A DfT Architecture and Tool Flow for 3-D SICs With Test Data Compression, Embedded Cores, and Multiple Towers

Christos Papameletis, Brion Keller, and Vivek Chickermane
Cadence Design Systems

Erik Jan Marinissen
IMEC

Said Hamdioui
Technische Universiteit Delft

Editor's notes:
This paper proposes a design-for-test architechture for efficient testing of 3-D ICs. The DfT architechture supports multiple dies, test data compression, and embedded cores. Commercial EDA tools are used to implement the DfT architechture.

—Dae Hyun Kim, Washington State University

■ **Three-dimensional stacked** integrated circuits (3-D SICs) using through-silicon vias (TSVs) and microbumps provide a solution to the growing demand for high-performance and low-power microelectronic products. Stacking ICs reduces their interconnect length, while the small dimensions of TSV-based interconnects offer high-density, low-latency, and low-power dissipation compared to conventional interchip interconnects.

As all microelectronic products, 3-D chips are susceptible to manufacturing defects and need to be tested. Specifically in 3-D integration, it is essential to enable test application to individual dies (prebond), partial die stacks (midbond), as well as complete die stacks (postbond) [1].

The design-for-test (DfT) infrastructure should provide modular test access to all components by elevating test control, instructions, and data up and down through the stack. This paper extends an existing 3-D DfT architecture [2], [3] and automates its implementation using Cadence EDA tools. The extensions pertain to large system-on-chip (SOC) designs and introduce support for test data compression (TDC) and wrapped embedded cores. In addition, stacks branching off into multiple towers are also supported, as such stack configurations start emerging.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.
Digital Object Identifier 10.1109/MDAT.2015.2424422
Date of publication: 20 April 2015; date of current version: 30 June 2015.

Original 3-D DfT architecture

This paper is based on the 3-D DfT architecture originally proposed for logic-on-logic stacks by Marinissen et al. [2]. In [3], this architecture was extended for memory-on-logic stacks, especially those containing JEDEC Wide-I/O Mobile DRAM [4]. The main component of the architecture is a die-level DfT wrapper, which can be based on either IEEE Std 1149.1 or IEEE Std 1500. Figure 1 shows a schematic view of the 3-D DfT architecture with an IEEE 1149.1-compliant package interface and die-level wrappers based on a 3-D enhanced version of IEEE 1500.

The architecture supports a modular test approach, in which dies and their interdie interconnects can be tested separately. This allows for: 1) targeted test pattern generation and reuse; 2) first-order fault diagnosis; and 3) yield attribution. Modular testing is enabled by a wrapper boundary register (WBR) which provides scan-based controllability and observability at all functional die inputs/outputs (I/Os).

The architecture has a mandatory 6-bit control bus `WSC` and a "serial" (1-bit) test access mechanism (STAM) `WSI`/`WSO` for instructions and low-bandwidth scan test data. There is also an optional "parallel" TAM (PTAM) `WPI`[n]/`WPO`[n] for higher throughput test access, in which the scan data are divided over n shorter parallel scan chains (with n user defined).

The architecture supports prebond, midbond, postbond, and final package test. Each die has two test ports: a primary test port (`WSC`, `WSI`, `WSO`, `WPI`[n], and `WPO`[n]) on the side of the stack's external I/Os, and a secondary test port (`WSCs`, `WSIs`, `WSOs`, `WPIs`[n], and `WPOs`[n]) toward the next die in the stack. An exception to this is the top die, which only has a primary test port, and the bottom die, which has an IEEE 1149.1-based primary test port. The secondary port of one die fits to the primary port of the next die in the stack. This way, the scan chains are concatenated together and form a configurable stack-wide TAM, which is controlled by a dedicated "turn/elevate" instruction bit per die. Both sequential and parallel test schedules are supported in which one die (respectively, multiple dies) is (are) tested at a time.

Prebond test access is provided either through direct probing of the fine-pitch microbumps [5] or through optional extra probe pads at the die-under-test [6]. As dedicated probe pads are costlier with respect to the silicon area, the architecture provisions a second, narrower PTAM of width m (with $m \leq n$). As long as their active dies form a linear sequence in their electrical connections, 2.5-D SICs are also supported.

The architecture relies heavily on reconfigurable scan chain concatenation. Per test mode (serial/

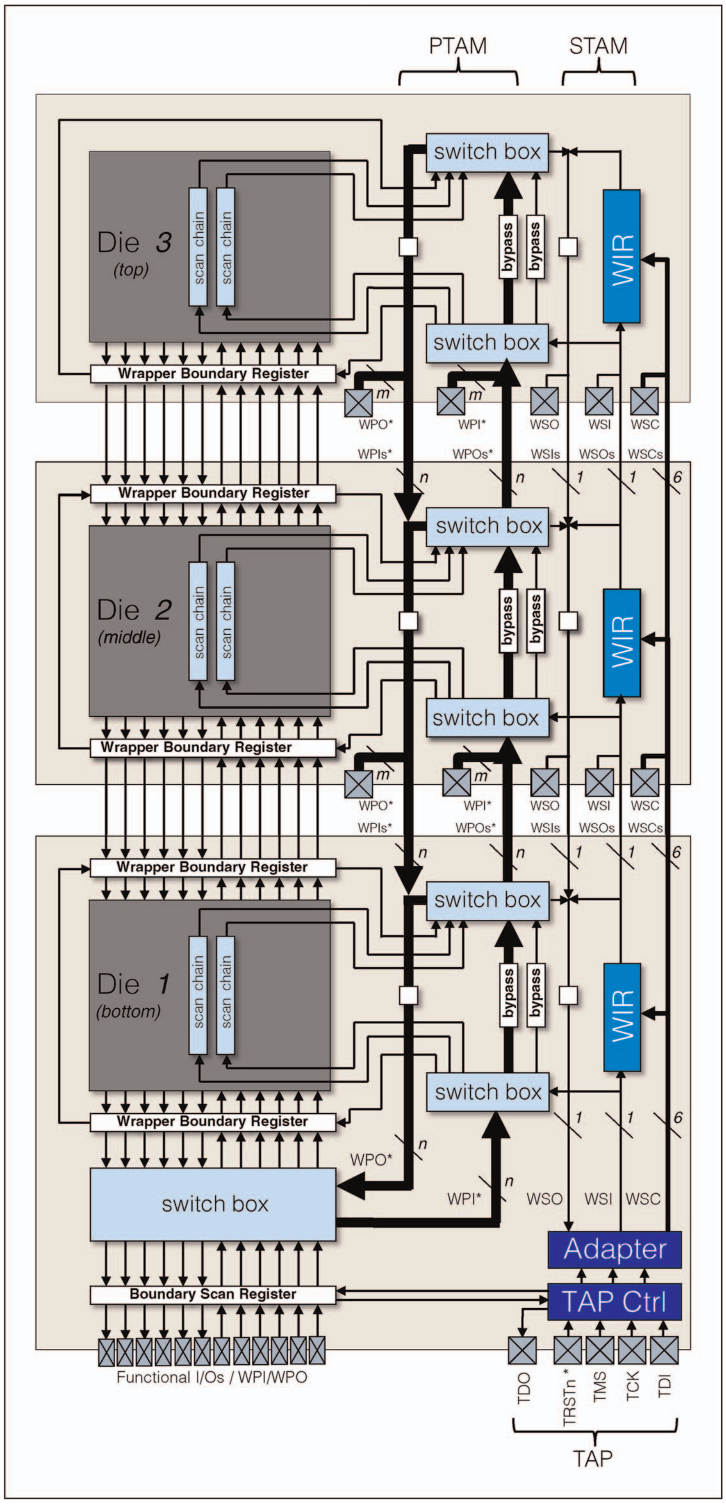

Figure 1. Schematic view of the basic 3-D DfT architecture.

parallel, pre/postbond, Bypass/ExTest/InTest, elevate/turn), the per-die scan resources [wrapper instruction register (WIR), bypass registers, WBR, and die-internal scan chains] are concatenated into 1, m, or n chains and subsequently accessed throughout the 3-D stack.

The implementation of this 3-D DfT architecture has been automated with Cadence RTL Compiler [7] and has been validated on industrial test chips, showing a negligible implementation cost below 0.03% silicon area [3].

Extensions for test data compression

TDC is used in most modern scan-testable SOCs to reduce both test data volume and test application time. At the input side, a decompressor block expands a small set of externally applied stimuli, while at the output side, a compressor block compresses the internally captured responses to a smaller set of externally observed responses [8].

A consequence of encompassing scan chains with decompression and compression blocks is that, when used in a daisy-chained context, the ability to shift arbitrary patterns through a compressed segment to other scan segments located either upstream or downstream in the scan chain is lost. Our basic 3-D DfT architecture, described in the Original 3-D DfT architecture section, uses this daisy-chain capability of scan chains for three purposes: 1) to allow concurrent execution of the InTests of multiple dies ("parallel schedule"); 2) to enable testing of interdie interconnects by allowing simultaneous scan test access to the WBRs on either side of the interconnect-under-test; and 3) width adaptation between serial (= 1 bit), prebond parallel (= m bit), and post bond parallel (= n bit) test modes. In order to accommodate dies with TDC in our 3-D DfT architecture, we need to work around this lost functionality.

We decided to drop concurrent execution of multiple InTests and only support sequential test schedules. In our experience, test-floor engineers prefer sequential test schedules over parallel test schedules. In sequential test schedules, a test failure can be directly associated to a specific die without further investigation, facilitating yield attribution to the various dies. Also, if a single die is tested at a time, it is easier to perform yield-driven test rescheduling to minimize the average test time in an abort-on-fail environment [9]. Moreover, the test length benefit of parallel schedules over sequential schedules is marginal at best, as the bulk of the test length is spent on scan-in and -out anyway. The test lengths t_{SEQ} and t_{PAR} of, respectively, sequential and full parallel schedules can be expressed as follows:

$$t_{\mathrm{SEQ}} = \sum_{i=1}^{d} (p \cdot ((i-1) + l_i + 1) + (l_i + (i-1))) \quad (1)$$

$$t_{\mathrm{PAR}} = p \cdot \left(\sum_{i=1}^{d} l_i (d-1) + 1 \right) + \left(\sum_{i=1}^{d} l_i + (d-1) \right) \quad (2)$$

where d is the number of stacked dies, p is the number of test patterns per die (assumed to be equal for all dies, representing the best case for parallel scheduling), and l_i denotes the scan length of die i. In most cases, the increase in test length due to dropping support for parallel test schedules is well below 1%, as can be seen in Figure 2, which shows, for $d \in \{2, 5, 10\}$, the relative test length benefit of a full parallel schedule over a sequential schedule as a function of the number of test patterns p (where we assumed $l_i = 1000$ for all $1 \leq i \leq d$).

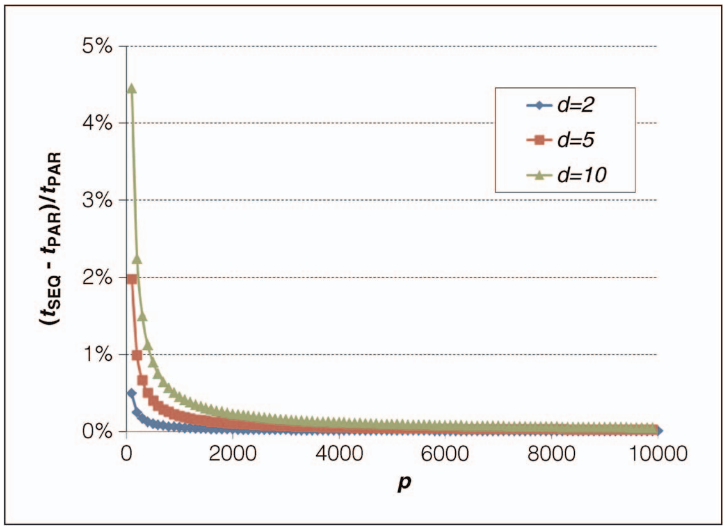

Figure 2. Test length benefit of full parallel over sequential test scheduling as a function of p for $d \in \{2, 5, 10\}$ and $l_i = 1000$ (for all $1 \leq i \leq d$).

To enable interdie interconnect testing, we require concatenated scan test access to the WBRs of all participating dies at both ends of the interconnects-under-test. For dies with TDC, this translates into a requirement to have an ExTest mode in which its WBR is excluded from the TDC envelope. To accommodate various TDC architectures [8], the starting point of our work is a "DfT-ready" die that is testable in a 2-D context, including scan chains and TDC decompression and compression blocks of the user's choice. A 3-D DfT requirement for the "DfT-ready" die is that it is equipped with a full die-level WBR, consisting of dedicated and/or shared wrapper cells. The WBR participates in inward-facing mode in the die's InTest (and hence, in that mode, it should be included in the TDC envelope), and participates without TDC in outward-facing mode in the die's ExTest. The latter ensures the ability to concatenate to the WBRs of neighboring dies for their interconnect test.

Due to the inability to concatenate scan chains in the presence of TDC, TAM width adaptation is implemented by dedicated serializers and deserializers, as opposed to scan chain concatenation in the original 3-D DfT architecture. These SerDes components can upconvert or downconvert between the various TAM widths and "bridge" them in a way independent from the internal circuitry and DfT components. The TAM widths are as follows (see Figure 3).

- 1-bit: The mandatory STAM, which uses `WSI`/`WSO`, and optional dedicated probe pads `WSI_pad`/`WSO_pad` for prebond testing.
- *n*-bit: The optional PTAM width at stack level, which uses `WPI`[*n*]/`WPO`[*n*].
- *m*-bit: The optional PTAM entry/exit point for prebond testing, which uses dedicated probe pads `WPI_pad`[m]/`WPO_pad`[*m*]. A conversion $m \leftrightarrow n$ is necessary to enable midbond testing of incomplete stacks still missing their bottom die.
- *k*-bit: The optional PTAM width within an individual die.
- *s*-bit: The optional PTAM width within the TDC envelope. For example, in case of 10× TDC, $s = 10 \times k$.

Only the 1-bit STAM and, the optional *n*-bit PTAM are global throughout the stack. Hence, only the value of *n* requires coordination between all die suppliers for the stack. The other parameters, *m*, *k*, and *s*, are local per die and can be freely chosen by the implementer of that particular die, potentially different from other dies that go into the same stack. Typically, $1 \leq m \leq n \leq k \leq s$. For parameters that are equal, no SerDes is required between them.

A complication introduced by the insertion of the SerDes is that the available TAM bandwidth remains the same, while the TAM width changes, therefore mandating an adaptation in the timing. Essentially, all (external) DfT before the deserializer and after the serializer belongs to one clock domain, while all (internal) DfT between the deserializer and the serializer belongs to another clock domain. The modified clock, generated by an on-chip clock control unit, only triggers the internal scan chains once the deserializer has been fully loaded and the serializer fully unloaded, and gates it off otherwise.

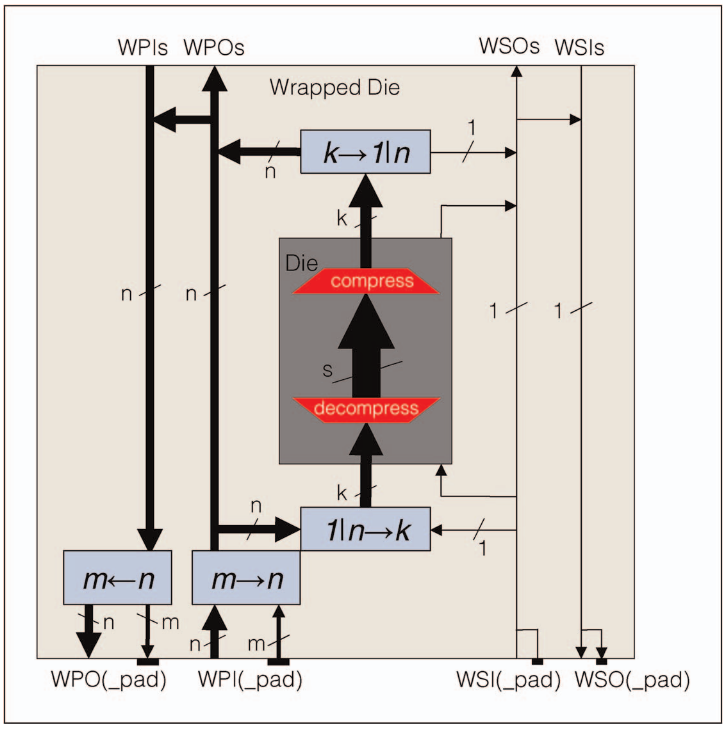

Figure 3. Generic width adaptation scheme in the presence of TDC, based on SerDes converters.

Extensions for multiple towers

As 3-D SICs grow larger and more complex, having the stack branch off into multiple towers is a natural evolution [2]. So, the extended 3-D DfT architecture has to support a scalable number of towers on top of a die. Hence, a die has to implement a variable number of secondary test ports for interfacing with the primary test ports of the die(s) directly on top of it; each secondary test port includes the `WSCs`, `WSIs`, and `WSOs` signals, and the `WPIs`[*n*] and `WPOs`[*n*] signals if a PTAM also exists.

A tower might still be incomplete or missing altogether at a certain midbond testing moment. In such a case, maintaining the continuity of the test instruction and test data paths requires being able to turn the paths at any given die level. To implement this, one additional WIR bit is introduced for each tower, as shown in Figure 4. These bits, named `IncTwr`*i*, include

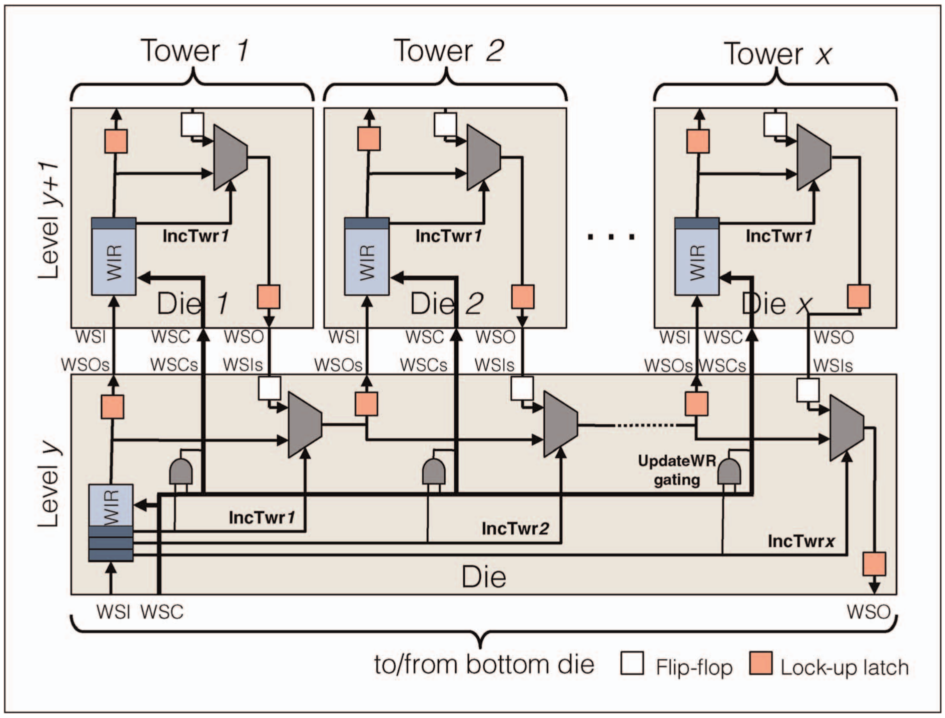

Figure 4. Detailed view of the WIR instruction path of a multitower stack.

or exclude, depending on their setting, tower *i* in/from the test path. If `IncTwr`*i* is deasserted, the `UpdateWR` signal of the corresponding excluded tower is gated off, so that its WIR state(s) will not be corrupted while other WIRs are programmed.

The fact that a WIR contains bits that determine whether other WIRs are part of the instruction path (similar to the selectable segments in IEEE Std 1149.1-2013 and the segment insertion bits in IEEE Std 1687) raises the need for incremental WIR programming. Upon reset, the accessible path only includes the bottom die to ensure its continuity. During the first iteration, only the WIR of the bottom die (first level) is included in the WIR path and can be programmed to include one or more WIRs of dies on the next level. During the second iteration, the WIR of the bottom die (first level) and previously included dies on the second level can be programmed, and so on. Programming the WIRs up to level *y* would thus require *y* programming iterations.

In addition, we protect every interdie scan chain crossing against clock skew by requiring every scan input to capture data on the rising scan clock edge and every scan output to launch data on the falling scan clock edge. As shown in Figure 4, we hereto insert pipeline flip-flops on `WSIs` and `WPIs`[*n*] inputs from all towers *i* and lockup latches on the inverse scan clock on `WSOs` and `WPOs`[*n*] outputs from all towers. That way, tolerance to interdie clock skew of up to a half clock cycle is achieved.

With the extensions for multiple towers, our architecture now also supports all 2.5-D SICs of which all external I/Os connect to only one side of a single active die.

Extensions for embedded cores

The basic 3-D DfT architecture targeted monolithic die designs only. Stacked dies are typically complex SOCs which often contain embedded cores. Hence, we extend the basic 3-D DfT architecture to handle such hierarchical SOC designs. In our model of a "DfT-ready" SOC we assume one or more embedded cores equipped with an IEEE 1500 wrapper. The test control bus `WSC` is broadcast to all embedded cores. The STAM (`WSI`/`WSO`) is daisy-chained through all embedded cores. The optional PTAM (`WPI`[*k*]/`WPO`[*k*]) serves the embedded cores in any arbitrary user-defined architecture.

Figure 5 shows a simplified example SOC consisting of a wrapped die containing two wrapped embedded cores. For such hierarchical SOCs, it is a requirement on the 3-D DfT architecture to provide simultaneous access to both die wrapper and core wrappers. The test mode where this is evident is the die's InTest mode, where the embedded core wrappers need to participate in their ExTest modes (Figure 5b); the reason is that the cores need to apply stimuli (S) to the top level and capture responses (R) from the top level of the die.

As is common in IEEE 1500, test control (`WSC`) is broadcast between dies and cores alike. For the STAM, which transports both test instructions and test data, concatenation using a daisy-chain architecture is the only interconnection option; it enables simultaneous test access to both die wrapper and embedded-core wrappers, and provides scalability for a varying number of cores. The proposed setup, for an example SOC containing two IEEE-1500-wrapped embedded cores, is shown in Figure 6.

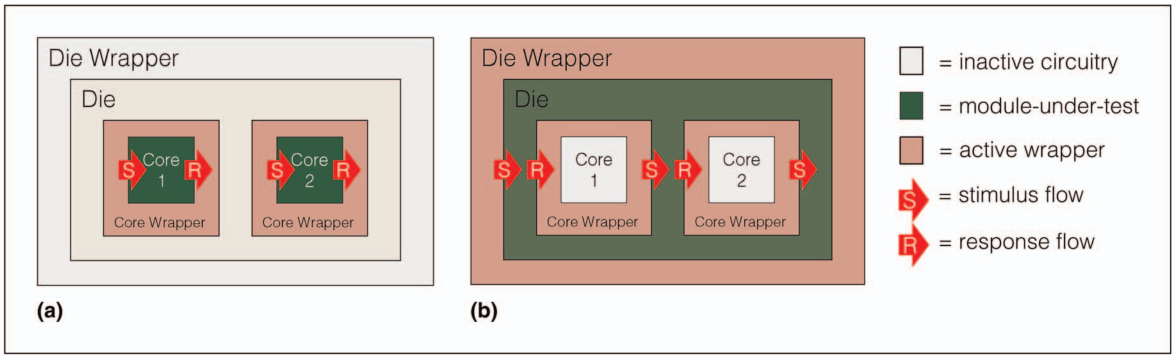

Figure 5. Example of a simplified hierarchical SOC and its test modes. (a) Cores' InTest. (b) Die InTest requires cores' ExTest.

To prevent unwieldy growth of the concatenated WIR length, we treat embedded cores in a fashion similar to stacked towers. The die-level WIR is extended with an extra control bit `IncCores` that governs inclusion of core wrappers in the test access path of this die. If deasserted, only the die-level WIR and data bypass register WBY are included in the STAM. If asserted, the die-level scan resources, as well as the core-level scan resources are included in the STAM. As depicted in Figure 6, this requires careful demultiplexing and multiplexing of the instruction and data paths of the STAM, in order to meet the constraint that at every wrapper boundary, the two are multiplexed into joint `WSI` and `WSO` terminals. A consequence of this mechanism is that the WIR programming has to be performed in an incremental fashion, similarly to multiple towers. Note that the same WIR bit `IncCores` can be used to gate off the cores' update signal (`UpdateWR`). Finally, lockup latches at clock domain crossings ensure the integrity of scan data.

Also for the PTAM, a consistent test data interface between the die top level and the cores needs to be created, so that the embedded cores can be included in the test data path. There are four potential interconnection topologies. In the multiplexing architecture, the full TAM width is exclusively provided to one core at a time. In the daisy-chain architecture, all components are concatenated to each other in the test path. In the distribution architecture, the available TAM width is distributed among the die top level and each one of the cores. Finally, in the broadcast architecture, stimuli are broadcast to all components, and responses are compressed to match the available TAM width. From these, the multiplexing architecture cannot be used, as it does not support simultaneous access to multiple components, which is necessary in this case. In a SOC including TDC, the

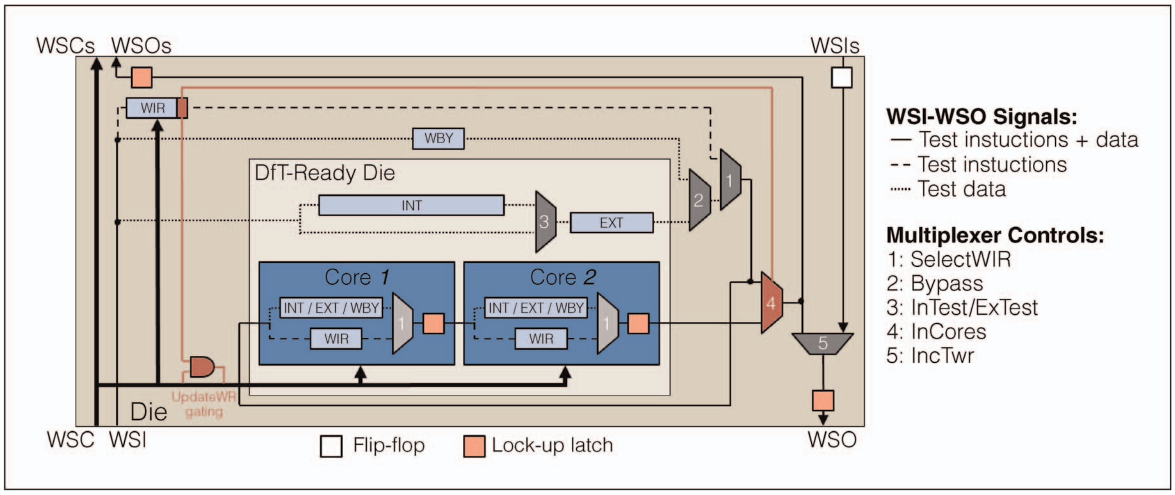

Figure 6. Test control and STAM for embedded cores.

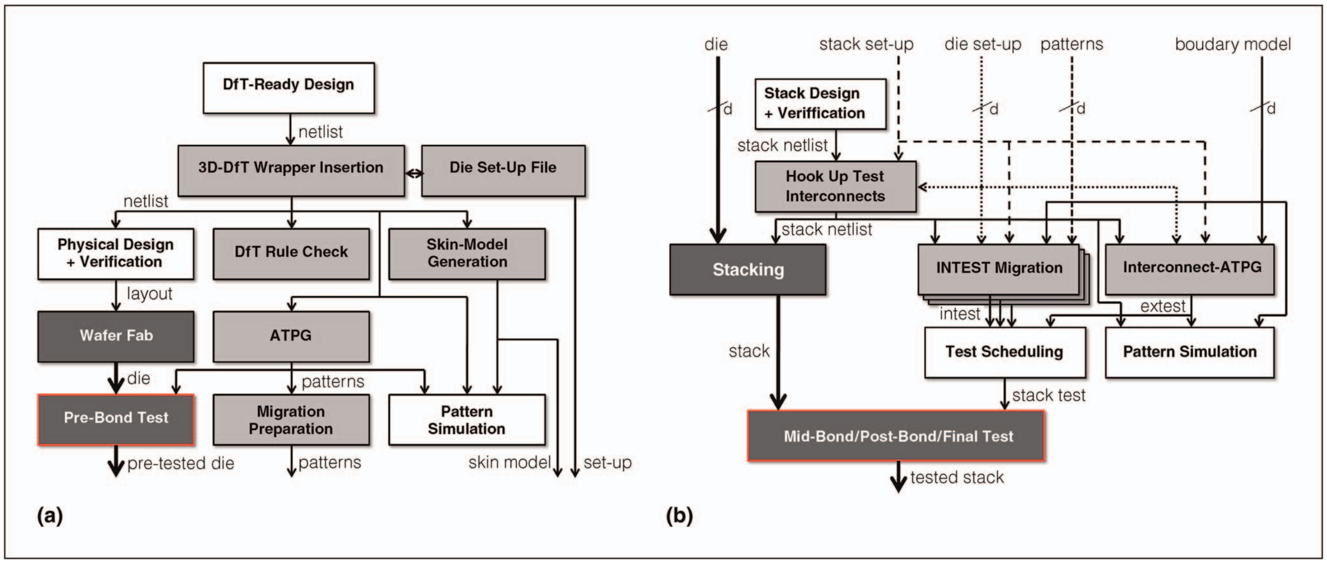

Figure 7. EDA tool flows. (a) Die-level flow. (b) Stack-level flow.

daisy-chain architecture has to be rejected for the same reason, as compressed chains cannot be concatenated. The remaining two interconnection architectures can be used by the die designer, who thus has a large design space available that allows him to flexibly configure one or more die InTest modes. The concept of a "DfT-ready" SOC makes it possible to provide full control of the InTest mode to the die designer as envisioned above.

EDA tool flows

EDA tool flows have been developed to automate the implementation of the extended 3-D DfT architecture and the generation of test patterns. We distinguish two flows: one for the die manufacturer and one for the stack manufacturer. Figure 7 depicts both tool flows. The light-gray boxes show the specific steps for 3-D DfT insertion and test generation. The white boxes denote other EDA steps. The dark-gray boxes represent physical operations. The end result of each flow, indicated with a red outline, is the ability to perform a test.

The die manufacturer's flow (Figure 7a) operates on the netlist of a "DfT-ready" die, which already contains the necessary DfT to be testable in a 2-D context. The execution of the flow is controlled by a die setup file, which specifies the number and naming of I/Os, the number of towers, the TAMs, any user-defined DfT test control signals, etc. The flow uses synthesis tool RTL Compiler to insert the 3-D DfT, producing a 3-D-wrapped die netlist. This netlist, after passing a DfT rule check, can move on to physical design (layout), verification, and actual wafer manufacturing. Automatic test pattern generation (ATPG) is performed with Encounter Test. The test patterns generated serve multiple purposes: verification of the DfT through pattern simulation, prebond testing by the die maker, and die InTest by the stack maker. In case the die maker does not want to disclose the full design details of the die with the stack maker, Encounter Test can prepare "migrateable patterns," together with a so-called "boundary model"; this is an abstract version of the die design, which includes all that is necessary for testing this die as part of an SIC, but excludes internal design IP. The boundary model allows the stack maker to generate their own (ExTest) interconnect patterns and to migrate the die's InTest patterns as provided by the die maker into a die test at the stack level. The boundary model contains just the logic necessary to run the wrapper interface, and hence it allows for a much smaller gate-count model for the SIC. In case the die maker does not mind disclosing the full design details of the die to the stack maker, migrateable InTest patterns and boundary model are not required and the full netlist of the die suffices.

In the stack manufacturer's flow shown in Figure 7b, it is assumed that the die makers want to hide their design details from the stack maker. For the various dies that together make up the stack, the deliverables of the die maker flow become the inputs in the stack maker flow: the dies themselves, the die

Table 1 Characteristics of our test case design.

Design	Level	Inputs	Outputs	Std cells	Flip-flops	TDC ratio	2D-DfT area	3D-DfT area
s400 (top)	2	3	6	62	21	–	+89.13%	+49.95%
s1196 (core)		14	14	271	18			
ac97_ctrl	2	56	46	19191	2302	2×	+11.13%	+3.24%
vga_lcd	1	87	99	163051	17265	4×	+9.06%	+0.69%

setup files, the die-level patterns, and the die boundary models. In addition, there is a stack setup file which describes the structure of the stack, optional partial stack configurations for midbond testing, and test modes of each stack. A stack test mode is composed of the test modes of the targeted dies. Information about each individual die is retrieved directly from the corresponding die setup file to make the flow more user-friendly and less error-prone. For instance, in order to generate the initialization sequence that configures the stack in a test mode, information about the stack structure and the test mode, included in the stack setup file, is combined with information about the structure of each die's WIR from the setup files of the various dies. The stack-level flow first creates the stack netlists and the initialization sequences for the various test modes. Then, depending on each test mode, it uses Encounter Test to migrate the pregenerated die test patterns to the stack boundary for intradie InTest and to perform interconnect ATPG for interdie ExTest [3], [7]. The stack-level test patterns can be simulated for verification purposes. The various tests need to be put in a test schedule and applied to the stack.

Splitting the ATPG process in two phases, the die level and the stack level, introduces various benefits. First, performing ATPG at the die level enables IP protection in the rather likely scenario where a die manufacturer and a stack manufacturer are different companies. Second, the task of achieving sufficient fault coverage lies with the die manufacturer, as it should, given that the stack manufacturer has no control over the DfT of the individual dies. Finally, the typically time-consuming InTest ATPG has to be performed only once for each test mode of each die, rather than for each test mode of the stack, and is likely to result in less overall test data volume [10].

Verification test case

The extended 3-D DfT architecture and the related automation flows were verified through application to a test case. The test case comprises three dies, organized in a stack with two towers. `vga_lcd` [11] served as the bottom die (level 1), with `ac97_ctrl` [11] and `s400` [12] (with `s1196` [12] as a wrapped embedded core) stacked on top of it (level 2). The IC designs that were used are presented in Table 1.

A 3-D DfT wrapper with a PTAM width $n = 8$ was automatically inserted around each die. Table 1 lists the relative area added by the 2-D DfT and the 3-D DfT to each die. The 2-D DfT area increase is measured with the functional design as base, while the 3-D DfT area increase is measured with the design containing the 2-D DfT as base. The reason for that is that 3-D DfT cannot exist in a design without 2-D DfT. The area costs are all relatively high, due to the small size of the test case. Nevertheless, it can be concluded that the area cost of the 3-D DfT is negligible and significantly smaller than that of the 2-D DfT for realistically sized designs.

Finally, the inserted 3-D DfT was verified by performing ATPG and simulating the application of the generated test patterns. All six possible stacks were created: one for each standalone die, one for each partial stack with either one of the two top dies missing, and one for the complete stack. Finally, all possible test modes for each stack were simulated to achieve exhaustive verification of the 3-D DfT.

This paper builds upon an existing 3-D DfT architecture that originally supported only single-tower, flat, noncompressed scan designs. In this paper, we extend this basic architecture to cover three realistic aspects of 3-D SICs:

- hierarchical SOC designs containing IEEE 1500-wrapped embedded cores;
- test data compression at die and/or core levels;
- stacks containing multiple towers.

The paper presents the 3-D DfT architecture extensions, as well as an automated DfT insertion and test generation flow with Cadence EDA tools. The concept of a "DfT-ready" die formulates the requirements which need to be met by the custom 2-D DfT incorporated in the various dies of the stack. ■

Acknowledgment

Part of the work of Erik Jan Marinissen has been performed in the project ESiP (winner of the ENIAC Innovation Award 2013), which was co-funded by the ENIAC Joint Undertaking (http://www.eniac.eu).

References

[1] M. Taouil et al., ''Test cost analysis for 3D die-to-wafer stacking,'' in *Proc. IEEE Asian Test Symp.*, Dec. 2010, pp. 435–441, DOI: 10.1109/ATS.2010.80.

[2] E. J. Marinissen et al., ''A DfT architecture for 3D-SICs based on a standardizable die wrapper,'' *J. Electron. Test., Theory Appl.*, vol. 28, no. 1, pp. 73–92, Feb. 2012, DOI: 10.1007/s10836-011-5269-9.

[3] S. Deutsch et al., ''DfT architecture and ATPG for interconnect tests of JEDEC wide-IO memory-on-logic die stacks,'' in *Proc. IEEE Int. Test Conf.*, Nov. 2012, DOI: 10.1109/TEST.2012.6401569.

[4] *Wide I/O Single Data Rate*, JEDEC Standard JESD229, JEDEC Solid State Technology Association, Dec. 2011http://www.jedec.org

[5] E. J. Marinissen et al., ''Direct probing on large-array fine-pitch micro-bumps of a wide-I/O logic-memory interface,'' in *Proc. IEEE Int. Test Conf.*, Oct. 2014, DOI: 10.1109/TEST.2014.7035314.

[6] E. J. Marinissen et al., ''Vesuvius-3D: A 3D-DfT demonstrator,'' in *Proc. IEEE Int. Test Conf.*, Oct. 2014, DOI: 10.1109/TEST.2014.7035332.

[7] S. Deutsch et al., ''Automation of 3D-DfT insertion,'' in *Proc. IEEE Asian Test Symp.*, Nov. 2011, pp. 395–400, DOI: 10.1109/ATS.2011.58.

[8] N. A. Touba, ''Survey of test vector compression techniques,'' *IEEE Design Test Comput.*, vol. 23, no. 4, pp. 294–303, Jul./Aug. 2006, DOI: 10.1109/MDT.2006.105.

[9] U. Ingelsson et al., ''Test scheduling for modular SOCs in an abort-on-fail environment,'' in *Proc. IEEE Eur. Test Symp.*, May 2005, pp. 8–13, DOI: 10.1109/ETS.2005.38.

[10] O. Sinanoglu et al., ''Test data volume comparison of monolithic testing vs. modular SOC testing,'' *IEEE Design Test Comput.*, vol. 26, no. 3, pp. 25–37, May/Jun. 2009, DOI: 10.1109/MDT.2009.65.

[11] C. Albrecht, ''IWLS 2005 benchmarks,'' in *Proc. Int. Workshop Logic Synthesis*, Jun. 2005. [Online]. Available: http://iwls.org/iwls2005/benchmark_presentation.pdf

[12] F. Brglez et al., ''Combinational profiles of sequential benchmark circuits,'' in *Proc. Int. Symp. Circuits Syst.*, May 1989, pp. 1929–1934, DOI: 10.1109/ISCAS.1989.100747.

Christos Papameletis is a DfT Engineer for Encounter Test at Cadence Design Systems, Endicott, NY, USA. His interests include boundary scan, iJTAG, LBIST, hierarchical test, and 3-D test, for which he is a member of the IEEE P1838 working group. Papameletis has an MSc in embedded systems from Delft University of Technology, Delft, The Netherlands.

Brion Keller is a Senior Architect for Encounter Test at Cadence Design Systems, Endicott, NY, USA. Keller has a BSc in computer science with a minor in chemical engineering from Penn State University. He is a Senior Member of the IEEE and has contributed to several IEEE standards.

Vivek Chickermane is a Distinguished Engineer and R&D Director for Encounter Test at Cadence Design Systems, Endicott, NY, USA. His research interests are in DfT, low-power design, and logic synthesis. Chickermane has an MSc and a PhD in computer engineering from the University of Illinois at Urbana-Champaign, Urbana, IL, USA.

Said Hamdioui is an Associate Professor in Computer Engineering at Delft University of Technology (TU Delft), Delft, The Netherlands. His research interests include dependable nanocomputing, reliability, hardware security, and test technology/design-for-test. Hamdioui has a PhD from TU Delft. He is a Senior Member of the IEEE and serves on the editorial board of IEEE Design & Test.

Erik Jan Marinissen is a Principal Scientist at IMEC, Leuven, Belgium. His research interests include all aspects of VLSI test and DfT. He has served as the Editor-in-Chief of the IEEE 1500 working group and chairs the IEEE P1838 working group. Marinissen has a PDEng in computing science from Eindhoven University of Technology, Eindhoven, The Netherllands. He is a Fellow of the IEEE and serves on the editorial board of IEEE Design & Test.

■ Direct questions and comments about this article to Christos Papameletis, Cadence Design Systems, Endicott, NY 13760 USA; christos@cadence.com.