Section 8: System Level Test

System level test (SLT) refers to exercising the components of a system as an integrated whole to validate correct system operation for its intended end-use applications. We confine our notion of systems to be primarily electronics-based. As a whole, a system is comprised of both physical hardware and software programs. Electronic hardware scales from single chips, multi-die integrated packages, printed circuit boards (PCB), on up to racks of PCBs. For interacting with the physical world, some system hardware components can fall into non-electronic domains such as mechanical, optical, chemical, and biological. Software includes firmware, device drivers, operating system, and applications. The steady march of Moore's Law in semiconductors has enabled the creation of ever more complex systems. These systems are finding new applications and winding their way ever deeper into our daily lives. This trend is expected to continue at an accelerated pace as exemplified by smart cars, homes, cities, factories, healthcare, agriculture, etc. loosely aggregated under the umbrella term internet-of-things (IoT), enabled by high-data-rate 5G wireless communications.

The purpose and use of SLT differs depending on the perspective of the adopter: a component supplier versus a system integrator. Running SLT is a way for the component supplier to minimize defect returns from the system customer. Having SLT enables earlier time-to-market for the component while buying more time to improve ATE-based production test. The system integrator uses SLT to assess the quality of incoming components from multiple sources as part of an overall system validation process. Complex systems can have many configuration parameters which can be tuned by software to achieve optimal performance. Variability of components and their interactions may require the system integrator to run SLT to tune each system individually.

This section addresses a number of topics addressing SLT:

1. What's driving the trend toward more use of SLT?
2. What are the various SLT approaches and flows being used today?
3. What are the challenges and issues in using SLT?
4. What are the opportunities for improving SLT?
5. How will future system applications affect SLT development?

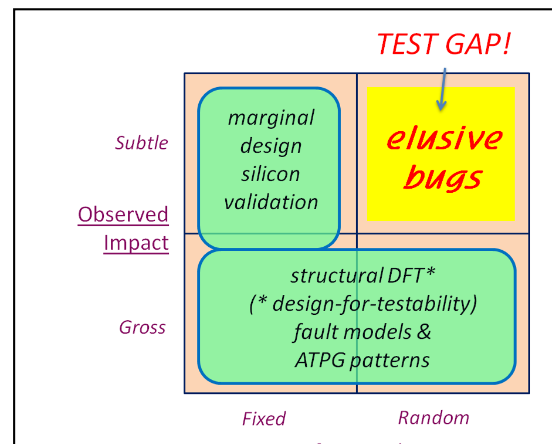

Figure 1. Marginal defects in advanced process nodes escape through test gap left open by existing test methods.

TREND TOWARD SLT ADOPTION

In the early days of simpler electronics, functional test predominated. As designs grew in complexity, propelled along the trajectory guided by Moore's Law, functional testing became untenable due to its high development cost and inadequate fault coverage to meet quality requirements. Scan-based structural test gained prominence since it enabled efficient automatic test pattern generation (ATPG) to achieve high fault coverage. As process dimensions shrank, fault models that underlie structural test kept pace with increasingly complex defect behaviors by evolving from stuck-at to transition-delay to cell-aware, delivering the required quality levels albeit with significantly higher test pattern size. However, the requirements are shifting yet again and the benefits of structural test may be hitting a limit. There is much anecdotal evidence that functional testing in the form of SLT is gaining more usage to augment structural test, in order to catch so-called "marginal" defects missed by ATE-based production test.

The notion of marginal defects arose in connection with inherent limits of optical lithography and process variation control in the manufacturing of devices at advanced nanometer nodes. Traditional test and debug methods are successful at catching defects with a gross observable impact that occur at random locations, or more subtle defects that are systematically localized. Left uncovered is a test gap for marginal defects that are both subtle in impact and randomly occurring (Figure 1). Exacerbating the issue is the growing divergence between test-mode versus in-system operation for today's multi-core system-on-chip (SoC) designs. For production test efficiency, complex clock and power domain interactions found during mission-mode are typically not exercised on ATE. Production test conditions can miss marginal defects which escape and cause failures under system operating conditions. These defects are difficult to resolve between field and factory, earning them the label of no-trouble-found (NTF).

Scan-based structural test is only applicable to synchronous digital logic. Specialized test schemes, usually executed via some form of BIST, exist for memory, high-speed I/O, and some analog functions. However, at the system level, defects that affect interactions among the constituent components are not well-covered by isolated

device/IP-level tests. Adding software to the mix further creates the possibility of soft failures when marginal devices are exercised under certain application scenarios that induce stress, triggering "application-enabled" defects. The trend toward product miniaturization in many end-use applications has driven packaging innovations to create multi-die system-in-package (SiP) solutions. For these advanced packages, testing to assure the structural and functional integrity of individual block-level parts may not be sufficiently comprehensive.

The test gaps identified above form the driving impetus towards SLT adoption. For the system integrator, performing SLT is par for the course, since the system is the end product. However, for the component supplier, SLT has not always been a necessity. To some extent, component functional test (in addition to structural test) performed on the ATE partially fulfills the purpose of SLT and is often good enough to ensure end-user system quality. But when both component and system complexities reach beyond a certain threshold, typical ATE-based functional test can no longer mimic the compendium of system-level interactions. For example, it is difficult to boot the entire Android operating system on a mobile phone SoC on the ATE. For the component supplier, it can be more cost-effective to build specialized SLT environments that can better mimic the end-user system and run a multitude of application scenarios to catch potential system-level defects. Thus, providers of sophisticated SoC and SiP components are seen to be early adopters of SLT.

SLT APPROACHES AND FLOWS

SLT practices vary widely, dictated by factors such as:

- System integrator vs. component supplier role
- Product market segment, unit volume, and lifetime
- Product quality and reliability requirements
- Competitive pressure in terms of time-to-market (TTM) and time-to-volume (TTV)
- Product profit margin

These factors determine how a company develops its own unique SLT approach, which can be characterized in a number of aspects:

- ATE vs. special SLT equipment
- Structural and functional SLT content
- SLT run time
- Full vs. sampled SLT
- SLT thermal, voltage, and cycling conditions
- SLT failure diagnosis

As shown in the example illustration (Figure 2) for a high-volume consumer mobile phone SoC, SLT today is inserted as the last step in the production test flow after wafer probe die test (WP), and packaged chip final test (FT). Typically, SLT is performed on special equipment distinct from WP and FT ATE. In this case, the SoC test board is comprised of the entire mobile phone system where the software stack from firmware to user applications can be run. The SLT flow is controlled by a PC-based test station with handler to load/unload multiple boards for parallel testing. If system components employ various scan-based test access standards such as IEEE 1149.x, 1500, and 1687 in a consistent and compatible manner, structural aspects of SLT to assess system assembly and connectivity can be executed more efficiently than relying purely on functional exercises. Reuse of modular BIST capabilities at the system level is another benefit.

Typical SLT run times can vary from seconds to tens of minutes for high-volume products. For low-volume and long-lifetime products requiring high reliability, the system integrator may run SLT for hours

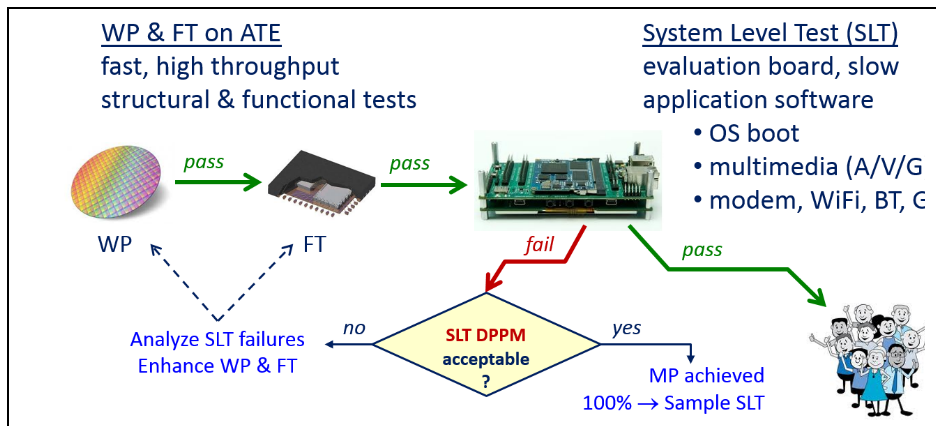

Figure 2. SLT flow for high-volume mobile phone SoC, initially 100% then reduced to sampling mode once WP/FT test quality has been enhanced to achieve acceptable SLT DPPM.

or even days. A component supplier can utilize SLT in the production test flow either as a permanent or a temporary step. Due to the relatively long run time of SLT, including it fully in production requires higher-cost SLT test equipment that can support very high concurrency in order to meet volume throughput demands. Full SLT typically applies to higher-end products selling at a premium, accompanied by higher user expectations. The cost structure of high-volume but lower-end products cannot support full SLT. Instead, to meet early TTM, full SLT is only utilized during initial production to deliver products of good-enough quality. Effort is spent on analyzing SLT failures to enhance WP & FT functional tests with the goal of reducing SLT DPPM. Once SLT DPPM reaches a level corresponding to acceptable customer quality expectations, SLT is shifted from full mode to occasional sampled monitoring. Quickly removing SLT as a throughput bottleneck enables the production TTV goal to be met.

Based on the experience of product deployment and in-field failures encountered, the system integrator may require the supplier to augment SLT with further preventive measures. For example, if power-up failure is noticeably significant, adding multiple cycles of OS boot under thermal stress to SLT may help flag problematic components. For safety-critical applications, it may require a serious and continuous effort in failure diagnosis to identify root-causes, leading to corrective actions including potential design and manufacturing changes.

There is increasing interest to employ big data analytics across the WP-FT-SLT flow to improve process efficiency and accuracy. These usually involve finding signature correlations among test measurements and failing behaviors which can be turned into predictive decisions as part of an adaptive test flow. Another approach being explored by industry is to move SLT upstream to WP, and for SLT to be ATE-based. This approach has two aspects: (1) replicate the full SLT environment and (2) add more SLT-like tests for the component. Being able to perform SLT at the wafer and/or die level will help provide Known-Good-Die (KGD) prior to subsequent multi-die integration steps. However, full replication of a highly complex SLT environment can be very expensive and is rarely justified except in special cases such as critical warfare electronics in a fighter jet. If system complexity is not high, it is possible to create the full system on the ATE load-board and run SLT using a low-cost tester (Figure 3). Besides the device-under-test (DUT), the load-board system includes flash memory to store SLT program components, a simple CPU as test controller, DRAM for system memory, and SLT self-checking circuitry to send DUT results to ATE. Such a "system functional test" approach has been used successfully to achieve exceptional quality for a line of low-cost consumer optical drive ICs.

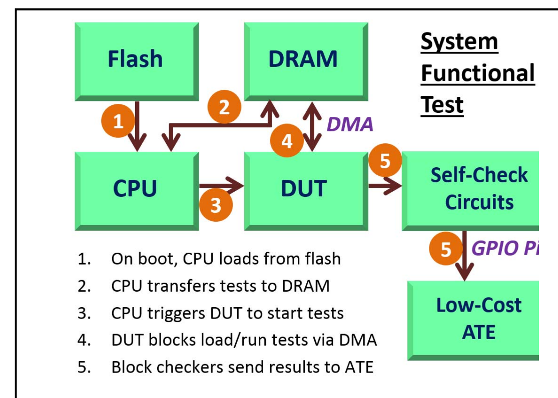

Figure 3. ATE-based system functional test.

Stand-alone component functional test executed on ATE can take advantage of on-chip processor and memory to run light-weight system programs to partially achieve the aims of SLT. Protocol-Aware (PA) testing is a recent development to ease the testing of modern device-to-device high-speed I/O interfaces that have complex non-deterministic protocols. It overcomes severe productivity challenges associated with traditional low-level fixed-timing ATE programming, and enables the test engineer to develop patterns at a higher level of system abstraction. Such patterns can cover more realistic and dynamic I/O protocol transactions to achieve better coverage.

SLT CHALLENGES AND ISSUES

Though its ability to catch defect escapes from WP/FT can be readily demonstrated, SLT's primary issue is the lack of quantifiable metrics to assess its effectiveness. Unlike structural test, SLT does not have well-developed fault models and an established statistical framework to link fault coverage to quality level. The fundamental reason underlying this widely-recognized issue has to do with the much more complex nature of SLT failures. Clearly, defects that manage to escape ATE test are harder to catch by definition. Reasons for failure in SLT are more likely to be murky and involve multiple contributing factors. For example, a compound failure may involve a piece of firmware code operating two interacting marginal devices in a manner that creates an unanticipated scenario that pushes system operation into an unsafe power-thermal zone. When running SLT, a common failure response is for the program to hang and time out without any indication of location and cause. Because SLT lacks the controllability and observability advantages of scan, there can be a long lag as in millions of clock cycles between when an error first occurs to when it propagates to a distant location where system failure is eventually observed.

Needed are methods of systematic SLT failure analysis to understand in more detail the defect mechanisms, error sequences, and associated statistical properties. That understanding then drives the development of effective test and

defect coverage strategies. Many elements of an SLT program are naturally derived from design verification where pattern development is guided by coverage of more abstract entities such as application scenarios, transactions and functions. Missing is a link from the higher-level abstraction to the underlying hardware in terms of covering where defects are more likely to occur, and de-emphasizing areas that are not problematic or already covered by WP/FT tests. Fault grading using traditional gate-level simulation is ineffectual due to severe run-time/capacity limitations and inability to model the entire system encompassing software, behavioral constructs, and non-digital hardware. Uniform treatment of low-level structural fault models without system context may waste computing resources and convey a false sense of adequate coverage. SLT program development, failure analysis, and enhancing ATE functional test remain largely a manual and inherently inefficient endeavor, contributing to its high cost of adoption.

In terms of test equipment, building an ATE-based full SLT environment in CP/FT faces ever higher barriers as system complexity increases. Even if it could be done, the cost is likely to be prohibitive in most situations. Thus, component manufacturers are more likely to choose dedicated SLT equipment used in an additional test insertion. Throughput then becomes the primary issue. A typical component SLT test station today is configured to handle four to six units in parallel. In order to meet high throughput demands under the constraint of long SLT run times, the manufacturer is forced to add more testers, more operators, and more factory floor footprint. Massively parallel SLT architectures are needed to raise throughput measured in unit-per-hour (UPH) by an order of magnitude or more. Being able to perform thermal stress testing is also gaining importance with more SoC devices implemented in FinFET nodes where self-heating is a concern as well as managing heat dissipation in tightly packed multi-die SiP components . Consequently, the SLT environment must support the evaluation of system thermal integrity.

OPPORTUNITIES TO IMPROVE SLT

Though SLT methods and practices are developed in a somewhat ad hoc fashion based on circumstances unique to each organization, many opportunities exist to introduce more formal and systematic approaches that can be shared across the industry to create commonality. Over time, an ecosystem can be built up based on standards, EDA tools, and equipment to enable more purpose-driven and efficient SLT flows.

As groundwork, research should be conducted, both theoretical and experimental, to develop fault models that embody emergent properties of complex component interactions. Complex systems are inherently hazardous since it's nearly impossible to anticipate all scenarios of failure, especially when software (rarely defect-free) and human behavior are integral to the system. Even when anticipated, failure mitigation measures constrained by cost may imply the acceptance of calculated risks during system design, calculations that may contain faulty assumptions. A system-level fault model should encompass, as a minimum, physical mechanisms of variation, noise, and aging, all the way up to application-induced stress scenarios that expose marginalities leading to heightened probability of failure. Having such a fault model will enable further development of system-level capabilities for fault grading, ATPG, and statistical correlation of coverage vs. quality. Unlike traditional device-level fault models, system-level faults may be difficult to explicate by single or simple root-causes. Thus, new approaches to fault diagnosis that consider compound causal relationships need to be investigated.

Since SLT is primarily function-oriented (as opposed to structural), much of the test patterns are procured from design verification (DV) which has also been facing growing design complexity as a serious challenge, to the point that DV now dominates the cost of SoC development. Starting from manually written tests to verify functionality, impressive progress has been made in the development of coverage metrics to assess quality, automated test generation such as constrained-random to improve coverage, and the standardized Universal Verification Methodology (UVM). Though successfully deployed widely, UVM has run into reuse scalability limitations: first in moving up the scope of integration beyond the level of verifying IP blocks to verifying complete systems with embedded software; and second in extending to non-simulation verification platforms such as in-circuit emulation, FPGA prototyping, and post-silicon bring-up.

The latest push by the DV community to overcome these limitations is to develop a common abstract model of verification intent named the Portable Test and Stimulus Standard (PSS). As shown in the PSS diagram (Figure 4), block-level abstract models can be combined directly to construct higher-level system verification tests. The same abstract stimulus model can drive test generation tools to target multiple verification platforms. PSS offers an opportunity for SLT to capitalize on major investments being made by EDA tool providers and users to realize a broad vision of verification reuse. As illustrated, the PSS abstract model can be extended to include system-level fault model and stress scenarios. Test generation tools can be enhanced to traverse the system activity graph, solving a set of constraints derived from SLT perspective to generate test patterns which can be run on production SLT

platforms. The test community should participate in the PSS Working Group (PS-WG) to promote the inclusion of SLT in its scope.

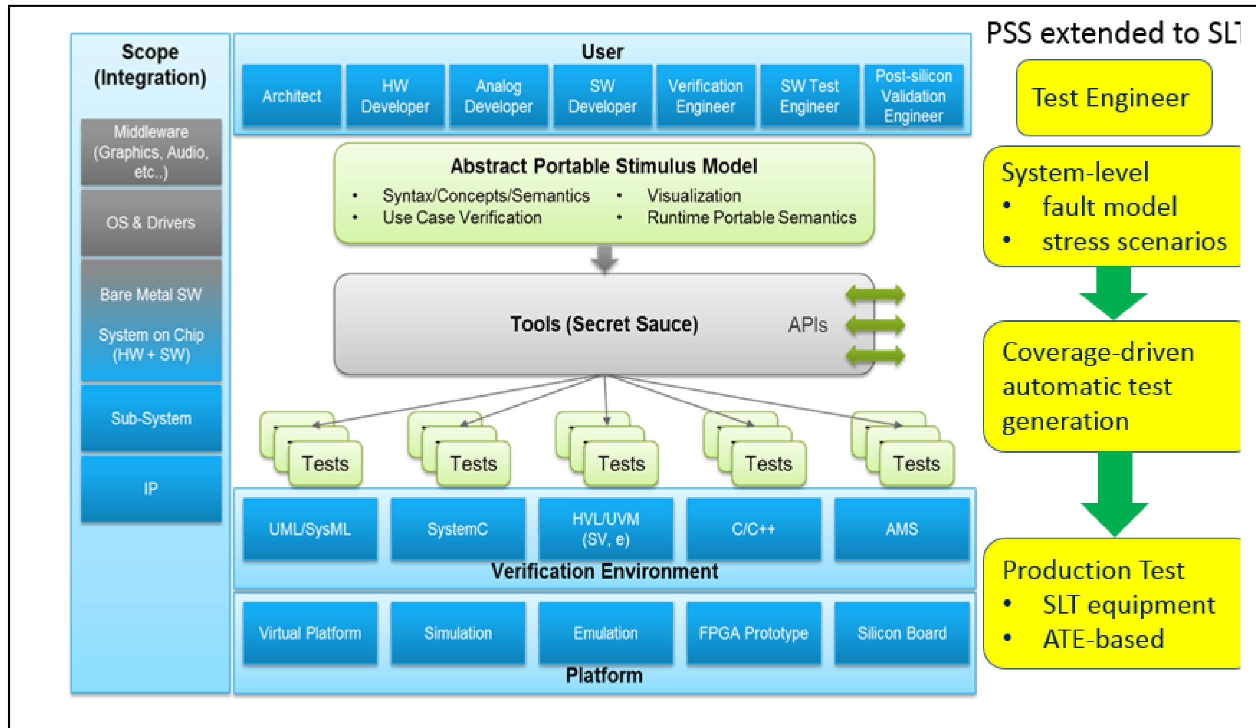

Figure 4. PSS serves as a framework to construct SLT flow that includes fault modeling and test pattern generation.

As with structural test, design modifications to increase controllability and observability should enable more effective and efficient SLT. Due to the close relationship between SLT and post-silicon validation, SLT design-for-testability (SLT-DFT) can draw on the latter's design-for-debug (DFD) practices as well as on-going industry and academic research to advance the state-of-the-art. In SoC designs, it is common to find on-chip instrumentation to support (1) profiling of performance aspects such as power consumption and speed, (2) monitoring of power grid noise and thermal excursions, and (3) debug and diagnosis of silicon failures by signal tracing. Instruments in hardware tend to favor observation so as to not disrupt normal system operation. Software also plays a crucial role in configuring initial silicon states, setting triggers for tracing, transporting logged data off-chip, and analyzing data for high-level debug. SLT can utilize data captured by embedded instruments during system operation to identify vulnerabilities in the silicon and improve failure diagnostic resolution. Of note is the recent appearance of commercial IP for embedded analytics (UltraSoC) and the growing popularity of USB as the device-under-debug to host interface. USB has the advantages of ubiquitous presence in SoC designs and high data transfer rate when compared to IEEE 1149.1 JTAG or IEEE 1687 IJTAG.

Another promising approach called Quick Error Detection (QED) aims to drastically reduce error latency via software transformations applied to existing validation tests. It has the option to insert hardware checkers (incurring a small overhead) to accelerate test execution time. Reduced error latency allows more precise localization of both functional and electrical bugs. QED transformations target bug scenarios abstracted from analysis of a diverse set of actual failures found in commercial multi-core SoCs. Thus, bug scenarios can be viewed as a kind of system-level fault model and the goal of QED is to generate tests to achieve high coverage of bug scenarios.

Embedded instruments provide a wealth of internal device data which can be exploited to optimize the entire production test flow. Adhering to the general concept of adaptive test (Fig. 5), upstream WP/FT data can drive downstream SLT decisions. For example, typical SLT failure rates (so called SLT DPPM) are quite low. Even a high SLT DPPM of 10,000 means 99% of devices will pass SLT. If one can predict based on WP/FT data even a modest portion, say 60%, of devices that are very likely to pass SLT, then limiting SLT to run only on the other 40% will realize large test cost savings. Furthermore, if WP/FT data can indicate potential weak spots in each device, SLT programs can be uniquely customized to target the weak spots and achieve more effective SLT screening, thus realizing higher quality. Customized SLT does require new flexibility for on-the-fly test program construction. To enable adaptive SLT, a learning phase is needed to find strong data correlations between WP/FT and SLT stages. The kinds of test data and correlation methods are active topics of investigation. For example, typical WP/FT test data may lack sufficient information. One approach applies delay-test scan patterns under non-destructive stress conditions on ATE to generate fine-grain localized internal health signatures which can be correlated against SLT pass/fail data to obtain predictive rules.

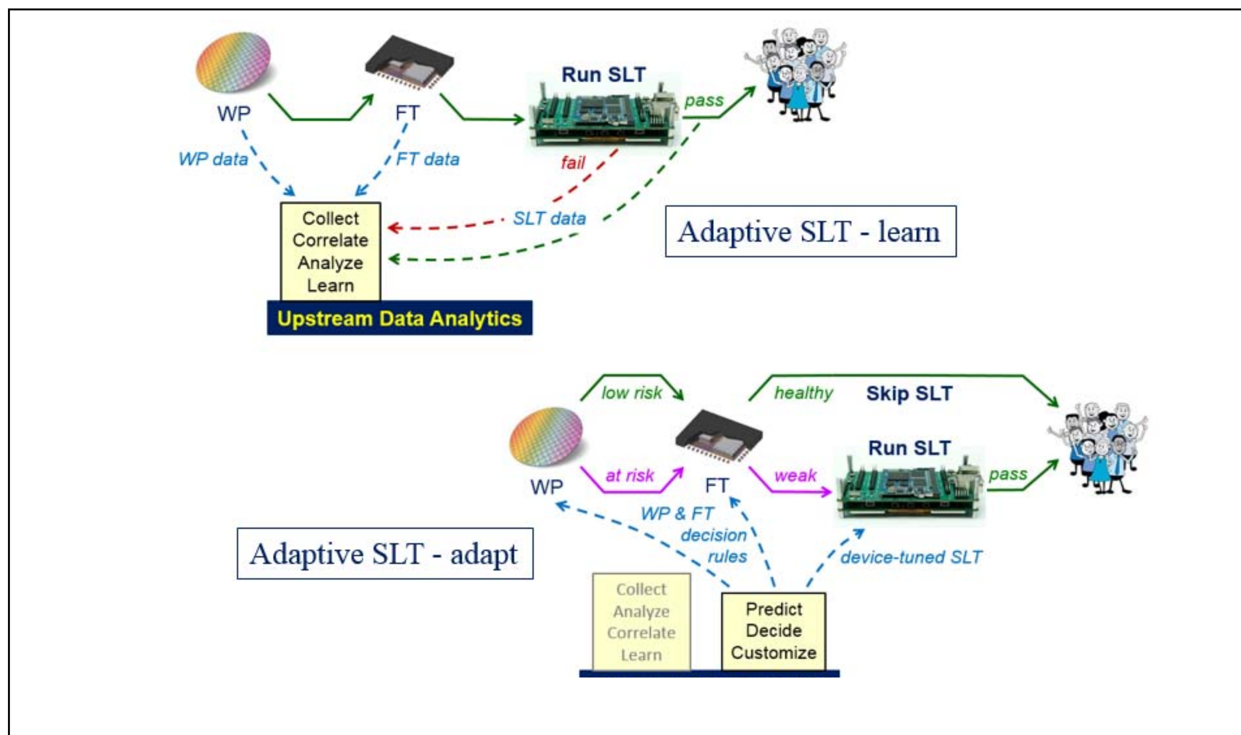

Figure 5. Adapt SLT flow using data analytics to optimize for cost efficiency and test effectiveness.

Sophisticated fault models such as cell-aware are being used today to target subtle system-level faults in WP/FT at great expense in test pattern count. Perhaps a more effective SLT can help off-load WP/FT to achieve an optimal and balanced production test flow. Creating a data loop between component suppliers' WP/FT and system integrators' SLT will require secure data exchange standards and means to protect sensitive proprietary information. Real-time data analytics also create opportunities for test equipment to add value by adopting machine learning accelerators.

IMPLICATIONS FOR SLT FROM FUTURE APPLICATIONS

The modern high-end car is a marvel of systems engineering with hundred-plus electronic control unit (ECU) sub-systems communicating over multi-level networks, all coordinated by software that can reach over 100M lines of code. Yet this level of complexity pales in comparison to the transformation currently underway that will take us into the future of self-driving electric vehicles. Withal the complexity lay the uncompromising requirements for reliability, safety, and resilience over an expected lifetime lasting ten years or more. This future robot on wheels brings together all of the key technological advances in 5G communications, artificial intelligence (AI) computing, and IoT applications. Needless to say, SLT will have to undergo a similar transformation to keep pace. The implications for SLT as applied to the robot car also extend to systems in other domains undergoing similar versions of the "smart" evolution.

Achieving the full potential of self-driving vehicles will require them to be part of an intelligent transportation system where wireless vehicle-to-vehicle/infrastructure (V2X) networking coordinates dynamic local operations. The key enabler is 5G millimeter wave (mmWave) beamforming which provides the fast network acquisition, short latency, and high data rate necessary for ad hoc network formation. Furthermore, due to space and weight constraints, the plethora of additional sensors dictate a portion of in-vehicle networks to also go wireless in support of various IoT-connected services. For 5G mmWave, the traditional cabled and controlled-environment testing approach no longer suffices. Instead, SLT has to be over-the-air (OTA) to check each individual antenna array element as well as the entire array's overall adaptive beamforming performance under realistic noisy conditions.

AI computing introduces software that is, in part, not written by humans, but instead trained by data in the form of deep-learning neural network (DNN) connection weights. The inexact nature of AI computation introduces a testing conundrum: Is a system failure due to defects in the DNN hardware implementation, or to insufficient DNN training that missed rare corner cases in the input data domain? Heavy AI computation also poses a power consumption challenge, especially if power comes mainly from stored energy, e.g., battery. Already inefficient conventional von-Neumann processors are being overtaken by highly parallel arrays of dedicated processing cores with local memory and data-flow accelerators. On the horizon are even more energy-efficient computation techniques spanning the spectrum from devices to algorithms such as analog, memristive, asynchronous, neuromorphic, and approximate. As structural testing techniques for synchronous digital designs do not easily extend to the above, it may be even more incumbent on SLT to fill the testing gap.

Future smart and energy-efficient systems may be architected from the ground up to be inherently redundant and error-tolerant much like the human brain. System construction will be based on complexity management principles of hierarchy, modularity, and abstraction. Such systems may have to operate continuously under extreme conditions with natural aging and degradation over time, thus requiring self-monitoring/testing/healing capabilities beyond T0 (defined as time of product shipment). Though component SLT may still be done as part of a less complex subsystem, SLT itself will evolve into the more encompassing built-in capability of self-aware systems.

Summary

SLT's rise in importance is inevitable as advances in semiconductor process and heterogeneous integration technology lead to ever more complex products. Its practice extends from system integrators to system component suppliers. Subtle software-hardware component interactions defy current production test screens, resulting in end-system application-dependent failures. Though SLT can help as an additional screen, there is much opportunity to improve its efficiency and effectiveness in methodology, automation, and test equipment. SLT also fits naturally within the overall adaptive test flow by utilizing cross-test data analytics. The test community should take advantage of the similarity between SLT and DV (particularly post-silicon validation) to push for PSS extensions to include SLT. Driven by upcoming 5G/IoT/AI-enabled transformations touching every aspect of our lives, SLT will most likely evolve into the self-monitoring/testing/healing capability of future smart systems.

References

S. Biswas and B. Cory, "An industrial study of system-level test," IEEE Design & Test, vol. 29, no. 1, pp. 19–27, February 2012.

P. G. Ryan et al., "Process Defect Trends and Strategic Test Gaps," International Test Conference, October 2014.

Z. Conroy et al., "A practical perspective on reducing ASIC NTFs," International Test Conference, November 2005.

A. Jutman, M. Sonza Reorda and H.-J. Wunderlich, "High Quality System Level Test and Diagnosis," Asian Test Symposium, November 2014.

H. H. Chen et al., "Predicting System-Level Test and In-Field Customer Failures using Data Mining," International Test Conference, September 2013.

Astronics Test Systems, "Massively Parallel System-Level Testing – The new Paradigm for Semiconductor Quality Control," https://www.astronics.com/test-systems/whitepaper---system-level-testing-for-semiconductors

Accellera Systems Initiative, Portable Stimulus Specification Working Group, http://www.accellera.org/activities/working-groups/portable-stimulus

P. Mishra et al., "Post-Silicon Validation in the SoC Era: A Tutorial Introduction," IEEE Design & Test, vol. 34, no. 3, pp. 68–92, May 2017.

M. Sadi et al., "An All Digital Distributed Sensor Network Based Framework for Continuous Noise Monitoring and Timing Failure Analysis in SoCs," Asian Test Symposium, November 2014.

D. Lin et al., "Effective Post-Silicon Validation of System-on-Chips Using Quick Error Detection," IEEE Trans. on Comp.-Aided Design of Integrated Circuits and Systems, vol. 33, no. 10, pp. 1573–1590, October 2014.

C. G. Shirley et al., “Board Manufacturing Test Correlation to IC Manufacturing Test,” International Test Conference, October 2014.
H. H. Chen et al., “Statistical Techniques for Predicting System-Level Failure using Stress-Test Data,” VLSI Test Symposium, April 2015.
H. H. Chen, “Data analytics to aid detection of marginal defects in system-level test,” Int. Symposium on VLSI Design, Automation and Test, April 2016.
A. Smith, “5G and IoT create a market for mmWave,” https://www.electronicsweekly.com/news/design/communications/5g-and-iot-create-a-market-for-mmwave-2017-06/