was proposed for Xilinx 4000 series FPGAs in [14], as shown in Fig. 2.11. This method is the first parity-based routing BIST approach. The TPG implementation is an N-bit counter which applies an exhaustive set of 2^N test vectors. The TPG also provides the parity bit for the count values and the parity bit is sent to the ORA, labeled as WUTs_Parity. As shown in Fig. 2.11, the ORA includes a parity generator which is used to generate another parity bit from the binary values at the destination end of the WUTs, labeled as PG_Parity. Both parity bits feed into a comparator where any mismatch in the two parity bits indicates a fault in WUTs. The ORA then latches up any mismatches and produces a pass/fail signal after all test vectors are applied.

This parity-based BIST approach is capable of detecting all single and *m* multiple faults, where 1≤*m*≤*N* [14]. The parity-based BIST approach has some advantages over the comparison-based BIST approach. The major advantages are the support of an odd number of WUTs and the fact that the WUTs are easier to route [2]. However, there are some important issues associated with this approach. It is assumed that the parity bit is transmitted on fault-free routing resources [14]. In addition, any fault that inhibits the count sequence may escape detection if the parity remains correct.

This approach was later adopted and modified for Atmel AT94K FPGAs in [15], as illustrated in Fig. 2.12. The TPG is configured as a 2-bit up-counter with even parity or a 2-bit down-counter with odd parity. The parity bits are considered as WUTs. And the TPGs drive multiple ORAs to increase the number of WUTs. The test patterns produced by combining the count-up TPG and count-down TPG ensures the detection of stuck-at faults on wires, open wires and shorted wires since (0,1) and (1,0) can be provided on any possible pair of the WUTs, as seens in Table 2.2. This method is able to detect not only stuck-open PIPs but also stuck-closed PIPs since both types of TPGs are used on opposite sides of a deactivated PIP to apply opposite logic values.

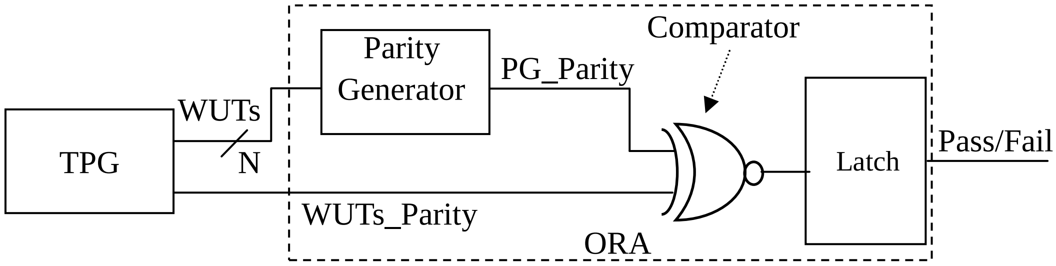


Figure 2.11 Parity-based BIST structure

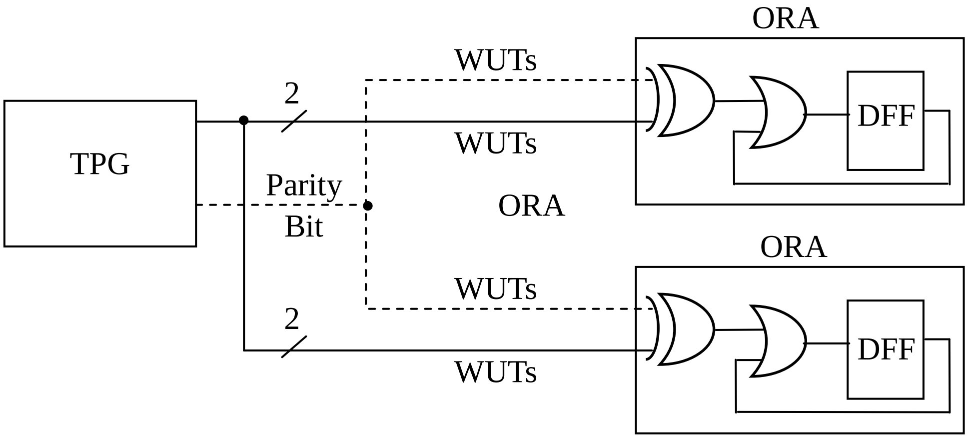


Figure 2.12 Modified parity-based BIST structure

2.4.2.4 Cross-Coupled Parity Appraoch

A cross-coupled parity approach was proposed for Virtex-4 FPGA routing resources in [16]. As illustrated in Fig. 2.13, a 2-bit up-counter initialized to all 0s and a 2-bit down-counter initialized to all 1s are used as the TPGs. The next state of the most significant bit of each counter is used as the parity bit and is cross-coupled to the ORAs with respect to the count values. Each ORA consists of a 3-bit exclusive-OR/exclusive-NOR gate for even/odd parity check, an OR gate, and a flip-flop.

In the original parity-based BIST approach, a fault affecting the counter may escape detection when the parity remains correct. This problem can be solved by reading the current state of the counter as well as the ORA results via partial configuration memory readback [16]. However, the cross-coupled parity approach makes it unnecessary to examine counter states. By cross-coupling the parity bits, any fault that affects one of the counters will be detected by ORAs. The test patterns are summarized in Table 2.2. As illustrated in this table, test vectors (0,1) and (1,0) can be formed from any pair of bits of the six bits of the test patterns which ensures the detection of shorted wires. Any one of test patterns produces both 0 and 1 logic values which ensures the detection of open wire faults and PIPs stuck-open faults.

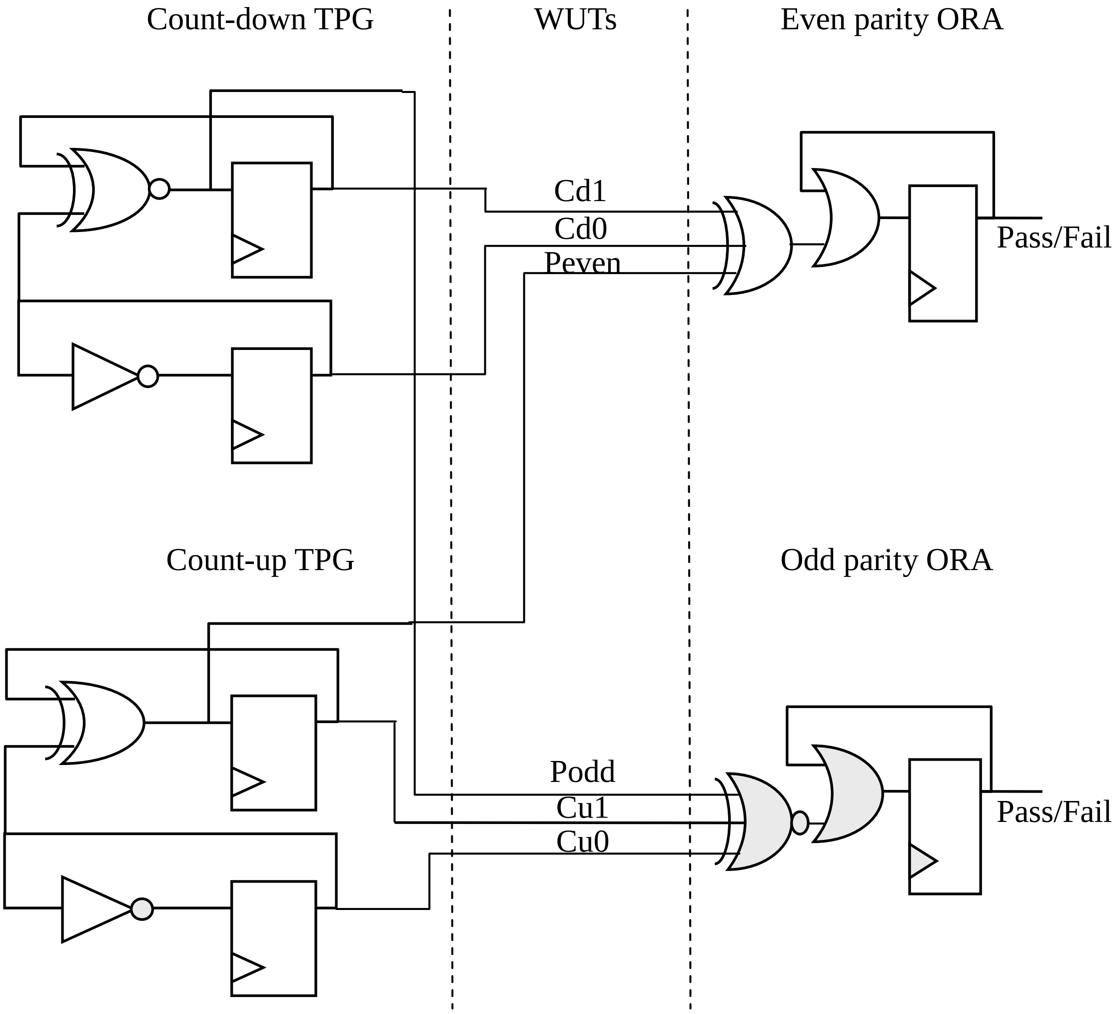


Figure 2.13 Cross-coupled parity approach structure

Table 2.2 Cross-coupled parity test patterns

Cu1	Cu0	P_{even}	Cd1	Cd0	P_{odd}
0	0	0	1	1	1
0	1	1	1	0	0
1	0	1	0	1	0
1	1	0	0	0	1

Analysis of the cross-coupled parity approach is given in [16] and [17]. Due to the large number of routing resources, routing BIST aims at maximizing the number of WUTs in any given test configuration [2]. The cross-coupled parity approach supports a larger number of WUTs since the TPG and ORA feedback connections can be considered as WUTs [16]. Comparison between cross-coupled parity approach and other commonly used BIST approaches such as LFSRs, celluar automata registers (CARs) as well as the previously discussed comparison and parity approaches was given in [16] and [17]. The cross-coupled parity approach was found to be the most effective in the terms of fault coverage, the number of WUTs and practical implementation. The cross-coupled parity approach was implemented for double lines in Virtex-4 FPGAs in [17].

2.5 Thesis Statement

The Cross-coupled parity BIST approach has been implemented for double lines in Virtex-4 FPGAs and proved to be the best method for testing Virtex-4 FPGAs routing resources [16][17]. This thesis aims at implementing the cross-coupled parity approach in the routing BIST for hex and long lines in Xilinx Virtex-4 FPGAs. BIST of hex lines and long lines in Virtex-4 FPGAs will be described in the subsequent chapters. The description of how the cross-coupled parity BIST approach is implemented for hex and long lines in Virtex-4 FPGAs will be presented in Chapter 3 and the implementation results will be given in Chapter 4.

CHAPTER 3

CROSS-COUPLED PARITY ROUTING BIST APPROACH IN VIRTEX-4

The cross-coupled parity routing BIST is implemented to test the hex lines and long lines in Virtex-4 LX and SX FPGAs. The detailed test architecture is presented in this chapter. The BIST of CLB column hex lines is presented in this order: north, south, west and east, followed by the BIST of non-CLB column hex lines in north and south directions. Long lines BIST is presented at the end of this chapter.

3.1 Hex Lines BIST

Virtex-4 FPGAs hex lines can be classified into CLB column hex lines and non-CLB column hex lines. Hex lines propagate in all four directions: north, south, east and west. For CLB column hex lines BIST in all four directions, TPGs and ORAs are configured in every CLB column. As for non-CLB column north and south hex lines BIST, for each non-CLB column, only its neighbor CLB column which is three columns to the left or right will be used for implementing TPGs and ORAs. Non-CLB column west and east hex lines do not have specific test configurations since they are tested during CLB column hex lines BIST as well as non-CLB column north and south hex lines BIST.

3.1.1 CLB Column North and South Hex Lines BIST

The structure of a CLB column north hex line is presented in Fig. 3.1. As shown in this figure, every hex line has one source CLB through BEG terminal and two destination CLBs via terminals MID and END. The BEG terminal offers connections coming out of the CLB while MID and END terminals provide paths fed into the CLB.

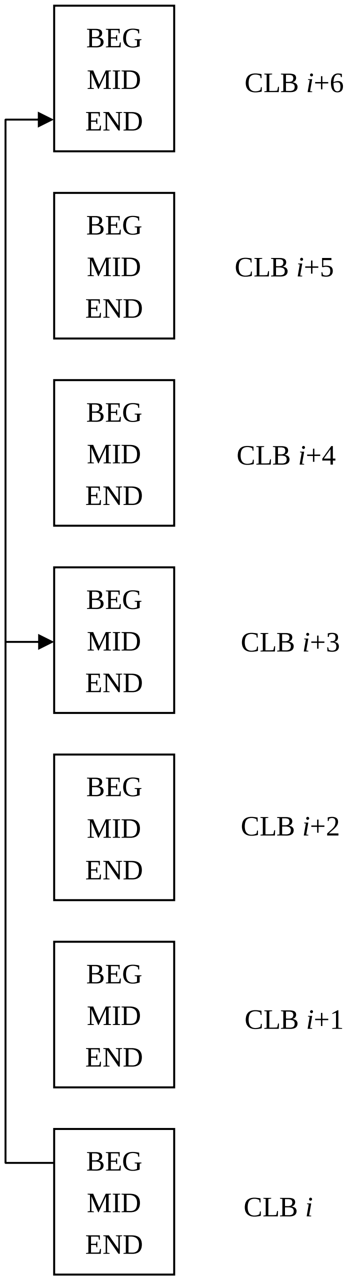


Figure 3.1 North CLB column hex line structure

The TPG and ORA structures for hex lines BIST are illustrated in Fig. 3.2. There are two types of TPGs, labeled as Te and To. Te is configured as a 2-bit up-counter initialized to all 0s, generating even parity, and To is configured as a 2-bit down-counter initialized to all 1s, generating odd parity. Two types of ORAs are used as well, labeled as Oe and Oo. The Oe ORA checks for even parity on the counter-bits from To (Cd1, Cd0) as well as the parity bit from Te (Peven). The Oo ORA checks for the even parity on the counter-bits from Te (Cu1, Cu0) along with the parity bit from To (Podd).

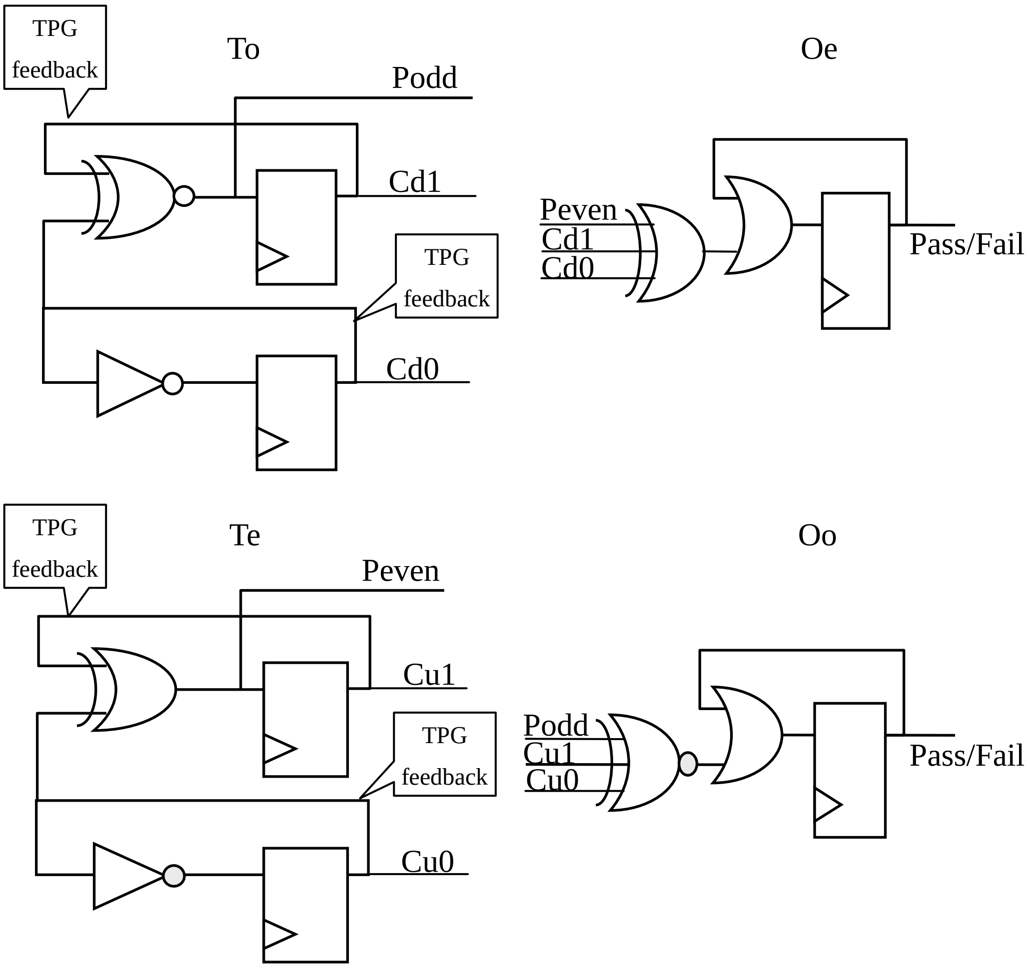


Figure 3.2 Cross-coupled parity TPG and ORA architectures

For any hex line which is sourced from any given CLB *i*, both types of TPGs are configured in CLB *i*, and both types of ORAs are configured in CLB *i*+3 and CLB *i*+6, with each TPG driving two ORAs, as illustrated in Fig. 3.3. In every CLB, slice 2 and slice 3 are configured as the TPGs; slice 0 and slice 1 are configured as the ORAs. The solid lines represent the test patterns sent from Te and the dashed lines represent the test patterns from To. The black lines represent the test patterns which are sourced from CLB *i* while the blue lines represent the test patterns which are sourced from CLB *i*+1, as illustrated in Fig. 3.3.

Four counter bits and two parity bits comprise six-bit test patterns. They are routed onto six north hex lines via BEG terminals in the switch box in CLB *i*, and fed into their respective ORAs via MID terminals in the switch box in CLB *i*+3 and END terminals in the switch box in CLB *i*+6. The parity bit, obtained from the next state of the most significant bits of each counter, is cross-coupled to the ORAs along with the counter bits generated from the other counter. The feedback paths in the TPGs as well as the six hex lines under test are considered as WUTs so that there are 18 WUTs in total in this cross-coupled parity BIST architecture [16].

One point to note is that the wire segments which are sourced from the same slice in adjacent CLBs are adjacent to each other. In order to avoid the same signal passing through adjacent wire segments, the locations of Te and To alternate in adjacent CLBs. Accordingly, the locations of Oe and Oo alternate in adjacent CLBs.

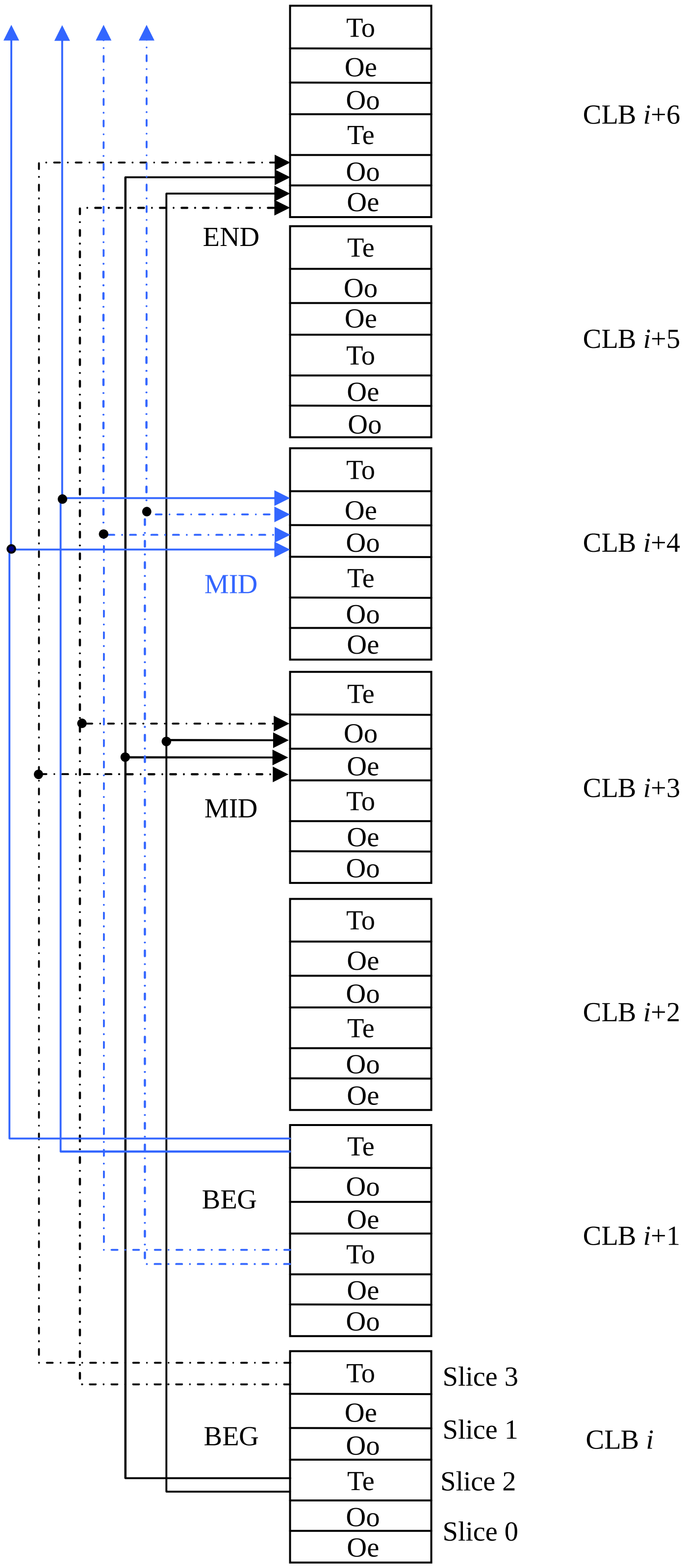


Figure 3.3 North CLB column hex line BIST architecture

Loopbacks are used at the top edges of the array to reroute the WUTs in the opposite direction using south hex lines. Fig. 3.4 presents the routing BIST architecture at the edges of the array. The solid lines represent north hex lines and the dashed lines represent south hex lines. As illustrated in Fig. 3.4, the WUTs are sourced from CLB *i*-3 and reach the top edge of the array. Loopbacks are used to route the WUTs onto the south hex lines and propagate in opposite direction. In CLB *i*-2, south hex line END-to-BEG connections are configured to make the WUTs propagate to the bottom edge of the array. These south hex line END-to-BEG connections are also used in the remaining CLBs along the way down to the bottom edge of the array. At the bottom edge of the array, the routing is turned back onto north hex lines via loopbacks. Finally, the WUTs feed into the ORAs via north hex line terminals, thereby completing the test architecture. Therefore, all BEG, MID and END terminals are tested along with their associated hex lines.

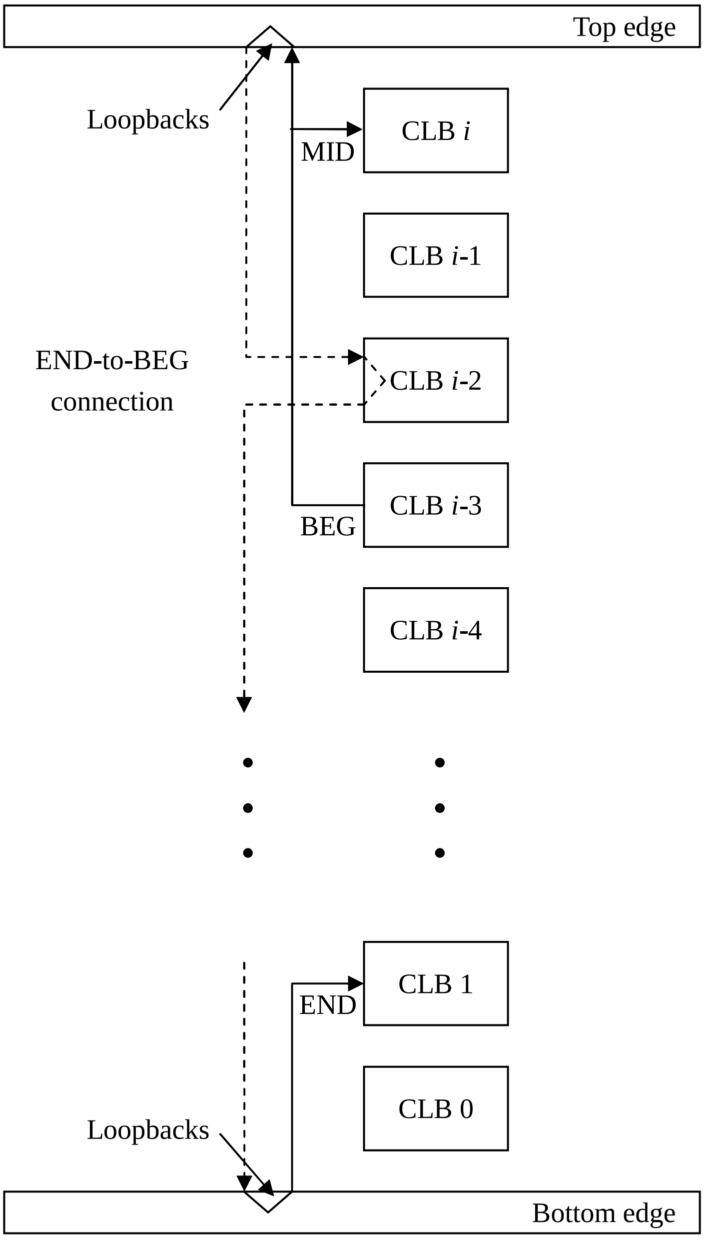


Figure 3.4 North CLB column hex lines BIST loopbacks implementation

There are only six WUTs, but ten hex lines need to be tested. At least two configurations are required to test the complete set of ten hex lines. However, the number of BIST configurations for north CLB column hex lines is four since MID terminals and END terminals are tested in separate configurations. This results from the limited connections in the switch box between hex line terminals and the LUTs which are due to the limited supply of available PIPs associated with hex lines. For example, north hex line wire segment 2 MID and END terminals (N6MID2 and N6END2) both have to pass through switch box connection OMUX4 into the LUTs. In order to avoid conflict, MID and END terminals need to be tested separately.

The four BIST configurations are summarized in Table 3.1. In the table, the first column shows which wire segment is under test. The BEG column shows which of the six test signals is connected to the BEG terminals, as well as the PIPs required to complete the connection, shown in the third column. The MID column represents which LUT input in slice 1 is connected to the MID terminal, along with the required PIPs shown in the next column. The END column and the following column follow the same principle. The four BIST configurations are assigned as follows: wire segments 3 through 8 along with their MID terminals are tested in the first configuration, while their END terminals are tested in the third configuration; wire segments 0 through 3 as well as 8 and 9 are tested in the second configuration along with their MID terminals, while their END terminals are tested in the fourth configuration. The MID and END terminals of wire segments 3 and 8 are connected to different LUT inputs in different configurations. The configuration number is indicated by labeling 1st, 2nd, 3rd, and 4th in the last four columns. For example, MID terminal of wire segment 3 is connected to LUT input F3-1 via connection N6MID3 -> E2BEG3 -> F3-1 in the first configuration. It is connected to LUT input G4-1 via connection N6MID3 -> E2BEG3 -> BYP_INT_BOUNCE4 -> BYP_BOUNCE4 -> G4-1 in the second configuration.

Table 3.1 CLB column north hex line BIST configurations

Wire	BEG	PIP	MID	PIP	END	PIP
0	Y2 (Peven)	OMUX 0	F3-1	W2BEG0 -> BYP_INT_BOUNCE2 -> BYP_BOUNCE2	G2-0	S2BEG0 -> BYP_INT_BOUNCE0 -> BYP_BOUNCE0
1	X2 (Cd0)		F4-1		G4-0	N2BEG1
2	YQ2 (Cu1)	OMUX 4	G3-1	W2BEG2	F2-0	E2BEG2 -> N2MID2 -> BYP_INT_BOUNCE6 -> BYP_BOUNCE6 BYP_INT_BOUNCE5 -> BYP_BOUNCE5
3	X3 (Cu0)		F3-1 (1st) G4-1 (2nd)	E2BEG3(1st) E2BEG3 -> BYP_INT_BOUNCE4 -> BYP_BOUNCE4 (2nd)	G3-0 (3rd) F3-0 (4th)	N2BEG3 (3rd and 4th)
4	X2 (Cd0)		G3-1	W2BEG4	F3-0	N2BEG4 -> BOUNCE1
5	Y2 (Peven)		G2-1		F2-0	N2BEG5
6	YQ2 (Cu1)	OMUX 9	F2-1	W2BEG6	G2-0	S2BEG6
7	YQ3 (Cd1)	OMUX 11	G1-1	W2BEG7 -> N2MID7	F4-0	E2BEG7 -> BYP_INT_BOUNCE1 -> BYP_BOUNCE1
8	Y3 (Podd)		F1-1 (1st) G1-1 (2nd)	E2BEG8 -> N2MID8 (1st and 2nd)	G1-0 (3rd) F1-0 (4th)	W2BEG8 (3rd and 4th)
9	YQ3 (Cd1)	OMUX 15	F1-1	W2BEG9 -> N2MID9	G1-0	E2BEG9

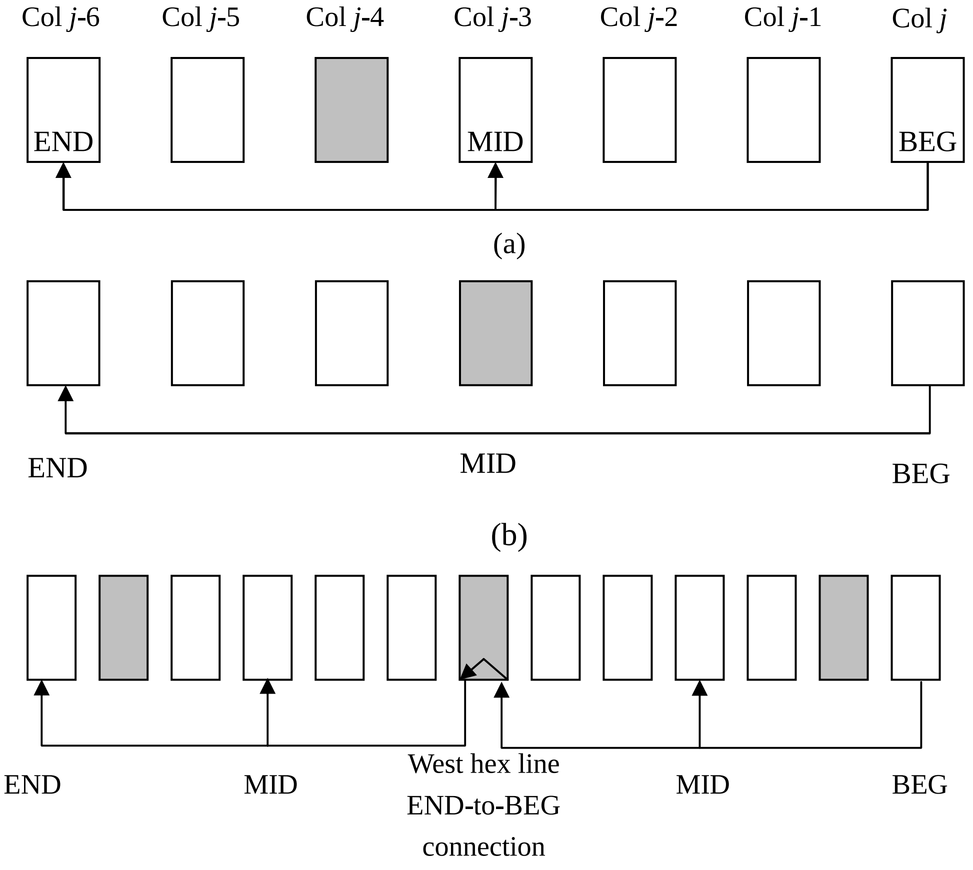
The BIST architecture for the south CLB column hex lines is similar to that for north CLB column hex lines except for the reversed signal flow direction. The test signals are routed onto south hex lines via BEG terminals from the TPGs in CLB *i* and via MID and END terminals into respective ORAs in CLB *i*-3 and CLB *i*-6.

Loopbacks are used at the bottom edge of the array to reroute the WUTs from south hex lines onto north hex lines and as well at the top edge of the array to turn the routing back to south hex lines. North hex line END-to-BEG connections are configured in every CLB. In a similar way, all BEG, MID and END terminals as well as associated south hex lines are completely tested. Four test configurations are required to test the ten south CLB column hex lines.

3.1.2 West and East Hex Lines BIST

In west and east CLB column hex lines BIST, the non-CLB columns must be taken into account, as shown in the west hex lines BIST example in Fig. 3.5. The non-CLB column is highlighted in gray. There are three strategies when a non-CLB column locates within the span of the hex lines under test. As illustrated in Fig. 3.5(a), in a normal situation, the non-CLB column does not influence the normal BIST structure. The test signals are routed onto the west hex lines via the BEG terminals and terminate at the CLBs, which are three columns and six columns to the west, respectively, via MID and END terminals into their respective ORAs. For the situation illustrated in Fig. 3.5 (b), the MID terminal connects to a non-CLB column; the test signals sourced through BEG terminals feed into the ORAs via END terminal along the west hex line under test. The MID terminal along the west hex line will be tested by the BIST configurations for north/south non-CLB column hex lines, which will be discussed later in this chapter. Fig. 3.5 (c) illustrates the situation when the END terminal connects to a non-CLB column. The END and BEG terminals in the switch box of the non-CLB column are connected to pass on the test pattern to the next ORAs.

Fig. 3.6 illustrates the usage of loopbacks in CLB column west hex lines BIST. At the western edge of the array, the test signals are routed onto east hex lines via loopbacks and propagate in the east direction. East hex line END-to-BEG connections are formed to pass the test signals to the eastern edge of the array. When reaching the eastern edge of the array, the routing is routed back onto west hex lines via loopbacks and complete the connections from the MID, END terminals to the ORAs, therefore completing the test architecture. The solid lines represent west hex lines and dashed lines represent east hex lines.



(c)

Figure 3.5 West CLB column hex lines BIST

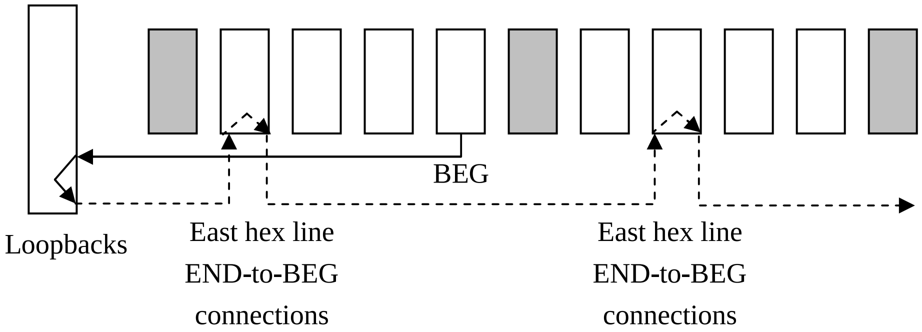


Figure 3.6 Loopbacks implementation in west hex lines BIST

The TPG and ORA structures used in west and east hex lines BIST are the same as illustrated in Fig. 3.2. Slice 2 and slice 3 are configured as the ORAs while slice 0 and slice 1 are configured as the TPGs. The arrangements of Te and To, as well as Oe and Oo, alternate in adjacent columns in order to avoid the same signal passing through adjacent lines. Four configurations are needed to completely test of all ten west hex lines. The four configurations for CLB column west hex lines are summarized in Table 3.2. Wire segments 2 through 6 plus 9 MID terminals and END terminals, which are highlighted in gray, are tested during the first and the third test configurations, respectively, while wire segments 0 through 2 plus 7 through 9 MID terminals and END terminals are tested during the second and the fourth test configurations, respectively. As for wire segments 2 and 9, since their MID terminals are tested in the first and the second configurations and their END terminals are tested in the third and the fourth configurations, the LUT inputs and the PIPs used in different configurations are distinguished by labeling 1st, 2nd, 3rd, and 4th.

Table 3.2 CLB column west hex line BIST configurations

Wire	BEG	PIP	MID	PIP	END	PIP
0	Y0 (Peven)		G1-2	S2BEG0	F1-3	N2BEG1
1	Y1 (Podd)	OMUX 2	F1-2	N2BEG1	G1-3	S2BEG2
2	YQ0 (Cu1)	OMUX 4	F1-2 (1st) G3-2 (2nd)	S2BEG2(1st) S2BEG2 -> BYP_INT_B0 -> BYP_BOUNCE0 (2nd)	G2-3 (3rd) F2-3 (4th)	N2BEG3 (3rd and 4th)
3	Y0 (Peven)		F2-2	N2BEG3	G1-3	S2BEG4 -> BOUNCE0
4	Y1 (Podd)		G1-2	S2BEG4 -> BOUNCE0	F1-3	E2BEG4 -> BOUNCE2 -> BYP_INT_B1-> BYP_BOUNCE1
5	X1 (Cu0)		F3-2	N2BEG5	G3-3	S2BEG6
6	X0 (Cd0)		G2-2	S2BEG6 -> BYP_INT_B1 -> BYP_BOUNCE1 -> BYP_INT_B2 -> BYP_BOUNCE2	F3-3	W2BEG6
7	X1		G4-2	N2BEG7	F4-3	E2BEG7 -> BYP_INT_B3 -> BYP_BOUNCE3
8	X0				G3-3	W2BEG8 -> BYP_INT_B5 -> BYP_BOUNCE5
9	YQ1 (Cd1)	OMUX 15	G3-2 (1st) F4-2 (2nd)	N2BEG9 -> BYP_INT_B5 -> BYP_BOUNCE5(1st) N2BEG5 (2nd)	F4-3 (3rd) G4-3 (4th)	E2BEG9 (3rd and 4th)

The east hex lines are tested similar to the west hex lines, where non-CLB columns also get involved. At the eastern edge of the array, the test signals are routed onto west hex lines using loopback connections, shown in Fig. 3.7. The test signals are passed through west hex lines to the western edge of the array by connecting the END and BEG terminals. The routing loops back to east hex lines and feed into the ORAs via MID and END terminals. The solid lines represent east hex lines and the dashed lines represent west hex lines in Fig. 3.7. The east hex lines BIST also require four test configurations to test all ten lines.

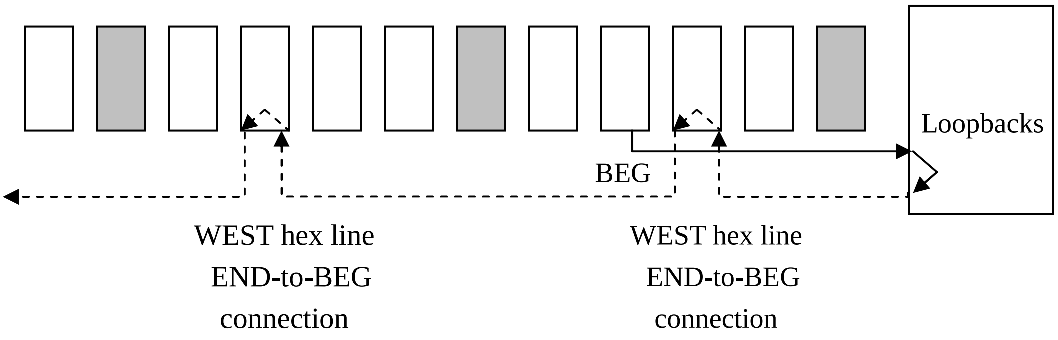


Figure 3.7 Loopbacks implementation in east hex lines BIST

3.1.3 Non-CLB Column Hex Lines BIST

The primary difference between CLB column north (south) hex lines BIST and non-CLB column north (south) hex lines BIST is that the nearby CLB columns are used to implement the TPGs and ORAs when testing non-CLB column hex lines. However, the number of BIST configurations for non-CLB column hex lines varies due to the different architectures of different devices. BIST for LX devices is presented first, followed by SX devices. MID and END terminals are tested separately due to the limited connections internal to the switch box, as discussed in CLB column hex lines BIST. However, additional limitations occur since double and hex lines in west and east directions are all used to complete the test for non-CLB column north and south hex lines. For example, some wire segment terminals (like W2BEG9 or

E2BEG1) which were originally used to connect WUTs with TPGs/ORAs in switch boxes can not be used in order to avoid conflicts.

Fig. 3.8 illustrates the BIST architecture for non-CLB column north hex line MID terminals. The columns highlighted in gray represent non-CLB columns while the other columns in white represent CLB columns used for implementing TPGs and ORAs. As illustrated in Fig. 3.8, for a given non-CLB column *j*, CLB column *j*+3 is configured into TPGs and ORAs. The TPGs are located in CLB column *j*+3 (according to the Virtex-4 LX FPGA architecture, column *j*+3 is a CLB column if column *j* is a non-CLB column). The test signals are routed onto west hex lines via BEG terminals and turned onto non-CLB north hex lines via W6MID-to-N6BEG connections. When reaching the MID terminals along north hex lines, the test signals are rerouted onto east hex lines via N6MID-to-E6BEG connections and feed into their ORAs via east hex line MID terminals. At the eastern edge of the chip, CLB column *n*-3 is used for the non-CLB column *n*, which is the rightmost column on the chip. The test signals are routed onto east hex lines to be connected to the WUTs, then turned onto west hex lines and fed into the ORAs. The solid lines represent the non-CLB column north hex lines and the dashed lines represent the east and west hex lines.

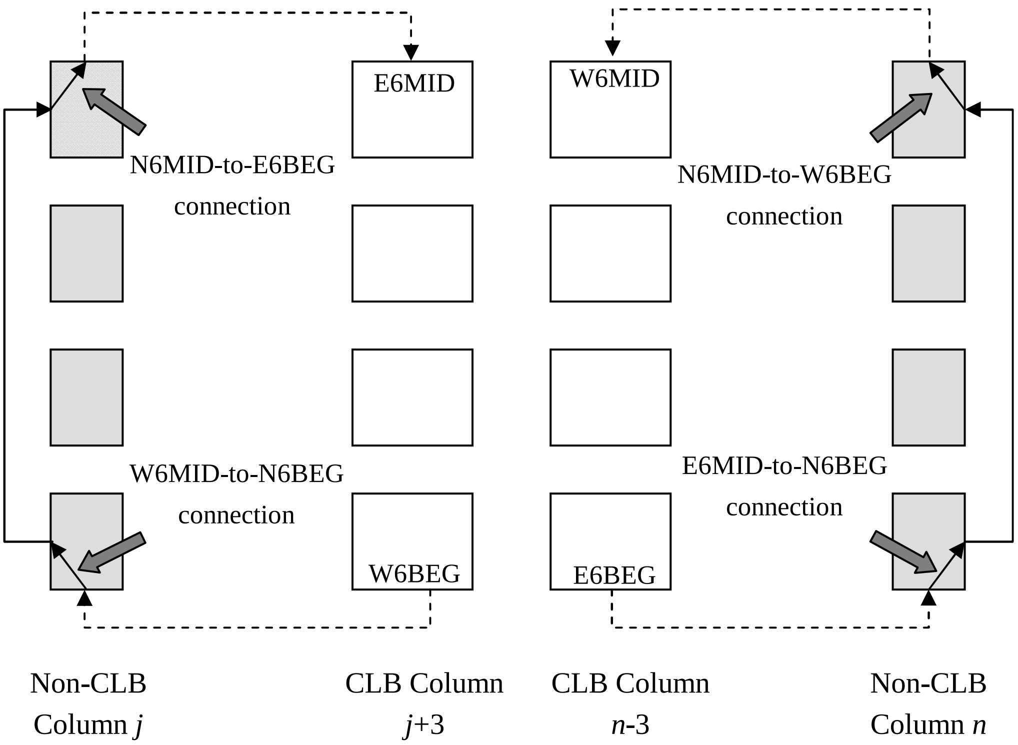


Figure 3.8 BIST architecture for non-CLB column north hex line MID terminals

Fig. 3.9 illustrates the BIST architecture for non-CLB column north hex line END terminals. It is similar to that for MID terminals except that N6END-to-W6BEG and N6END-to-E6BEG connections are used to connect the WUTs with TPGs and ORAs, respectively. One point to note is that some north hex line END terminals do not connect to either west or east hex lines. For these wire segments, their END terminals are routed onto east and west double lines, via N6END-to-E2BEG and N6END-to-W2BEG connections, as illustrated in Fig. 3.10. The dashed lines in Fig. 3.9 and Fig. 3.10 represent west and east hex and double lines used for connecting the WUTs with TPGs/ORAs. The solid lines represent non-CLB column hex lines under test.

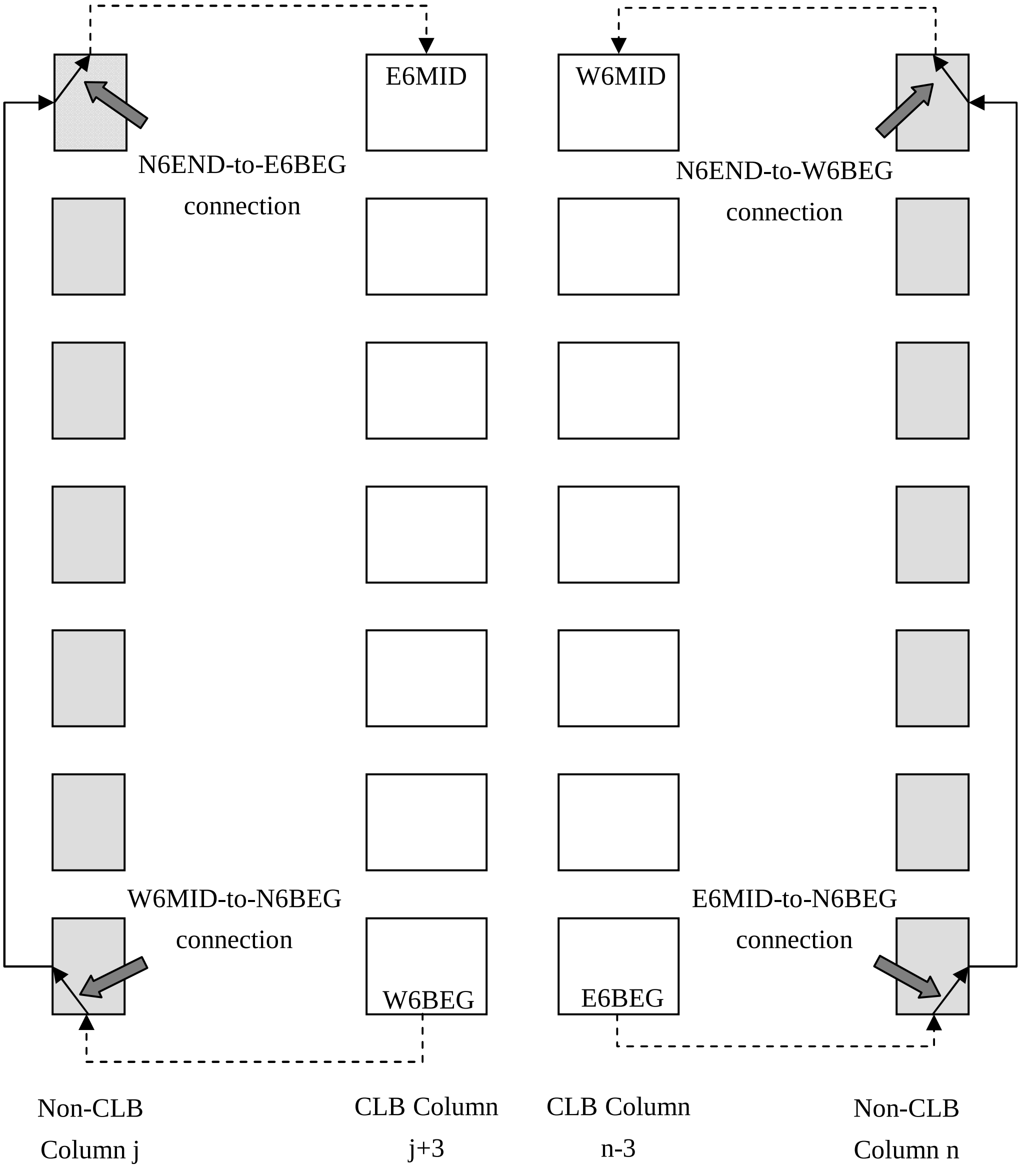


Figure 3.9 BIST architecture for non-CLB column north hex line END terminals

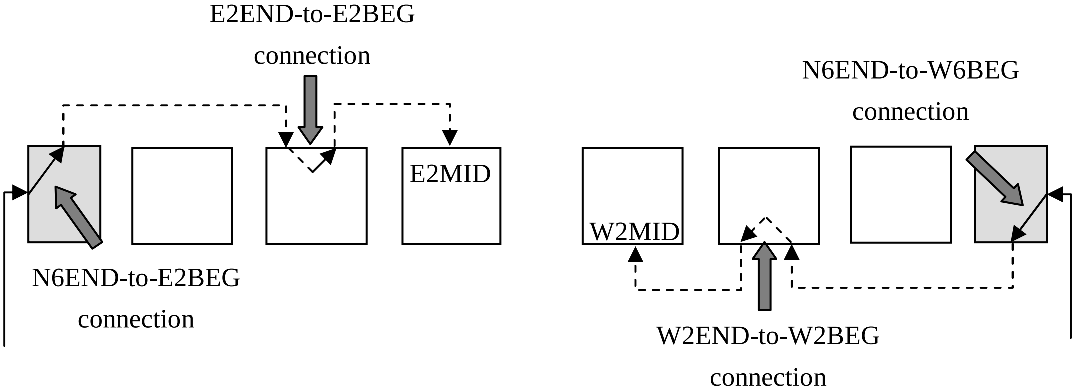


Figure 3.10 BIST architecture for non-CLB column north hex END terminals(cont.)

Fig. 3.11 illustrates the loopbacks at the edges of the array in the BIST for non-CLB column MID terminals. The loopback structure is similar to that of CLB column north hex lines. The loopbacks at the top edge of the array make the WUTs, shown as the solid lines, route onto non-CLB column south hex lines, shown as the dashed lines. The END and BEG terminals are connected to pass the test patterns to the bottom edge of the array through south hex lines. The test patterns are then routed back onto non-CLB column north hex lines via the loopbacks at the bottom edge of the array. The similar structure is applied to END terminals based on the BIST structure shown in Fig. 3.9.

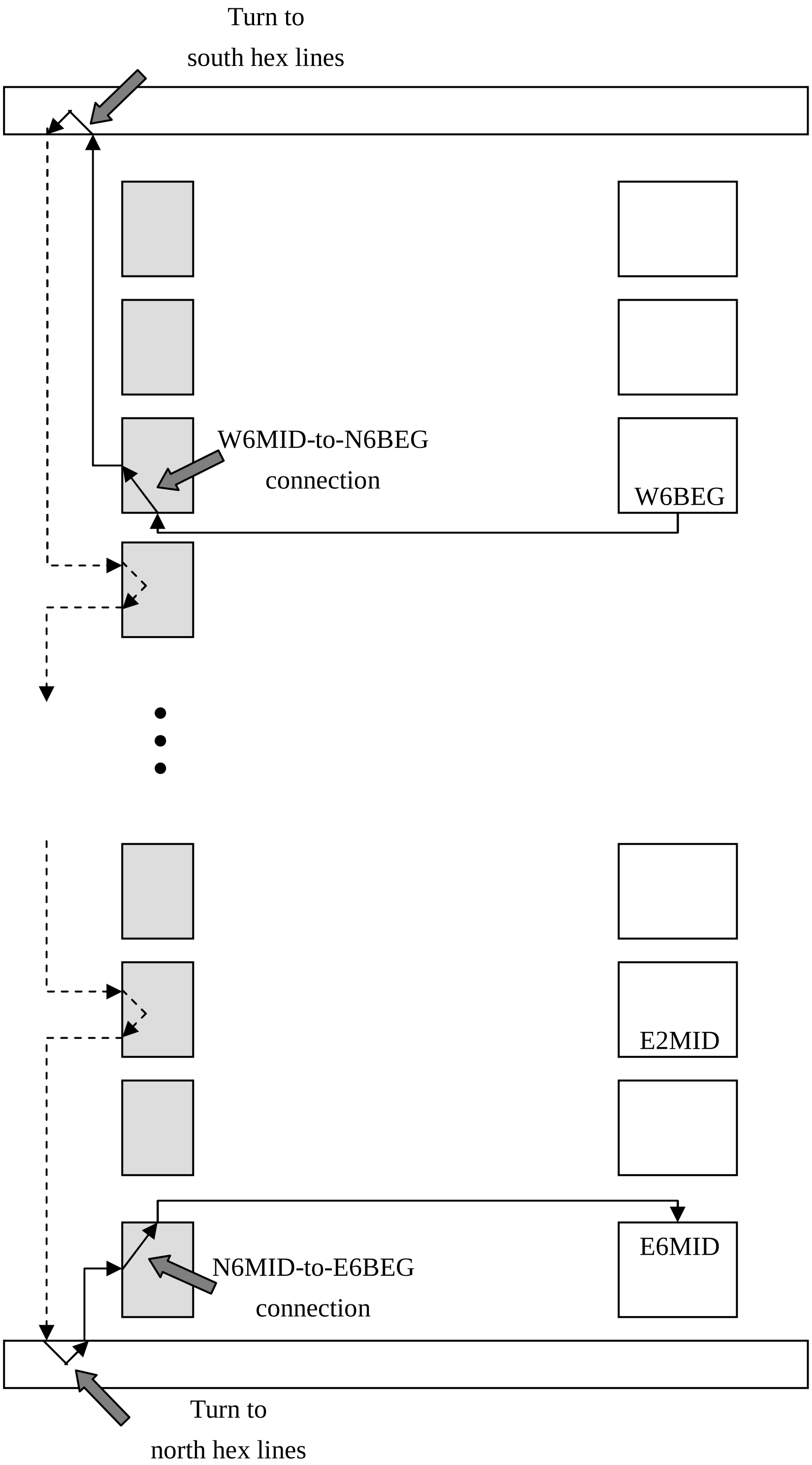
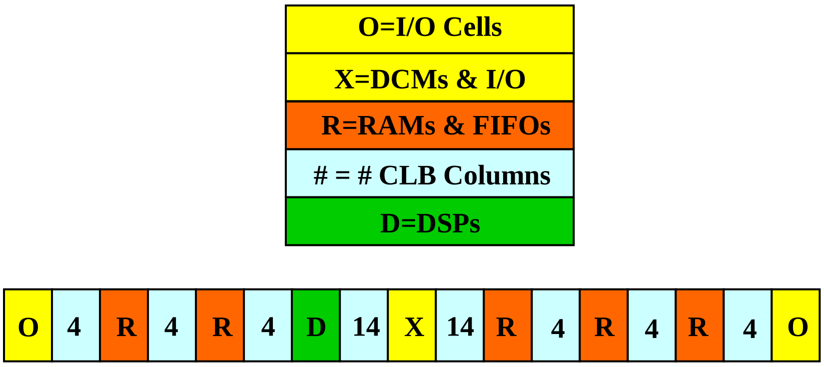


Figure 3.11 Loopbacks in the BIST for non-CLB north hex line MID terminals

The BIST structure illustrated above only works for Virtex-4 LX devices since the number of CLB columns between two closest non-CLB columns is equal to or larger than four, as illustrated in Fig. 3.12 (a). For SX devices, BIST architecture and the total number of configurations change since non-CLB columns appear every third column or every fifth column, as shown in Fig. 3.12 (b). This structure prevents the BIST architecture on a portion of the non-CLB columns and requires alterations which lead to a large increase in the number of BIST configurations.

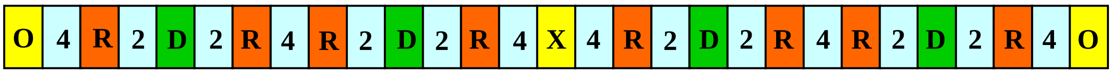
In SX devices, for any non-CLB column *j*, if column *j*+3 or *j*-3 is a CLB column, BIST architectures can be implemented as illustrated in Fig. 3.8 and Fig. 3.9. Only the four DSP columns can not follow the BIST architecture and therefore require additional configurations. The BIST architecture for those DSP columns is illustrated in Fig. 3.13. The east and west hex lines END terminals are used to connect the ORAs and TPGs instead of the MID terminals. The number of BIST configurations for north hex non-CLB lines increases to 8 in SX25/35 devices and increases to 20 in SX55 devices.



O=I/O Cells
X=DCMs & I/O
R=RAMs & FIFOs
 # = # CLB Columns
D=DSPs

O	**4**	**R**	**4**	**R**	**4**	**D**	**14**	**X**	**14**	**R**	**4**	**R**	**4**	**R**	**4**	**O**

(a) LX60



O	**4**	**R**	**2**	**D**	**2**	**R**	**4**	**R**	**2**	**D**	**2**	**R**	**4**	**X**	**4**	**R**	**2**	**D**	**2**	**R**	**4**	**R**	**2**	**D**	**2**	**R**	**4**	**O**

(b) SX55

Figure 3.12 Virtex-4 LX and SX column structures

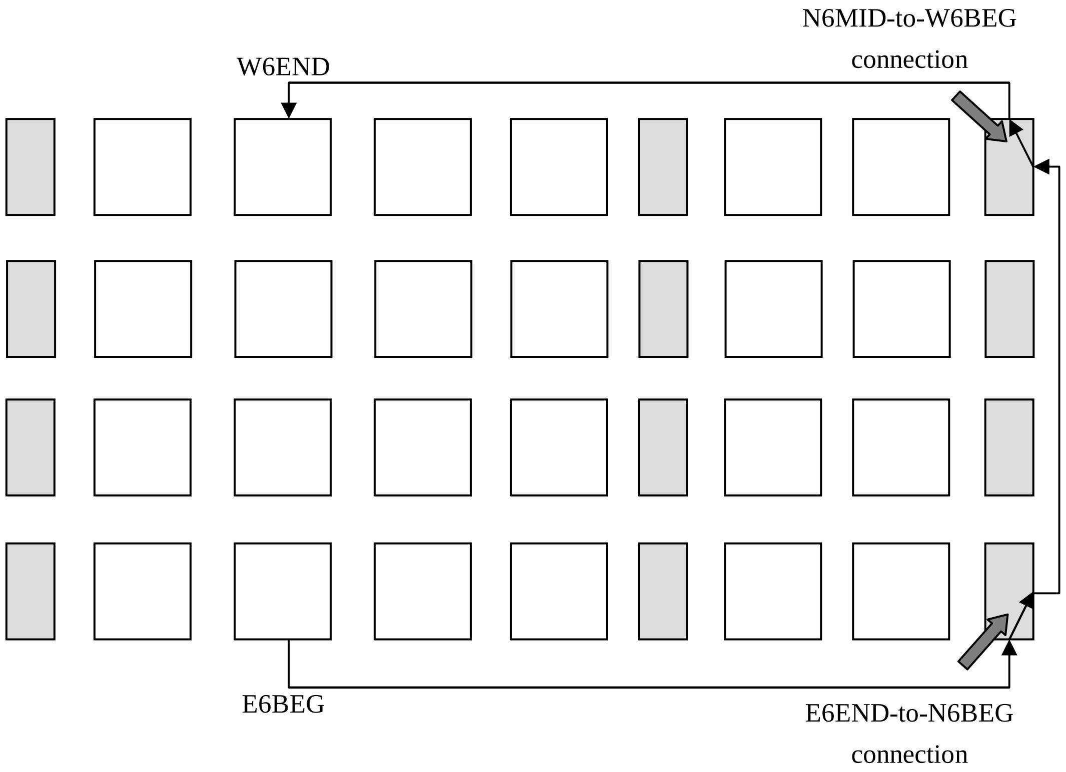
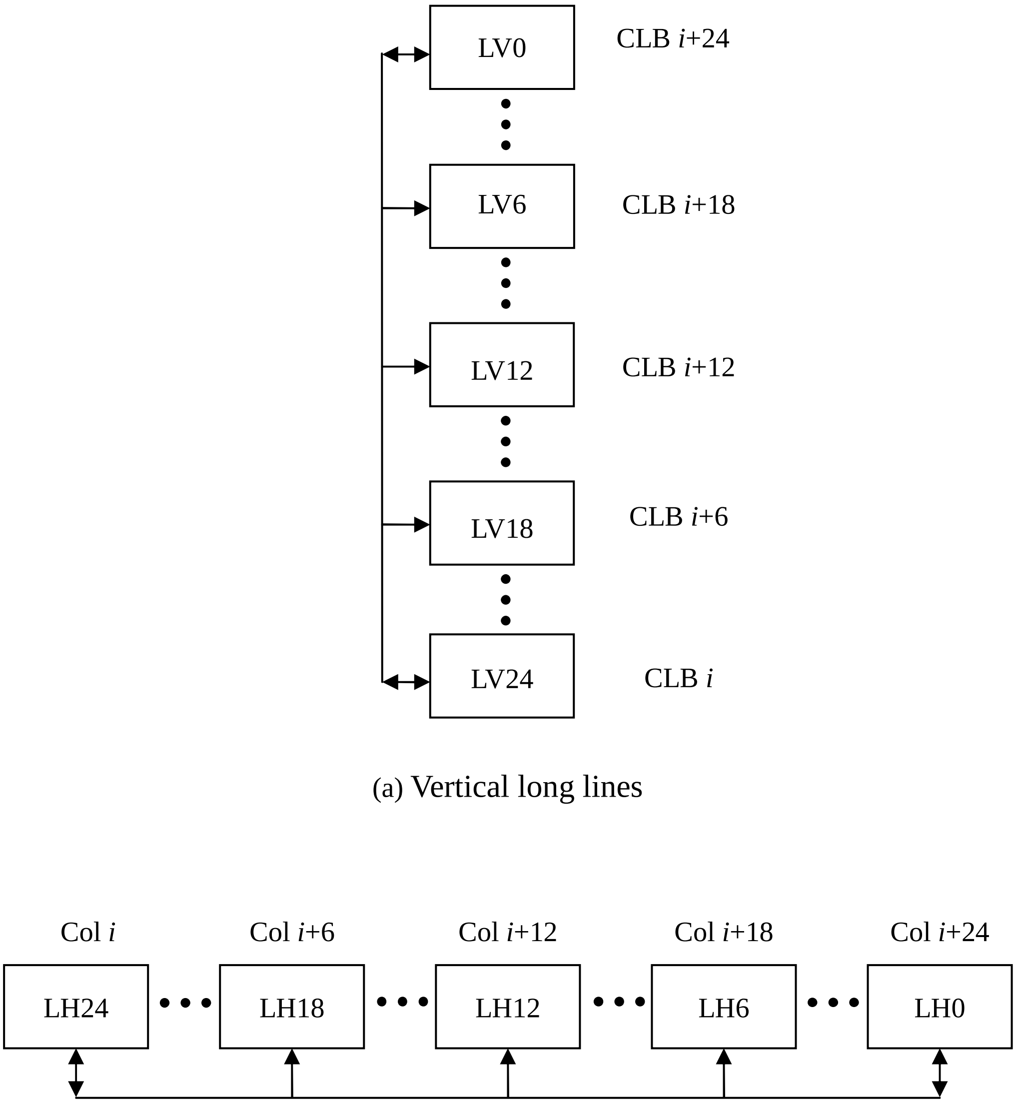


Figure 3.13 The BIST architecture of additional configurations for SX devices

The BIST architecture for non-CLB column south hex lines is the same except for the reversed signal flow direction. In addition, there is no need to set up additional test configurations for non-CLB column east and west hex lines since their BEG and END terminals are tested during the BIST for east and west CLB column hex lines and their MID terminals are tested during the BIST for non-CLB column north and south hex lines.

3.2 Long Lines BIST

Long lines span 24 CLBs with five terminals along its span. Long lines propagate horizontally and vertically. There are five horizontal long lines and five vertical long lines associated with each switch box; thereby there are ten long line terminals connected to each switch box. Among the five terminals along the long line, the three terminals in the middle can only be used as inputs into the switch boxes while the two end terminals can be used as both inputs and outputs. The horizontal and vertical long lines structures are shown in Fig. 3.14. Long lines are regarded as bi-directional since the two end terminals serve as both input and output terminals.



(a) Vertical long lines

(b) Horizontal long lines

Figure 3.14 Long line structures

3.2.1 CLB Column Vertical Long Lines BIST

Since there are five long lines tied to each switch box, there are five long line terminals, LV0, LV6, LV12, LV18 and LV24, associated with each switch box. As for each switch box, the TPG signal is sourced from terminal LV24 onto the long line for the north direction, and from terminal LV0 onto the long line for the south direction. For the north direction, the other four long lines along with the terminals, LV0, LV6, LV12, and LV18, are observed by ORAs. For the south direction, the other four long lines along with the terminals, LV6, LV12, LV18 and LV24, are observed by ORAs. Accordingly, the TPG structure is changed into a 3-bit up-counter or down-counter with even parity and the ORA structure is altered to examine four inputs and latch up any parity errors, as illustrated in Fig. 3.15. Each TPG requires two slices and each ORA requires one full slice. The test pattern sequences are summarized in Table 3.3. The parity bits from the up-counter TPGs and down-counter TPGs are not cross-coupled into the ORAs since both TPGs produce even parity bits. However, since each TPG only sources one of the four test patterns, any fault affecting a given TPG will produce failures in all four ORAs the TPG drives. By examining the four ORAs, the faulty TPG will be detected.

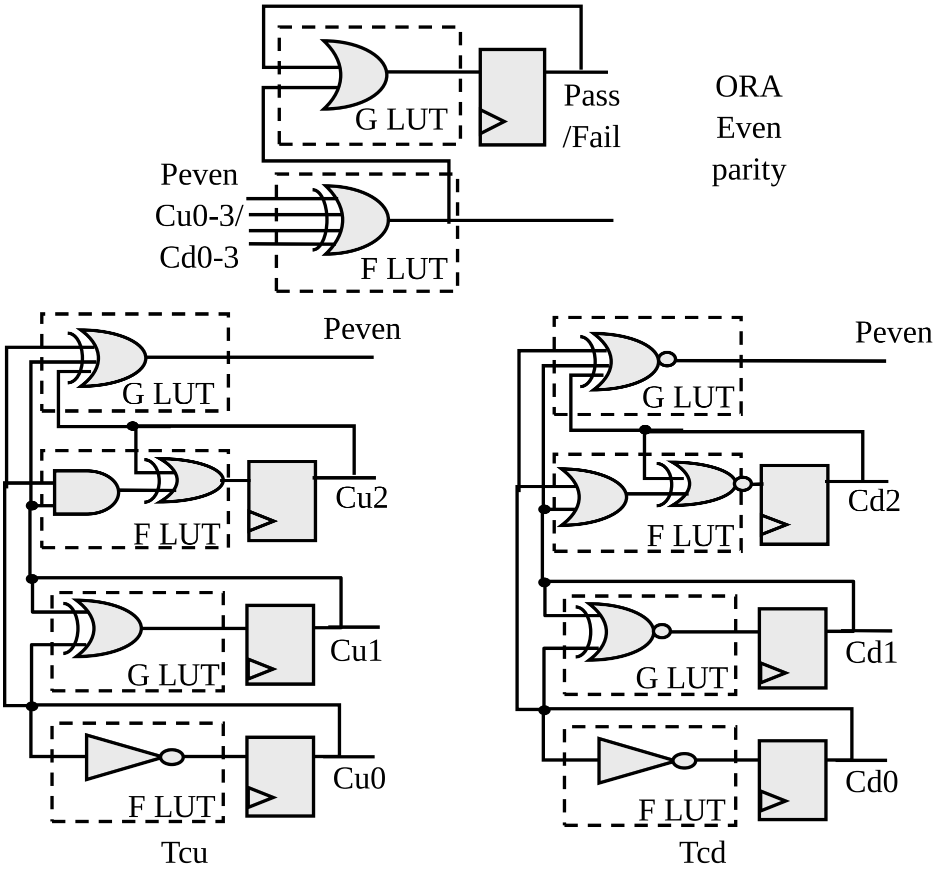


Figure 3.15 Long line BIST TPG and ORA structures

The long line BIST architecture for the north direction is illustrated in Fig. 3.16. For each CLB, one TPG and one ORA are configured, taking up three full slices in total. Each TPG drives four ORAs and the TPG at every sixth CLB sends out one of the test patterns, including Peven, Cu0/Cd0, Cu1/Cd1, or Cu2/Cd2. In addition, every set of adjacent six CLBs is configured to send out the same set of test patterns. As illustrated in Fig. 3.16, CLB *i* through *i*+5 all send out the parity signal while these six CLBs are configured into alternating count-up TPGs and count-down TPGs. In this way, the bridging faults on adjacent lines can be tested because the even parity bits of the two types of TPGs produce opposite logic values, as can be seen in Table 3.3. The same situation happens to the adjacent CLBs which are configured to send out count values.

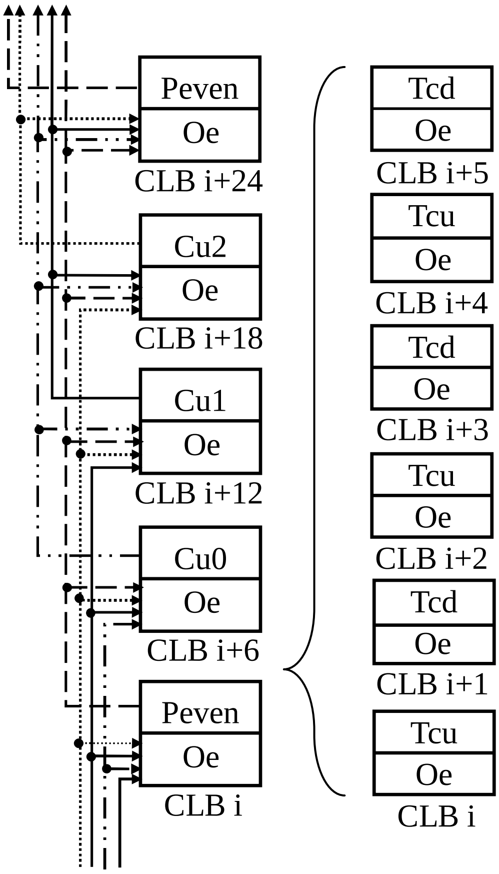


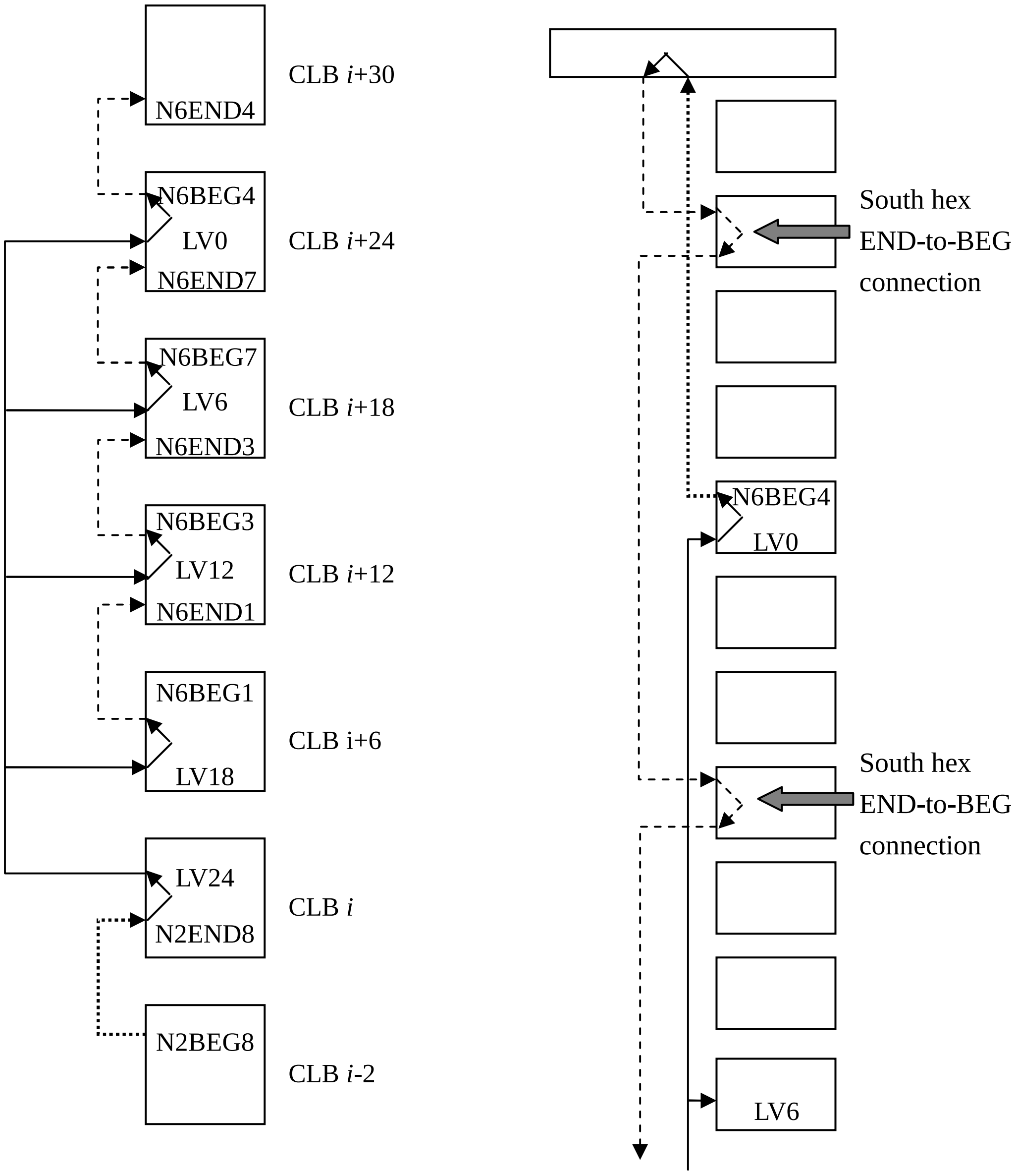
Fig 3.16 CLB column north long line BIST architecture

Table 3.3 Long lines BIST test pattern sequence

TPG count-up even parity (Tcu)		TPG count-down even parity (Tcd)	
Cu2Cu1Cu0	P_{even}	Cd2Cd1Cd0	P_{even}
0 0 0	0	1 1 1	1
0 0 1	1	1 1 0	0
0 1 0	1	1 0 1	0
0 1 1	0	1 0 0	1
1 0 0	1	0 1 1	0
1 0 1	0	0 1 0	1
1 1 0	0	0 0 1	1
1 1 1	1	0 0 0	0

The BIST architecture for CLB column north long lines is illustrated in Fig. 3.17 (a). The dashed lines used in Fig. 3.17 (a) represent the hex lines, the dotted lines represent the double lines and the solid lines represent the WUTs. Due to the limitation of the connections between long line terminals and the LUTs, north double and hex lines are used to connect the WUTs to TPGs/ORAs. The TPG is configured in CLB *i*-2 instead of CLB *i*, the test signals pass through north double lines and then turn to long lines through terminal LV24 because terminal LV24 does not have connections from the LUTs. Since there are no paths provided in the switch box from terminals LV18, LV12, LV6 and LV0 into the LUTs, these terminals are connected to north hex lines and then fed into the ORAs which are located at CLB *i*+12, CLB *i*+18, CLB *i*+24 and CLB *i*+30. Each long line terminal is connected to specific hex line wire segments, as shown in Table 3.4. This table illustrates how the long line terminals are routed into the ORAs as well as the feedback connections in the TPGs and ORAs. For example, terminal LV0 is routed onto north hex line wire segment 1 and fed into the LUTs in slice 0 through the F2 input via wire segment connections N6END1, N6BEG1 and N2BEG1.

At the edges of the array, loopbacks are used via hex lines as well as double lines to route the test signals to the opposite edge of the array. In Fig. 3.17 (b), loopbacks are used at the top edge of the array and the test signals turn onto south hex lines to the bottom edge of the array. The ORA for terminal LV0 is placed at the bottom of the array. The dotted lines used in Fig. 3.17 (b) represent the north hex lines, the solid lines represent north long lines under test and the dashed lines represent south hex lines. The BIST architecture for CLB column south long lines is arranged in the same way as the north long lines. Both north and south directions require one test configuration.



(a) (b)

Figure 3.17 CLB column north long line BIST architecture (cont.)

Table 3.4 North CLB column long line BIST configurations

	Start	PIPs	End
Long Lines Terminals	XQ2(CU0/CD0)/ YQ2(CU1/CD1)/ XQ3(CU2/CD2)/ Y3(Peven)	OMUX9 -> N2BEG8-> N2END8	LV24
	LV18	N6END1 -> N6BEG1 -> N2BEG1	F4-0
	LV12	N6END3 -> N6BEG3 -> E2BEG3	F3-0
	LV6	N6END7 -> N6BEG7 -> E2BEG7	F2-0
	LV0	N6END4 -> N6BEG4 -> S2BEG4 -> BOUNCE3	F1-0
TPG Feedback	XQ2	F2-2	OMUX6
		G2-2	OMUX6
		F2-3	OMUX6
		G2-3	OMUX6
	YQ2	G1-2	OMUX2
		F1-3	OMUX2
		G1-3	OMUX2
	XQ3	F4-3	OMUX13
		G4-3	OMUX13
ORA Feedback	X0	G1-0	
	YQ0	G4-0	OMUX0 -> S2BEG0

3.2.2 Horizontal Long Lines BIST

According to long line architectures, the BIST implementation for CLB column west and east long lines is illustrated in Fig. 3.18. The TPG and ORA designs remain the same as illustrated in Fig. 3.15. As illustrated in Fig. 3.18, each TPG drives four ORAs. For the west long lines which source from CLB *i*, the TPG have to be located at CLB *i*+2 since terminal LH0 is not connected to LUT inputs. Therefore, the TPG is located in CLB *i*+2 and west double lines, shown as dotted lines in Fig. 3.18, are used to route the TPG to terminal LH0. The ORAs are configured at CLB *i*-12, *i*-18, *i*-24 and *i*-30 due to the lack of connections between LH terminals and LUT inputs. The dashed lines represent the west hex lines used to route the long line terminals into the ORAs. The WUTs are represented by the solid lines. For example, west hex lines are used to route test patterns into ORAs since LH terminals are only connected to hex line BEG terminals. An example of such connections, LH6-to-W6BEG0 connection, is shown in Fig. 3.18. In similar way, there are LH12-to-W6BEG4 connection, LH18-to-W6BEG6 connection, LH24-to-W6BEG8 connection.

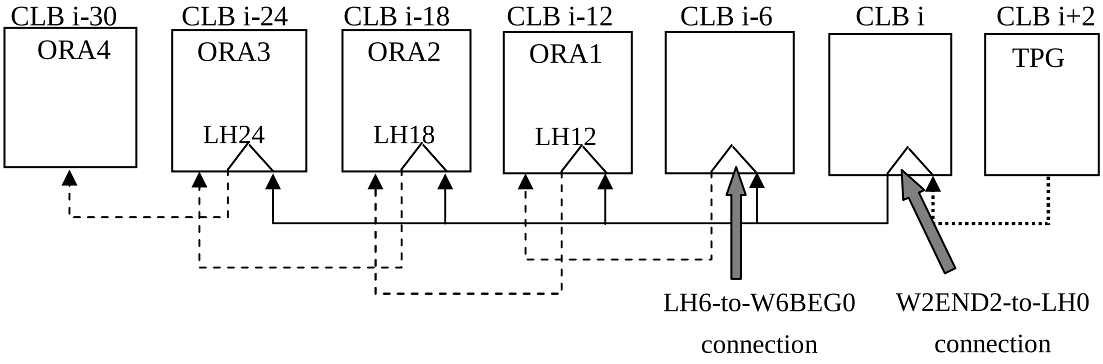


Figure 3.18 BIST implementation for CLB column west long lines

Three test configurations are needed to test all west long lines. For any CLB column *i*, terminal LH0 is tested if column *i*+2 is a CLB column, terminal LH6 is tested if column *i*-12 is a CLB column, and LH12 through LH24 are tested if columns *i*-18, *i*-24, *i*-30 are CLB columns, respectively. The long lines sourced from a CLB column will be tested in the first configuration. For any non-CLB column *i*, the long lines sourced from column *i* will be tested in the second configuration. If column *i* is a CLB column while column *i*+2 is a non-CLB column, the long lines sourced from column *i* will be tested in the third configuration. In the third configuration, TPGs are configured in column *i*+4. The terminals which are not tested in the first and the second configurations are also tested in the third configuration.

3.2.3 Non-CLB Column Vertical Long Lines BIST

For non-CLB column north and south long lines BIST, adjacent CLB columns are used for implementing TPG and ORA, as illustrated in Fig. 3.19. TPG signals are sent from the TPG located in CLB *i* in adjacent CLB column *j*+1 and routed onto the WUTs via double line wire segment 8. In non-CLB column *j*, W2MID8 is connected to terminal LV24 to pass TPG signals onto the WUTs. North hex line terminals are then connected to long lines terminals. Four ORAs are configured in CLB *i*+12, *i*+18, i+24 and *i*+30, respectively. The loopbacks are the same as those for CLB column long lines. In Fig.3.19, the dotted, dashed, and solid lines stand for east (west) double lines, long lines and north hex lines under test, respectively.

The BIST architecture for non-CLB column long lines changes slightly when it comes to the non-CLB column at the eastern edge of a chip. As illustrated in Fig. 3.20, for the non-CLB column at the right edge of a chip, the adjacent CLB column is used to implement TPGs and ORAs. CLB *i*-1 is configured as TPGs, and CLBs *i*+12, *i*+18, *i*+24 and *i*+30 are configured as ORAs. TPG signals are sent out through north double lines, connected to east double lines and then routed onto long lines through terminal LV24. When reaching terminal LV18, test signals propagate onto north hex lines, arriving at CLB *i*+12 where north hex lines are connected to west double lines to feed test signals into ORAs in CLB columns. The dotted, dashed and solid lines stand for double lines (east, west and north), north hex lines and north long lines under test, respectively.

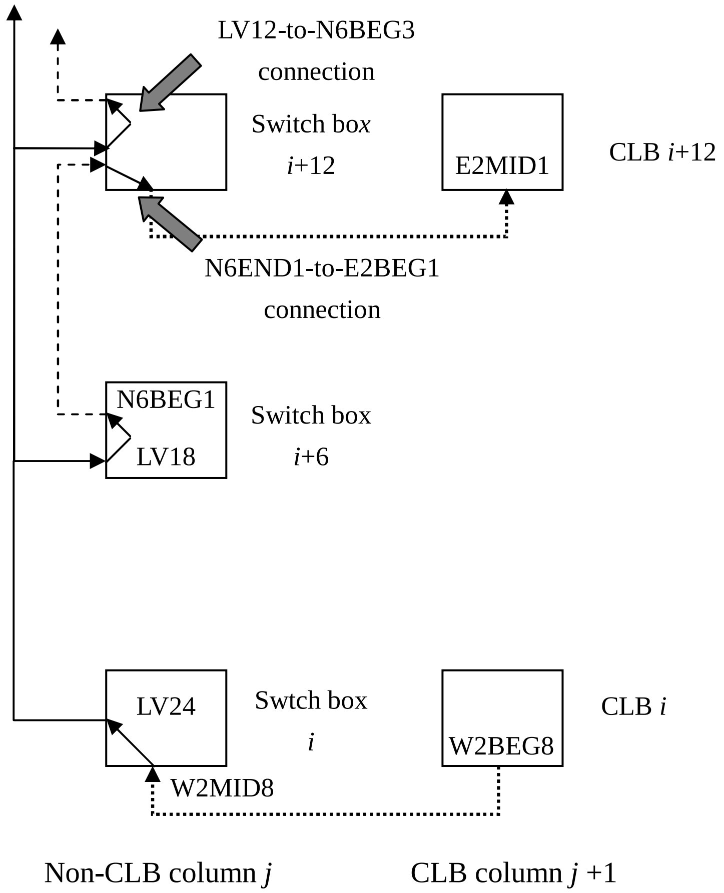


Figure 3.19 BIST architecture for non-CLB column north long lines

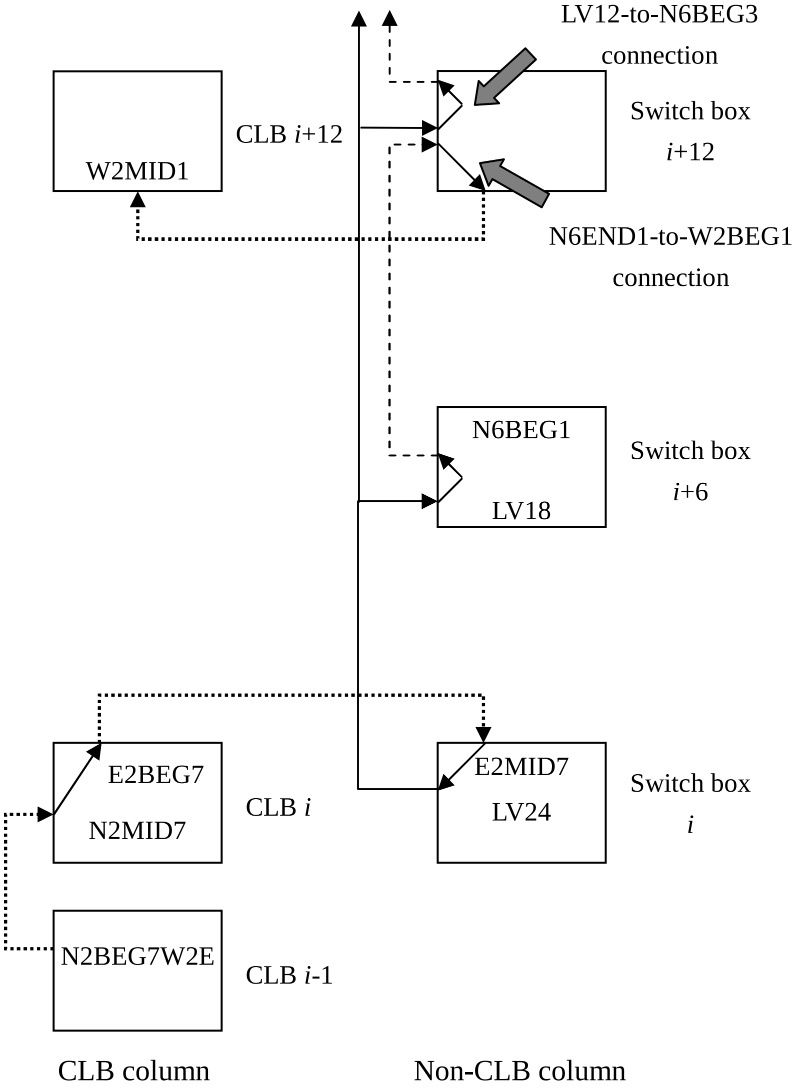


Figure 3.20 BIST architecture for non-CLB column long lines at the eastern edge of array

3.3 BIST Configuration Summary

The number of BIST configurations is summarized in Table 3.5. The BIST for CLB column hex lines in all four directions requires 16 configurations in total, four configurations for each direction. The number of BIST configurations for non-CLB column hex lines vary among different devices due to different arrangements of non-CLB columns. Two configurations are required for CLB column north and south long lines, as well as for non-CLB column north and south long lines. As for CLB column east and west long lines, each requires three configurations since non-CLB columns are involved.

Table 3.5 BIST configuration summary

Routing Resource		**Direction**				**Total Configs**
		N	**S**	**W**	**E**	
CLB hex lines		4	4	4	4	16
Non-CLB column hex lines	LX	4	4	╳	╳	8
	SX25/35	8	8	╳	╳	16
	SX55	20	20	╳	╳	40
CLB long lines		1	1	3	3	8
Non-CLB column long lines		1	1	╳	╳	2
Total BIST Configurations						**LX=34** **SX25/35=42** **SX55=66**

CHAPTER 4

VIRTEX-4 ROUTING BIST IMPLEMENTATION RESULTS

The actual implementation screen images from the Xilinx design tool--FPGA editor for BIST configurations are given in this chapter. The implementation results of the cross-coupled parity approach for Virtex-4 FPGA EasyPath parts are presented, focusing on CLB column hex and long lines as well as non-CLB column hex and long lines. This is followed by evaluations of the routing BIST implementations, including timing analysis and memory usage.

4.1 Virtex-4 FPGAs Routing BIST Implementation Results

Detailed Virtex-4 FPGAs routing BIST architectures for hex and long lines have been described in Chapter 3. By presenting intuitive implementation images of routing BIST configurations from FPGA Editor, Virtex-4 routing BIST implementation results are illustrated. All the implementation results illustrated are based on a Virtex-4 FPGA LX60 device.

4.1.1 CLB Column North and South Hex Lines BIST

For each CLB column north and south hex line, the MID terminal and the END terminal are tested in separate BIST configurations. Fig. 4.1 illustrates the implementation of BIST for CLB column north hex line MID terminals. Internal to each CLB, slices 2 and 3 are configured as the TPGs and slice 1 is configured as the ORAs. Fig. 4.2 illustrates the BIST implementation for CLB column north hex lines END terminals. Internal to each CLB, slices 2 and 3 are configured as the TPGs and slice 0 is configured as the ORAs. For the loopback connections used at the top edge of array for north hex lines, Fig. 4.3 illustrates the actual implementation in Virtex-4 FPGAs.

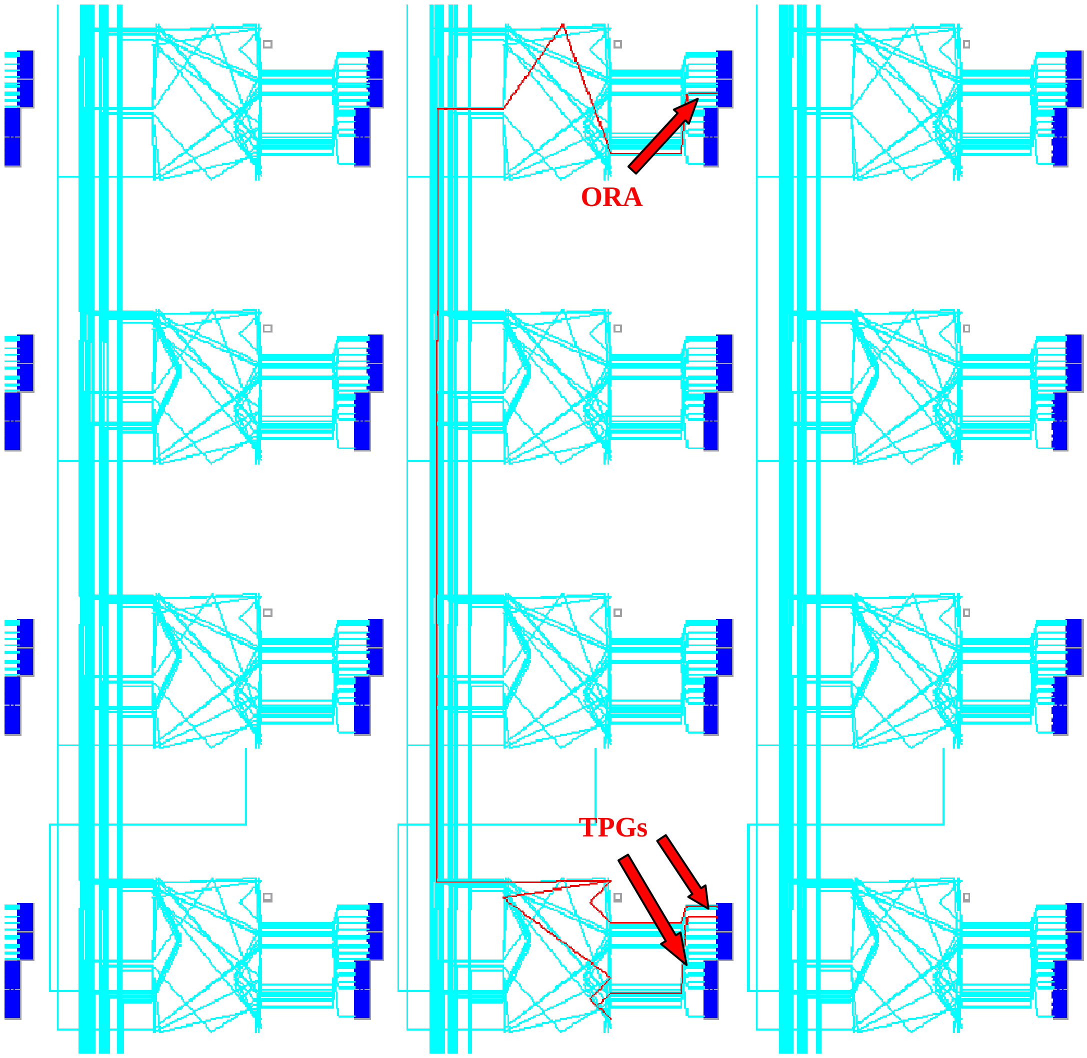


Figure 4.1 BIST for CLB column north hex line MID terminal.

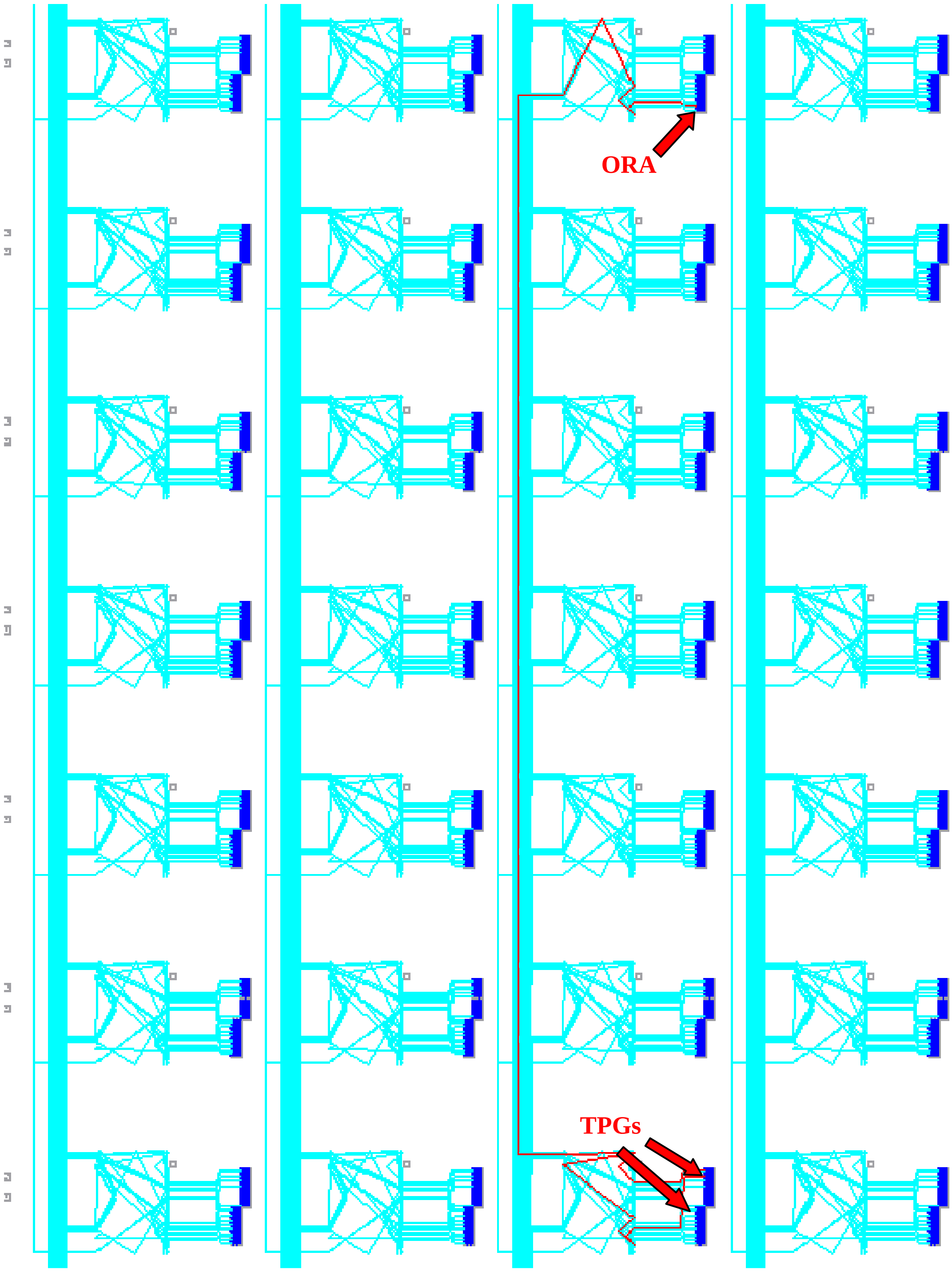


Figure 4.2 BIST for CLB column north hex line END terminal.

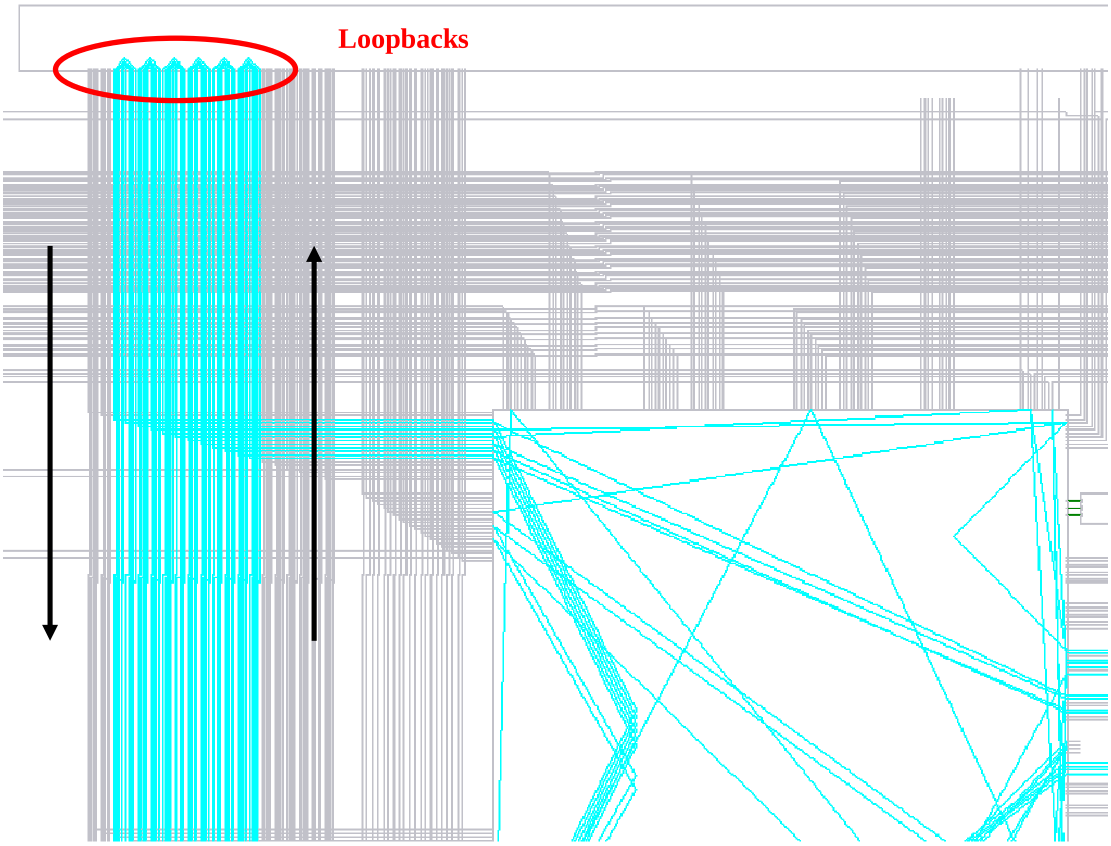


Figure 4.3 Loopback connections for CLB column north hex line.

The BIST architecture for CLB column south hex lines is similar to that for CLB column north hex lines except for the reversed signal flow direction. An illustration of the BIST structure for CLB column south hex line MID terminal is given in Fig. 4.4.

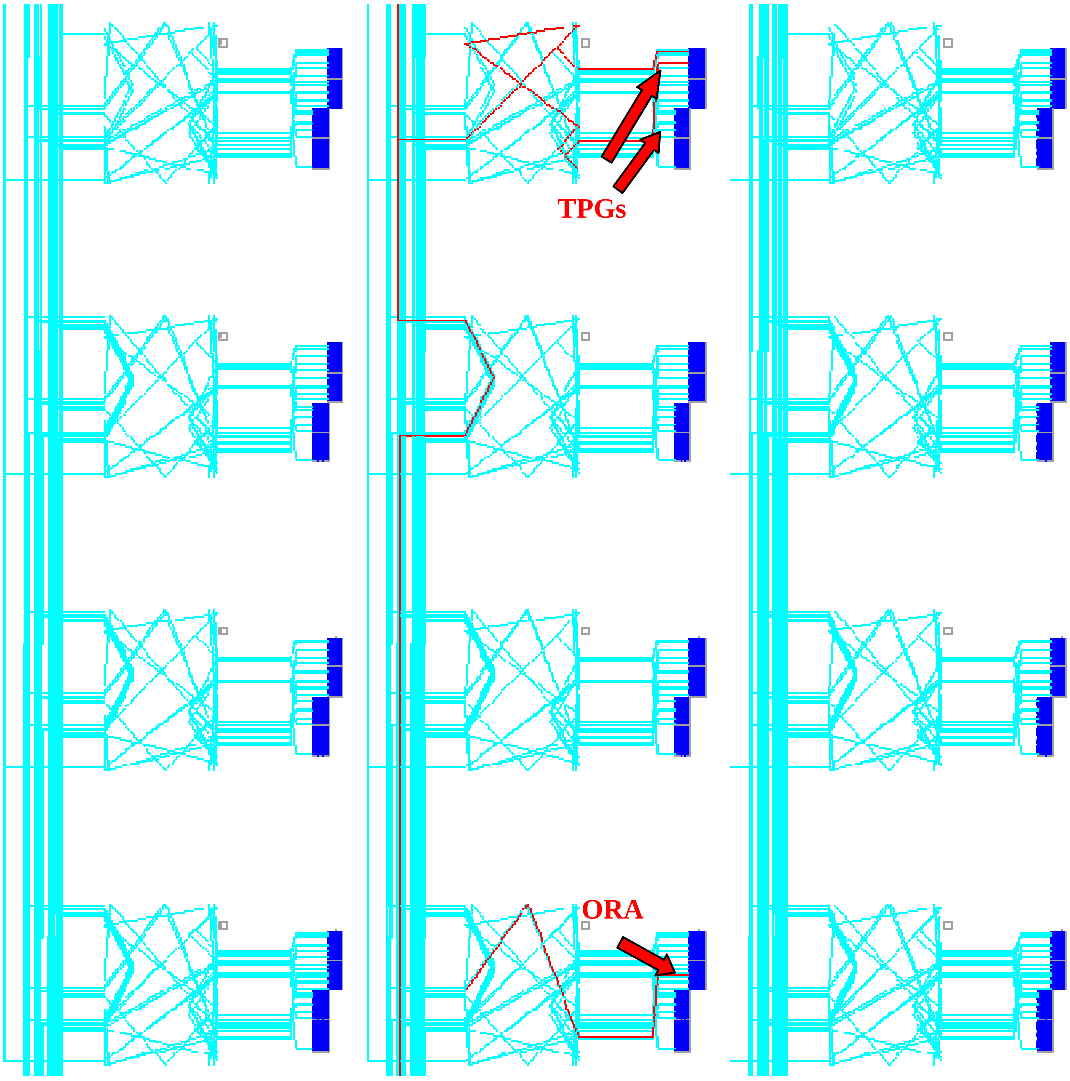


Figure 4.4 BIST for CLB column south hex line MID terminal.

4.1.2 West and East Hex Lines BIST

The main difference between north, south hex lines and west, east hex lines is the involvement of non-CLB columns. Fig. 4.5 illustrates the implementation of the strategies for bypassing non-CLB columns by directly connecting END and BEG terminals in non-CLB column switch boxes. Fig 4.6 shows an example of loopback connections at the left edges of the chip for west hex lines. Fig 4.7 shows an example of loopback connections at the right edges of the chip for east hex lines. The arrows in

Fig. 4.6 (c) and Fig. 4.7 (c) indicate the signal flow direction of loopback connections.

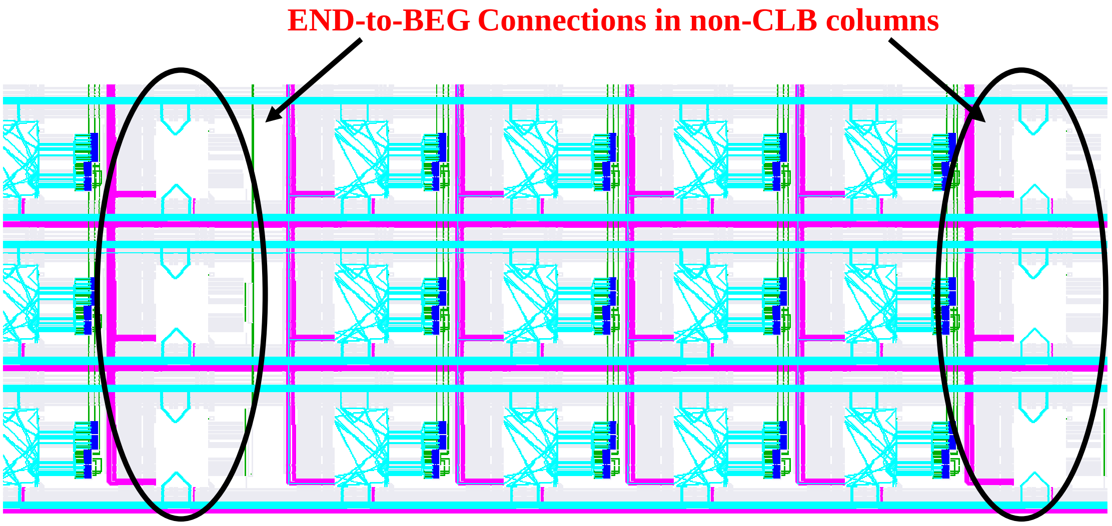
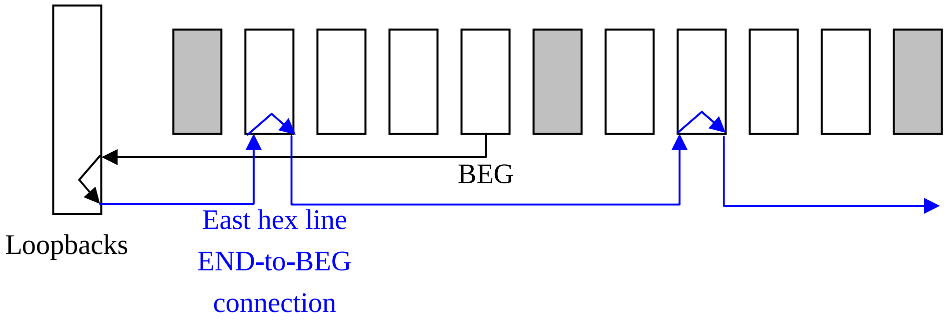
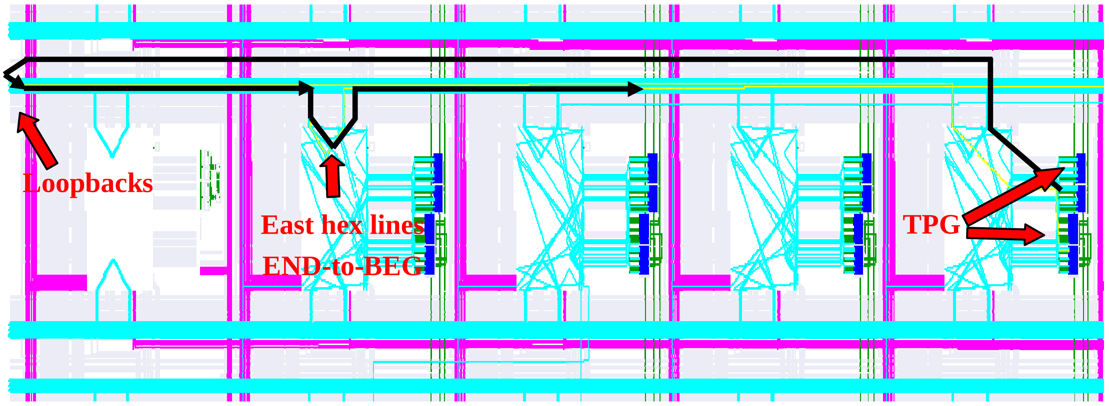


Figure 4.5 END-to-BEG connections in non-CLB column switch boxes

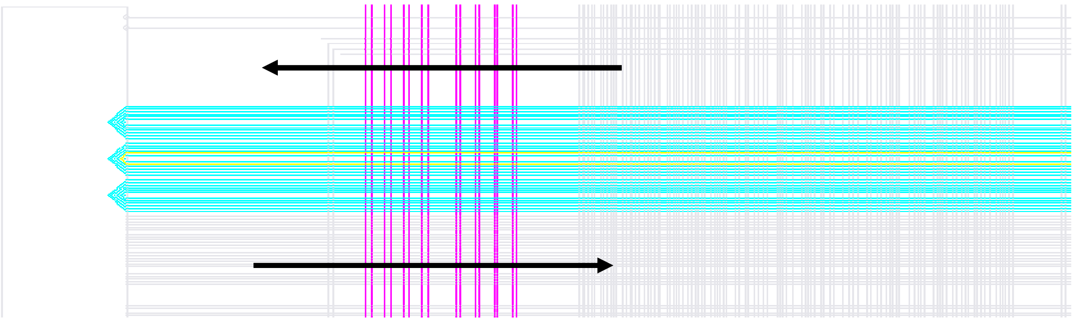


(a) BIST structure



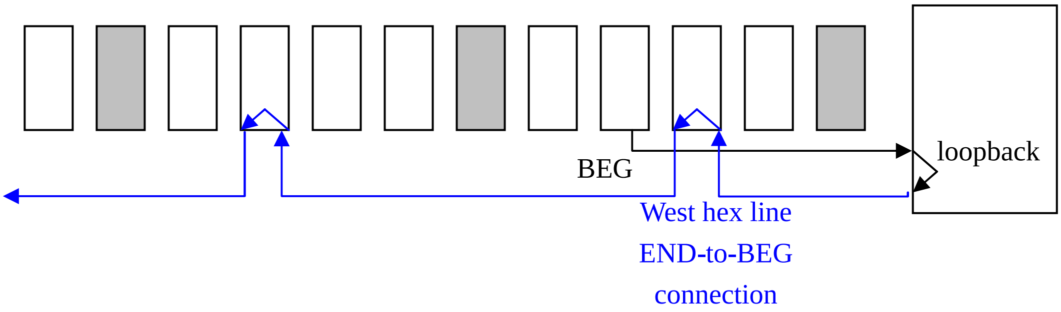
(c) Actual implementation of (a)

Figure 4.6 Examples of loopback connection implementations for west hex lines.

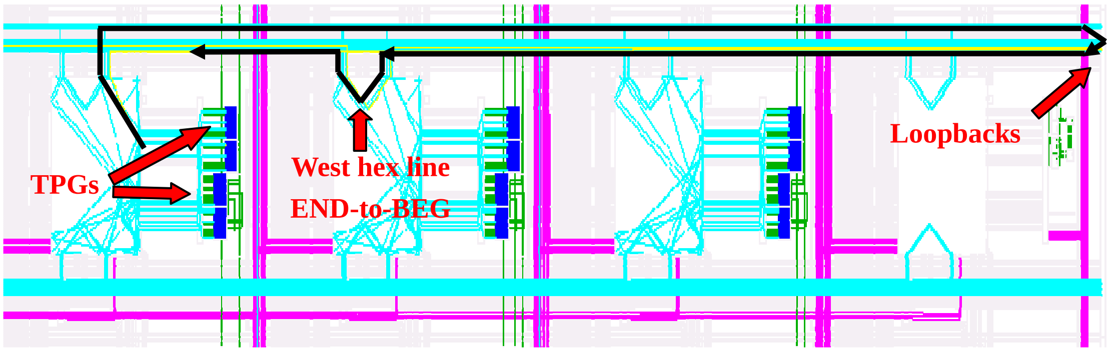


(c) Detailed loopback connections

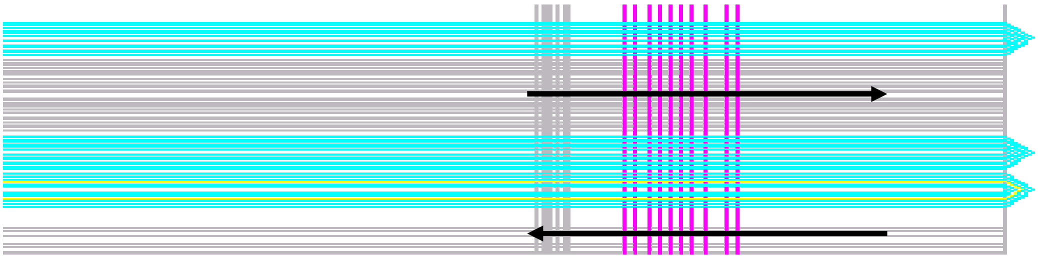
Figure 4.6 Examples of loopback connection implementations for west hex lines.



(a) BIST structure with loopback connections



(b) Actual implementation of (a)

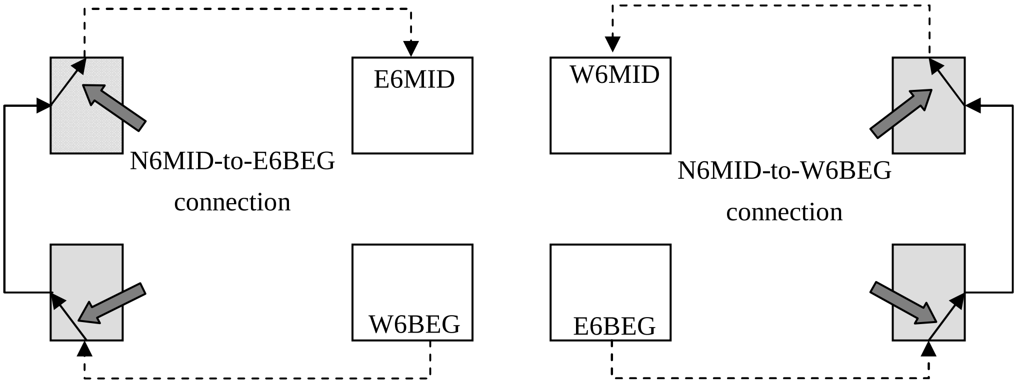


(c) Detailed loopback connections

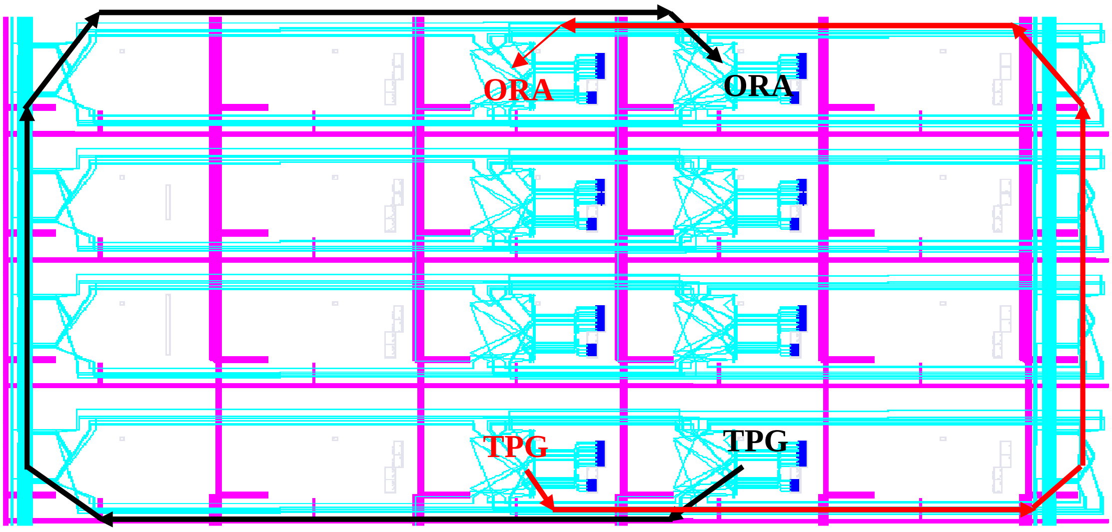
Figure 4.7 Example of loopback connection implementations for east hex lines

4.1.3 Non-CLB Column Hex Lines BIST

The primary difference between CLB column hex lines BIST and non-CLB column hex lines BIST is that neighboring CLB columns are used to implement TPGs and ORAs when testing non-CLB column hex lines. Except for the right most non-CLB column which uses the CLB column three columns to the left, other non-CLB columns use the CLB column three columns to the right. Fig. 4.8 illustrates the BIST structure and the actual implementation image. The arrows in Fig. 4.8 (b) indicate the signal flow direction.



(a) BIST structure

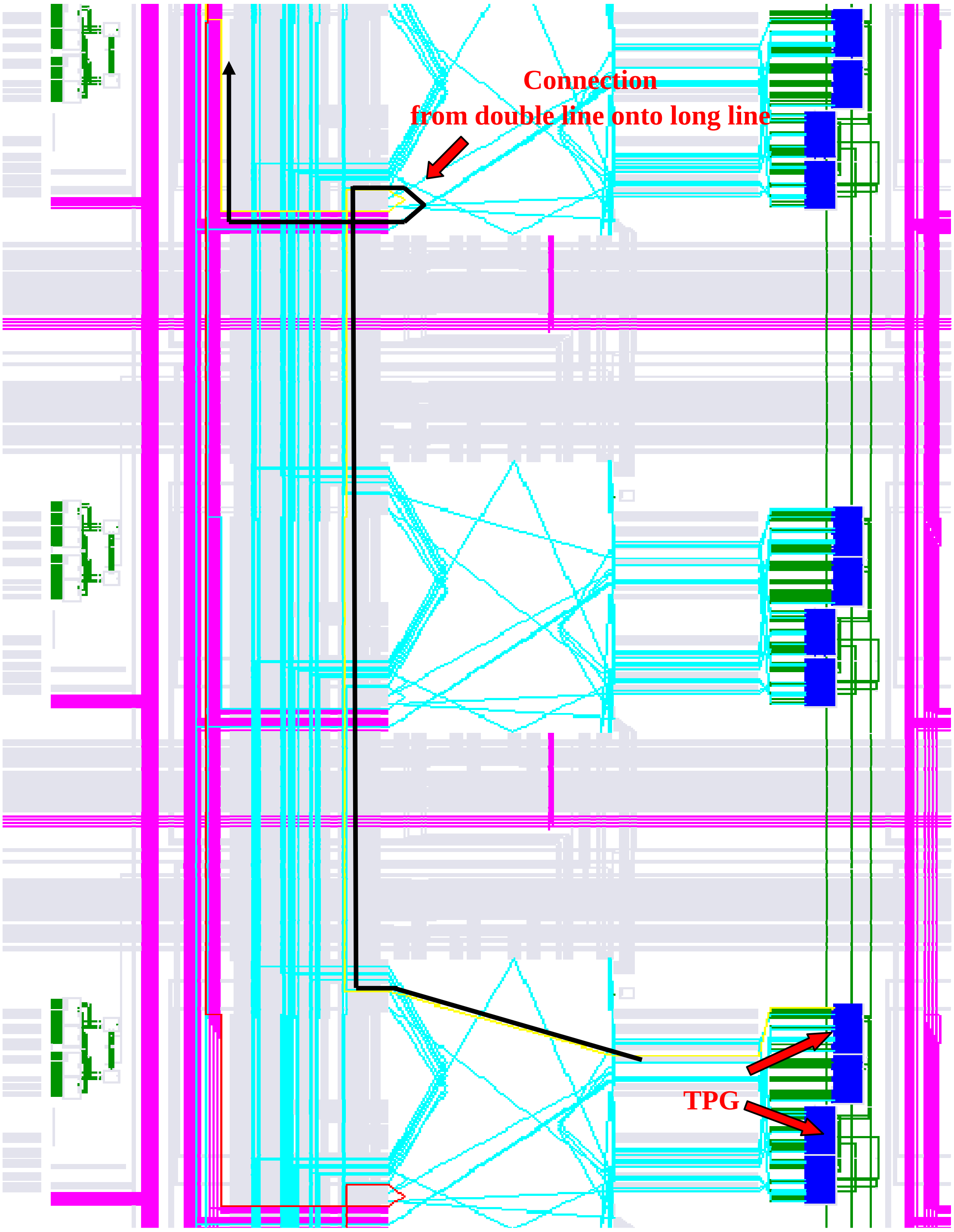


(b) Actual implementation of (a)

Figure 4.8 BIST implementation for non-CLB column hex lines (north)

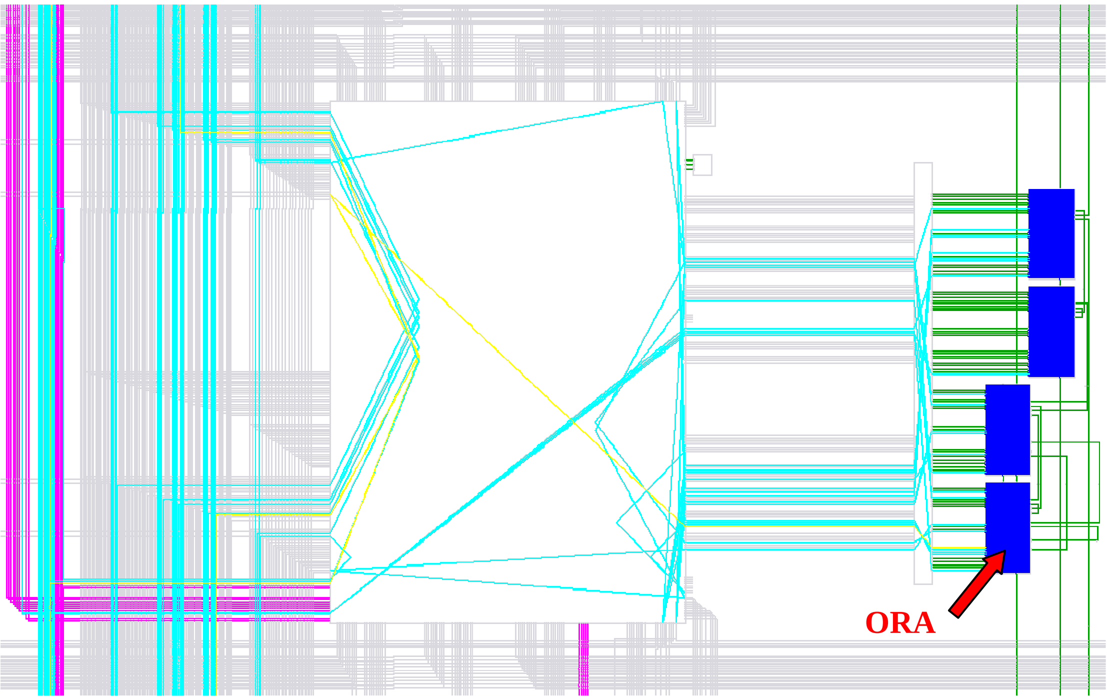
4.1.4 CLB Column North and South Long Lines BIST

Fig. 4.9 and Fig. 4.10 illustrate the BIST structure for CLB column north long lines. Fig. 4.9 (a) shows that the TPG is first connected to a north double line and then routed onto a long line; Fig. 4.9 (b) shows the routing from north hex lines to the ORAs. Fig. 4.10 shows the loopback connections at the top edge of array, including double lines and hex lines. Both double and hex lines are used for connecting the targeted long lines under test with TPGs and ORAs.



(a) Connections from TPG to double line and turn to long line

Figure 4.9 BIST structure for CLB column north long lines



(b) Connections from hex line into the ORA

Figure 4.9 BIST structure for CLB column north long lines

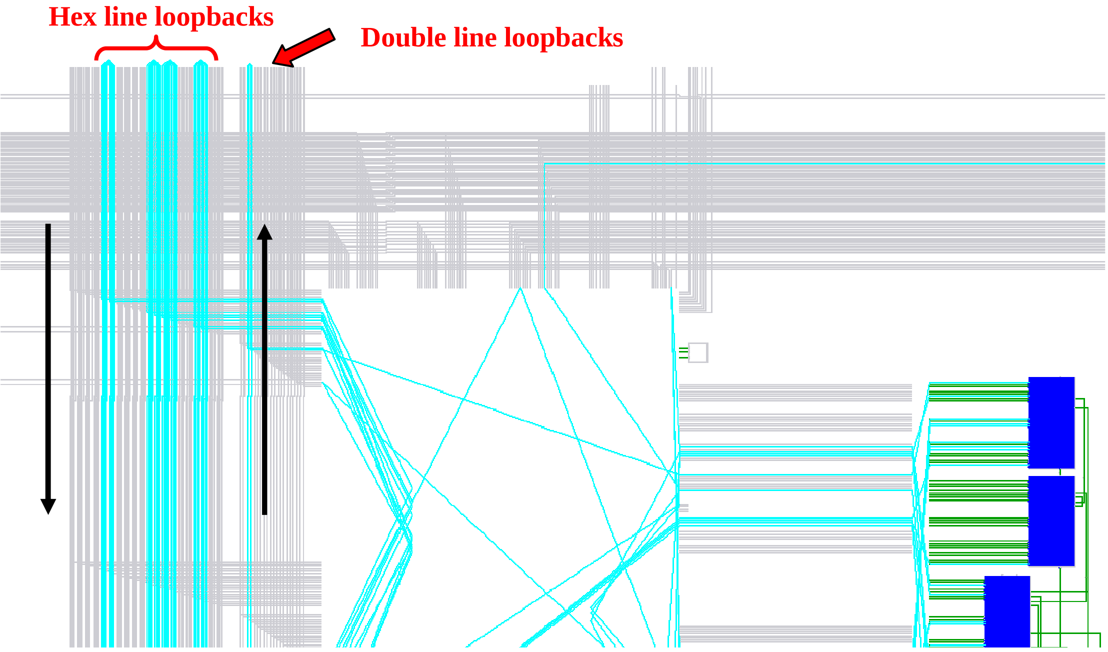
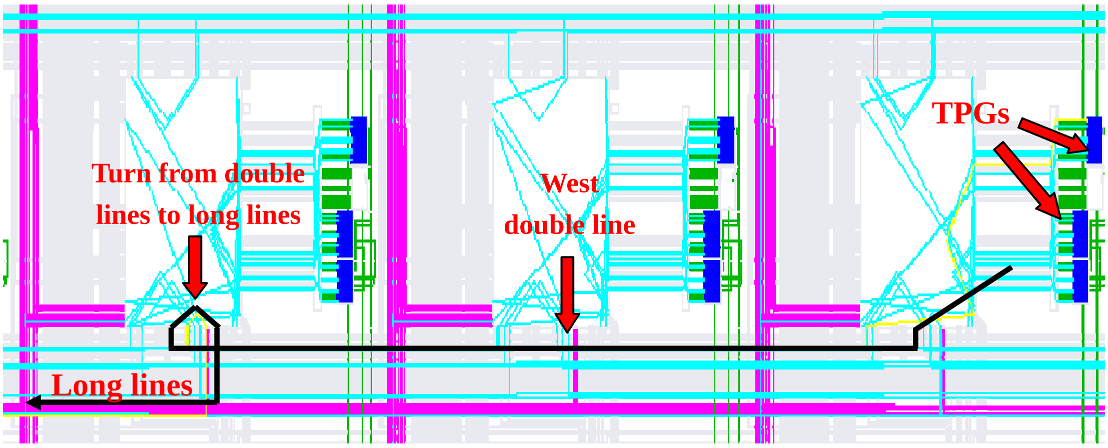


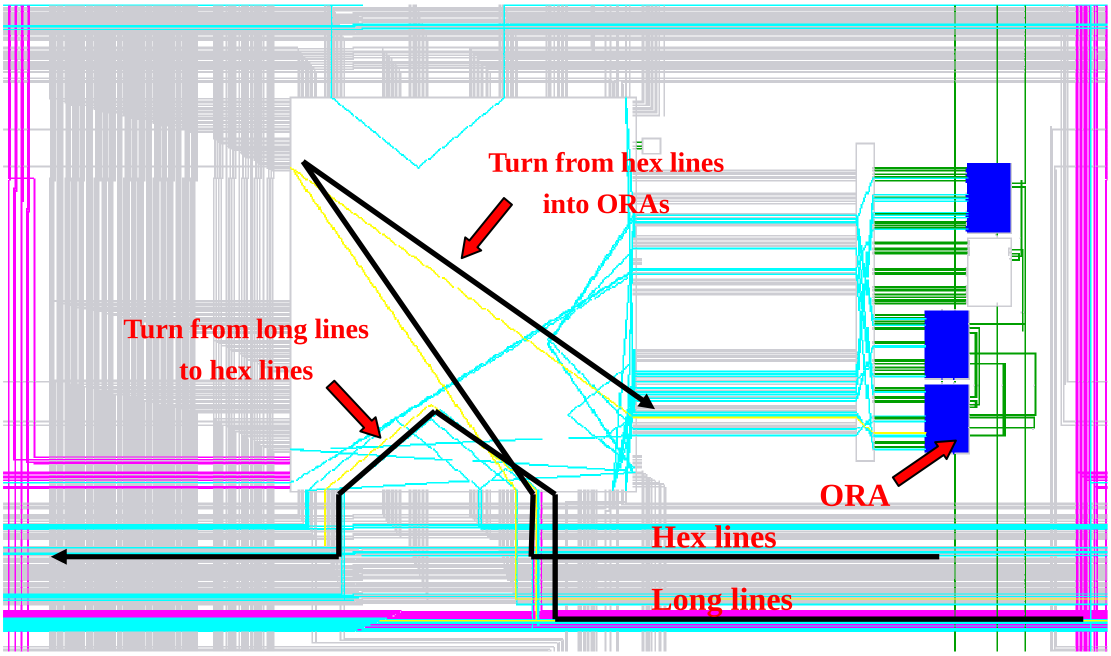
Figure 4.10 Double line and hex line loopback connections

4.1.5 CLB Column West and East Long Lines BIST

For CLB column west long lines BIST, TPGs and WUTs need to be connected via west double lines; WUTs are routed into ORAs via west hex lines. For CLB column east long lines BIST, east double lines and east hex lines are used. This is due to the limited connections between horizontal long line terminals and LUTs. Fig. 4.11 illustrates the usage of west double and hex lines in west long lines BIST.



(a) Connections between TPGs and WUTs via west double lines

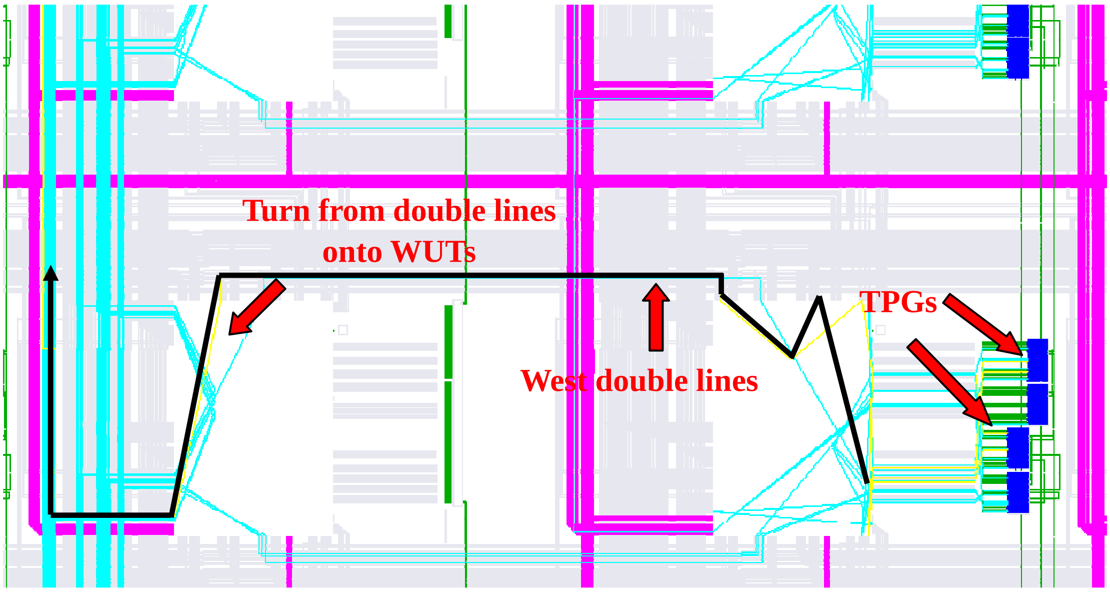


(b) Connections between WUTs and ORAs via west hex lines

Figure 4.11 BIST structure for CLB column west long lines

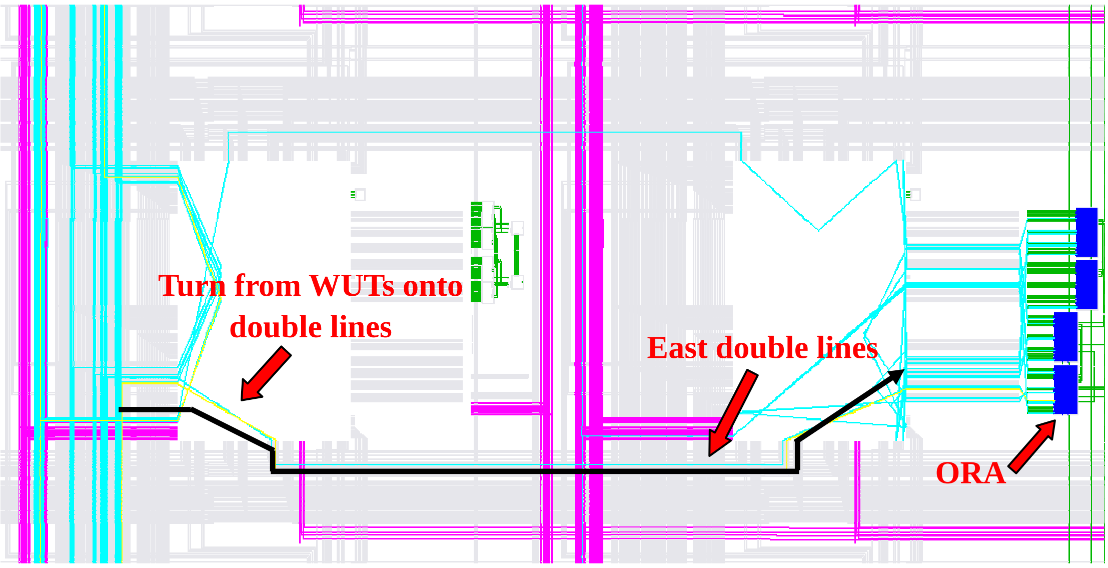
4.1.6 Non-CLB Column North and South Long Lines BIST

Non-CLB column long line BIST needs to implement TPGs and ORAs in adjacent CLB columns. The rightmost non-CLB column uses the CLB column which is one column to the left; and all other non-CLB columns use the CLB columns which are one column to the right. Test signals are routed onto east or west double lines, and turned to long lines, as illustrated in Fig. 4.12 (a). The routing turns from long lines onto west (or east) double lines and feed into the ORAs, as illustrated in Fig. 4.12 (b).



(a) West double lines are used between TPGs and WUTs

Figure 4.12 BIST structure for non-CLB north long lines



(b) East double lines are used between WUTs and ORAs

Figure 4.12 BIST structure for non-CLB north long lines

4.2 Experimental Results on Virtex-4 FPGA EasyPath Parts

Virtex-4 FPGA EasyPath parts are devices with existing faults which failed Xilinx manufacturing tests before going to the market. All developed routing BIST configurations were executed on Virtex-4 FPGA EasyPath Parts, including nine LX60 devices and four SX35 devices.

The routing BIST results are listed in Table 4.1 and Table 4.2. All configurations are listed in the table column subheadings with pass or fail indication in table entries for all EasyPath parts. The letter P indicates that the configurations passed on specific EasyPath parts. The letter F indicates that the configurations failed on specific EasyPath parts. As shown in Table 4.1 and Table 4.2, all routing BIST configurations passed on nine LX60 EasyPath parts (EP-1 through EP-9) and three SX35 EasyPath parts (EP-15, EP-17 and EP-19), while they failed on one SX35

EasyPath part (EP-16). In other words, the ORAs indicated failures in each routing BIST configuration on the EP-16 chip.

Table 4.1 Routing BIST results on LX60 EasyPath parts

	CLB Col Hex Lines				Non-CLB Col Hex Lines		CLB Col Long Lines			Non-CLB Col Long Lines	
	N	S	W	E	N	S	N	S	W	N	S
	1-4	1-4	1-4	1-4	1-4	1-4	1-4	1-4	1-3	1	1
EP-1	P	P	P	P	P	P	P	P	P	P	P
EP-2	P	P	P	P	P	P	P	P	P	P	P
EP-3	P	P	P	P	P	P	P	P	P	P	P
EP-4	P	P	P	P	P	P	P	P	P	P	P
EP-5	P	P	P	P	P	P	P	P	P	P	P
EP-6	P	P	P	P	P	P	P	P	P	P	P
EP-7	P	P	P	P	P	P	P	P	P	P	P
EP-8	P	P	P	P	P	P	P	P	P	P	P
EP-9	P	P	P	P	P	P	P	P	P	P	P

Table 4.2 Routing BIST results on SX35 EasyPath parts

	CLB Col Hex Lines				CLB Col Long Lines			Non-CLB Col Long Lines	
	N	S	W	E	N	S	W	N	S
	1-4	1-4	1-4	1-4	1-4	1-4	1-3	1	1
EP-15	P	P	P	P	P	P	P	P	P
EP-16	F	F	F	F	F	F	F	F	F
EP-17	P	P	P	P	P	P	P	P	P
EP-19	P	P	P	P	P	P	P	P	P

Based on BIST diagnosis, the possible fault locations were narrowed down to row one on the left half of the array. For example, the BIST results from the configurations for CLB column north hex lines shows that ORA failures appear at row four as well as row 7 on the left half of the arrays. These ORA failures indicate that the faults reside in row one on the left half of the array. The same diagnosis results were achieved on other BIST configurations. Further experiments showed that the configuration memory write circuitry was faulty. The faulty circuitry wrote logic 1s into bits 22, 23 and 24 of all frames in row one in the left half of the array, resulting in multiple stuck-closed PIPs, which is consistent with the failing BIST results.

4.3 Virtex-4 Routing BIST Analysis

Analysis of Virtex-4 FPGA routing BIST configurations is given in terms of timing analysis and memory resource usage for partial reconfiguration.

4.3.1 Timing Analysis

Timing analysis was performed on all developed routing BIST configurations on a Virtex-4 FPGA LX60 device. The maximum BIST clock frequency for each configuration is recorded in Table 4.3. The routing BIST configuration for non-CLB column south long line has the lowest maximum BIST clock frequency.

For all Virtex-4 FPGA LX and SX devices, the number of rows is larger than the number of columns. This makes the maximum clock frequency of the BIST configurations for vertical hex and long lines lower than that of the BIST configurations for horizontal hex and long lines, as illustrated in Table 4.3. In addition, compared to hex line BIST architectures, long line BIST architectures are more complex and consume more routing resources. Especially, non-CLB column long lines BIST consume the most routing resources since additional connections between each non-CLB column and its neighbor CLB column are needed. Therefore, the configurations for non-CLB column long lines result in the lowest maximum BIST clock frequency.

Table 4.3 Maximum BIST clock frequency for routing BIST configurations on LX60 device

			Max frequency (MHz)
CLB Col Hex Line	**North**	**1**	130.993
		2	133.351
		3	132.608
		4	130.907
	South	**1**	130.617
		2	126.056
		3	133.565
		4	129.333
	West	**1**	151.08
		2	155.4
		3	158.003
		4	155.521
	East	**1**	151.24
		2	158.755
		3	158.655
		4	159.084
Non-CLB Col Hex Line	**North**	**1**	121.183
		2	120.322
		3	123.885
		4	123.993
	South	**1**	121.699
		2	120.788
		3	123.732
		4	124.673
Long Line	**North**	**1**	67.412
	South	**1**	66.494
	West	**1**	97.79
		2	95.776
		3	95.429
Non-CLB Col Long Line	**North**	**1**	65.945
	South	**1**	64.375

Further timing analysis was done on the BIST configuration for non-CLB column south long line. The maximum BIST clock frequency for each LX and SX device is summarized in Table 4.4. Since LX100, LX160 and LX200 devices have the same number of rows, which is the largest number among all LX and SX devices, they have the same lowest maximum clock frequency, as illustrated in Table 4.4 and Fig. 4.13.

Table 4.4 Maximum BIST clock frequency on LX and SX devices

Device	Max frequency (MHz)
LX15	103.125
LX25	79.271
LX40	64.375
LX60	64.375
LX80	54.195
LX100	46.792
LX160	46.792
LX200	46.792
SX25	103.125
SX35	79.271
SX55	64.375

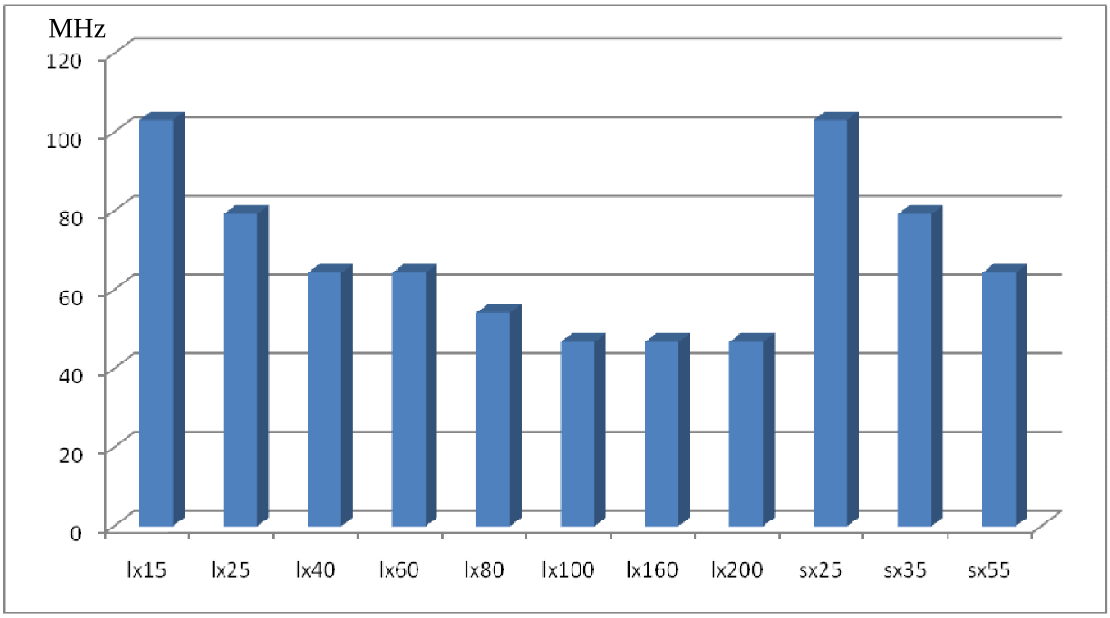


Fig 4.13 Maximum BIST clock frequency on LX and SX devices (Cont.)

4.3.2 Memory Resource Usage for Partial Configurations

For vertical hex and long lines, each column is regarded as a BIST unit and the BIST implementation is repetitive among columns. For horizontal hex and long lines, each row is regarded as a BIST unit and the BIST implementation is repetitive among rows. Therefore, it is the most efficient to generate compressed routing BIST configuration bit files due to the repetitiveness of BIST application.

An even more efficient way is to generate compressed partial configuration bit files. By generating partial configuration bit files, memory usage can be saved due to the similarity between routing BIST configurations. Based on an initial download compressed full configuration bit file, each of the rest of the BIST configuration bit files is generated, including only the configuration frame differences between itself and the reference configuration. Take CLB column hex line BIST configurations for example; all CLBs are configured into TPGs and ORAs. The only differences between different configurations are the configuration memory bits controlling LUT functions and WUTs, which account for a smaller portion of overall configuration memory bits. As illustrated in Table 4.5, the reference configuration is the first configuration for CLB column north hex lines. The order of download files should be in the same order as listed in Table 4.5 (from the top to the bottom). Xilinx design tool BITGEN.EXE is used to generate all partial configurations, and all partial configurations should operate correctly in theory if all of them are downloaded in the correct order. However, the third configuration of CLB column north hex lines failed to operate correctly. This is due to some issues in BITGEN.EXE: partial configuration generation does not work correctly when there are too many changes on routing resources involved.

Table 4.5 lists the comparison on memory usage between full configuration bit files and compressed full configuration bit files, based on LX60 device. 83.3% memory usage is saved by generating compressed full configuration bit files. The comparison on memory usage between compressed full configuration bit files and compressed partial configuration bit files is also included, based on LX60 device. 45% memory usage is saved by generating compressed partial configuration bit files.

Table 4.5 Memory usage for routing BIST configurations on LX60 device

			Full configuration bit files (bits)	**Compressed full configuration bit files (bits)**	**Compressed partial configuration bit files (bits)**
CLB Col Hex Line	**North**	**1**	17717632	2653248	2653248
		2	17717632	2653248	1603424
		3	17717632	2666304	1603424
		4	17717632	2653248	1603424
	South	**1**	17717632	2654880	1683040
		2	17717632	2653248	1454144
		3	17717632	2666304	1683040
		4	17717632	2666304	1522176
	West	**1**	17717632	2744640	1808704
		2	17717632	2731584	1464128
		3	17717632	2826816	1731648
		4	17717632	2839872	1606976
	East	**1**	17717632	2744640	1795648
		2	17717632	2744640	1464128
		3	17717632	2817600	1788992
		4	17717632	2839872	1606976
Non-CLB Col Hex Line	**North**	**1**	17717632	3029472	2201824
		2	17717632	3015456	952224
		3	17717632	2939424	948928
		4	17717632	2976480	864928
	South	**1**	17717632	3064800	1087040
		2	17717632	3007488	922784
		3	17717632	2915616	925088
		4	17717632	3106944	1065536
CLB Col Long Line	**North**	**1**	17717632	2653248	1848544
	South	**1**	17717632	2651616	1561984
	West	**1**	17717632	4904064	3884320
		2	17717632	4165536	2812672
		3	17717632	3490656	1329568
Non-CLB Col Long Line	**North**	**1**	17717632	2932704	1925312
	South	**1**	17717632	2943072	883776
Total			549246592 (65.47MB)	91353024 (10.88 MB)	50287648 (5.99 MB)

CHAPTER 5

SUMMARY AND CONCLUSION

5.1 Summary of Cross-coupled Parity Approach

The cross-coupled parity approach was first developed and implemented in the routing BIST for double lines for Virtex-4 FPGAs in [16]. It is proved to be the most effective method to test the global routing resources in Virtex-4 FPGAs. The cross-coupled parity approach supports larger numbers of wires under test and has advantages over previously developed routing BIST approaches.

The cross-coupled parity BIST approach has a major advantage over the comparison-based BIST approach: any fault affecting the TPG will be detected. This is due to the fact that the parity bits from both up-counter TPG and down-counter TPG are cross-coupled into the ORAs with respect to the count bits.

The cross-coupled parity BIST approach overcomes some problems which reside in the parity-based routing BIST approach proposed in [14]. It is assumed that the parity bits are transmitted on fault-free routing resources in [14], where the routing resources used to transmit the parity bits are considered as WUTs in the cross-coupled parity approach. In addition, the possibility of fault escape is reduced. For example, if the TPG affected by a fault skips some test patterns but the parity bit remains correct, fault escape would occur in the parity-based approach. But this problem is solved in the cross-coupled parity approach by cross-coupling the parity bits generated from two TPGs.

5.2 Summary of Routing BIST in Virtex-4 FPGAs

The work in this thesis presented routing BIST in Virtex-4 LX and SX devices, focusing on CLB column and non-CLB column hex lines and long lines. The number of BIST configurations for all LX devices is 34. The number of BIST configurations increases to 42 for SX 25/35 devices and to 66 for SX 55 device. The different arrangements of non-CLB columns in LX and SX devices lead to the variation on total number of BIST configurations.

Hex lines are classified into CLB column hex lines and non-CLB column hex lines. Hex lines propagate in all four directions: north, south, west and east. There are 10 hex lines for each direction as well as 3 terminals along each hex line, associated with each switch box. Therefore, there are 120 hex line terminals in total associated with each switch box. Four configurations are required to completely test all CLB column hex lines in each direction, where two configurations are for testing MID terminals and two configurations are for testing END terminals. The number of BIST configurations for non-CLB column hex lines are varying between different devices. SX devices require more configurations compared to LX devices since non-CLB columns are arranged more densely in SX devices than in LX devices. As for the BIST for non-CLB column hex lines, only north and south directions require specific test configurations since west and east hex lines are tested in the BIST configurations for CLB column hex lines and non-CLB column north and south hex lines. Four configurations are required to test non-CLB column hex lines for each direction.

Long lines are divided into CLB column long lines and non-CLB column long lines. Long lines propagate horizontally and vertically. There are 5 horizontal long line terminals and 5 vertical long line terminals tied to each switch box. The total number of BIST configurations for long lines is 10 for all LX and SX devices. Since long line terminals do not connect to LUT inputs, double and hex lines are used for connecting WUTs with TPGs and ORAs. This leads to more complex BIST architecture for long lines than hex lines.

5.3 Future Work

The developed BIST architecture was based on Virtex-4 LX and SX devices. Further work needs to be done for FX devices in the area where PowerPC cores are located, since the PowerPC cores prevent the normal propagation of global routing resources.

Complete testing of the switch boxes also remains to be done for all Virtex-4 FPGAs. The developed routing BIST configurations only test a portion of PIPs internal to each switch box. Only the PIPs as well as OMUX connections used for forming signal paths in the switch box are tested at this time.

The developed program for Virtex-4 FPGAs could be modified for Virtex-5 FPGAs. The cross-coupled parity approach may need alterations corresponding to the structural changes in global routing resources in Virtex-5 FPGAs.

BIBLIOGRAPHY

[1] S. Brown, R. Francis, J. Rose, and Z. Vranesic, Field-Programmable Gate Arrays, Boston, MA: Kluwer Academic Publishers, 1992.

[2] L. Wang, C. Stroud, and N. Touba, System On Chip Test Architectures, Amsterdam: Elsevier, 2007.

[3] J. Rose, A. El Gamal, and A. Sangiovanni-Vincentelli, "Architecture of Field Programmable Gate Arrays," *Proc. of IEEE*, vol. 81, no. 7, pp. 1013-1029, 1993.

[4] C. Stroud, S. Wijesuriya, C. Hamilton, and M. Abramovici, "Built-In Self-Test of FPGA Interconnect," *Proc. IEEE International Test Conf.*, pp. 404-411, 1998.

[5] C. Stroud, J. Nall, M. Lashinsky, and M. Abramovici, "BIST-Based Diagnosis of FPGA Interconnect," *Proc. IEEE International Test Conf.*, pp. 618-627, 2002.

[6] *Virtex-4 Family Overview*, v 2.0, Xilinx Inc., San Jose, CA, 2007.

[7] V. D. Agrawal, R. Charles and K. Saluja, "A Tutorial on Built-In Self-Test, Part 1: Principles", *IEEE Design and Test of Computers*, vol. 10, no.1, pp. 73-82, 1993.

[8] C. Stroud, *A Designer' Guide to Built-In Self-Test*, Boston, MA: Springer, 2002.

[9] M. Abramovici and C. Stroud, "BIST-based Test and Diagnosis of FPGA Logic Blocks," *IEEE Trans. Very Large Scale Intergration Systems*, vol. 9, no. 1, pp. 159-172, 2001.

[10] T. Liu, F.Lombardi, and J. Salinas, "Diagnosis of Interconnects and FPICs Using a Structured Walking-1 Approach," *Proc. IEEE VLSI Test Symp.*, pp. 256-261, 1995.

[11] F. Lombardi, D. Ashen, X. Chen, and W. K. Huang, "Diagnosing Programmable Interconnect Systems for FPGAs," *Proc. ACM/SIGDA International Symp. on FPGAs*, pp. 100-106, 1996.

[12] M. Renovell, J. Figueras, and Y. Zorian, "Test of RAM-Based FPGA: Methodology and Application to Interconnect," *Proc. IEEE VLSI Test Symp.*, pp. 203-237, 1997.

[13] H. Michinishi, T. Yokohira, T. Okamoto, T. Inoue, and H. Fujiwara, “A Test Methodology for Interconnect Structures of LUT-Based FPGAs,” *Proc. IEEE Asian Test Symp.*, pp. 68-74, 1996.

[14] X. Sun, J. Xu, B. Chan, and P. Trouborst, “Novel Technique for BIST of FPGA Interconnects,” *Proc. IEEE International Test Conf.*, pp. 795-803, 2000.

[15] J. Sunwoo and C. Stroud, “Built-In Self-Test of Configurable Cores in SOCs Using Embedded Processor Dynamic Reconfiguration,” *Proc. International System-on-Chip Design Conf.*, pp. 174-177, 2005.

[16] B. Dixon and C. Stroud, “Analysis and Evaluation of Routing BIST Approaches for FPGAs,” *Proc. IEEE North Atlantic Test Workshop*, pp. 85-91, 2007.

[17] B. Dixon, “Built-In Self-Test of The Programmable Interconnect In Field Programmable Gate Arrays,” M.S. Thesis, Auburn University, Auburn, AL, 2008.

[18] *Virtex-4 User Guide*, v 2.0, Xilinx Inc., San Jose, CA, 2005.

[19] *PowerPC 405 Processor Block Reference Guide*, v 2.3, Xilinx Inc., San Jose, CA, 2008.

[20] *Xilinx ISE Software Manuals and Help*, ISE 9.2i, Xinlinx Inc., San Jose, CA, 2007.

[21] M. Renovell, J. M. Portal, J. Figures and Y. Zorian, “Testing the Interconnect of RAM-Based FPGAs,” *IEEE Design and Test of Computers*, vol. 15, no.1, pp. 45-50, 1998.

[22] I. Harris and R. Tessier, “Interconnect Testing in Cluster-Based FPGA Architectures,” *Proc. AMC/IEEE Design Automation Conf.*, pp. 49-54, 2000.

[23] I. Harris and R. Tessier, “Diagnosis of Interconnect Faults in Cluster-Based FPGA Architectures,” *Proc. IEEE International Conf. on Computer Aided Design*, pp. 472-476, 2000.

[24] M. Abramovici, J. Emmert, and C. Stroud, “Roving STARs: An Integrated Approach to on-Line Testing, Diagnosis of FPGAs In Fault-Tolerant Applications, ” *Proc. International Test Conf.*, pp. 973-982, 1999.