Analysis of Test Access Mechanisms for Improving Scan Compression and Test Time

Vijay Sontakke
VLSI Design, Retym Incorporation, Austin, TX, USA

***Abstract*—System-on-Chip (SoC) devices now integrate dozens—and sometimes hundreds—of heterogeneous embedded IP cores, each of which must be verified after fabrication. Industry therefore relies on modular testing so that every core can be exercised and validated without revealing its internal implementation, and so that designers can reuse test patterns efficiently. A persistent challenge is the mismatch between the limited scan-in/scan-out bandwidth at the chip boundary and a much larger channel capacity required if all cores were tested simultaneously. Widespread use of scan-compression schemes, such as Embedded Deterministic Test (EDT), offers several features for channel count selection, and this also needs to be considered during bandwidth allocation across cores. The multiple requirements are met using Test Access Mechanism (TAM), and over the past two decades, researchers have proposed many TAM architectures that move well beyond simple pin multiplexing, each balancing wiring overhead, concurrency, pattern compression, and scheduling complexity in different ways. However, a combined study of their effectiveness considering multiple aspects is not available. This study reviews the principles, algorithms, and architectures of TAM and test scheduling techniques. A classification of the techniques is provided, based on the method used and the area of application. The goal of the study is to create a platform for the future development of test access mechanisms. The study is believed to be helpful to both industry and academia.**

***Keywords*—Compression ratio; scan bandwidth; TAM; test coverage; test scheduling; test time**

I. Introduction

The rapid scaling of semiconductor technology to sub-10 nm nodes, combined with the transition toward 3-Dimensional (3D) integrated circuits, has significantly influenced both chip design and testing methodologies. Modern System-on-Chip (SoC) designs now integrate over a billion transistors and operate at gigahertz frequencies. These complex systems incorporate a diverse mix of digital, analog, mixed-signal, memory, optical, microelectromechanical, and radio-frequency cores. Testing such multifaceted designs presents a considerable challenge. The widespread adoption of System-on-Chip (SoC) technology has led to a dramatic increase in testing costs, primarily due to the difficulty of accessing embedded cores, the lengthy process of developing and applying test patterns, and the large volume of test data involved. Network-on-chip facilitates core communications, but it complicates SOC testing [1][2][3][4].

As integrated circuits scale to billions of transistors, designing and testing them as a flat is almost impractical. SoCs address this complexity by partitioning functionality into different cores and integrating them. Each such core is typically designed, and verified independently before being integrated into the system. One of the most widely adopted strategies for managing test complexity in System-on-Chip (SoC) devices is on-chip test compression. In this approach, test patterns are delivered to the chip pins in compressed form and then decompressed on-chip before being applied to scan chains. This technique requires dedicated on-chip infrastructure, including test access mechanisms (TAMs) and test wrappers.

Beyond the hardware infrastructure, efficient SoC testing also relies on effective test scheduling to manage the concurrent testing of multiple cores and the use of shared test resources. Coordinated optimization of TAM allocation and test scheduling can significantly reduce test time, minimize data volume, and control testing costs. Core-based System-on-Chip (SoC) architectures are dominant in modern industry, with many designs containing hundreds of physical cores—often including multiple instances of identical Intellectual Property (IP) blocks. This modular reuse of independently verified cores is a key reason for the success and scalability of SoC-based development. Using multiple identical cores also provides opportunities for test optimization, as test patterns can be reused. Illinois Scan [5] applies test patterns simultaneously. This technique can be used to further improve efficiency. Broadcast method for applying scan stimuli is also used by [6][7][8][9].

Despite these advantages, SoC-level test planning is constrained by several critical factors, including limited test pins, power consumption constraints during testing, and design-for-test (DFT) routing and layout constraints. Each core in a large System-on-Chip (SoC) requires dedicated input/output test channels; however, the finite number of chip-level test pins makes it impossible to access all core-level channels simultaneously. As a result, hierarchical test strategies and pattern retargeting are crucial for managing access and ensuring test efficiency. In this context, hierarchical testing has emerged as the most scalable and effective solution, enabling the systematic testing of complex SoCs while accounting for the limitations imposed by modern design and manufacturing processes. In this testing, ATPG is performed at the core level, and then patterns are retargeted to the top level [9] [10]. Hierarchical test offers two major benefits: 1) Pattern generation is done at the core level, which requires a smaller memory footprint and shorter Central Processing Unit (CPU) runtime; 2) Pattern generation and verification can be done as soon as the core is ready.

While many researchers listed various TAM design techniques in their papers' background and literature review, they did not cover all aspects of the techniques and focused only on a relevant aspect, e.g., pattern independence or area improvement. There is no survey published till now that

compares several representative TAM solutions, examining how best to partition the available TAM width, assign cores to those partitions, tune EDT parameters in concert with TAM design, and ultimately minimize total test time while preserving high fault coverage. This research is the first of its kind to present this study.

Table I below gives the full form of abbreviations and acronyms used in this study.

TABLE I. ABBREVIATIONS AND ACRONYMS

Abbreviations	Full Form
ATPG	Automatic Test Pattern Generation
DFT	Design for Test
SOC	System On Chip
EDT	Embedded Deterministic Test
TAM	Test Access Mechanism
I/O	Input Output
IP	Intellectual Property
ATE	Automated Test Equipment
CPU	Central Processing Unit
LPC	Low Power Controller
SDI	Scan Data Input
SDO	Scan Data Output
PCIe	Peripheral Component Interconnect Express
GPIO	General Purpose Input/Output

II. BACKGROUND

Embedded Deterministic Test (EDT) was introduced as a means to significantly reduce both the volume of scan test data and the total test time [11]. EDT comprises two complementary components: on-chip hardware and deterministic Automatic Test Pattern Generation (ATPG) software. The on-chip hardware is inserted along scan paths and works in conjunction with ATPG tools that produce highly compressed test patterns tailored to this architecture. As illustrated in Fig. 1, the EDT hardware includes a continuous-flow decompressor that expands a small set of scan channel inputs into test stimuli for a large number of internal scan chains, as well as a compactor that compresses the scan responses from these chains into a reduced set of outputs.

The concept of modular testing is structured around three key components: the test pattern source, the test response sink, and the infrastructure composed of the Test Access Mechanism (TAM) and the test wrapper. In modern large-scale SoCs, embedded cores are heavily used. These cores cannot be directly accessed via the chip's primary inputs and outputs, necessitating a specialized TAM for system-level testing. The TAM serves as a communication bridge, routing test data between the chip I/O (Input Output) and the internal cores. The test wrapper, on the other hand, creates a standard interface between each core and its external test environment.

TAM plays a critical role in modular testing, as its configuration and efficiency directly affect the overall test time of the SoC [12] [13]. The IEEE 1500 standard provides a defined structure for wrapper design [14]. However, it does not prescribe specific methods for TAM optimization. As a result, the design and optimization of TAM architectures remain an important area of ongoing research aimed at improving test efficiency in complex SoC environments. Initially, TAMs were considered as a medium for moving data from chip pins to cores and for observing the core's response on pins [15][16]. Test wrappers are used as an interface to access cores from the soc level [17][18][19]. A simpler form of TAM can be channel broadcasting that can be used for identical cores [8] or channel sharing, which can be used for non-identical cores [20].

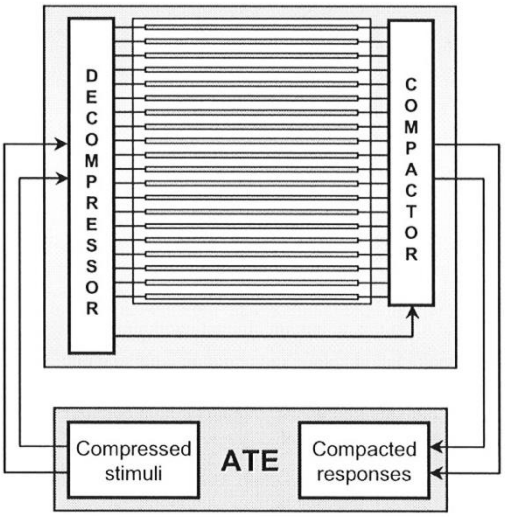

Fig. 1. EDT architecture [11].

Numerous SoC testing strategies proposed in the literature rely on dedicated on-chip infrastructure, such as test access mechanisms (TAMs), test wrappers, and a variety of test pattern scheduling algorithms.

Along with TAM, effective test scheduling is required to reduce SOC testing costs [21]. A typical SOC contains multiple cores and large pattern sets. Test scheduling is typically an NP-complete problem because it is formulated as a combinatorial open-shop scheduling problem with a fixed number of processors [22] or as two- or three-dimensional bin packing [23].

When multiple cores are exercised simultaneously to reduce overall chip testing time, power consumption during scan testing increases. Many design and pattern-generation algorithms have been used to reduce power during the shift [24] and capture [25] phases of the scan pattern. Many test scheduling flows are also proposed, which help to reduce power during pattern application [26][27][28][29][30].

III. CLASSIFICATION

Flows that have efficient TAM and test scheduling significantly improve test data volume, test time, and ultimately test cost [31]. Over the years, researchers have presented various test access mechanisms. This survey covers those approaches.

They are divided into categories based on the testing parameters. Following is a list of such categories:

- TAM with improved EDT injectors
- TAM with On-Chip Compare
- TAM with Efficient Pattern Generation at Higher-Level
- Channel Sharing and Broadcasting
- Dynamic Bandwidth Management with Channel Sharing
- TAM Optimization Using Pattern Suite
- Test-pattern Independent TAM
- Using High Frequency for Test Channels
- Test Schedule for Low-Power Designs

IV. TAM Design and Test Scheduling Methodologies

A. TAM with Improved EDT Injector

1) Channel-Utilization issues: Even with the flexible assignment of Automated Test Equipment (ATE) channels supported by Embedded Deterministic Test (EDT), many patterns remain poorly encoded. Consider a set that needs 8 input channels: its fill rate can drop from 1.7% to as little as 0.3 %. Such patterns consume a large number of channels while using only a tiny fraction of the available bandwidth, leading to under-utilization. This inefficiency is evident across all patterns, regardless of the initial channel allocation, and the fill rate declines sharply.

The green curve in Fig. 2 estimates the theoretical minimum number of channels each pattern actually requires. The gap between this reference (green) and the observed usage (blue) represents the scope for further channel reduction. Ideally, a TAM configuration that keeps channel counts to a minimum while still guaranteeing successful encoding would track the green profile in Fig. 2.

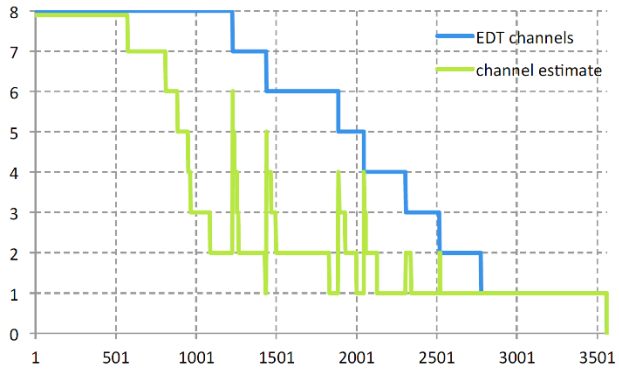

Fig. 2. Test cube channel demands [32].

2) Enhancing encoding efficiency through injector placement: The leading cause of the sharp drop in EDT encoding efficiency can be mitigated by either selecting input channels in a more planned order or by arranging the channel injectors within the EDT logic in a carefully chosen pattern. The injectors are selected according to predetermined rules [11]. Janicki et al. [32] demonstrated that rearranging these injectors yields higher compression and improved bandwidth utilization. The spread of seed variables through the decompressor significantly affects the number of input channels required for a given pattern and, consequently, the attainable compression ratio. Their distribution depends on the interplay among injector locations, feedback connections, and phase-shifter taps, with the greatest benefits arising when injector placement is optimized for the inputs the ATPG solver uses most often. Those heavily exercised inputs must, therefore, be equipped with injectors that promote vigorous circulation of seed variables.

Consider a conventional EDT decompressor implemented as a 16-bit ring generator with four input channels, each containing two injectors (illustrated in Fig. 3). In this structure, the two middle channels—channels 2 and 3—have injectors spaced widely enough that freshly injected seed variables spread rapidly across every stage of the ring generator. This produces a balanced distribution throughout the scan chains. By contrast, channels 1 and 4 place their injectors much closer together, limiting seed movement and resulting in poorer distribution.

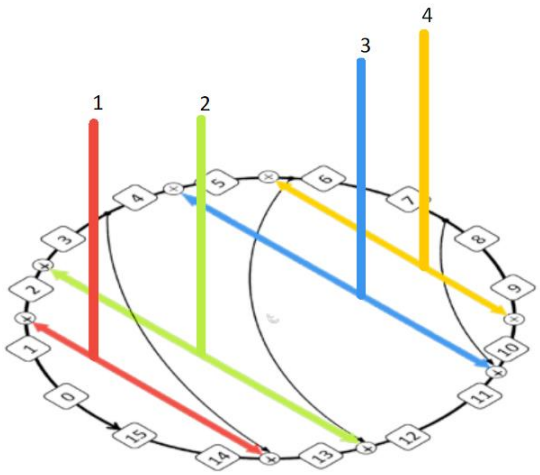

Fig. 3. Conventional EDT injector placement [32].

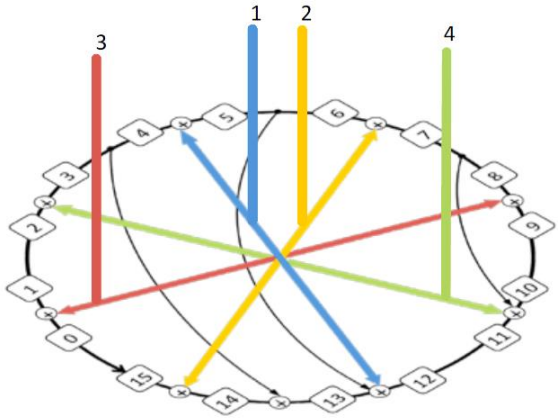

Fig. 4. Evenly distributed EDT Injectors [32].

Using these observations, the authors proposed a heuristic that selects each new input channel to be as far away from the

channel chosen in the preceding step as possible (see Fig. 4). The algorithm maintains an ordered channel list, repeatedly swapping two randomly chosen channels and retaining the swap only if it increases the sum of pair-wise distances between adjacent channels. In parallel, it rearranges injectors to promote a uniform spread of seed variables. Channels that the ATPG solver uses most frequently receive injector placements that speed seed circulation and reduce the number of clock cycles required to deliver variables to the scan chains.

Table II summarizes the results for five standalone industrial cores after applying this test data reduction strategy. Compared with the method in [33], the new scheme achieves significantly higher compression efficiency.

TABLE II. CORE-BASED AVERAGE CHANNEL DEMAND [32]

Core	Test Patterns	EDT Channels	Average Channel Demands	
			EDT of [33]	*Uniform Injectors*
C1	3504	20	4.99	3.76
C2	3634	16	4.83	4.27
C3	2531	12	3.97	3.14
C4	4074	8	3.25	2.78
C5	1358	6	2.4	2.11

B. TAM with On-Chip Compare

1) On-Chip comparison for parallel testing of identical cores: In this scheme, a single set of scan-in data is broadcast simultaneously to several identical processor cores, allowing them to be exercised in parallel. Each core's response is then evaluated on-chip. The comparators match the captured outputs either against a golden reference pattern, also supplied by the tester, or against one another. Any core whose response deviates is flagged as faulty and can subsequently be repaired or disabled. Grady et al. [7] showed that because identical cores should produce identical responses when isolated from external data sources, on-chip comparators can both deliver test stimuli and compact the resulting pass/fail information. Thus, the TAM not only transports test data to and from the embedded cores but also acts as an effective compression mechanism, collapsing the test outcomes of all cores into the scan-out channels of a single representative core. Microprocessors tend to use multiple instances of a CPU core [34][35][36][37], and the scheme proposed by the author could be quite useful for testing such microprocessors. Earlier, [38] showed how to test multiple identical processor cores in parallel by broadcasting the scan data inputs to all of them.

2) Pipelined TAM architecture with full-rate self-compare mode: Fig. 5 illustrates a simplified configuration of a basic pipelined TAM architecture for a chip containing three identical processor cores, represented as three large blocks. In this design, each core receives the same test stimulus in a staggered sequence. The TAM circuitry operates with a continuous free-running clock signal (TAM_Clk) at the frequency required to shift scan data through the cores' scan chains. Regardless of whether the cores are actively shifting data, all TAM pipeline registers update every cycle with the TAM_Clk.

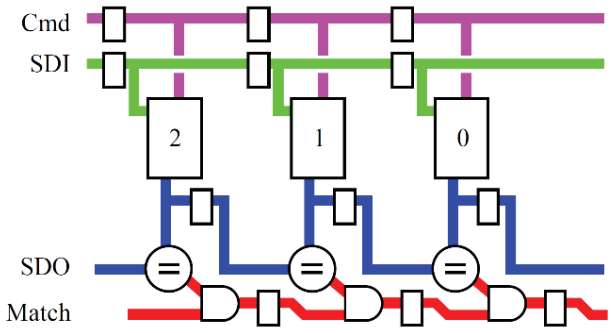

Fig. 5. TAM usage in a SOC [7].

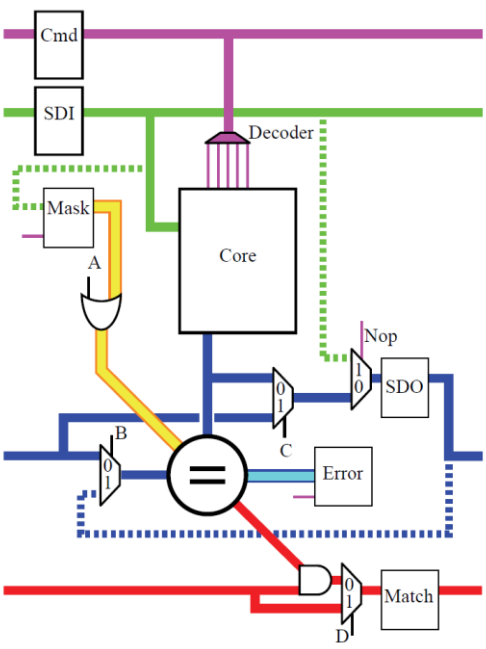

Fig. 6. TAM connections in a Core [7].

Within the scan data output (SDO) lane, each core's scan output is routed to a comparator. This comparator compares the output either with the upstream core's output or with data held in the SDO pipeline register. The comparator produces bitwise results (shown in light blue in Fig. 6), which are captured by an error register. These error registers are "sticky", meaning once an error is recorded, it remains latched until explicitly cleared.

The architecture can operate in four distinct modes. One key configuration, referred to as Full-Rate Self-Compare Mode, repurposes all scan output pins as inputs for expected pattern data. In this mode, each core's output is compared directly to the expected results loaded via the TAM. Control signals B and C are asserted (set to logic 1) for each core, activating the full-rate self-compare pipeline structure shown in Fig. 5.

During each shift-type operation cycle, each core sequentially compares its output against the expected pattern and propagates its local match result down a match pipeline. The match output at the chip boundary indicates, on a cycle-by-cycle

basis, whether a mismatch occurred across any channel. Simultaneously, each core's sticky error register records whether it ever failed during testing. This mode enables testing of all N cores in parallel within the time required to test just a single core, effectively increasing test throughput by a factor of N, while still delivering individual pass/fail results.

In [39], the author used the TAM architecture for AMD's quad-core Opteron processor. It ran the TAM in inter-core-compare mode. The ATE directly monitored only one reference core, and the remaining cores were verified on-chip by comparing their scan responses to that of the reference. When their outputs matched, they were considered fault-free. Results showed that this strategy allowed the entire quad-core device to be tested with only a 23% increase in test time compared to a single-core test. The TAM adds one flop each for command, SDI, mask, error, and SDO, as well as for all scan channels. However, the Opteron processor contains 5000 flops per channel [40], so this overhead is insignificant.

C. TAM with Efficient Pattern Generation at Higher-Level

1) Hybrid test methodology using shared wrappers: Modular SOC testing is beneficial for large SOCs where not all cores can be loaded and tested due to processing capacity constraints and other concerns, such as power consumption. The concept of modular testing was introduced in [37], which uses test wrappers. Hierarchical test methods [41][42] add scan chains and compression logic in every core [43][44].

However, system-on-chip devices typically integrate a mix of large and small cores. Wrapping every small core individually for hierarchical pattern retargeting can be wasteful. So, a common alternative is to group several small cores under a single wrapper and run ATPG across the whole cluster. The patterns generated for this cluster are then retargeted directly to the chip level. This strategy is known as the hybrid test methodology. Running ATPG over a cluster rather than over each core separately can yield a more compact overall pattern set.

2) Control and data separation: To address the limitation imposed by a restricted number of top-level test pins in SoC designs, techniques such as channel sharing among non-identical modules and channel broadcasting to identical modules are commonly employed. Guoliang Li et al. [45] presented a comparative analysis of various hybrid hierarchical and modular test strategies and highlighted the trade-offs and efficiencies for different SoC configurations.

One of the most widely adopted EDT compactor implementations is the Xpress Compactor, which directly sources the control bits from external input channels. To reduce shift power during scan operations, optional Low Power Controllers (LPCs) are added. These are also driven by bits from external input channels. The control bits and test data can be separated onto different input channels [46]. Fig. 7 illustrates an example EDT IP that includes both an Xpress Compactor and an LPC. In this setup, two EDT input channels feed the decompressor, and the control bits for both the compactor and the LPC are distributed evenly between these two channels.

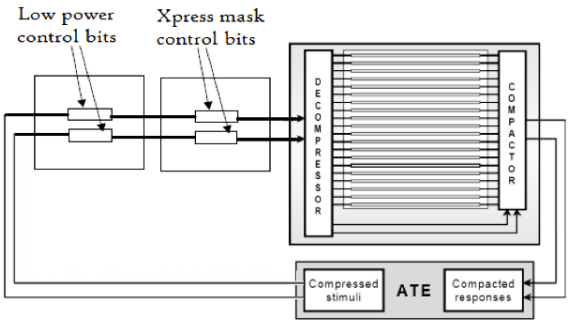

Fig. 7. An EDT without separated control bits and test data [45].

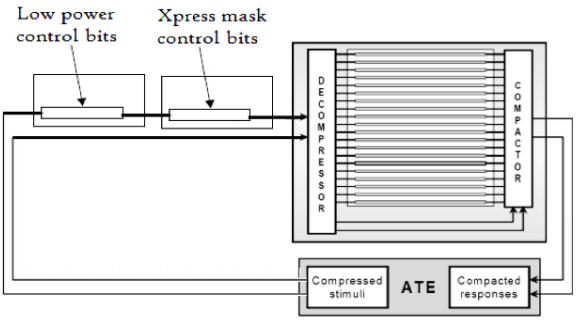

Fig. 8. An EDT with separated control bits and test data [45].

Improved flexibility and shorter shift lengths can be achieved by separating control bits and test data onto different input channels. Such an architecture is shown in Fig. 8, where all control bits are allocated to a dedicated channel, leaving the remaining channels exclusively for test data. This separation enables more efficient scheduling and resource sharing.

However, without such separation between control bits and test data, input channel sharing across different modules is generally not feasible—except in cases involving identical modules. Thus, the ability to decouple control and data streams is crucial for enabling channel sharing in more generalized, non-uniform System-on-Chip (SoC) environments.

3) Scenario variations for channel sharing: Fig. 9 illustrates a modular test setup in which ATPG targets a pair of EDT blocks grouped within a single core. Once control and test data are separated on separate input channels, the data lanes can be shared among different EDT blocks, as shown in Fig. 10. Channel sharing need not be confined to a single core or even to one hierarchy level. It can be arranged entirely within a core, across the SoC, or at intermediate boundaries. A mixed configuration of internal and external sharing can also be created. The most flexible arrangement could have data-channel sharing across distinct cores coupled with broadcast-style reuse of channels among identical cores. In this combined scheme, control channels remain dedicated whenever the participating cores are not identical; however, both control and data lanes can be reused freely when the cores are identical.

The author conducted an evaluation on a large industrial System-on-Chip (SoC) containing 26 cores, each with its own EDT logic. These cores were manually divided into six function-

based groups, labeled A through F. Automatic Test Pattern Generation (ATPG) was then performed under five distinct configurations:

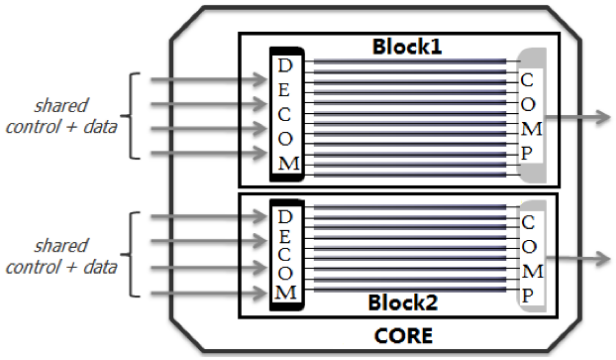

Fig. 9. Non-channel sharing modular test [45].

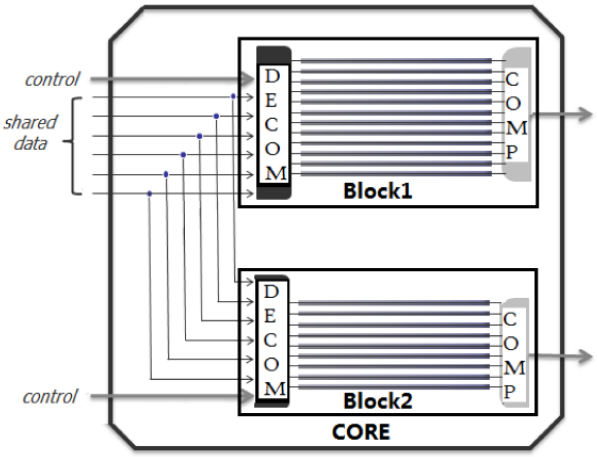

Fig. 10. Channel sharing across two blocks [45].

- Scenario I – Conventional hierarchical flow: Every core uses EDT, and control and data share the same channels.
- Scenario II – As in Scenario I, but each core's EDT keeps control and data on separate channels.
- Scenario III – Hybrid hierarchical/modular flow: ATPG runs across all cores in a given group, yet control and data remain combined within each core's EDT.
- Scenario IV – Builds on Scenario III by separating control and data channels inside each core's EDT.
- Scenario V - Extends Scenario IV by enabling channel sharing among cores within each group.

The results from this evaluation show that incorporating channel sharing into the hybrid modular-hierarchical flow (Scenario V) delivers the greatest efficiency. The total test cycles drop by roughly 20% compared to the baseline traditional hierarchical approach, which does not separate the control and data channels (Scenario I).

D. Channel Sharing and Broadcasting

Broadcasting and channel sharing can be used to share SOC pins as test pins for a group of cores. All control and data channels can be shared among identical cores [8] [47]. Such sharing is usually called broadcasting. Channel sharing refers to sharing data channels only, but not control channels [20] [48]. This is done for non-identical cores. A mix of broadcast and channel-sharing methodology is also used to optimize test time [49].

Broadcasting scan stimuli to identical instances of core has also been shown in [7][8][9][50]. Xiao Liu et al. and colleagues extended the channel-sharing approach described in [45] and reported their findings in [51]. At the SoC level, engineers must choose between two test strategies: 1) hierarchical pattern retargeting or 2) chip-level ATPG that combines channel broadcasting with channel sharing. Factors such as design scale, available compute resources, ATPG run-time budget, scalability goals, and the effort required to design core wrappers determine the optimal choice. The use of a combination of broadcasting and channel sharing across different groups of cores has also been shown in [52].

TABLE III. CHANNEL SHARING VS. NON-CHANNEL SHARING ATPG [51]

Fault Type	Configuration	Test Coverage	Compression Improvement
Stuck-at	Non-channel sharing	97.65%	1X
	Channel sharing	97.70%	1.68X
Transition	Non-channel sharing	92.02%	1X
	Channel sharing	92.11%	1.97X

The author compared the allocation of input and output channels across all core instances for two configurations—without channel sharing and with channel sharing under a fixed budget of 12 input channels and 14 output channels at the chip boundary. In both cases, the total pin count stays constant, and the number of output channels is identical. ATPG results for stuck-at and transition faults (see Table III) show that channel sharing delivers slightly higher fault coverage. When sharing is absent, each core receives only a few input channels, limiting encoding capacity and leaving some faults untested. Under tight pin constraints, combining channel sharing with broadcasting effectively exploits the limited bandwidth. Without sharing, each core's sparse channel allocation nearly doubles the number of patterns required to achieve comparable coverage.

E. Dynamic Bandwidth Management with Channel Sharing

Under traditional or static bandwidth management, all cores are partitioned into several groups. Each group represents a single test configuration in which the selected cores run in parallel while the groups themselves are exercised sequentially. The partitioning algorithm selects one core at a time and packs as many additional cores as possible into the same group, subject to the System-on-Chip (SoC)'s pin budget and power constraints.

By contrast, dynamic bandwidth management breaks each core's pattern set into smaller segments and redistributes those segments across multiple groups. A given core may, therefore, be tested in several non-contiguous windows, allowing the scheduler to exploit idle bandwidth more aggressively.

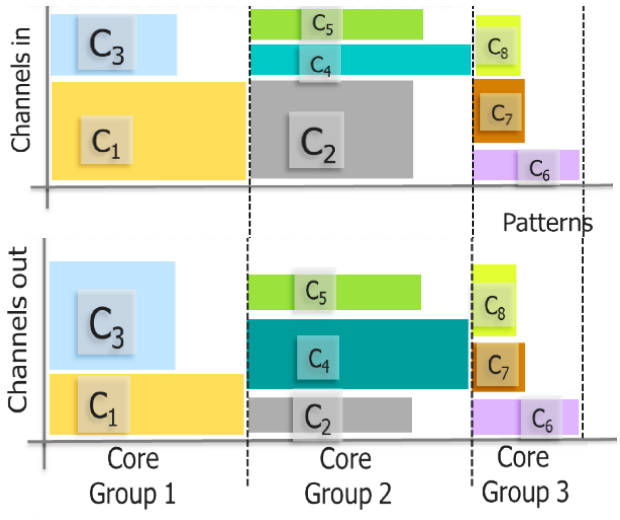

Fig. 11. Scheduling using static bandwidth management [53].

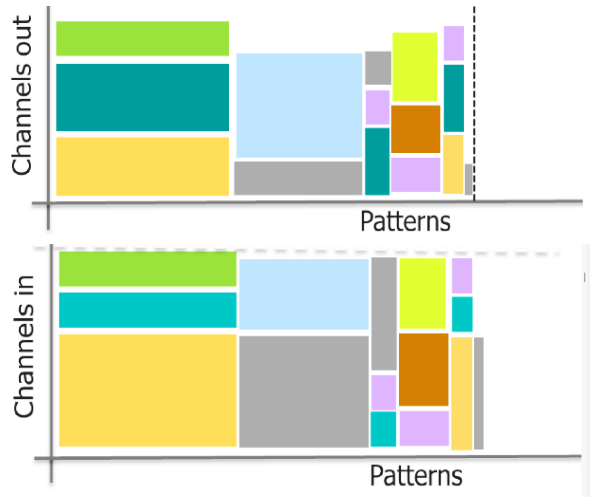

Fig. 12. Scheduling using dynamic bandwidth management [53].

Yu Huang et al. [53] evaluated the effectiveness of both approaches—static versus dynamic—and further examined how channel sharing influences each strategy's bandwidth utilization and overall test throughput.

Each core is represented as a pair of rectangles. The rectangle's width corresponds to the number of test patterns that the core requires, while its height represents the number of input or output channels it consumes. Scheduling the tests becomes a two-dimensional bin-packing problem. These rectangular pairs must be arranged within the fixed "SoC bin" to minimize the total pattern count. Fig. 11 and Fig. 12 illustrate schedules generated by static and dynamic bandwidth management, clearly demonstrating that the dynamic approach reduces the overall pattern volume by allowing each core's patterns to be distributed across multiple configurations.

For the evaluation, the author selected an industrial SoC containing 13 cores and 30 scan-in and 30 scan-out channels. Test coverage and pattern volume were measured under four configurations: static bandwidth management with and without channel sharing, and dynamic bandwidth management with and without channel sharing. The results show that channel sharing, coupled with broadcasting, sharply reduces the pattern count by approximately 1.86 times under static scheduling (a decrease from 173 to 93 patterns) and by an even larger factor of 2 under dynamic scheduling (from 123 to 62 patterns). Dynamic bandwidth management itself proves advantageous. Relative to static scheduling, it reduces the pattern total by approximately 1.4 times when channel sharing is absent and by roughly 1.5 times when sharing is enabled. The most significant savings are achieved when channel sharing, broadcasting, and dynamic compaction are combined, making this strategy the most effective for reducing test patterns while maintaining coverage.

F. TAM Optimization Using Pattern Suite

Janicki et al. [54] described a Test Access Mechanism that takes advantage of specific properties of Embedded Deterministic Test (EDT). Their approach combines test data reduction methods with a scheduling strategy and introduces new TAM structures for both the stimulus and response paths.

Experiments show that test cubes contain very little specified data. This is observed even when they are produced by advanced dynamic-compaction techniques that target multiple faults over several clock compressions. Initially, the fill rate is typically only 1% to 5%, and after the first few vectors, it often falls well below 1%. This sparseness makes modern test-data compression methods highly effective. However, with a static assignment of decompressor input channels, most of that capacity remains idle after the early patterns are applied, resulting in a significant drop in encoding efficiency. Keeping a fixed number of inputs, therefore, injects thousands of useless bits. In practice, far fewer ATE channels are needed to keep every vector encodable. By adjusting the number of active decompressor inputs to match the declining fill rate, the ATE bandwidth can be used far more efficiently.

Flexible, demand-driven channel allocation can significantly boost both compression and encoding efficiency. By distributing ATE channels to each core only when—and only in the quantity—its fill rate requires, multiple cores can be exercised in parallel, reducing overall test time. Staggering the operation of individual decompressors into well-chosen time slots further lowers the number of external channels needed, often far below the aggregate total of all cores' EDT inputs.

1) Test Scheduler: The authors' proposed scheduling flow begins once ATPG has generated test cubes for every targeted fault and merged them into complete patterns. Each pattern is then fed to an encoding solver that drives the on-chip decompressor with the minimum possible number of EDT input channels. After encoding, the patterns undergo fault simulation to identify which faults propagate to which outputs of the compaction logic and to locate any unknown (X) states that slip past X-masking. Using these simulation results, the algorithm pinpoints the smallest set of output channels that can still indicate every detected fault, even in the presence of X states. It then pairs, for every pattern, the fewest required input channels with this minimal set of observation outputs, creating a concise "signature" of channel needs. Patterns that share identical input-output signatures are clustered into classes, simplifying subsequent scheduling. Finally, these classes are

handed to the test scheduler, which, taking into account the TAM architecture, the available ATE pin budget, power limits, and other system-level constraints, assigns specific input and output channels to the relevant cores for pattern application.

After clustering, the scheduler assigns the external tester's channels. Input and output channels are handled independently because a cluster's demand for stimulus pins can differ from its need for response pins. For each cluster, the algorithm maps the required number of ATE inputs to the cores' decompressor ports and the outputs to the chosen observation sites, while always respecting the global pin budget and any other SoC-level constraints. The test vectors are applied to selected cores in phases. Other cores are frozen during such a pattern application phase using clock gating or chain blocking at the decompressor methods [55].

Only a small fraction of patterns need every output pin to catch faults. As testing progresses and the fault count per pattern declines, later vectors can be observed through far fewer outputs. Fig. 13 illustrates this effect for ten cores in an industrial SoC: each bar shows what share of patterns for that module succeed with just one, two, three, four, or five output channels.

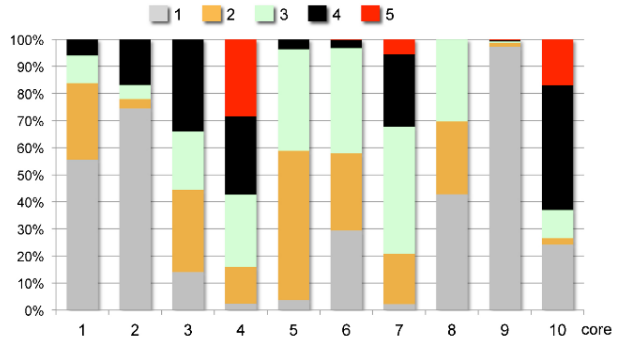

Fig. 13. Percentage of patterns with different numbers of output channels [54].

Being able to adjust the observation width is essential for conserving bandwidth on the output side. The scheduler chooses which pins to monitor by examining how faults display on each output, how they mask one another, and how unknown (X) states propagate. For each pattern group, it selects the smallest set of observation sites that still reveals every targeted fault, allowing the remaining channels to be reassigned to where they are most useful. Fig. 14 shows the test schedule derived using this method.

2) TAM design: Once the test schedule is finalized, the flow drives the TAM hardware design. The input network is constructed from n demultiplexers—one for each fixed ATE input channel—and the output network comprises m multiplexers, corresponding to the number of ATE output channels. Each demux (or mux) connects its channel to a single core input (or output). An address register loaded with every compressed test vector tells the switch which core to serve. Because this control data is small, it is sent uncompressed alongside the test data. The compressed outputs of the cores are connected to the output switching network [56].

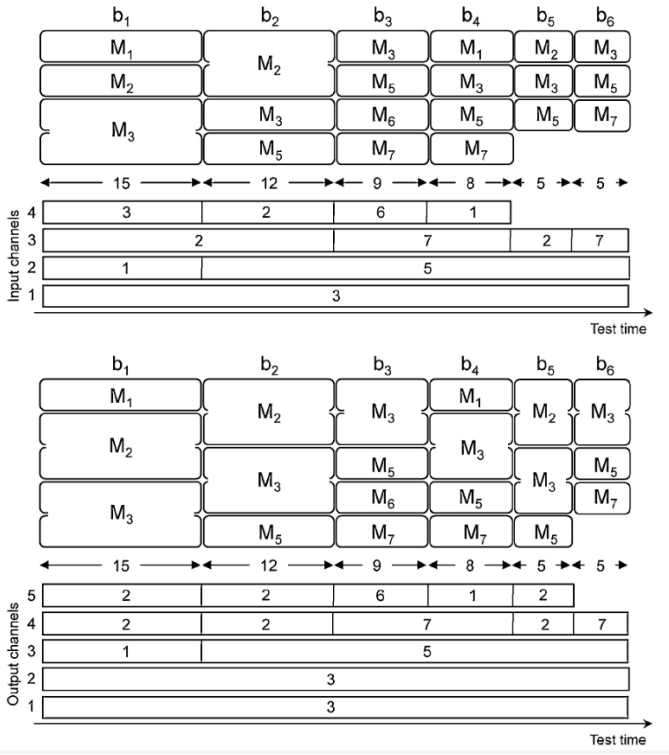

Fig. 14. Base clusters and test scheduling [54].

A demultiplexer must fan out to at least as many cores as it supports, though it may feed multiple EDT inputs of the same core. Conversely, several demultiplexers (i.e., multiple ATE channels) can converge on a single EDT input, with OR gates at the core boundary enabling this shared connection.

Wiring is synthesized from the scheduler's assignments using a simple greedy rule: identify the ATE channel that appears most frequently, hard-wire it to EDT input 1 everywhere it is needed, and then repeat for the next most frequent channel and the following available EDT input. This is continued until every required channel is routed. This strategy minimizes the total wiring and keeps the OR-gate fan-in low.

In earlier work [33], the author examined a comparable strategy and reported the results. For each design and allocation method, it lists the minimum number of ATE channels—and the corresponding reduction factor—required to maintain test application time virtually unchanged. The data show that the number of EDT inputs can be reduced by up to a factor of five without lengthening the schedule. Pushing bandwidth even lower introduces a useful trade-off. Although fewer input channels gradually extend the run time, this is offset by compression gains of up to 5×. Accordingly, effective EDT compression for the two designs, D1 and D2, increases from 78× and 393× to 392× and 1,770×, respectively.

G. Test-pattern Independent TAM

The method described in Section IV F is tightly coupled to the final test patterns. With it, an optimal TAM layout cannot be designed until ATPG is complete. That reliance can delay the design timeline, postponing the physical layout until late in the flow. To avoid this bottleneck, paper [54] also introduces a

generic, nearly optimal TAM architecture that is independent of both the test schedule and pattern set, enabling designers to finalize the TAM early and generate schedules later without revisiting those hardware decisions.

Unlike the earlier pattern-dependent approach, the input test-access network is fixed here early in the design flow, before any ATPG work begins. Because a core's lower-order EDT inputs are used most frequently, the architecture connects those pins to multiple ATE channels, providing the scheduler with additional routing flexibility.

This regular, pre-wired network is suitable for SoCs whose cores are fully isolated. Non-isolated blocks that must be tested together may run into conflicts. For example, two two-input cores cannot be driven in parallel if their shared ATE channels would need to feed both cores at once.

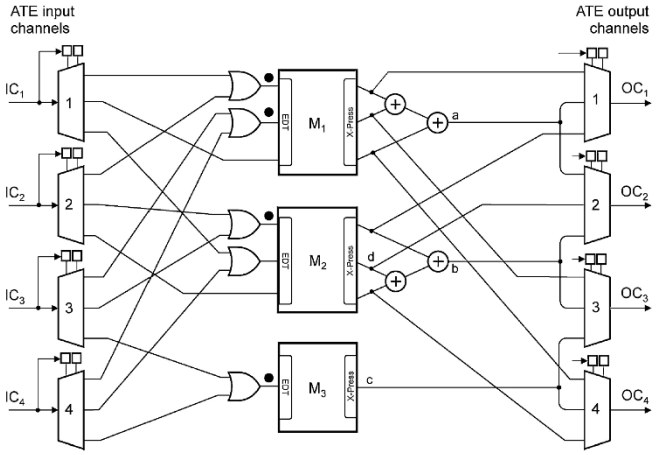

Fig. 15. TAM design [54].

The scheme is far simpler than the sophisticated crossbars used in parallel processors or packet switches. It delivers test-data streams of varying width, so individual EDT inputs within a core may see uneven traffic. The same principles apply on the output side (see Fig. 15). An output switch aggregates each core's compressed responses into a limited set of ATE output channels. The scheme uses the Hopcroft-Karp algorithm for channel allocation.

The study presented results for 5 SOCs (D1, D2,… D5) that showed the number of EDT inputs can be reduced by as much as 8 to 1, while still delivering the full pattern set within the original schedule. Pushing the interface bandwidth even lower introduces a trade-off. Test time rises gradually as channels are removed, but this is offset by sizable compression gains (up to 6×). Accordingly, effective EDT compression climbs from 64× and 61× to 328× and 391× for designs D1 and D2, respectively.

For SoCs whose cores are fully isolated, input- and output-channel counts can be reduced by a factor of four without extending the test. For designs D3, D4, and D5, the scheme increased effective compression from 65×, 78×, and 50× to 323×, 364×, and 320×, respectively.

The authors also presented a similar scheme in [57] for test-pattern-independent scheduling. It showed that compression ranging from 156× to 320× could be achieved using it.

H. Using High Frequency for Test Channels

Mangilal et al. presented an architecture [58] that leverages a high-speed Peripheral Component Interconnect Express (PCIe) bus for structural testing. This high-speed data transfer enables high throughput, resulting in shorter test times. The system allows structural testing directly on the platform, eliminating the need for expensive test equipment. Leveraging the PCIe bus also ensures portability across platforms, supporting both system-level and in-field testing. Thus, the platform enables quick detection of structural faults.

The authors tested the system on SoCs containing multiple chiplets. At the chiplet level, connections are established using the PCIe functional stack. This configuration supports both wafer and die testing. In multi-die packages, data transfers to other chiplets via a functional interface. A Network-On-Chip Controller manages chiplet interconnections.

Soomro et al. also presented an architecture [59] that uses high-speed IO to design a test access mechanism (TAM) for 3D-based SOC.

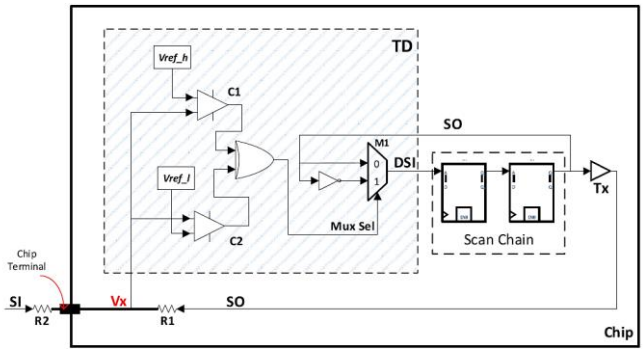

Fig. 16. Test channel with ternary encoding and decoding [59].

Typically, a chip terminal is used at a given time to either send or receive data from a chip. The author presented a method that uses the chip terminal to simultaneously send and receive data. It uses ternary encoding bidirectional signaling. Simultaneous bidirectional signaling is used with full-duplex mode on the chip pins. A single electrical path instead of two could be used with this system, doubling the test channel count and increasing data transfer speed. The system internally uses two virtual unidirectional IOs. The send and receive data are encoded into ternary at the chip boundary, and the information is converted back to binary using a decoder. Fig. 16 shows the design of such a channel. Two resistors of equal value are connected in series. This connection creates a voltage divider that encodes binary values into ternary levels, representing whether both terminals are low, high, or opposite. Using the system, experiments were performed on 4 chips, and the study reports an improvement of up to 53.6%.

I. Test Schedule for Low-Power Designs

In [60], the authors presented a test scheduler for a multi-core, multi-voltage system of chips. It considers power constraints while scheduling the test. Multiple voltages are used in low-power design circuits. A voltage-dependent defect becomes active when a specific voltage is applied. In particular, when a large number of cores operate with many voltage levels, test scheduling becomes quite complex. These settings and

complexity lead to a large test data volume and increase overall test cost. The authors proposed a test scheduling method that uses a Time-Division Multiplexing-based schedule to increase parallelism. The scheduler includes a Graphical User Interface that provides easier control over parameters such as voltage levels, core selection, and power constraints. Test engineers and researchers could use the system to define a test schedule for multi-voltage SOCs.

V. Discussion

The benefits of various approaches, along with the main challenges in test access mechanisms design and test scheduling methodologies, are discussed in this section. It also discusses a direction for future work.

- TAM with improved EDT injectors: This technique improves the encoding efficiency of EDT, thereby improving its performance further. It preserves all the benefits of EDT with no negative impact on design, test, and manufacturing costs.
- TAM with On-Chip Compare: The TAM presented here adds hardware to perform scan output data on chip. This scheme is efficient, especially for chips containing multiple instances of cores. It also provides various modes which are useful at different stages of the product life cycle, and could be useful to collect failure data and yield learning.
- TAM with Efficient Pattern Generation at Higher-Level: This technique proposes a hybrid test methodology and revisits the benefits of generating patterns for groups of blocks/cores. Having its own compression logic for smaller blocks offers advantages, such as reducing routing overhead and the number of power-isolation cells and/or level shifters when the blocks operate across multiple power domains.
- Channel Sharing and Broadcasting: This method uses the same channels for multiple cores. Such a simpler mechanism could improve compression by 2× for medium-sized SOC.
- Dynamic Bandwidth Management with Channel Sharing: Results presented using this method showed that channel sharing, along with broadcasting, must be used wherever possible. Further dynamic bandwidth management is better than static bandwidth management. Combining channel sharing/ broadcasting with dynamic bandwidth management yields the best results.
- TAM Optimization Using Pattern Suite: This scheme allocates channels dynamically using a test scheduling algorithm and configurable TAM. Channels are allocated to cores based on clusters created for their pattern suite. The scheme uses a best-fit strategy for channel assignment, yielding a remarkably smaller test data volume.
- Test-pattern Independent TAM: This is a generic scheme that could be used to find a test schedule even when no pattern data is available. It uses a new solver, test-scheduling algorithms, and TAM design schemes to dynamically allocate channels.
- Using High Frequency for Test Channels: These techniques use high-speed pins for transporting test data. Low-speed operation could be allowed for the logic inside the chip. Having a high transfer rate externally effectively increases overall throughput. Additionally, this technique uses system pins (e.g., PCIe), which are already present on the chip and would otherwise remain unused and wasted if only General Purpose Input/Output (GPIO) (which are slow) were used for test data transport.
- Test Schedule for Low-Power Designs: With the growing demand for low-power chips and their use for low-cost applications, the cost of testing them has become a challenge. Using multiple voltages increases test data volume due to testing requirements across voltage combinations. The technique presented provides optimal test scheduling across various voltage and core combinations, saving time for the SOC test engineer developing such a test schedule using tedious methods.

VI. Conclusion

The electronics industry's aggressively shrinking chip features and its move toward three-dimensional integrated circuits have had a dramatic impact on SoC design and test procedures. A diverse mix of digital, analog, mixed-signal, memory, optical, microelectromechanical, and radio-frequency cores continues to pose challenges for effective testing. ATE channel bandwidth management for SoC designs plays a key role in increasing test data compression, reducing test time, and thereby reducing test cost. This study presented a survey of methods for allocating channels to cores. A few of these techniques have been implemented by commercial tool developers, making them easier to integrate into the flow. However, the unique advantages of the other techniques are not available in commercial tools. A framework is required to effectively use a combination of techniques, such as TAM with Efficient Pattern Generation at a higher level, along with On-Chip Compare. In addition, shrinking geometries, lower power-consumption specifications, and growing SOC sizes will require testing devices with newer fault models, and that too, with increased frequency specifications. These requirements call for further exploration of TAM design and scheduling techniques.

This study discussed approaches for designing TAMs and test scheduling algorithms, and their efficiency in reducing test time. Designing TAM for both types of flows—hierarchical and module design — is discussed. It also presented various EDT features that could be used to design an efficient TAM for SOC. Further, it summarized the advantages and challenges of each approach. Finally, this study summarizes the challenges and future research directions.

References

[1] A. M. Amory, E. Briao, E. Cota, M. Lubaszewski and F. G. Moraes, "A scalable test strategy for network-on-chip routers," *IEEE International Conference on Test, 2005.*, Austin, TX, USA, 2005, pp. 9 pp.-599.

[2] A. M. Amory, K. Goossens, Erik Jan Marinissen, M. Lubaszewski and F. Moraes, "Wrapper Design for the Reuse of Networks-on-Chip as Test

Access Mechanism," *Eleventh IEEE European Test Symposium (ETS'06)*, Southampton, UK, 2006, pp. 213-218.

[3] K. Stewart and S. Tragoudas, "Interconnect testing for networks on chips," *24th IEEE VLSI Test Symposium*, Berkeley, CA, USA, 2006, pp. 6 pp.-107.

[4] M. Richter and K. Chakrabarty, "Test pin count reduction for NoC-based Test delivery in multicore SOCs," *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Dresden, Germany, 2012, pp. 787-792.

[5] Kuen-Jong Lee, Jih-Jeen Chen and Chen-Hua Huang, "Using a single input to support multiple scan chains," *1998 IEEE/ACM International Conference on Computer-Aided Design. Digest of Technical Papers (IEEE Cat. No.98CB36287)*, San Jose, CA, USA, 1998, pp. 74-78.

[6] Yu Huang, S. M. Reddy and Wu-Tung Cheng, "Core-clustering based SoC test scheduling optimization," *Proceedings of the 11th Asian Test Symposium, 2002. (ATS '02).*, Guam, USA, 2002, pp. 405-410.

[7] G. Giles, J. Wang, A. Sehgal, K. J. Balakrishnan and J. Wingfield, "Test access mechanism for multiple identical cores," *2009 International Test Conference*, Austin, TX, USA, 2009, pp. 1-10.

[8] M. Sharma, A. Dutta, W. -T. Cheng, B. Benware and M. Kassab, "A novel Test Access Mechanism for failure diagnosis of multiple isolated identical cores," *2011 IEEE International Test Conference*, Anaheim, CA, USA, 2011, pp. 1-9.

[9] J. Janicki, "EDT bandwidth management - Practical scenarios for large SoC designs," *2013 IEEE International Test Conference (ITC)*, Anaheim, CA, USA, 2013, pp. 1-10.

[10] B. Keller, "Efficient testing of hierarchical core-based SOCs," *2014 International Test Conference*, Seattle, WA, USA, 2014, pp. 1-10.

[11] J. Rajski, J. Tyszer, M. Kassab and N. Mukherjee, "Embedded deterministic test," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 5, pp. 776-792, May 2004.

[12] Vikram Iyengar, Krishnendu Chakrabarty and E. J. Marinissen, "Test access mechanism optimization, test scheduling, and tester data volume reduction for system-on-chip," in *IEEE Transactions on Computers*, vol. 52, no. 12, pp. 1619-1632, Dec. 2003.

[13] Q. Zhou and K. J. Balakrishnan, "Test Cost Reduction for SoC Using a Combined Approach to Test Data Compression and Test Scheduling," *2007 Design, Automation & Test in Europe Conference & Exhibition*, Nice, France, 2007, pp. 1-6.

[14] "IEEE Standard Testability Method for Embedded Core-based Integrated Circuits," in *IEEE Std 1500-2022 (Revision of IEEE Std 1500-2005)* , vol., no., pp.1-168, 12 Oct. 2022.

[15] Z. S. Ebadi and A. Ivanov, "Design of an optimal test access architecture using a genetic algorithm," *Proceedings 10th Asian Test Symposium*, Kyoto, Japan, 2001, pp. 205-210.

[16] A. Sehgal, S. K. Goel, E. J. Marinissen and K. Chakrabarty, "IEEE P1500-compliant test wrapper design for hierarchical cores," *2004 International Conferce on Test*, Charlotte, NC, USA, 2004, pp. 1203-1212.

[17] V. Iyengar, K. Chakrabarty and E. J. Marinissen, "Test wrapper and test access mechanism co-optimization for system-on-chip," *Proceedings International Test Conference 2001 (Cat. No.01CH37260)*, Baltimore, MD, USA, 2001, pp. 1023-1032.

[18] E. J. Marinissen, S. K. Goel and M. Lousberg, "Wrapper design for embedded core test," *Proceedings International Test Conference 2000 (IEEE Cat. No.00CH37159)*, Atlantic City, NJ, USA, 2000, pp. 911-920.

[19] N. A. Touba and B. Pouya, "Testing embedded cores using partial isolation rings," *Proceedings. 15th IEEE VLSI Test Symposium (Cat. No.97TB100125)*, Monterey, CA, USA, 1997, pp. 10-16.

[20] EDN, "Product How To: DFT strategy for ARM processor-based designs", www.edn.com, Jan 2013.

[21] U. Ingelsson, S. K. Goel, E. Larsson and E. J. Marinissen, "Test scheduling for modular SOCs in an abort-on-fail environment," *European Test Symposium (ETS'05)*, Tallinn, Estonia, 2005, pp. 8-13.

[22] K. Chakrabarty, "Test scheduling for core-based systems using mixed-integer linear programming," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 10, pp. 1163-1174, Oct. 2000.

[23] Yu Huang, "Optimal core wrapper width selection and SOC test scheduling based on 3-D bin packing algorithm," *Proceedings. International Test Conference*, Baltimore, MD, USA, 2002, pp. 74-82.

[24] Vijay Sontakke, John Dickhoff, "Developments in scan shift power reduction: a survey", Bulletin of Electrical Engineering and Informatics (BEEI), 2023.

[25] Vijay Sontakke, John Dickhoff, "A survey of scan-capture power reduction techniques", International Journal of Electrical and Computer Engineering (IJECE), 2023.

[26] Zhiyuan He, Zebo Peng and Petru Eles, "A heuristic for thermal-safe SoC test scheduling," *2007 IEEE International Test Conference*, Santa Clara, CA, 2007, pp. 1-10.

[27] Z. He, Z. Peng, P. Eles, P. Rosinger and B. M. Al-hashimi, "Thermal-Aware SoC Test Scheduling with Test Set Partitioning and Interleaving," *2006 21st IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, Arlington, VA, USA, 2006, pp. 477-485.

[28] S. Samii, M. Selkala, E. Larsson, K. Chakrabarty and Z. Peng, "Cycle-Accurate Test Power Modeling and Its Application to SoC Test Architecture Design and Scheduling," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 5, pp. 973-977, May 2008.

[29] Yu Xia, M. Chrzanowska-Jeske, Benyi Wang and M. Jeske, "Using a distributed rectangle bin-packing approach for core-based SoC test scheduling with power constraints," *ICCAD-2003. International Conference on Computer Aided Design (IEEE Cat. No.03CH37486)*, San Jose, CA, USA, 2003, pp. 100-105.

[30] C. Yao, K. K. Saluja and P. Ramanathan, "Partition Based SoC Test Scheduling with Thermal and Power Constraints under Deep Submicron Technologies," *2009 Asian Test Symposium*, Taichung, Taiwan, 2009, pp. 281-286.

[31] Vikram Iyengar, Krishnendu Chakrabarty and E. J. Marinissen, "Test access mechanism optimization, test scheduling, and tester data volume reduction for system-on-chip," in *IEEE Transactions on Computers*, vol. 52, no. 12, pp. 1619-1632, Dec. 2003.

[32] Jakub Janicki, Jerzy Tyszer, Grzegorz Mrugalski, Janusz Rajski, "Bandwidth-aware test compression logic for SoC designs", *2012 17th IEEE European Test Symposium (ETS)*, 2012, pp. 1-6.

[33] M. Kassab, G. Mrugalski, N. Mukherjee, J. Rajski, J. Janicki and J. Tyszer, "Dynamic channel allocation for higher EDT compression in SoC designs," *2010 IEEE International Test Conference*, Austin, TX, USA, 2010, pp. 1-10.

[34] I. Parulkar, T. Ziaja, R. Pendurkar, A. D'Souza and A. Majumdar, "A scalable, low cost design-for-test architecture for UltraSPARC/spl trade/ chip multi-processors," *Proceedings. International Test Conference*, Baltimore, MD, USA, 2002, pp. 726-735.

[35] M. Riley, L. Bushard, N. Chelstrom, N. Kiryu and S. Ferguson, "Testability features of the first-generation CELL processor," *IEEE International Conference on Test, 2005.*, Austin, TX, USA, 2005, pp. 9 pp.-119.

[36] P. J. Tan, T. Le, K. -h. Ng, P. Mantri and J. Westfall, "Testing of UltraSPARC T1 Microprocessor and its Challenges," *2006 IEEE International Test Conference*, Santa Clara, CA, USA, 2006, pp. 1-10.

[37] R. Molyneaux, T. Ziaja, Hong Kim, Shahryar Aryani, Sungbae Hwang and A. Hsieh, "Design for testability features of the SUN microsystems niagara2 CMP/CMT SPARC chip," *2007 IEEE International Test Conference*, Santa Clara, CA, USA, 2007, pp. 1-8.

[38] S. Makar, T. Altinis, N. Patkar and J. Wu, "Testing of Vega2, a chip multi-processor with spare processors.," *2007 IEEE International Test Conference*, Santa Clara, CA, USA, 2007, pp. 1-10.

[39] K. J. Balakrishnan, G. Giles and J. Wingfield, "Test Access Mechanism in the Quad-Core AMD Opteron Microprocessor," in *IEEE Design & Test of Computers*, vol. 26, no. 1, pp. 52-59, Jan.-Feb. 2009.

[40] T. Wood, "The Test Features of the Quad-Core AMD Opteron-Microprocessor," *2008 IEEE International Test Conference*, Santa Clara, CA, USA, 2008, pp. 1-10.

[41] J. Remmers, M. Villalba and R. Fisette, "Hierarchical DFT methodology - a case study," *2004 International Conferce on Test*, Charlotte, NC, USA, 2004, pp. 847-856.

[42] D. Trock and R. Fisette, "Recursive hierarchical DFT methodology with multi-level clock control and scan pattern retargeting," *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Dresden, Germany, 2016, pp. 1128-1131.

[43] P. Wohl, J. A. Waicukauski, J. E. Colburn and M. Sonawane, "Achieving extreme scan compression for SoC Designs," *2014 International Test Conference*, Seattle, WA, USA, 2014, pp. 1-8.

[44] C. Barnhart, V. Brunkhorst, F. Distler, O. Farnsworth, B. Keller and B. Koenemann, "OPMISR: the foundation for compressed ATPG vectors," *Proceedings International Test Conference 2001 (Cat. No.01CH37260)*, Baltimore, MD, USA, 2001, pp. 748-757.

[45] G. Li, "Hybrid Hierarchical and Modular Tests for SoC Designs," *2015 IEEE 24th North Atlantic Test Workshop*, Johnson City, NY, USA, 2015, pp. 11-16.

[46] Y. Huang, "Test Compression Improvement with EDT Channel Sharing in SoC Designs," *2014 IEEE 23rd North Atlantic Test Workshop*, Johnson City, NY, USA, 2014, pp. 22-31.

[47] G. Giles, J. Wang, A. Sehgal, K. J. Balakrishnan and J. Wingfield, "Test access mechanism for multiple identical cores," *2009 International Test Conference*, Austin, TX, USA, 2009, pp. 1-10.

[48] Y. Huang, "Test Compression Improvement with EDT Channel Sharing in SoC Designs," *2014 IEEE 23rd North Atlantic Test Workshop*, Johnson City, NY, USA, 2014, pp. 22-31.

[49] X. Liu, C. Yu, Y. Qi, Y. Huang and J. Fu, "Case Study of Testing a SoC Design with Mixed EDT Channel Sharing and Channel Broadcasting," *2016 IEEE 25th North Atlantic Test Workshop (NATW)*, Providence, RI, USA, 2016, pp. 12-17.

[50] Yu Huang, S. M. Reddy and Wu-Tung Cheng, "Core-clustering based SoC test scheduling optimization," *Proceedings of the 11th Asian Test Symposium, 2002. (ATS '02).*, Guam, USA, 2002, pp. 405-410.

[51] X. Liu, C. Yu, Y. Qi, Y. Huang and J. Fu, "Case Study of Testing a SoC Design with Mixed EDT Channel Sharing and Channel Broadcasting," *2016 IEEE 25th North Atlantic Test Workshop (NATW)*, Providence, RI, USA, 2016, pp. 12-17.

[52] G. Li, "Hybrid Hierarchical and Modular Tests for SoC Designs," *2015 IEEE 24th North Atlantic Test Workshop*, Johnson City, NY, USA, 2015, pp. 11-16.

[53] Y. Huang, "Case study of bandwidth management in SoC testing," *2017 IEEE North Atlantic Test Workshop (NATW)*, Providence, RI, USA, 2017, pp. 1-6.

[54] J. Janicki, M. Kassab, G. Mrugalski, N. Mukherjee, J. Rajski and J. Tyszer, "EDT Bandwidth Management in SoC Designs," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 12, pp. 1894-1907, Dec. 2012.

[55] D. Czysz, G. Mrugalski, N. Mukherjee, J. Rajski and J. Tyszer, "Reduced ATE Interface for High Test Data Compression," *2011 Sixteenth IEEE European Test Symposium*, Trondheim, Norway, 2011, pp. 99-104.

[56] J. Rajski, J. Tyszer, G. Mrugalski, W. -T. Cheng, N. Mukherjee and M. Kassab, "X-Press: Two-Stage X-Tolerant Compactor With Programmable Selector," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 1, pp. 147-159, Jan. 2008.

[57] J. Janicki, "EDT channel bandwidth management in SoC designs with pattern-independent test access mechanism," *2011 IEEE International Test Conference*, Anaheim, CA, USA, 2011, pp. 1-9.

[58] K. J. Mangilal, M. Yilmaz, V. Agarwal, S. Sarangi and K. Narayanun, "A Scalable & Cost Efficient Next-Gen Scan Architecture: Streaming Scan Test via NVIDIA MATHS," 2024 IEEE International Test Conference (ITC), San Diego, CA, USA, 2024, pp. 400-406.

[59] I. A. Soomro, M. Samie and I. K. Jennions, "Test Time Reduction of 3-D Stacked ICs Using Ternary Coded Simultaneous Bidirectional Signaling in Parallel Test Ports," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 39, no. 12, pp. 5225-5237, Dec. 2020.

[60] F. Vartziotis, "TDMS Test Scheduler: An Integrated Framework for Test Scheduling of DVFS-based SoCs with Multiple Voltage Islands," 2021 IEEE European Test Symposium (ETS), Bruges, Belgium, 2021, pp. 1-2.