System-on-Chip Testability Using LSSD Scan Structures

Kamran Zarrineh
Sun Microelectronics

Vivek Chickermane
IBM Microelectronics

Shambhu J. Upadhyaya
State University of New York at Buffalo

A technology-independent test synthesis tool extends the basic level-sensitive scan design (LSSD) boundary scan methodology. It reuses functional storage elements wherever possible and introduces minimal test logic overhead and delay.

■ Testing densely packaged very large-scale integration (VLSI) circuits has become challenging. Embedded memories and reusable cores have become common because they reduce the design time to market for complex systems. Scan design is the most commonly practiced approach to enhancing design testability. This approach lets design storage elements be configured into shift registers during test mode, thereby enhancing the logic's controllability and observability. When a design has numerous embedded macros, there are two ways to improve the overall testability:

- Make the embedded macro use the same scan design methodology as its surrounding logic. This allows simultaneous testing of the surrounding and embedded logic using the same test methodology.
- Isolate the embedded logic and test it in a separate test phase using a separate test methodology from that of the surrounding logic.

Designing an embedded macro to be merged and tested with the rest of the design is clearly preferable but not always feasible. Memory arrays such as static RAMs, dynamic RAMs, ROMs, and so forth, are examples of macros that are extensively used but seldom designed to be mergeable. Wherever these macros are embedded, they can be isolated during test with a set of scannable cells. These scan cells are connected to form one or more scan chains around each embedded macroblock. The generated scan chains allow the control and observation of the embedded macroblock's I/O ports by using a few chip-level test ports.

An integrated system can often have a set of embedded macroblocks that depend on the state or response of another embedded macroblock in a given design. Isolating and independently testing each embedded macroblock is not desirable since it introduces unacceptable signal delays in this type of configuration. Using different design for test (DFT) methodologies to satisfy test requirements results in different test protocols and an increase in test cost.

To test such integrated systems using scan methodology, DFT engineers use a scan structure that works within the rules of an adopted test methodology. For instance, a new usage of level-sensitive scan design (LSSD) scan structures would enhance the testability of random logic trapped between embedded macroblocks in a design. The scan structures should be designed so a macroblock's latched I/O ports are reused and the available functional storage elements are transformed to obtain the desired scannable elements. This flexibility results in a

0740-7475/01/$10.00 © 2001 IEEE

large test area reduction. Embedded memories, cores, and chips on multichip modules and boards can also benefit from such unified, flexible DFT methodology, which allows testing of the entire system with a single test protocol.

The LSSD boundary scan approach uses the LSSD test protocol.[1,2] In LSSD, storage elements are designed as shift register latches (SRLs). A shift register is designed either as a double-latch configuration for functional flip-flops or as a single-latch configuration for transparent latches. A double- or single-latch shift register consists of a master and slave latch. The difference between a double- and a single-latch shift register is that the double latch's functional output port is on the slave latch. In contrast, the single latch's functional output is on the master latch.

The master latch is a two-port latch with one data port and one scan data port. A data clock (conventionally called the C clock) activates the data port . A shift clock called A clock activates the scan port; B clock activates the slave clock. These clocks are derived from the oscillator using a clock splitter. During system operation, the clock splitter generates a C clock pulse during the off phase of the system oscillator and a B clock pulse during the oscillator's on phase. This permits the SRL to behave like an edge-triggered flip-flop or a transparent D latch. In test mode, the clock splitter permits control of the SRL from three system primary inputs: LSSD_A, LSSD_B, and LSSD_C clock ports. During the scan shift operation, nonoverlapping A and B clocks enable the scan data's safe shifting. The test data is scanned in through the master latch's scan port and scanned out of the slave latch's data output. During the test mode's system cycle phase, the B clock is used to launch the test data from the slave latch. A subsequent C clock captures the test response in all the SRLs.

The boundary scan methodology

- supports the ability to test a chip with a tester that contacts only a small percentage of chip I/Os (boundary cells are inserted between the uncontacted pins and the system logic they feed or are fed by);
- permits interconnect test of boards and systems consisting of chips that support some form of boundary scan; and
- isolates a chip from the board in which it is embedded when its internal logic is being tested by chip-specific scan patterns or self-test.

Boundary cells are built in one of two ways. In high-performance chips, it is common to place inline latches between the functional logic and chip I/Os. The inline latches can be transformed to scannable latches to form boundary scan cells. If inline latches are not available, then boundary cells are added to exclusively support test. These cells are transparent during functional operation.

While LSSD boundary scan provides one approach, the IEEE Std 1149.1 boundary scan methodology is also commonly used.[3] This standard's methodology uses a four- or five-pin test access port (TAP) to control and access the boundary scan registers. While IEEE 1149.1 boundary scan is very attractive for board-level test, it has a significant overhead that makes it less attractive than LSSD boundary scan for embedded cores and memories. Furthermore, LSSD boundary scan provides capabilities such as reduced pin count and I/O wrap test methodologies for testing the I/Os and the chip interconnects. A compromise is to use LSSD boundary scan for the embedded macros and an LSSD-compatible IEEE 1149.1 methodology for chip interconnect testing.[4] This permits effective use of LSSD for all chip-internal tests and hybrid IEEE 1149.1 for all chip-external tests.

The insertion of LSSD boundary scan structures is labor intensive and requires expertise. The process often requires

1. identification and transformation of functional storage elements to scannable SRLs,
2. design and insertion of the scan-only cells necessary to satisfy the test requirements,
3. insertion and transformation of I/O cells, and
4. design and insertion of the necessary logic for I/O optimization.

A test synthesis tool based on extended LSSD boundary scan automatically inserts the necessary LSSD scan structures for embedded memories, cores, and chips. The tool uses several multipurpose scan structures, which are designed as technology-independent blocks (TIBs). For technology-independent designs, the inserted TIBs are synthesized with the functional logic, which results in a better optimized and timed design. The functional storage elements that enhance the testability of the macro blocks and/or their surrounding logic are determined and automatically transformed into an appropriate scan element. Since boundary scan structures are inserted around the chip's I/O cells, a technology-independent approach (I/O management) is needed to insert these I/O cells on the chip boundary.[5] This test synthesis tool has been developed and implemented in the IBM TestBench system and has been used to insert scan structures in several IBM designs.

Enhancing system-on-chip testability

We review techniques for enhancing the testability of embedded memories, cores, and chips using the LSSD boundary scan methodology.

Enhancing embedded memory testability

Embedded memory arrays can be divided into *dependent* and *independent* memories.[6] If an embedded memory's data, address, or control signals are supplied by another embedded memory, it is called a dependent embedded memory; otherwise it is independent. The embedded memory that supplies the necessary data to dependent memories is the source, and the embedded memory that receives the data is the sink. LSSD boundary scan can enhance the testability of both independent and dependent embedded memories and their surrounding logic.

For independent embedded memories, this DFT methodology isolates and enhances the testability of each embedded memory. The controllability and/or observability of each latched port are enhanced by transforming its functional storage elements to scannable cells, and scan-only cells are inserted for unlatched ports. The

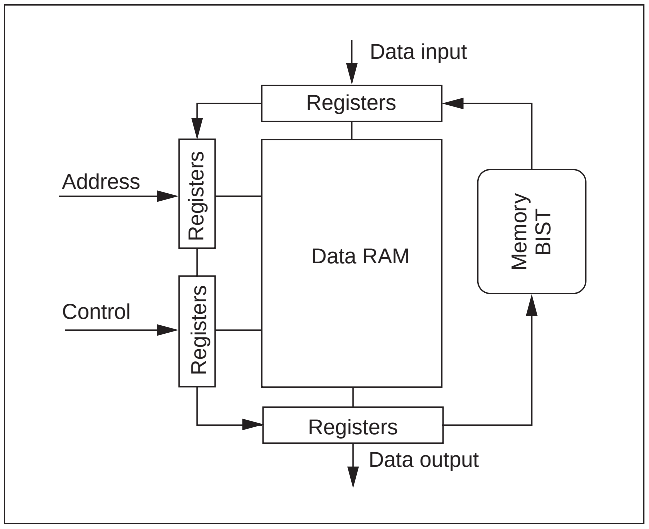


Figure 1. Testing an embedded memory through shared scan chains.

test patterns and addresses are loaded into the scan cells of the data input and address ports of the embedded memory. The scan cells inserted around the embedded-memory control ports act as a gate that lets read and/or write operations occur. The embedded memory's response to the test data is captured in the data output scan cells and is observed via a scan-out operation. The added scan cells can be chained together to create a scan chain, as shown in Figure 1. The size of the embedded memory that can be tested using this method is limited because the necessary data are loaded serially via the scan chain. We can test a larger memory by creating three or more separate scan chains consisting of data I/O, address, and control scan cells.

Loading and unloading embedded-memory test patterns via a tester is inefficient because of the large number of test patterns. A scan-based embedded-memory built-in self-test (BIST) unit is designed to load the necessary data, address, and control signals, and compare the embedded memory's response with the input data patterns, using the LSSD test protocol.

Dependent embedded memories (most macroblocks, in general) pose two additional test problems: accumulative delay and test area overhead due to scan structures added to each embedded block and pockets of random logic trapped between these memories. One approach to testing this embedded memory

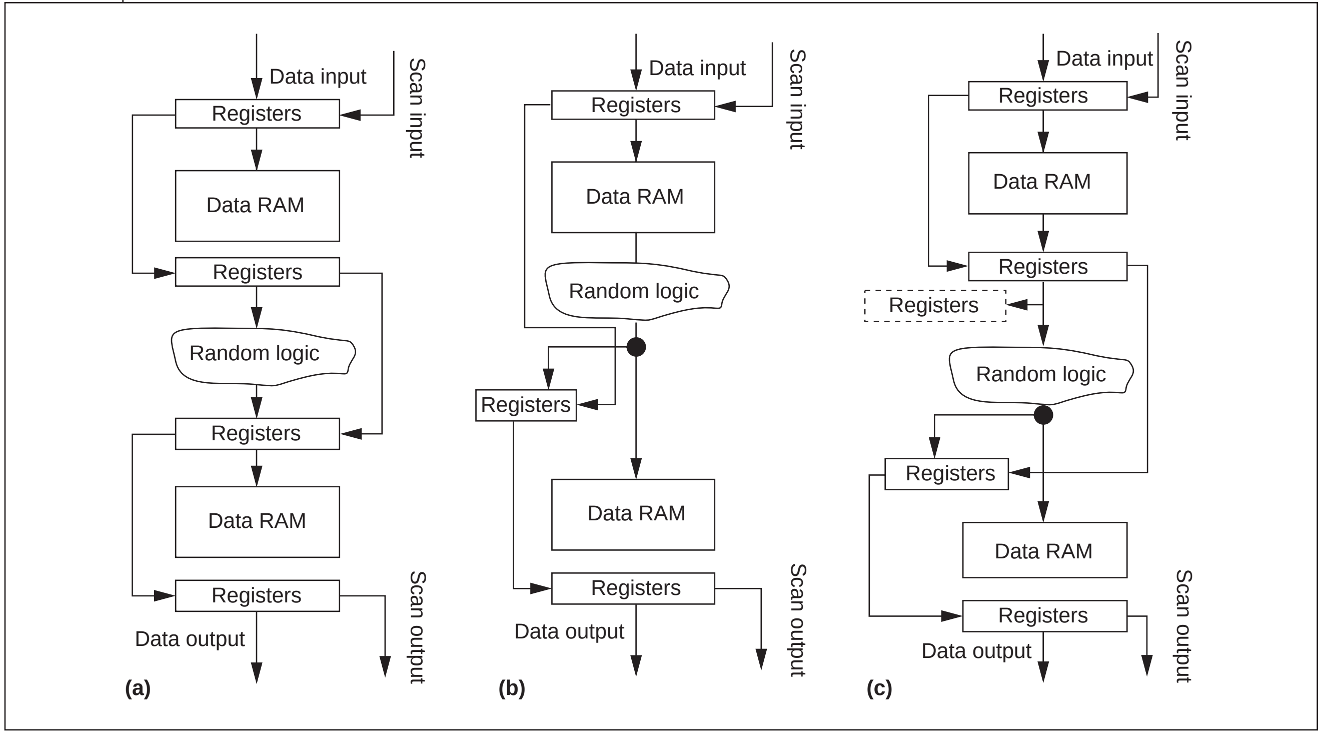


Figure 2. Testing dependent memories where memory blocks have both controllability and observability (a), only controllability and enhanced observability (b), and controllability and enhanced observability (c).

type is to isolate and test each embedded memory independently using scannable elements, as described earlier. The inserted scan structures also enhance the surrounding logic's testability, as shown in Figure 2a. If the pocket of random logic's behavior is fully known, DFT engineers can test the entire dependent-memory structure in a way similar to testing a sequential circuit.

One disadvantage of this approach is that it cannot locate the failed component. To overcome this problem, a set of observation-only pseudo-output cells can be added to the sink embedded memory's input ports, as shown in Figure 2b. To further enhance the dependent embedded memory's testability and diagnostic capability, storage elements at the source embedded memory's output can be transformed to scannable elements or a set of scan-only latches, as shown with a dotted line in Figure 2c. If necessary, functional storage elements in the random logic pockets between sources and embedded memory sinks can also be transformed to scannable elements to further enhance testability. The memory BIST unit can control and observe each embedded memory using a functional I/O subset of the embedded memory. Thus several scattered embedded memories in the design can share one memory BIST unit, incurring only a very low wiring overhead.

Enhancing embedded-core testability

Embedded cores—based on a description such as register transfer-level (RTL) and netlists—could be divided into soft, firm, and hard cores.[7,8] Soft cores are the most flexible and transparent to users; hard cores, the least. Furthermore, embedded cores that can be absorbed by their surrounding logic are called *mergeable*; those that cannot be absorbed are *unmergeable*.[9]

Recently, researchers have proposed several methods for testing embedded cores. We describe and compare selected methods with the LSSD boundary scan test methodology.

Marinissen has proposed a structured and scalable mechanism for providing test access to embedded reusable cores.[10] This method consists of a TestShell, an intellectual property (IP) core wrapper; TestRail, a mechanism to transport the data to the IP; and a test control mechanism (TCM) to control TestShell operation. Users can scale TestShell and TestRail based on test requirements and the design's topology. The core's test protocol is translated into the TCM.

Our method also permits scaling the number of test pins to best satisfy design test requirements, as was done in previous work.[10] However, in our method appropriate functional registers are reused to minimize test logic overhead. Furthermore, we use one test protocol to test the entire chip, making translation unnecessary.

Whetsel has proposed a scan test architecture that provides test connectivity, communication, and embedded-core control within the system ICs.[11] This scan architecture divides the scannable elements into parallel scan distributors (PSD), parallel scan collectors (PSC), and a parallel scan path (PSP). The PSD and PSC provide a more efficient method for application of test data to the PSP than using an external tester. This architecture lets designs use shorter scan chains, decreasing power dissipation and reducing test time.

Our method can be configured to form the PSPs of appropriate length, then the PSD and PSC units can be satisfied by a pseudorandom pattern generation (PRPG) and a multiple-input signature register (MISR) if necessary. Lowering the test clock rate or creating shorter scan chains can also achieve low-power testing.

DFT engineers divide the testing of embedded cores into testing

- the internals of the core or testing by applying a set of precomputed test patterns at the next higher integration level,
- the interconnections, and
- the user-defined logic or the random logic around a core.[8]

Tests patterns could be applied to the I/Os of a core in parallel. Parallel access has a high routing penalty and wiring overhead; serial access has a higher test time and might introduce arbitrary switching during the scan operation. Parallel access' routing penalty and wiring overhead increase as designs become denser. A reasonable addition of test logic overhead can improve penalties associated with serial access without wiring overhead of parallel access method; therefore, serial access method is used for test pattern application. Not all test data can be loaded via the scan-in operation. Therefore, from a test perspective, the core ports can be divided into ports that should be controlled in parallel—called *parallel test ports* (such as test clock and other control signals)—and ports that can be accessed in series, or *serial test ports* (such as functional data ports). If necessary, functional data ports can be added to parallel test ports and accessed accordingly.

To enhance the core internal-logic's testability, full-scan DFT methodologies such as LSSD can be used. This methodology identifies and transforms the core's storage elements into scan structures.[12] Then, based on the storage elements' clock domains, allowable number of scan elements per scan chain, or their physical location, the transformed scan structures are connected to form one or more scan chains. For unmergeable cores, the core designer can perform this process and a set of test patterns can be provided with the core. For mergeable cores, users perform DFT activities and compute the necessary test patterns.[9]

In either case, the necessary test patterns should be loaded into a set of scan structures surrounding the core. Integrating mergeable cores with user-defined logic in system-on-chip (SOC) designs might not always be feasible because of restrictions on the core, such as power consumption. The test complexity and the merged-design size might require partitioning the design and solving each partition's test problem separately.

I/O core ports are usually buffered with functional latches to synchronize the data. For mergeable cores, these functional latches can be potentially transformed and reused as test structures, providing the necessary controllability and observability of ports with minimum overhead. Any functional storage element that is not on a fan-out branch around an embedded core can be selected and transformed into a scannable element. The logic on the embedded core's boundary has to be analyzed to find the most suitable functional storage element for

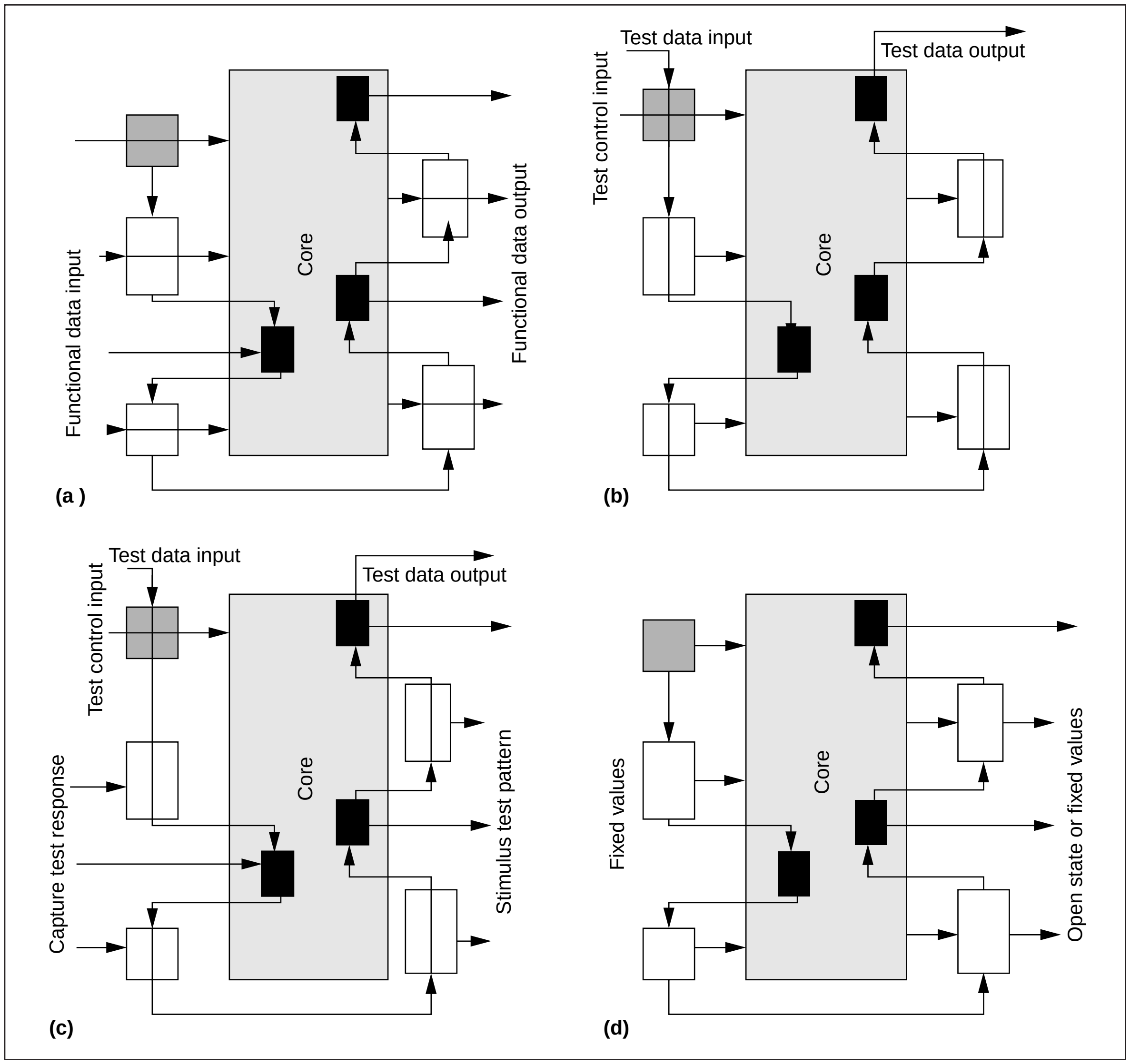


Figure 3. Embedded core DFT logic for functional (a), internal (b), and interconnect (c) testing. DFT logic can also isolate the core (d).

transformation and reuse. In the case of unlatched I/Os on critical paths, input ports can be gated (if not already buffered with internal receivers), which adds negligible delay. These input ports cannot be used during test mode. Similarly, using scan-only cells can enhance output port observability.

In addition, to prevent the core under test's test activity from being disturbed by or disturbing other cores, LSSD ensures that each core can be tested in isolation. Test isolation requirements are also necessary to protect low-power cores from accidental damage during test mode and to hold the core's state for I_{DDQ} testing. In this case, dedicated scan structures have the logic to provide the necessary isolation. For transformed functional storage elements, the necessary logic is added around these elements to satisfy the isolation requirement.

The different test modes of the inserted and transformed scan structures that enhance a core's testability are shown in Figure 3. Observation-only scan structures inserted to enhance the testability of parallel test ports are shown at the core's top left corner. Reference is made to the gray box. White boxes represent the scan structures inserted to enhance an unbuffered port's controllability and observ-

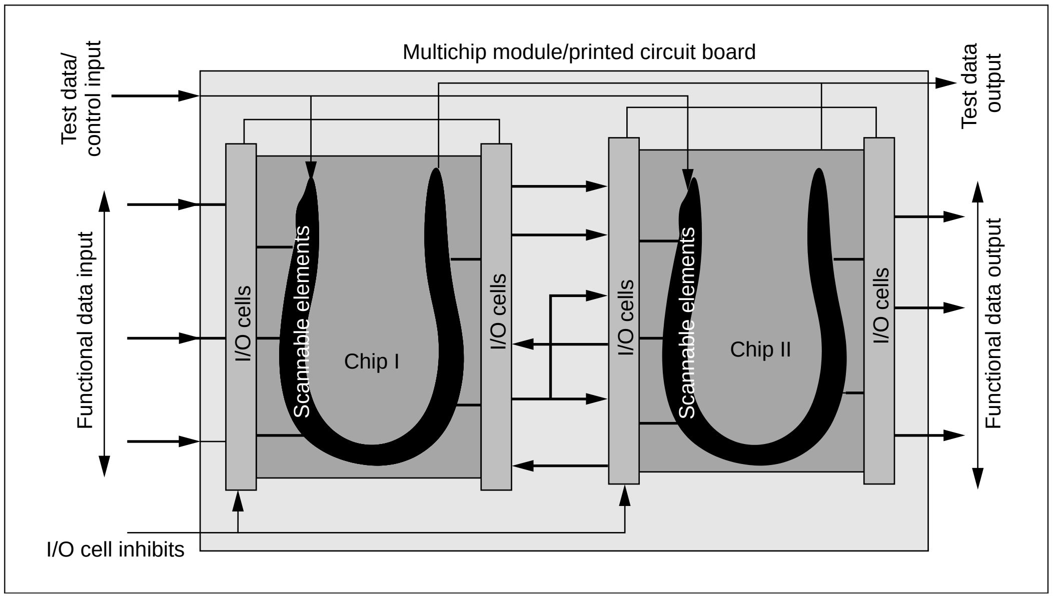


Figure 4. Testing the internal and external parts of a chip using boundary scan methodology.

ability. Black boxes represent transformed functional latches. Figure 3a shows the functional mode of the core's operation. In this mode, the functional path is active and data passes through the inserted test logic. In test mode, shown in Figure 3b, precomputed test data are scanned to test the core under test's internal logic while the functional paths are deactivated. In Figure 3c, the test data to test the interconnection of the embedded cores are loaded in the scan structures. In the last mode of operation, shown in Figure 3d, the core under test is isolated and therefore the input ports are at a fixed logic value while the core's output ports are fixed to a specific logic value or are at a high-impedance state.

Enhancing chip testability

The LSSD test methodology enhances chip testability by

- enhancing chip interconnect testing,
- providing access for I/O cell parametric tests, and
- providing controllability and observability to a chip's I/Os.

Although both LSSD and IEEE 1149.1 methodologies enhance the chip testability, there are fundamental differences between them.[3] LSSD is a simpler protocol, because it does not require using an internal test controller and an instruction-driven test protocol. It also permits different scan configurations and test modes, which permits testing of the test logic as well as functional logic. LSSD boundary scan allows reuse of functional storage elements as boundary scan cells where possible. The test protocol can accommodate more complex boundary scan structures to test a chip's I/O cells without contacting them with the tester probe.

A hybrid boundary scan method has LSSD's advantages and is IEEE 1149.1 compliant. It has been used for many LSSD-based designs, as described previously.[4] This hybrid method involves generating appropriate LSSD clocks based on IEEE 1149.1 clock/control signals. For example, the hybrid controller has to derive shift clocks (A and B) and capture clocks (C), based on the IEEE 1149.1 TCK clock.

Suppose chips I and II are interconnected and reside on a board as shown in Figure 4. The test-only transformed scan structures are shown as the black horseshoe objects in both chips. To test each chip's internal logic, the necessary test data is loaded via the test data/control input port while the functional data ports are disabled. The response of each chip's internal logic is captured by its pseudo-outputs and observed via a test data output port. Since the surrounding scan cells from other chips on the board isolate these chips during internal test mode, all chips can be tested simultaneously. For testing the interconnection of chips I and II, the stimuli patterns are applied from the pseudo-output of chip I, and the response is captured by the pseudo-input of chip II. Furthermore, to protect the chips from unsafe test patterns, each chip's I/O cells have disable or inhibit circuitry. An inhibit receiver and an inhibit driver signal are connected to the corresponding I/O cells in series. The series configuration lets the I/O cells turn on or off one at a time if the I/O cell's inhibit signals are activated. These I/O cells will be at a fixed value of high impedance, which creates the necessary isolation for the chips, especially on the output side.

Random-logic built-in self-test

Random-logic BIST can be easily implemented using LSSD structures.[13] The most commonly used LSSD-based random-logic BIST method is called Stumps, which stands for self-test using MISR and a parallel SRSG (shift register sequence generator).[14] In the Stumps architecture, a design's scannable storage elements are serially connected to form short shift registers or Stumps channels. Each Stumps channel contains no more than a predefined number of scannable storage elements (512, for example) to ensure that the test time is practical. When test mode begins, the PRPG, MISR, and channels are initialized with appropriate values using the LSSD test protocol. The Stumps test protocol alternates between loading and unloading the channels and capturing the circuit's response. The test data is generated by the PRPG based on its initial state; the test data is then loaded in each Stumps channel in parallel. The MISR generates a signature based on the Stumps channel's shifted-out contents. At the end of test mode, the MISR contents are compared with the expected signature.

To ensure that the process is repeatable for a fault-free design and that it produces a predictable signature, the design under test's I/Os must be isolated. This can be achieved by using boundary cells for isolation and to launch or capture test data into or from the application logic.

For unmergeable cores, the inserted Stumps architecture is controlled from a test controller embedded in user-defined logic. The isolation logic ensures a complete separation of the core from user-defined logic during Stumps mode. For mergeable cores, the PRPG, MISR, and the Stumps test controller are removed from the core while leaving the channel connections intact. User-defined logic and the core are merged during this test mode, and the core's channels become a subset of the encompassing Stumps channels. The overall design's test logic overhead is a minimum if the encompassing logic also uses LSSD test methodology.

Test synthesis methodology

Our test synthesis tool can process a hierarchical design or work on a flattened design. For a hierarchical design, isolation logic is first added to appropriate instances of the embedded-cores processing entity. Next, memory BIST macros are generated and inserted to test any embedded memories in the entity being processed or to test lower-level entities if necessary. Each entity's functional storage elements are transformed to scannable elements based on a given test methodology. A statistical testability analysis is performed to identify random-pattern-resistant logic pockets and insert scan structures to improve the testability of these logic pockets. In addition, appropriate test I/O pins are added to the entity interface and the necessary connections are made.

Once all entities are processed, the appropriate LSSD test ports are added to the top-level entity. These test I/Os could be shared with functional I/Os to reduce test cost. The test synthesis tool adds the necessary test logic based on the test and functional port types.[15] For example, sharing a test clock with a functional input requires additional test logic. An I/O management scheme inserts, transforms, and chains the I/O cells (pads) for the I/O ports of the top-level entity, which is from a technology-independent description. The I/O cells are necessary to satisfy the LSSD boundary scan requirements.[5]

If required, a random-logic BIST controller is added, and the muxes to form the Stumps channels are created. Boundary scan test logic is added to enhance the testability of the top-level entity (the chip), and all test logic connections are made. The core isolation methods we described earlier are used to isolate hard cores in the design.

A design undergoes different types of test during its fabrication and assembly. The semiconductor manufacturing site defines these tests. Each test type is defined in a test mode file that the automatic test pattern generation (ATPG) framework uses to compute appropriate test patterns. The first step in test pattern generation is ensuring the correct structure and connection of the additional test logic for each defined test mode. Therefore, for each test mode, the inserted test logic's correctness is verified through a test structure verification process. This step could also evaluate the design's testability.

The macro structure verification (MSV) process verifies the isolation logic surrounding the embedded cores and the ability to control and observe the core's I/Os. This uses a hierarchical process, which lets a user specify different controllability and observability requirements on the core's I/Os to satisfy different test modes. Controllability and observability requirements could be the initialization and activation patterns of a BIST macro inside a hard core or the patterns necessary to shut-off a core during the test, that is, ground the output of phase-locked loops during I_{DDQ} test. Controllability and observability requirements could also be used to specify the patterns necessary to test the core. Furthermore, the MSV and macro test generation (MTG) processes take user-specified information regarding the core's test order to produce appropriate test patterns. These test patterns could be loaded in the tester by themselves or as a part of the test patterns for the entire chip. Once this step is complete, test patterns for the entire design are generated. Figure 5 (next page) shows the described 15-step method.

The test synthesis tool can be invoked in two different design scenarios, *early* and *late* modes.[13] In early mode, the scan structure insertion occurs just after high-level synthesis but prior to logic synthesis and technology mapping.[16] The test structures are inserted using technology-independent primitives, optimized and timed together with functional logic. Late mode works with technology-dependent insertion, where the design is fully technology-mapped at the netlist level. This requires that the test logic be inserted and optimized without affecting the already-synthesized logic.

Our test synthesis tool's flexibility supports the design of almost any type of LSSD scan structures to satisfy different test requirements. The most commonly used cells in this DFT methodology are interception, observation-only, muxed, and I/O wrap scan cells.

Interception scan cells

Interception scan cells are used to enhance a port's controllability and observability. Any functional latch that resides on the boundary of the embedded memory, core, or chip and represents a root (input ports) or a sink (output ports) of any fan-out can be transformed into an interception cell. The transformation algorithm extracts and considers the functional storage elements' characteristics to ensure correct system behavior in functional mode after scan cell insertion.[13] The functional data I/O nets of the functional storage element are connected to their corresponding data I/O of double- or single-latch SRLs. In addition, the power-on initialization of the functional storage elements can be done via the scan path. This design practice reduces the amount of wiring in the system. Our test synthesis tool extracts the specified power-on initialization sequence from the design description and adds the necessary inversions to the scan path. These inversions let the design be initialized by asserting the scan input port to logic value 0 and asserting 1 at the A and B clocks, or a predefined time. For ports where functional storage elements are not available, transparent scan-only (*flushed*) SRLs, are inserted in the interception configuration. Figure 6a(page 93) illustrates an interception SRL.

Observation-only scan cells

Observation-only SRLs only enhance a

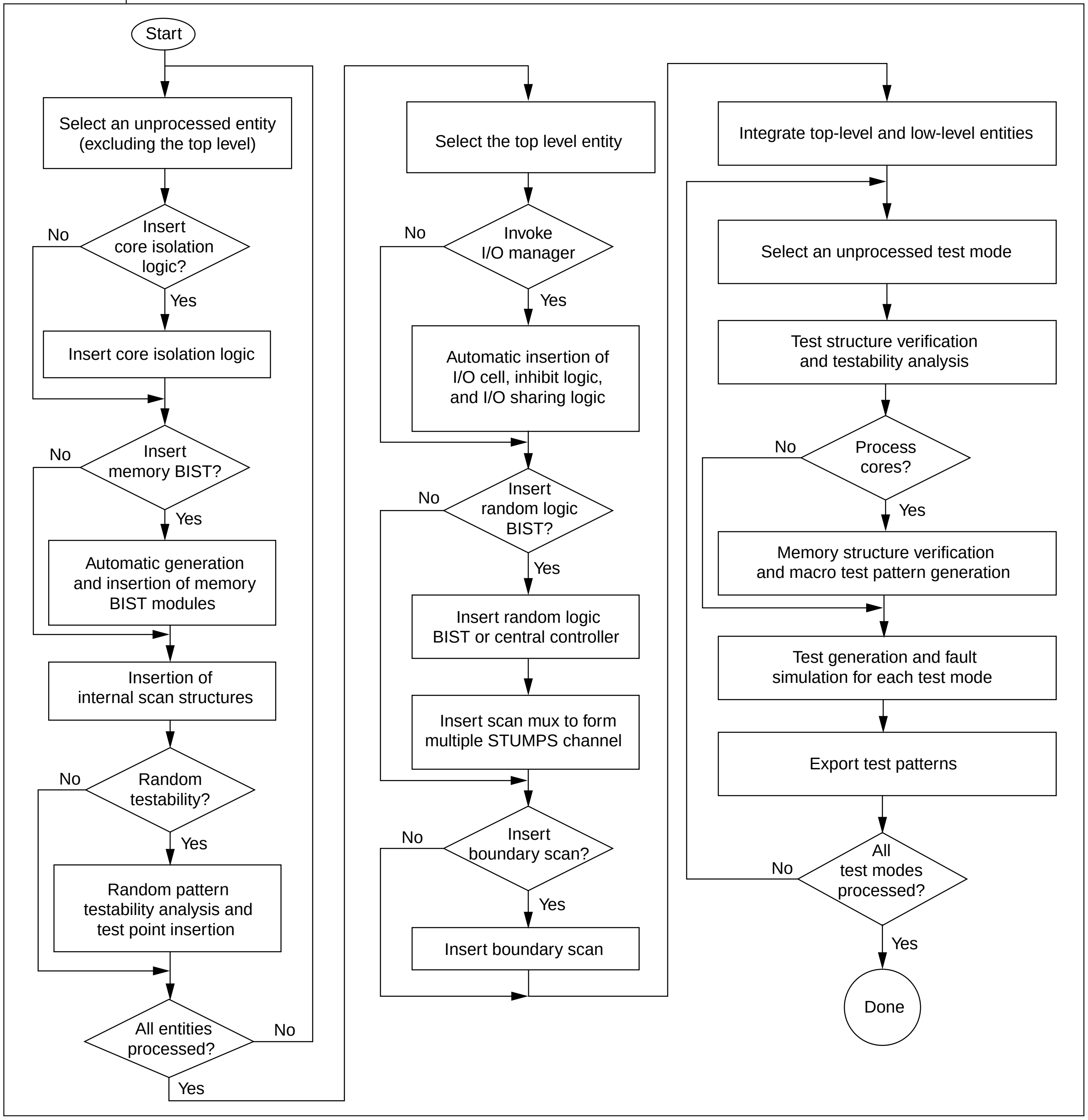

Start
Select an unprocessed entity (excluding the top level)
Insert core isolation logic?
No
Yes
Insert core isolation logic
Insert memory BIST?
No
Yes
Automatic generation and insertion of memory BIST modules
Insertion of internal scan structures
Random testability?
No
Yes
Random pattern testability analysis and test point insertion
All entities processed?
No
Yes
Select the top level entity
Invoke I/O manager
No
Yes
Automatic insertion of I/O cell, inhibit logic, and I/O sharing logic
Insert random logic BIST?
No
Yes
Insert random logic BIST or central controller
Insert scan mux to form multiple STUMPS channel
Insert boundary scan?
No
Yes
Insert boundary scan
Integrate top-level and low-level entities
Select an unprocessed test mode
Test structure verification and testability analysis
Process cores?
No
Yes
Memory structure verification and macro test pattern generation
Test generation and fault simulation for each test mode
Export test patterns
All test modes processed?
No
Yes
Done

Figure 5. Flow of the described test synthesis tool.

port's observability. These SRLs allow capturing and observing the signal activities on a port with negligible delay overhead. The function of the port whose observability needs to be enhanced determines the connection of the observation-only SRL. For example, to enhance the C clock port's observability, connect it to the TIB SRL's C clock port. Similarly, to enhance the data port's observability, connect it to the data port of an observation-only SRL, as shown in Figure 6b. The design tool extracts information about the port from the design description. Our tool extracts information about the chip I/O function from the design description and then inserts the TIB SRL and makes the appropriate connections.

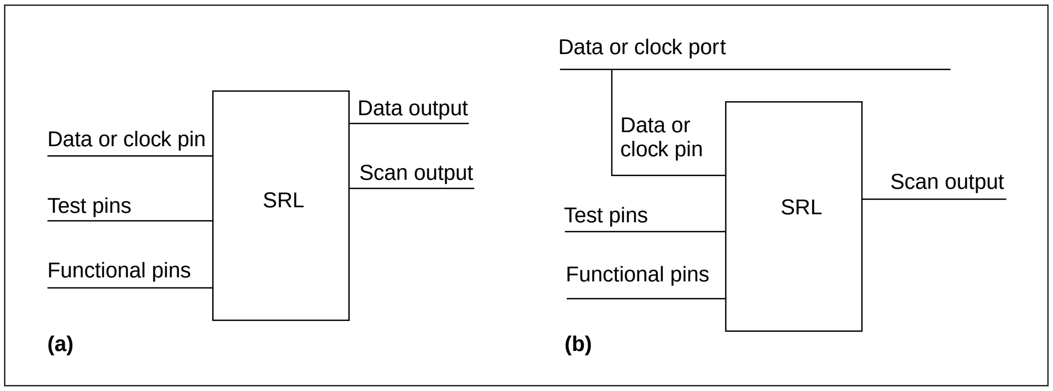


Figure 6. TIB SRL in interception (a) and observation-only (b) configurations.

Muxed SRL cells

Muxed SRLs can serve as an alternative to interception scan cells to enhance an unlatched port's controllability and observability. Figure 7 is a technology-independent representation of the I/O interface and structure of a muxed SRL. The data I/O and clock ports are the cell's system ports; the rest are test ports. The control signals in muxed SRLs determine the cell's mode of operation. Pseudo-input mode disables the data input port, and the SRL acts as the input to the functional path. In pseudo-output mode, muxed SRLs can capture signals on the functional path.

The correct insertion of a muxed SRL requires correctly identifying each functional port and the test requirements. Figure 8 (next page) illustrates two modes of operation for muxed SRLs. For use with embedded memories and cores, one muxed SRL is inserted for each port. To test a chip's boundary, one muxed SRL is inserted for each port of the I/O cells. A common muxed SRL could be used to control a common enable signal of a set of tri-state drivers. Therefore, our test synthesis tool inserts only one muxed SRL at the source of the enable signal fan-out node. In early mode test synthesis, the inserted logic is first transformed into its specified components and then optimized with the rest of the logic. If the multiplexer in the muxed SRL cannot be optimized with its surrounding logic and the target technology supports muxed SRLs, the inserted muxed TIB SRL will be directly mapped to a technology-specific muxed SRL cell. Otherwise, its components (AND gates, OR gates, and the SRL) are mapped to their corresponding technology cells.

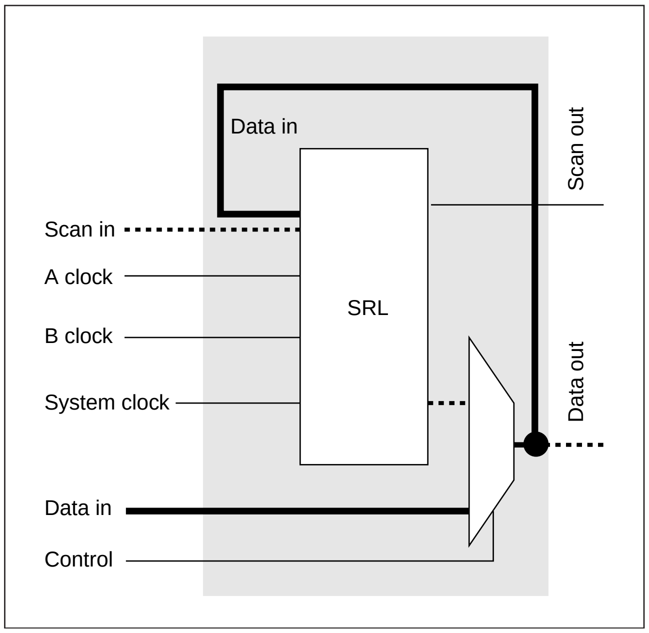


Figure 7. Muxed SRL represented in TIB, pseudo-input, and pseudo-output.

I/O wrap scan cells

I/O wrap scan cells are used for both static and delay testing of the I/O cells without physically contacting their pads with the tester probes.[17] Basically, an I/O wrap scan cell launches test data out of the I/O cell and then captures the response. This is possible if the I/O cells of all noncontacted pads are bidirectional in test mode.

I/O wrap scan cells must maintain normal behavior of the I/O cells in functional mode. Correct insertion of I/O wrap scan cells requires identifying a port's functionality—whether it is a receiver, is a two- or three-state driver, or is

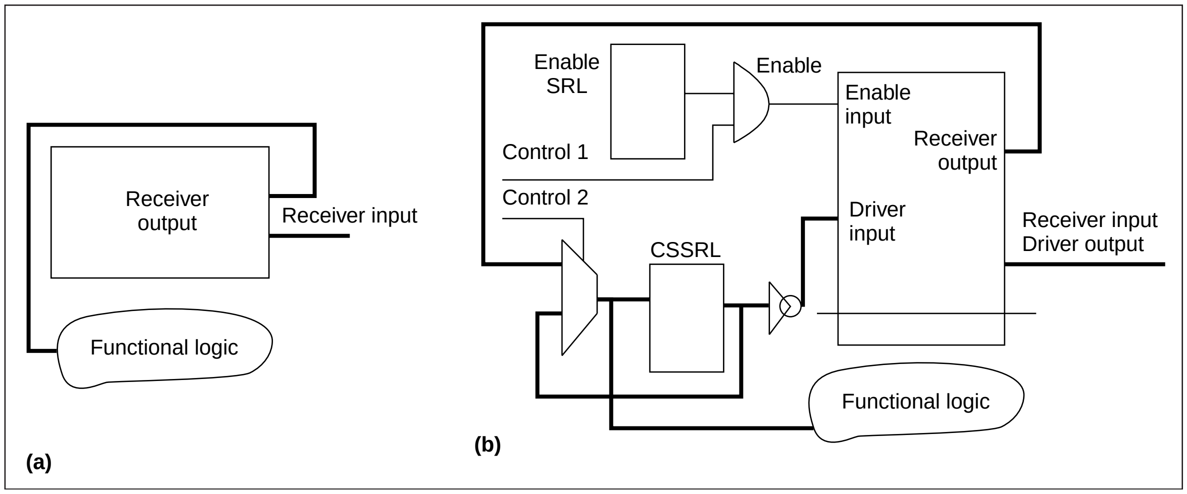


Figure 8. Functional receiver (a) with an I/O wrap scan cell can operate in functional, I/O wrap test, and control functional logic (b) test modes.

bidirectional—and this is not possible unless the design description includes all the I/O cells.

I/O cells can be inserted from a technology-independent description, which supports design reuse across different technologies.[5] Suppose I/O wrap capability needs to be added to the functional receiver, as shown in Figure 8a. If the I/O receiver is technology independent, our test synthesis tool directly adds the necessary pins to the I/O receiver TIB and transforms it to a bidirectional I/O cell. A technology-mapped I/O receiver is unmapped and the necessary ports are added to its TIB representation. Then, the transformed TIB is remapped to an appropriate technology I/O cell. Once this step is successfully completed, an I/O wrap scan cell for the functional receiver is inserted, which results in the circuit shown in Figure 8b.

During functional mode, control$_1$ and control$_2$ are set to logic value 0. This turns the I/O cell driver off and selects the data on the I/O cell's receiver output port, as shown in Figure 8b. In test mode, control$_1$ is set to logic value 1, and the scan-only enable SRL (E-SRL) controls the I/O cell's enable port. The master and slave latches of the capture/stimulus SRL (CS-SRL) act as the capture and stimulus latches. By setting control$_2$ to logic value 0, the value loaded in the CS-SRL's slave latch launches the stimulus data to the I/O cell. The inverter ensures that the stimulus and captured data are different. Some internal scan test methodologies (weighted and random-pattern testing) require that I/O receivers be set to a known value during test. This requirement is satisfied by setting control$_2$ to 1 and letting the value loaded in CS-SRL control the functional logic.

A library of I/O wrap scan cells were designed for functional receivers, two and three-state drivers, and common I/Os or bidirectional I/O cells by using our test synthesis tool's primitives. During logic synthesis, these I/O wrap scan cells are flattened and their primitives are optimized and mapped to the target technology with the functional logic.

Experimental results

TestBench is IBM's test generation and DFT synthesis software. Its test generation capability supports different variations of internal scan design, such as LSSD and mux-scan. It also supports different types of boundary scan, including different types of LSSD and IEEE 1149.1 methodologies. It includes application programs to generate static and dynamic tests for logic and embedded cores. Engineers have used TestBench to generate test patterns for test methodologies such as stored-pattern logic, weighted random pattern, built-in self-test, scan, interconnect, I/O cell, and I_{DDQ}.[18–20] It also has diagnostics to help debug the design. Test synthesis provides test logic insertion capability for internal scan, boundary scan, and random logic BIST methodologies; TestBench also automatically inserts I/O cells.[5,13,21] In addition, TestBench's test structure verification system ensures the correct structure and functioning of inserted test logic.[6]

LSSD boundary scan has been implemented in TestBench's design-for-test-synthesis (DFTS) application. To quantify the speed-up gains from DFTS, we ran experiments using LSSD boundary scan insertion with five small VHDL designs. These designs had bus sizes of four to 64 bits. The experiments were done on a 150-MHz PowerPC desktop workstation with an AIX 4.2 operating system. We assumed that the I/O cells of five VHDL designs are successfully inserted. We used DFTS to insert muxed scan structures for a set of designs with 50 to 710 I/O ports and 30 to 390 functional flip-flops.

Table 1. LSSD boundary scan insertion for circuits using a muxed SRL line.

Circuit name	No. of I/Os	No. of functional registers	No. of expected registers	No. of actual registers	Time (s)
P4	50	30	48	33	1.9
P8	94	54	92	67	3.6
P16	182	102	180	132	7.3
P32	358	198	356	255	17.5
P64	710	390	708	610	48.0

In the experiment's first part, an LSSD boundary scan structure with muxed scan cells was inserted. For the experiment's second part, we used I/O wrap scan cells. To determine the minimum and maximum logic overhead for LSSD boundary scan, we reused the appropriate functional latches as boundary scan cells. In the second part, the functional registers were hidden from DFTS. The minimum area overhead in LSSD boundary scan is somewhat ambiguous since in one design, all core and chip I/Os could be registered, while in another design no registers are logically close to the core and chip boundary. Although both cases could reuse all registers as LSSD boundary scan cells, the latter case affects the interconnect test coverage.

In this experiment, we consider functional latches whose logical location makes them good candidates for reuse as LSSD scan cells, provided the location of the sequential elements did not affect the I/O interconnect test coverage. I/O optimization helps minimize the LSSD methodology's I/O overhead.[15] However, boundary-scan-dedicated C, A, and B clock ports have been added to the circuit for both muxed and I/O wrap boundary scan methodologies. The tester has to contact a minimum of seven I/O ports. These ports are scan input, scan output, A clock, B clock, C clock, *I/O cell inhibit*$_1$, and *I/O cell inhibit*$_2$ ports. Table 1 summarizes the total number of I/Os, functional buffers, expected and actual registers, SRLs, and test logic insertion time. Expected registers are registers necessary for the design based on port type. Actual registers are test-only registers inserted in the design. Only one scan cell for the common enable signal and one observation scan-only cell for the functional clock have been added. We observe the following:

- Using functional latches as boundary scan cells reduces the number of inserted SRLs by 18 to 20%. Sharing the enable latches also contributed to an overall reduction in the test logic overhead. For example, the LSSD methodology using muxed SRLs for P32 with 352 outputs requires 356 boundary scan cells. The actual number is 255 registers because we reuse functional latches whenever possible.
- The LSSD boundary scan test logic was added to the circuit in less than one minute for P64 (710 ports).
- By inserting the test logic early in the design cycle—at RTL—the logic optimization and timing correction was performed on the entire design. Therefore, the resulting netlist is a better optimized and timed design.

In the experiment's second part, we added I/O wrap scan structures. Although DFT methodology lets us use functional registers in place of the dedicated, test-only scan structures in the I/O wrap scan methodology, this experiment does not reuse functional registers as scan structures. Table 2 summaries our results.

Table 2. I/O Wrap boundary scan insertion.

Circuit name	No. of I/Os	No. of functional registers	No. of expected registers	No. of actual registers	Time (s)
P4	50	30	137	137	1.8
P8	94	54	269	269	3.3
P16	182	102	533	533	6.6
P32	358	198	1,061	1,061	14.7
P64	710	390	2,117	2,117	37.0

We can extend the LSSD boundary scan methodology beyond its original intended use—testing internal and external chip logic on boards—to enhance the testability of embedded memories, embedded cores, and their surrounding random logic. All scan structures are connected together to form a set of long (maximum allowable length) scan chains. Each scan chain can be divided into multiple short scan chains, enhancing the testability and diagnosability of different macroblocks, such as embedded memories and cores. The scan chain configuration depends on the design's test mode. Regardless of the test mode, the scan structures are level sensitive and the entire design can use the LSSD test protocol.

The work described here has been adopted by several IBM ASIC design groups to fully automate their SOC DFT methodologies. It has also evolved into the IOSpecDFT tool that the IBM ASIC organization supplies to its worldwide design centers for push-button insertion of top-level I/O and test structures.[22] IOSpecDFT also offers a high degree of customization necessary for supporting the complexities of leading edge ASIC designs. ■

Acknowledgment

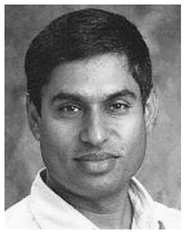
This work was supported and developed at IBM, Endicott, N.Y.

■ References

1. R.W. Bassett et al., "Boundary Scan Design Principles for Efficient LSSD ASIC Testing," *Proc. Design Automation Conf.,* IEEE CS Press, Los Alamitos, Calif., 1977, pp. 462-468.
2. E.B. Eichelberger and T.W. Williams, "A Logic Design Structure for LSI Testability," *J. Design Automation and Fault-Tolerant Computing*, vol. 2, 1978, pp. 165-178.
3. *IEEE Std 1149.1a-1993*, *Test Access Port and Boundary-Scan Architecture*, IEEE Press, Piscataway, N.J., 1993.
4. S.F. Oakland, "IEEE 1149.1 Boundary Scan in IBM CMOS ASICS," *IBM ASICS Application Note SA14-2282-04*, IBM Microelectronics, Essex Junction, VT, Nov. 1997.
5. K. Zarrineh and V. Chickermane, "A DFT Perspective on I/O Management," *Proc. Int'l Conf. Computer Design*, IEEE CS Press, Los Alamitos, Calif., 1996, pp. 46-53.
6. L. Ternullo Jr. et al., "Deterministic Self-Test of a High-Speed Embedded Memory and Logic Processor Subsystem," *Proc. Int'l Test Conf.*, IEEE CS Press, Los Alamitos, Calif., 1995, pp. 33-44.
7. Y. Zorian, "Test Requirements for Embedded Core-Based Systems and IEEE P1500," *Proc. Int'l Test Conf.*, IEEE CS Press, Los Alamitos, Calif., 1997, pp. 191-199.
8. A.M. Rincon et al., "Core Design and System-on-a-Chip Integration," *IEEE Design & Test of Computers*, Oct.-Dec. 1997, pp. 26-35.
9. R.K. Gupta and Y. Zorian, "Introducing Core-based System Design," *IEEE Design & Test of Computers*, Oct.-Dec. 1997, pp. 14-25.
10. E.J. Marinissen et al., "A Structured and Scalable Mechanism for Test Access to Embedded Reusable Cores," *Proc. Int'l Test Conf.*, IEEE CS Press, Los Alamitos, Calif., 1998, pp. 284-293.
11. L. Whetsel, "Core Test Connectivity, Communication and Control," *Proc. Int'l Test Conf.*, IEEE CS Press, Los Alamitos, Calif., 1998, pp. 303-312.
12. V. Chickermane and K. Zarrineh, "Addressing Early Design-For-Test Synthesis in a Production Environment," *Proc. Int'l Test Conf.*, IEEE CS Press, Los Alamitos, Calif., 1997, pp. 246-255.

13. M.P. Kusko et al., "Microprocessor Test and Test Tool Methodology for the 500 MHz IBM S/390 G5 Chip," *Proc. Int'l Test Conf.*, IEEE CS Press, Los Alamitos, Calif., 1998, pp. 717-726.
14. P.H. Bardell, W.H. McAnney, and J. Savir, "Built-In Test for VLSI: Pseudo Random Techniques," Wiley Interscience, New York, 1987.
15. S.F. Oakland et al., "An ASIC Foundry View of Design and Test," *Proc. Int'l Test Conf.*, IEEE CS Press, Los Alamitos, Calif., 1994, pp. 865-868.
16. W.H. Joyner et al., "Technology Adaptation in Logic Synthesis," *Proc. Design Automation Conf.*, ACM Press, New York, 1986, pp. 94-100.
17. P. Gillis et al., "Delay Test of Chip I/Os Using LSSD Boundary Scan," *Proc. Int'l Test Conf.*, IEEE CS Press, Los Alamitos, Calif., 1998, pp. 83-92.
18. R. Kapur et al., "Design of an Efficient Weighted Random Pattern Generation System," *Proc. Int'l Test Conf.*, IEEE CS Press, Los Alamitos, Calif., 1994, pp. 491-500.
19. B.L. Keller and T.J. Snethen, "Built-In Self Test Support in IBM Engineering Design System," *IBM J. Research and Development*, Mar.-May 1990, pp. 406-415.
20. S. Chakravarty and P.J. Thadikaran, *Introduction to I_{DDQ} Testing,* Kluwer Academic Publishers, Boston, Mass., 1997.
21. P.S. Gillis et al., "Test Methodologies and Design Automation for IBM ASICs," *IBM J. Research and Development*, vol. 40, no. 4, July 1996, pp. 461-474.
22. V. Chickermane et al. "Automated Chip-Level I/O and Test Insertion Using IBM Design-For-Test Synthesis," *IBM MicroNews*, vol. 6, no. 2, May 2000.

Kamran Zarrineh is in charge of design for test issues for the UltraSparc V microprocessor at Sun Microelectronics in Chelmsford, Mass. His research interests include design and test of VLSI systems. Zarrineh has a BS in electrical engineering from the University of Utah, an MS in computer science from the State University of New York at Binghamton, and a PhD in electrical engineering from the University at Buffalo. He is a member of the IEEE.



Shambhu J. Upadhyaya is an associate professor of computer science and engineering at the University at Buffalo. His research interests include VLSI testing, fault diagnosis, fault-tolerant computing, information assurance techniques, and diagnostic reasoning. Upadhyaya has a PhD in electrical and computer engineering from the University of Newcastle, Australia. He is a senior member of the IEEE.

Vivek Chickermane leads the design for test synthesis project at the IBM Test Design Automation group in Endicott, N.Y. His research interests include testing, synthesis, and verification of VLSI systems. Chickermane has a BTech in electrical engineering from the Indian Institute of Technology, New Delhi, and an MS and a PhD in electrical engineering from the University of Illinois, Urbana-Champaign. He is a member of the IEEE.

■ Direct questions and comments about this article to Kamran Zarrineh, Sun Microsystems, DFT Architecture, 5 Omni Way, Chelmsford, MA 01824; zarrineh@east.sun.com.