Improved Core Isolation and Access for Hierarchical Embedded Test

Benoit Nadeau-Dostie, Saman M.I. Adham, and Russell Abbott
LogicVision

Editor's note:
IEEE Std 1500 enables automation and hence allows for easier and faster integration of embedded cores into an SoC. This article describes an automated test development system based on the concept of embedded test.
—Erik Jan Marinissen, IMEC

IEEE Std 1149.1 is a widely accepted testability standard that has been in use since the early 1990s.[1] Although the standard was primarily focused on board-level assembly verification testing via the boundary scan register, its test access port (TAP) and many optional provisions make it usable for a broader range of applications. Since its inception, researchers have proposed numerous extensions and applications that allow designers to use the standard's TAP for general system-level test and maintenance tasks and for accessing chip-level testability features.

Until the late 1990s, chip-level applications used TAP for accessing a chip's scan design or on-chip BIST features. However, it became apparent that increasing chip complexity would force new design and test methods. On the design side, circuits are now partitioned into blocks to facilitate synthesis and layout. Iteration turnaround times can be maintained at a reasonable level. Concurrent engineering enables several design teams to develop the various blocks in parallel. Some preexisting blocks can be reused, especially for standard functions, such as the PCI bus interface. Finally, several circuits exploit the architecture's regularity by using several instances of the same block.

These new design methods have introduced completely new test constraints.[2,3] Unlike ICs, cores can't be manufactured and tested individually. Direct access to core ports is virtually impossible. The problem is similar to testing a fully assembled printed circuit board (PCB) without having previously tested the individual components. Test and diagnosis of such a board is possible only if each IC comes with a standard test interface such as IEEE Std 1149.1, which has internal and external test modes. IEEE P1500 was proposed to similarly address the testing of core-based circuits. The internal test information is encapsulated in the form of scan test vectors or BIST procedures, and no design knowledge is necessary. This is especially useful when the core is provided by a third party. Each core's external test information makes it possible to test the connections and logic between cores.

In this article, we describe a new hierarchical embedded-test methodology developed at LogicVision for core-based designs. The wrapper serial port (WSP) of IEEE Std 1500 is at the heart of the test architecture.[4] Each core is equipped with a WSP and controlled by the top-level TAP to provide access to all core test controllers. An improvement to the standard, called *shared isolation*,[5] makes it possible to significantly reduce core isolation costs while enabling at-speed testing of core interface signals. Our methodology provides a fixed core test interface regardless of the number of test resources under its control. The star architecture allows verification and testing of cores independently, enabling concurrent engineering and better diagnostic resolution.

Hierarchical test architecture

Figure 1 shows an example IC using LogicVision's hierarchical test methodology. This simple circuit

0740-7475/09/$25.00 © 2009 IEEE Copublished by the IEEE CS and the IEEE CASS

has three cores; two of them contain logic and memories, and the third contains logic and serializer and deserializer inputs and outputs. Typically, each core corresponds to a physical region—that is, a portion of the circuit that is laid out separately. In this simple example, all circuitry is located in the cores. Therefore, the top-level functional circuit description contains only the core instances and their interconnections.

The root of the test control hierarchy is the TAP implemented according to IEEE Std 1149.1. Each core's WSP is at the second level of the hierarchy, which is implemented according to IEEE Std 1500. Finally, each WSP controls all the test controllers of the corresponding core. These controllers constitute the last level of the test hierarchy. A test controller can implement BIST, test compression, or functional debug functions. Debug functions can be as simple as a test data register or as complex as an on-chip logic analyzer.

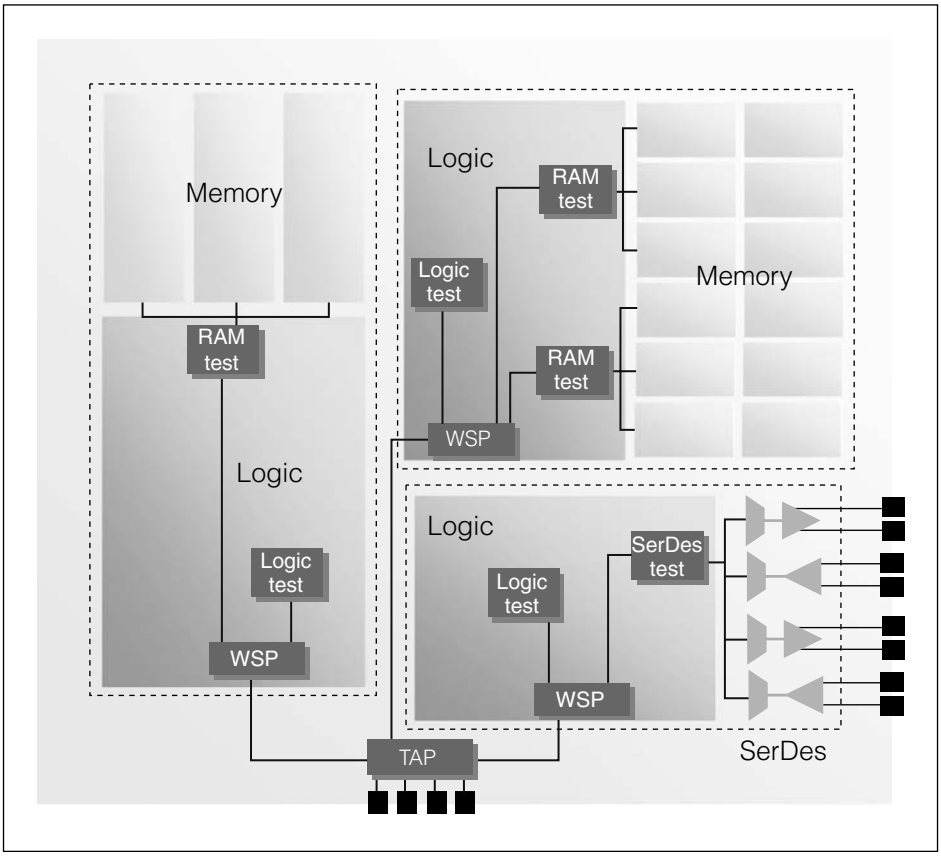

Figure 1. Hierarchical test architecture. (SerDes: serializer and deserializer; TAP: test access port; WSP: wrapper serial port.)

In this case, test controllers are sometimes referred to as *instruments*. One of the scan chains can be the wrapper boundary register (WBR) mandated by IEEE Std 1500. This scan chain controls the core inputs and observes the core outputs during an internal test. Conversely, during an external test, it controls the core outputs and observes the core inputs. The standard requires that each input and output have a flip-flop associated with it, to provide control and observation functions.

A star configuration controls WSPs and test controllers (TCs) at all levels of the hierarchy. Figure 2 shows a more detailed view of how we connect WSPs and TCs to the top-level TAP or a parent WSP. Any number of WSPs and TCs can be connected to addressable test ports. The serial input is shared by all test ports, but each test port has a dedicated enable (*enTP1* to *enTP5*) and serial output (*fromTP1* to *fromTP5*). WSPs share a set of control signals (*CTL_WSP*), whereas TCs share a different set (*CTL_TC*). There are currently seven control signals in each set. (The *CTL_WSP* and *CTL_TC* control signals are documented elsewhere.[4,6])

We selected a star configuration, as opposed to a daisy-chain one, for various reasons, from both a design and test perspective. In a star configuration, each core can be exercised and verified independently from other cores in the context of the entire circuit without having to modify the testbenches or vectors. This is an advantage in a concurrent design environment, where the models (such as RTL or gate-level netlists) of other cores might not be available or are constantly changing because of different test trade-offs. It also makes it possible to add entire cores without changing the method used to access test resources in other cores. From a test perspective, a single issue in one of the daisy chain's WSPs would prevent access to all cores, complicating the silicon debug process enormously. These two reasons alone justify the two dedicated wires required for each test port.

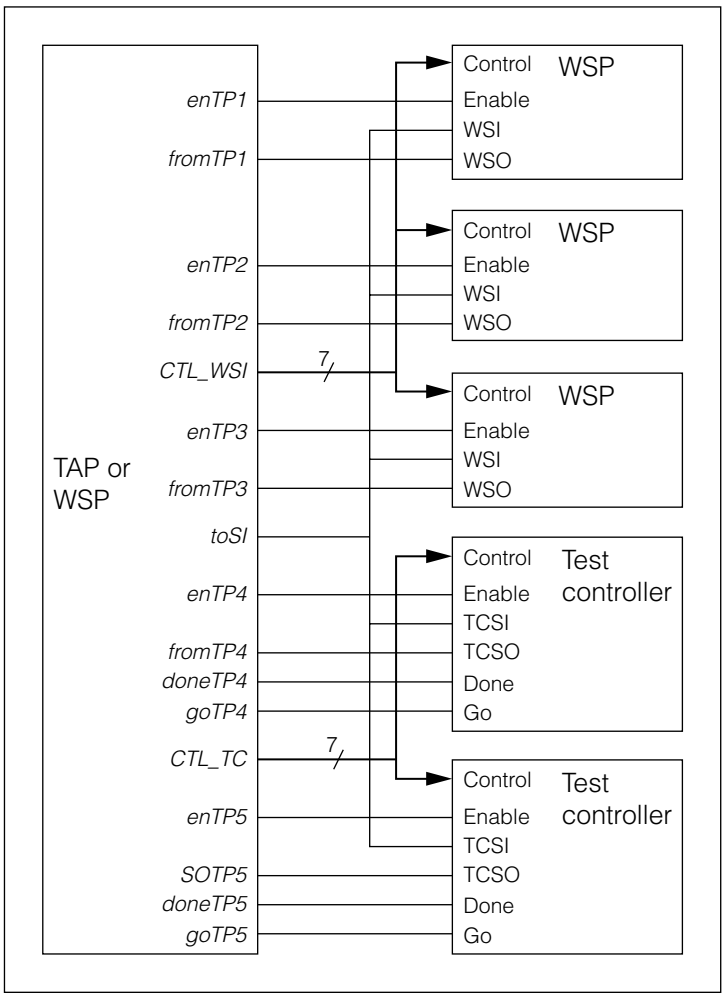

Figure 2. Test port connections. (TC: test controller; TCSI: test controller serial input; TCSO: test controller serial output; TP: test port; WSI: wrapper serial input; WSO: wrapper serial output.)

One of the main advantages of the architecture we have described is that the test interface of each core is completely independent of the number and type of its test resources. It allows defining the interface early in the design process, at the RTL, and this interface remains constant throughout the physical-design process. It also allows simultaneously running test controllers in different cores using different test parameters under the control of the selected core's instruction. Some blocks can even continue to operate in functional mode while others are in test mode.

Our WSP interface is identical to the one described in IEEE Std 1500 except for two signals: the enable (*enTPn*) and a control signal (*extTM*), which is part of the *CTL_WSP* bus, configuring the core in an external test mode. When those signals are tied to a logic high and logic low, respectively, then the interface is identical to the standard one. The enable signal is essential to implement the star configuration. The enable gates the standard signals *CaptureWR*, *UpdateWR*, and *ShiftWR* inside each core so that the cores can be independently accessed. This gating must be done inside, not outside, the core. Several design methodologies do not allow discrete logic gates to appear at the top level of the circuit or core. An alternative is to perform the gating inside the TAP or parent WSP. However, this approach requires two additional wires for each test port, which unnecessarily increases the routing congestion around the TAP or parent WSP.

The *extTM* control signal makes the WBR accessible, either as a single or multiple segments, at the core boundary so that it becomes part of the scan test of the next level up. This eliminates the need to load an instruction (`WS_EXTEST` or `WP_EXTEST`) in each WSP at the beginning of the scan test. There is no provision in IEEE Std 1500 to standardize the binary code of the `WS_EXTEST` instruction selecting the external mode. This is different from IEEE Std 1149.1, which mandates a binary code of all 1s for the `EXTEST` instruction selecting the *boundary scan register*, which is the WBR's chip-level counterpart. The *extTM* control signal also prevents a malfunction in any of the WSPs from interfering with the scan test of the next level up, which is useful from a diagnostics perspective.

Core isolation

Shared isolation is an improvement to IEEE Std 1500 that can minimize the amount of test-dedicated circuitry required for core isolation and enable at-speed testing within and between cores.[5]

Each core requires an isolation mechanism—that is, a boundary register that controls its inputs and observes its outputs during internal tests—as well as control of its outputs and observation of its inputs during external tests. Figure 3 shows examples of cells that serve to implement this isolation. These cells are called *dedicated-isolation cells* because their circuitry is dedicated to implement test functions only; they don't have a functional mode of operation. During an internal test, the *intTM* signal is asserted, and the *extTM* is de-asserted. The core inputs are controlled by the input flip-flops of the boundary register and the core outputs observed by the output flip-flops. During an external test, the *extTM* signal is

asserted, and the *intTM* signal is de-asserted. The core outputs are controlled by the output flip-flops and the core inputs observed by the input flip-flops. The scan-enable signal is controlled by *intSE* during the internal test, and by *extSE* during the external test. During the functional mode of operation, all test control signals are de-asserted, and the input and output cells are transparent. Also, the cells' capture mode is disabled to avoid unnecessary switching of the input and output flip-flops and false timing violations. The capture mode can be disabled by gating the clock or the data input of the flip-flops.

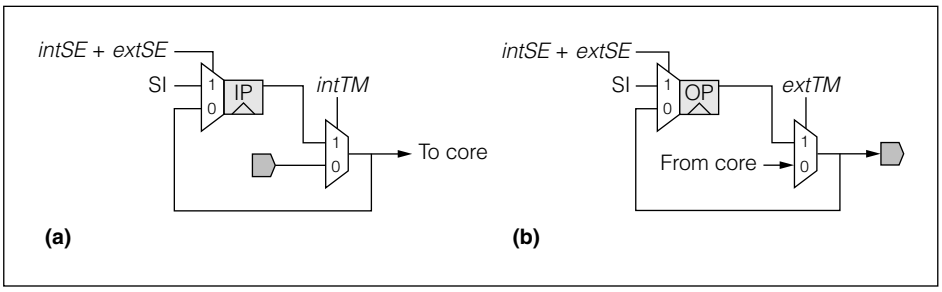

Figure 3. Dedicated-isolation cells: input cell (a) and output cell (b). (SE: scan enable; SI: scan input; TM: test mode.)

The advantage of these cells is that they do not require any knowledge of the function, their insertion is easy to automate, and the core model is easy to generate for the next level of circuit hierarchy because both the inputs and outputs are registered during test. These cells constituted our first generation of isolation cells.[7]

However, there are two major disadvantages to dedicated isolation cells. First, the amount of test-dedicated circuitry is considerable for most cores. For example, large cores with hundreds and even thousands of input and output pins are not rare. Second, it is impossible to perform an accurate at-speed test for the first layer of functional flip-flops located in the fan-out of input pins or the fan-in of output pins. This is because the path from a source-functional flip-flop of a first core to a destination-functional flip-flop of a second core is segmented into three overlapping parts:

- source flip-flop to output cell,
- output cell to input cell of second core, and
- input cell to destination flip-flop.

Overcoming these limitations requires using functional flip-flops located near the core inputs and outputs to implement the isolation. Figure 4 shows examples of input and output shared-isolation cells; the flip-flops are shared by the functional and test modes. During the scan-insertion process, functional flip-flops are substituted with conventional scanable flip-flops, but the scan-enable input is a function of *intSE*, *extSE*, *intTM*, and *extTM*. The functionality is exactly the same as for the dedicated-isolation cells. However, there is a negligible amount of test-dedicated circuitry, and functional paths between cores can be tested at speed.

It is rare that all inputs and outputs of a core are registered. At best, some design methodologies mandate registration of either outputs or inputs, but not necessarily both. Although it is advantageous to use functional flip-flops that have their inputs or outputs directly connected to a core input or output, respectively, and to use dedicated isolation cells for the rest, our concept of shared isolation does not restrict functional flip-flops to be directly connected to core inputs and outputs. Our software automatically identifies the first layer of functional flip-flops that can be used to implement the boundary register to minimize the number of dedicated-isolation cells.

Figure 5 shows a small core with seven inputs and five outputs and the 10 functional flip-flops forming the boundary register. Some of the flip-flops are directly connected to core inputs and outputs (IP1, IP3, OP1, and OP4). However, there are others that are connected through combinational logic (IP2, IP4, IP5, OP2, and OP3). The algorithm that classifies the flip-flops into input and output cells is performed in two phases:

1. Identify input cells, which are all flip-flops with a core input in their fan-in.

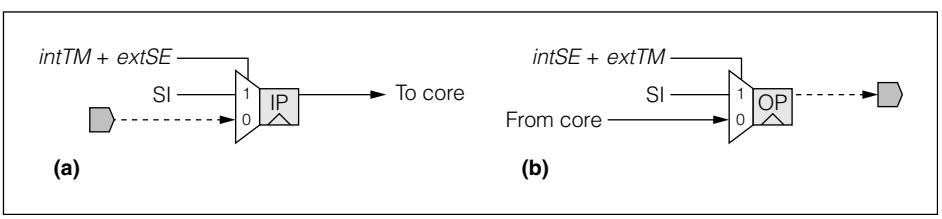

Figure 4. Shared isolation cells: input cell (a) and output cell (b).

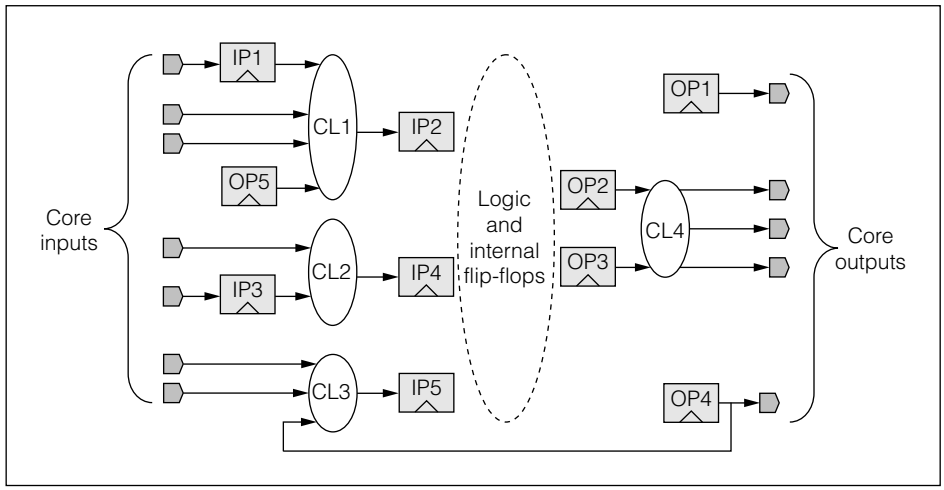

Figure 5. Shared isolation. Identification of isolation flip-flops and shell model. (CL: combinational logic.)

2. Identify output cells, which are all flip-flops that are not input cells but have either a core output or an input cell in their fan-out.

In Figure 5, OP5 is a special case because it has no core input and output in its fan-in or fan-out, and yet it is part of the boundary register because it is needed to control all inputs of combinational logic CL1 during the external test. CL2 through CL4 are also tested during this external test. A shell model is automatically extracted and saved in the database associated with the core. The model considerably simplifies the parent module's test generation because it only includes the WBR and the logic between the core inputs and outputs and the WBR.

The shared-isolation technique does not completely eliminate the need for dedicated-isolation cells. It is preferable to use those cells for core inputs with a large fan-out. For example, a reset input can fan out to all the core's flip-flops. In that case, all functional flip-flops are identified as input cells, and the shell model contains the entire core, which of course defeats the purpose of the hierarchical method. This is why the software automatically identifies all core inputs with large fan-out and sorts them in descending fan-out order. We can easily identify the few inputs that require dedication isolation, because they will have hundreds or thousands of gates in their fan-out. Typically, enough dedicated isolation cells are inserted to cover most of the core logic (for example, more than 90%) during internal test. The remaining logic is tested as part of the parent module's scan test.

Figure 6 shows a simplified chip that contains two embedded cores (A and B) and some logic at the top (CL2, CL5, and one flip-flop). CL1 and CL4 constitute the logic between the boundary register and the core inputs and outputs that was not tested during core A's internal test. The same is true for core B's CL3 and CL6. During the chip top-level test, cores A and B are put in their external test mode by asserting their

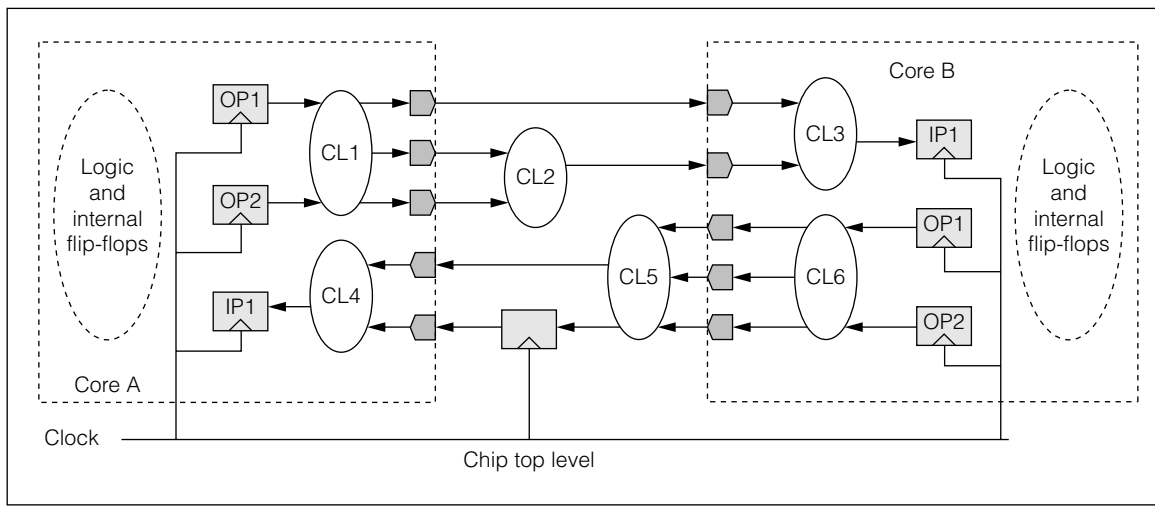

Figure 6. Chip top-level view.

extTM control signal, and their boundary registers become part of the top-level scan chains. Because all the logic is bounded by functional flip-flops operating from a common clock, it is possible to apply an accurate at-speed test in ATPG, compression, or logic BIST modes. Figure 6 shows different possible paths between the two cores' boundary register or between the cores and the top-level logic.

It is impossible to achieve the same timing accuracy if strictly enforcing the rule of IEEE Std 1500, which does not allow logic to be located between the core inputs and outputs and the boundary register. Paths need to be tested in two or three segments and require manipulation of the clock period. Also, dedicated-isolation cells must be added, increasing area. For example, a path from core A's OP1 to core B's IP1 is tested in three segments when not using shared isolation: core A OP1 to core A DI (dedicated isolation), core A DI to core B DI, and core B DI to core B IP1. Because each path segment is incomplete, a shorter clock period is needed during the scan test's capture cycle. This period is different for a core's different path segments and requires testing the path segments in several steps, complicating the test and increasing test time. Even if all segments are measured by varying the clock period, some difficult calibration is necessary to account for the DI cells' setup time and their clock-to-output time, which are not part of the paths to be measured. In the particular case in which the cores have registered inputs and outputs—in our example, core A OP1 would be directly connected to an output, and core B IP1 would be directly connected to an input—the timing paths are measured accurately. However, for the core to be compliant with the standard, all inputs and outputs would need to be registered, which is extremely rare.

Shared- and dedicated-isolation cells are automatically configured to enable at-speed test of circuits with multicycle and cross-domain paths for internal and external modes.[8] Because we support a launch-off-shift scan test methodology, both the external and internal scan-enable signals are pipelined in each clock domain within the core.

Implementation examples

We can illustrate the benefits of shared isolation using two cores with different characteristics. Core 1 is relatively small, with a relatively high number of registered inputs and outputs and no high fan-out inputs. Core 2 is large, with relatively few registered inputs and outputs and several high fan-out inputs. For each core, we evaluated three trade-off scenarios (cases A, B, and C), which show the influence of the various core characteristics on the results:

- *Case A*. Dedicated-isolation cells are used for all core inputs and outputs. This is the most straightforward method for core isolation, but it also the most expensive.
- *Case B*. Shared-isolation cells are used for inputs and outputs that are connected directly (or through buffers) to functional flip-flops. This method requires slightly more sophisticated software support, but it improves the at-speed coverage of paths sourced from or terminated at the core inputs and outputs. Also, it yields the best implementation that is supported by IEEE Std 1500.
- *Case C*. The use of shared-isolation cells is maximized, and dedicated-isolation cells are used only for high fan-out core inputs such that at least 90% of the logic is tested during the core internal test. The rest of the logic is tested during the parent module's test (core or chip top level).

Tables 1 and 2 show the results for cores 1 and 2, respectively. The cores with all their testability features—WSP, logic test controller, scan chains, test points (if using logic BIST), and memory BIST controllers—have been synthesized to gates for all cases. In the technology we used, a flip-flop was equivalent to 7.5 gates. As expected, the maximum use of shared-isolation cells (case C) for core 1 yields an overall gate reduction of 6.3% and 1.6% when compared to cases A and B, respectively. The small reduction from case B to case C is due to core 1's relatively high number of registered inputs and outputs (approximately 75%). This percentage is often far smaller, and the gate reduction achieved by case C is much better. Even so, a gate count reduction of 1.6% is still significant for a high-volume chip. There is a similar gate-count reduction (1.5% and 1.2%, respectively) for core 2.

For cases A and B, IEEE Std 1500 requires that all faults be tested during internal test mode. In case C, we relaxed this requirement to enable at-speed testing of the core interface logic. The tables show that 4% (core 1) to 5% (core 2) of the logic is tested during the cores' external test. Only each core's shell model is required to perform test generation and fault simulation. As we explained earlier, proper

Table 1. Core 1 results.

	No. of gates			No. of faults			No. of cells in boundary register		
Case	**Combinational**	**Sequential**	**Total**	**Internal**	**External**	**Total**	**Dedicated**	**Shared**	**Total**
A	71,521	112,362	183,883	414,310	0	414,310	894	0	894
B	66,857	108,864	175,720	414,310	0	414,310	227	667	894
C	65,269	107,673	172,942	397,792	16,518	414,310	0	902	902
C – A (absolute)*	–6,252	–4,689	–10,941	–16,518	16,518	0	–894	902	8
C – A (%)*	–9.6	–4.4	–6.3	–4.0	4.0	0	–6.2	6.3	0.1
C – B (absolute)*	–1,587	–1,191	–2,778	–16,518	16,518	0	–227	235	8
C – B (%)*	–2.4	–1.1	–1.6	–4.0	4.0	0	–1.6	1.7	0.1

* C – A and C – B represent the reduction (or increase) in the number of gates, faults, or boundary cells achieved by case C compared to case A and case B, respectively, in absolute and relative terms. All percentages are expressed relative to the corresponding value of case C.

at-speed testing of interface logic is impossible for cases A and B.

For core 1, the total length of its boundary register does not vary significantly, so there is no impact on the parent module's scan chain organization, test time, or data. However, the elimination of dedicated isolation cells also reduces the number of flip-flops to scan during the internal test by 894 (about 6.2% compared to case A), which helps reduce test data by a similar amount.

More important, all paths to and from the boundary register can be tested at speed during external test. For core 2, only 210 dedicated-isolation cells are needed in the boundary register out of the 1,972 for case A and 1,656 for case B, which explains the reduction in the total number of gates we previously indicated. The small number of dedicated-isolation cells means that the large majority of the paths can be tested at speed in external mode. This small number also results in a slight reduction in the number of flip-flops to scan in internal mode (1.7% compared to case A, and 1.4% compared to case B).

The greatest impact on the external mode is a higher number of flip-flops in the boundary register: 4,819 (case C) instead of 1,972 (cases A and B). However, the amount of logic in external mode is usually

Table 2. Core 2 results.

	No. of gates			No. of faults			No. of cells in boundary register		
Case	**Combinational**	**Sequential**	**Total**	**Internal**	**External**	**Total**	**Dedicated**	**Shared**	**Total**
A	812,104	784,784	1,596,888	3,036,565	0	3,036,565	1,972	0	1,972
B	809,951	783,233	1,593,184	3,036,565	0	3,036,565	1,656	316	1,972
C	798,667	775,106	1,573,772	2,885,050	151,515	3,036,565	210	4,609	4,819
C – A (absolute)*	–13,437	–9,678	–23,116	–151,515	151,515	0	–1,762	4,609	2,847
C – A (%)*	–1.7	–1.2	–1.5	–5.0	5.0	0	–1.7	4.5	2.8
C – B (absolute)*	–11,284	–8,127	–19,412	–151,515	151,515	0	–1,446	4,293	2,847
C – B (%)*	–1.4	–1.0	–1.2	–5.0	5.0	0	–1.4	4.2	2.8

* C – A and C – B represent the reduction (or increase) in the number of gates, faults, or boundary cells achieved by case C compared to case A and case B, respectively, in absolute and relative terms. All percentages are expressed relative to the corresponding value of case C.

small, and the impact on the overall chip test data and test time is moderate.

As complexity increases, more situations are arising in which cores are embedded within cores. We are actively working on solutions to optimize the isolation of embedded cores and retarget patterns to test controllers within those embedded cores. We are also working on integration issues encountered in circuits containing cores with a mixture of test interfaces; some of the cores use an IEEE Std 1500 WSP, and others use an IEEE Std 1149.1 TAP. ■

Acknowledgments

We thank Steve Pateras, Fadi Maamari, and J.-F. Côté for their contributions to this article.

References

1. *IEEE Std 1149.1-2001, Standard Test Access Port and Boundary Scan Architecture,* IEEE, 2001.
2. Y. Zorian, E.J. Marinissen, and S. Dey, "Testing Embedded-Core Based System Chips," *Proc. Int'l Test Conf.* (ITC 98), IEEE CS Press, 1998, pp. 130-143.
3. R.K. Gupta and Y. Zorian, "Introducing Core-Based System Design," *IEEE Design and Test,* vol. 14, no. 4, Oct.-Dec.1997, pp. 15-25.
4. J.-F. Côté and B. Nadeau-Dostie, *Test Access Circuit and Method of Accessing Embedded Test Controllers in Integrated Circuit Modules,* US patent 6,760,874, to LogicVision Inc., Patent and Trademark Office, 2004.
5. B. Nadeau-Dostie et al., "Hierarchical Design and Test Method and System, Program Product Embodying the Method and Integrated Circuit Produced Thereby," US patent 6,615,392, to LogicVision Inc., Patent and Trademark Office, 2003.
6. *IEEE Std 1500-2007, Standard Testability Method for Embedded Core-Based Integrated Circuits,* IEEE, 2007.
7. B. Nadeau-Dostie ed., *Design for At-Speed Test, Diagnosis and Measurement,* Kluwer Academic, 2000.
8. S. Pateras, "Achieving At-Speed Structural Test," *IEEE Design & Test,* vol. 20, no. 5, 2003, pp. 26-33.

Benoit Nadeau-Dostie is chief scientist at LogicVision. His research interests cover all aspects of logic and memory test of ICs as well as test-access methods and standards. He has a BScA and a PhD in electrical engineering from the University of Sherbrooke. He is a senior member of the IEEE and a member of the IEEE Computer Society.

Saman M.I. Adham is a senior director of engineering at LogicVision. His research interests include DFT and BIST of SoCs, fault modeling and diagnosis, memory test algorithms, and electronic design automation. He has a BS and an MS from the University of Baghdad and a PhD from Queens University, all in electrical engineering. He is a senior member of the IEEE and a member of the IEEE Computer Society.

Russell Abbott is a senior design engineer at Logic Vision. His research interests include logic test of ICs and electronic design automation. He has a BS in computer engineering from the University of Manitoba.

■ Direct questions and comments about this article to Benoit Nadeau-Dostie, LogicVision (Canada), 1565 Carling Ave., Suite 508, Ottawa, Ontario, K1Z 8R1, Canada; benoit@logicvision.com.