INTERNATIONAL TECHNOLOGY ROADMAP FOR SEMICONDUCTORS

2011 EDITION

DESIGN

THE ITRS IS DEVISED AND INTENDED FOR TECHNOLOGY ASSESSMENT ONLY AND IS WITHOUT REGARD TO ANY COMMERCIAL CONSIDERATIONS PERTAINING TO INDIVIDUAL PRODUCTS OR EQUIPMENT.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

DESIGN

SCOPE

Design technology (DT) enables the *conception*, *implementation*, and *validation* of microelectronics-based systems. Elements of DT include *tools*, *libraries*, *manufacturing process characterizations*, and *methodologies*. DT transforms ideas and objectives of the electronic systems designer into manufacturable and testable representations. The role of DT is to enable profits and growth of the semiconductor industry via cost-effective production of designs that fully exploit manufacturing capability. In the 2011 ITRS, the International Technology Working Group (ITWG) for Design is responsible for the Design and *System Drivers* chapters, along with models for clock frequency, layout density, and power dissipation in support of the Overall Roadmap Technology Characteristics. Specific DT challenges and needs are mapped, as appropriate, to System Drivers. Readers of this chapter are encouraged to also review previous editions of the ITRS Design Chapter, which provide excellent and still-relevant summaries of DT needs.

The main message in 2011 remains—*Cost (of design) is the greatest threat to continuation of the semiconductor roadmap.* Cost determines whether differentiating value is best achieved in software or in hardware, on a programmable commodity platform, or on a new IC. Manufacturing non-recurring engineering (NRE) costs are on the order of millions of dollars (mask set + probe card); design NRE costs routinely reach tens of millions of dollars, with design shortfalls being responsible for silicon re-spins that multiply manufacturing NRE. Rapid technology change shortens product life cycles and makes time-to-market a critical issue for semiconductor customers. Manufacturing cycle times are measured in weeks, with low uncertainty. Design and verification cycle times are measured in months or years, with high uncertainty.

Previous editions of the ITRS have noted a *design productivity gap*—the number of available transistors growing faster than the ability to meaningfully design them. This gap impacts IC product value, placing at risk foundry amortization, return on investment (ROI) for supplier industries, and indeed the entire semiconductor investment cycle. Yet, investment in process technology continues to dominate investment in design technology. The DT roadmap enables control of design costs, as shown in Figure DESN1. In the figure:

1) Hardware aspects of design continue to witness soaring verification team sizes and test costs, and design for manufacturability (DFM) issues now permeate the design flow. DT innovations keep hardware design costs in check, at an estimated $25.7M in 2011 for the consumer portable system-on-chip (SOC-CP) defined in the *System Drivers Chapter*, versus around $7,708M had DT innovations between 1993 and 2009 not occurred.

2) Software aspects of IC design can now account for 80% or more of embedded systems development cost, with the industry having fully entered the era of multi-core designs, with both homogeneous and heterogeneous architectures. In 2011, with the addition of hardware-related software development cost, overall design cost reaches $39.8M. Many DT innovations required in the next 15 years address software aspects of design; indeed, the design of concurrent software is noted as a long-term "Grand Challenge" for DT in the 2011 ITRS *Executive Summary*.

Failure to effectively develop and deploy the roadmap of DT innovations will break the long-standing trend of progress in the semiconductor industry. Hence, we view these DT gaps as crises that must be addressed in the next 15 years.

This chapter first presents *silicon complexity* and *system complexity* challenges, followed by five *crosscutting challenges* (productivity, power, manufacturing integration, interference, and error tolerance) that permeate all DT areas. The bulk of the chapter then sets out detailed challenges in the form of design technology requirements and solutions tables that comprise a *quantitative* design technology roadmap. The organization follows a traditional landscape of DT areas: design process; system-level design; logical, circuit and physical design; design verification; design test, and design for manufacturability.[1] These challenges are discussed at a level of detail that is actionable by management, R&D, and academia in the target supplier community, particularly the electronic design automation (EDA) industry. When appropriate, the detailed challenges are mapped to the microprocessor (MPU), system on chip (SOC, consumer or networking), analog/mixed-signal (AMS), and memory system drivers. Most challenges map to MPU and SOC, reflecting today's EDA technology and market segmentation. A brief unified overview of AMS-specific DT is also

[1] *Additional discussion of analog/mixed-signal circuits issues is contained in the System Drivers Chapter (AMS Driver). Test equipment and the test of manufactured chips are discussed in the Test Chapter, while this chapter addresses design for testability, including built-in self test (BIST).*

provided. The overall approach to the chapter organization and discussion reflects the rise of application- and driver-specific DT.

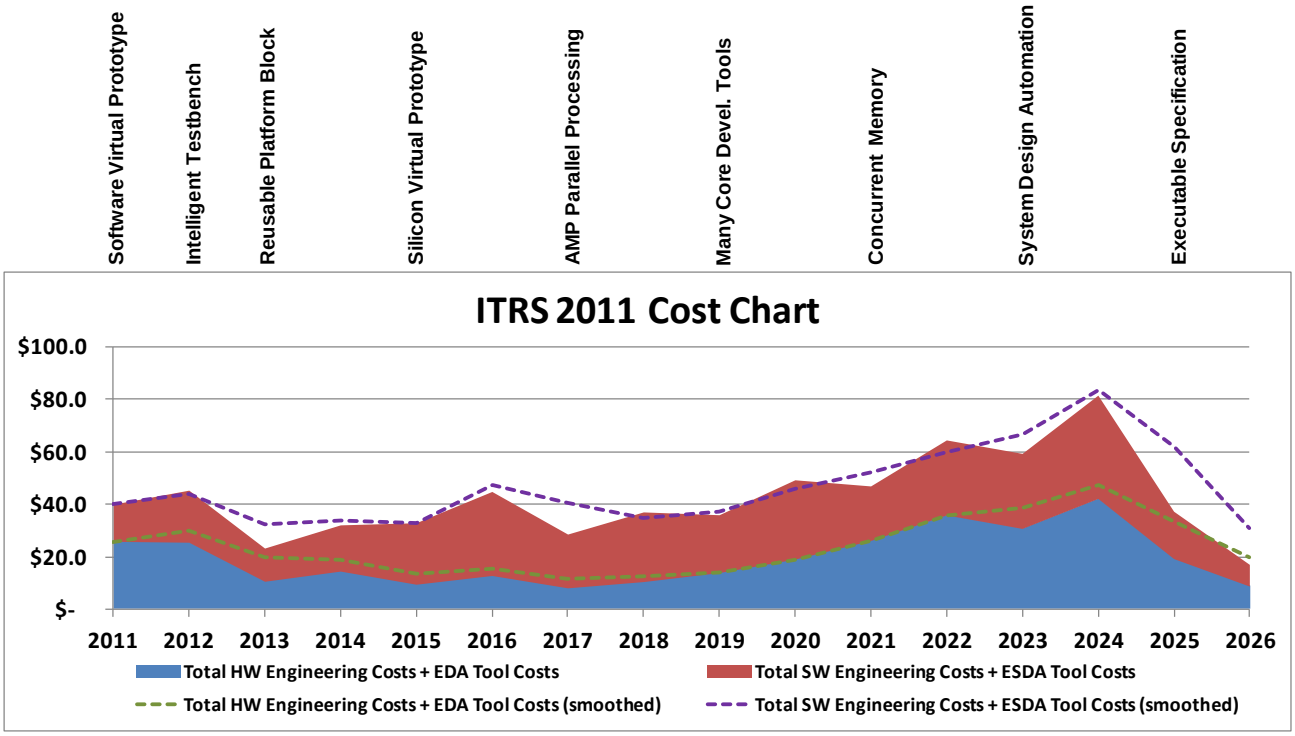

Figure DESN1 Impact of Design Technology on SOC Consumer Portable Implementation Cost

Roadmapping of DT is different from roadmapping of manufacturing technology. Manufacturing technology seeks to implement a set of requirements, and faces limits imposed by physical laws and material properties. In contrast, DT seeks to optimize designs so that they will meet their market requirements, and faces limitations imposed by computational intractability, the unknown scope of potential applications, and the multi-objective nature of design optimization. Because underlying optimizations are intractable, heuristics are inherent to DT, as are practical tradeoffs among multiple criteria such as density, speed, power, testability, or turnaround time. Evaluation of DT quality is thus context-sensitive, and dependent on particular methodologies or design instances. Furthermore, while ITRS technology generations occur discretely when all needed technology elements are in place, DT improvements can increase productivity or quality even "in isolation," and are thus deployable when developed.

OVERALL CHALLENGES

DT faces two basic types of complexity—*silicon complexity* and *system complexity*—that follow from roadmaps for ITRS manufacturing technologies.

Silicon complexity refers to the impact of process scaling and the introduction of new materials or device/interconnect architectures. Many previously ignorable phenomena now have great impact on design correctness and value:

- *Non-ideal scaling of device parasitics and supply/threshold voltages* (leakage, power management, circuit/device innovation, current delivery)
- *Coupled high-frequency devices and interconnects* (noise/interference, signal integrity analysis and management, substrate coupling, delay variation due to cross-coupling)
- *Manufacturing variability* (statistical process modeling and characterization, yield, leakage power, with implications for, and impacts on, library characterization, analog and digital circuit performance, error-tolerant design, layout reuse, reliable and predictable implementation platforms)

- *Complexity of manufacturing handoff* (reticle enhancement and mask writing/inspection flow, NRE cost)
- *Scaling of global interconnect performance* relative to device performance (communication, synchronization)
- *Decreased reliability* (gate insulator tunneling and breakdown integrity, joule heating and electromigration, single-event upset, general fault-tolerance)
- *Codesign* (additional complexity of codesign of multiple chips and packages, especially with the introduction of 3D stacking technologies with Through-Silicon Vias (TSVs))

Silicon complexity places long-standing paradigms at risk, as follows: 1) system-wide synchronization becomes infeasible due to power limits and the cost of robustness under manufacturing variability; 2) the CMOS transistor becomes subject to ever-larger statistical variabilities in its behavior; and 3) fabrication of chips with 100% working transistors and interconnects becomes prohibitively expensive. Available implementation fabrics (from direct-mapped custom through general-purpose software-programmable) easily span four orders of magnitude in performance metrics (e.g., GOps/mW), and there is added opportunity to leave value on the table via ill-advised guardbands, abstractions, or other methodological choices. These challenges demand more broadly trained designers and design technologists, as well as continued mergers between traditionally separated areas of DT (synthesis-analysis, logical-physical, etc.).

System complexity refers to exponentially increasing transistor counts enabled by smaller feature sizes and spurred by consumer demand for increased functionality, lower cost, and shorter time-to-market.[2] Many challenges are facets of the nearly synonymous *productivity* challenge. Additional complexities (system environment or component heterogeneity) are forms of *diversity* that arise with respect to system-level SOC integration, and 3D integration. Design specification and validation become extremely challenging, particularly with respect to complex operating contexts. Tradeoffs must be made between all aspects of value or quality, and all aspects of cost. (A simplistic example: "Moore's Law" for performance might suggest a tradeoff of design time (= time-to-market) for performance at roughly 1% per week.) Implied challenges include:

- *Reuse*—support for hierarchical design, heterogeneous SOC integration (modeling, simulation, verification, test of component blocks) especially for analog/mixed-signal
- *Verification and test*—specification capture, design for verifiability, verification reuse for heterogeneous SOC, system-level and software verification, verification of analog/mixed-signal and novel devices, self-test, intelligent noise/delay fault testing, tester timing limits, test reuse
- *Cost-driven design optimization*—manufacturing cost modeling and analysis, quality metrics, co-optimization at die-package-system levels, optimization with respect to multiple system objectives such as fault tolerance, testability, etc.
- *Embedded software design*—predictable platform-based electronic system design methodologies, codesign with hardware and for networked system environments, software analysis and verification
- *Reliable implementation platforms*—predictable chip implementation onto multiple circuit fabrics, higher-level handoff to implementation
- *Design process management*—design team size and geographic distribution, data management, collaborative design support, "design through system" supply chain management, metrics and continuous process improvement

Together, the silicon and system complexity challenges imply *superexponentially increasing complexity* of the design process. To deal with this complexity, DT must provide concurrent optimization and analysis of more complex objectives and constraints, acknowledge added considerations such as design reuse and manufactured system cost in the design optimization, and encompass added scope such as embedded software design and interfaces to manufacturing. The sheer breadth of silicon and system complexities is also a challenge to roadmapping of DT and the electronic design automation (EDA) industry.

Five *crosscutting challenges*—1) design productivity (a near-term "Grand Challenge" for DT noted in the 2011 ITRS *Executive Summary*), 2) power management (a near-term Grand Challenge), 3) design for manufacturability (a near-term Grand Challenge), 4) interference, and 5) reliability and resilience (a long-term Grand Challenge)—underlie the design cost "meta-challenge" and have potential solutions that span all areas of DT. Three are highlighted as difficult challenges in the ITRS Executive Summary. *Design productivity* (a "cost-effective manufacturing" challenge) is closely linked to system and design process complexity, and is the most massive and critical DT, challenge both near and long term.

[2] *A "Law of Observed Functionality," notorious in consumer electronics, states that transistor count increases exponentially while the system value (utility) increases linearly (see T. Claasen, "The Logarithmic Law of Usefulness," Semiconductor International, July 1998).*

Power management (an "enhancing performance" challenge) oscillates between a performance-driven active power crisis and a variability-driven leakage power crisis in the near term. *Design for Manufacturing* (a "cost-effective manufacturing" challenge) is required for the industry to produce chips in large quantities at acceptable cost and on economically feasible schedules: the past focus on lithography hardware limitations will broaden as variability in its many forms becomes a crisis, and deep integrations of DFM with yield management and design for test are needed. Table DESN1 summarizes key aspects of the crosscutting DT challenges.

Crosscutting Challenge 1: Design Productivity. To avoid exponentially increasing design cost, overall productivity of designed functions on chip must scale at > 2× per technology generation. Reuse productivity (including migration and analog, mixed-signal, and RF (AMSRF) core reuse) of design, verification and test must also scale at > 2× per technology generation. Implied needs are in: (1) verification, which is a bottleneck that has now reached crisis proportions, (2) reliable and predictable silicon implementation fabrics that support ever-high level electronic system design handoff, (3) embedded software design, which has emerged as the most critical challenge to SOC productivity, (4) particularly for the MPU context, improved productivity of large, distributed design organizations that work with tools from a variety of sources, and (5) automated methods for AMS design and test, which are required by the SOC and AMS system drivers. And (6) methods for optimal codesign of 3D systems, particularly 3D systems integrated using Through-Silicon Vias (TSVs) and emerging monolithic 3D technologies. These improvements require metrics of normalized design quality as a function of design quality, design NRE cost, manufacturing NRE cost, manufacturing variable cost, and semiconductor product value. Metrics of design technology quality such as stability, predictability, and interoperability must be improved as well. Time-to-market of new design technology must be reduced, e.g., via standards and platforms for interoperability and DT reuse.

Table DESN1 Overall Design Technology Challenges

Challenges ≥ 22nm	*Summary of Issues*
Design productivity	System-level: high level of abstraction (HW/SW) functionality spec, platform-based design, multi-processor programmability, system integration, AMS codesign and automation Verification: executable specification, ESL formal verification, intelligent test bench, coverage-based verification Logic/circuit/physical: analog circuit synthesis, multi-objective optimization Logic/circuit/physical: SIP and 3D (TSV-based) planning and implementation flows Heterogeneous component integration (optical, mechanical, chemical, bio, etc.)
Power consumption	Logic/circuit/physical: dynamic and static, system- and circuit-level power optimization
Manufacturability	Performance/power variability, device parameter variability, lithography limitations impact on design, mask cost, quality of (process) models ATE interface test (multi-Gb/s), mixed-signal test, delay BIST, test-volume-reducing DFT
Interference	Logic/circuit/physical: signal integrity analysis, EMI analysis, thermal analysis
Reliability and resilience	Logic/circuit/physical: MTTF-aware design, BISR, soft-error correction

Challenges <22nm	*Summary of Issues*
Design productivity	Verification: complete formal verification of designs, complete verification code reuse, complete deployment of functional coverage Tools specific for SOI and non-static logic, and emerging devices Cost-driven design flow
Power consumption	Logic/circuit/physical: SOI power management Logic/circuit/physical: Reliability and resilience- and temperature-constrained 3D physical implementation flows
Manufacturability	Uncontrollable threshold voltage variability Advanced analog/mixed signal DFT (digital, structural, radio), "statistical" and yield-improvement DFT Thermal BIST, system-level BIST
Interference	Interactions between heterogeneous components (optical, mechanical, chemical, bio, etc.)
Reliability and resilience	Autonomic computing, robust design, SW reliability and resilience

ATE—automatic test equipment BISR—built-in self repair BIST—built-in self test DFT—design for test EMI—electromagnetic interference ESL—Electronic System Level HW/SW—hardware/software MTTF—mean time to failure SOI—silicon on insulator

Crosscutting Challenge 2: Power Management. Non-ideal scaling of planar CMOS devices, together with the roadmap for interconnect materials and package technologies, presents a variety of challenges related to power management and current delivery. (1) Both the MPU and Consumer Portable drivers in the *System Drivers Chapter* require flat active and standby power, even as logic content and throughput continue to grow exponentially. DT must address the resulting *power management gap*. (2) Increasing power densities worsen thermal impact on reliability and resilience and performance, while decreasing supply voltages worsen leakage currents and noise. These trends stress on-chip interconnect resources (such as to control V = IR power supply drop in light of the *Assembly and Packaging* roadmap for bump count and passivation opening size), ATE equipment limits, and burn-in paradigms. The emergence of 3D technologies further complicates this problem as chips have to be codesigned to deliver power to co-stacked sub-systems while managing the number and location of TSVs. (3) Integration of distinct high-performance, low operating power (LOP), and low standby power (LSTP) devices demands power optimizations that simultaneously exploit many degrees of freedom, including multi-V_{th}, multi-T_{ox}, multi-V_{dd} coexisting in a single core—while guiding additional power optimizations at the architecture, operating system, and application software levels. (4) Leakage power varies exponentially with key process parameters such as gate length, oxide thickness and threshold voltage; this presents severe challenges to both analysis and optimization in light of both scaling and variability.

Crosscutting Challenge 3: Design for Manufacturing. "Red bricks," that is, technology requirements for which no known solutions exist, are increasingly common throughout the ITRS. At the same time, challenges that are impossible to solve within a single technology area of the ITRS may be solvable (more cost-effectively) via appropriate synergies with DT. New interconnections between design and all other manufacturing-related disciplines lead to the rise of design for manufacturability (DFM), to which this DT roadmap devotes an entire section. Indeed, the feasibility of future technology nodes will come to depend on this communication. Several examples are as follows. (1) Tester equipment cost and speed limitations may be addressed by more rapid adoption of new fault models (for example, crosstalk, path delay), along with corresponding automatic test pattern generation (ATPG) and BIST techniques. (2) System implementation cost, performance verification, and overall design turnaround time (TAT) may be improved through die-package-board co-optimization and analysis, as well as DT for system-in-package design. (3) CD control requirements in the *Lithography*, *Process Integration, Devices, and Structures* (PIDS), *Front-End Processing* (FEP), and *Interconnect* technology areas may be relaxed by new DT for correctness under manufacturing variability (e.g., variability-aware circuit design, regularity in layout, timing structure optimization, and static performance verification). (4) Manufacturing non-recurring costs can be reduced by more intelligent interfaces to mask production and inspection flows.

Crosscutting Challenge 4: Interference. Resource-efficient communication and synchronization, already challenged by global interconnect scaling trends, are increasingly hampered by noise and interference. Prevailing signal integrity methodologies in logical, circuit and physical design are reaching their limits of practicality. These methodologies include repeater insertion rules for long interconnects, slew rate control rules, power/ground distribution design for inductance management, etc. Scaling and SOC integration of mixed-signal and (radio frequency) RF components will require more flexible and powerful methodologies. Issues include noise headroom (especially in low-power devices and

dynamic circuits), large numbers of capacitively and inductively coupled interconnects, supply voltage IR drop and ground bounce, thermal impact on device off-currents and interconnect resistivities, and substrate coupling. A basic DT challenge is to improve characterization, modeling, analysis, and estimation of noise and interference at all levels of design.

Crosscutting Challenge 5: Reliability and Resilience. Relaxing the requirement of 100% correctness for devices and interconnects may dramatically reduce costs of manufacturing, verification, and test. Such a paradigm shift will likely be forced in any case by technology scaling, which leads to more transient and permanent failures of signals, logic values, devices, and interconnects. Several example issues are as follows. (1) Below 65nm, single-event upsets (soft errors) impact field-level product reliability, not only for embedded memories, but for logic and latches as well. (2) Methods for accelerated lifetime testing (burn-in) become infeasible as supply voltages decrease (resulting in exponentially longer burn-in times); even power demands of burn-in ovens become overwhelming. (3) Atomic-scale effects can demand new "soft" defect criteria, such as for non-catastrophic gate oxide breakdown or highly resistive vias. In general, automatic insertion of robustness into the design will become a priority as systems become too large to be functionally tested at manufacturing exit. Potential solutions include automatic introduction of redundant logic and on-chip reconfigurability for fault tolerance, development of adaptive and self-correcting or self-healing circuits, and software-based fault-tolerance.

DETAILED DESIGN TECHNOLOGY CHALLENGES

The remainder of this chapter gives an overview of design methodology, followed by quantified challenges and potential solutions in five main areas of DT. As noted above, most challenges map to SOC, reflecting today's EDA technology and market segmentation.

DESIGN METHODOLOGY

The process of designing and implementing a chip requires a large collection of *techniques*, or *tools*, and an effective *methodology* by which a designer's input predictably leads to a manufacturable product.[3] While considerable attention is given to the tools needed, the equally important subject of design methodology is often neglected. Each technology generation requires designers to consider more issues; hence, new analysis methods and tools must be developed to evaluate new phenomena and aid the designer in making critical design decisions. An even greater challenge is to determine the most effective sequence in which issues should be considered and design decisions made, so as to minimize iterations.

With the transition from microelectronics to nanoelectronics along "More Moore," "More than Moore," and "Beyond CMOS" trajectories come inevitable paradigm shifts in the design of silicon systems. These affect all levels of the design process, and require enormous effort toward new methodologies and tools. DT must enable the creation of highly complex yet cost-efficient silicon system solutions, while exploiting all available opportunities afforded by nanoelectronics. For innovative applications (see the *System Drivers Chapter* for more details) to become affordable, daunting EDA challenges must be overcome. Shrinking of silicon devices and related fabrication processes has been the foundation for ever more powerful integrated silicon solutions that permeate daily life. Delivery of such silicon solutions has relied on the availability of EDA and design technologies that smoothly transform specification data into manufacturing-ready layout data, and perform necessary verifications at different levels of abstraction (see Figure DESN2). However, the continued availability of such design technologies can no longer be taken for granted.

Cost and time-to-market of new SOCs requires DT that spans all parts of a complex design process (Figure DESN2) which consists of two main elements: the implementation path (left) and the verification path (right). The figure shows the so-called V-Cycle of a design system architecture integrating both "More Moore" and "More than Moore" design aspects. The arrows indicate available (**green**) and partially available (**yellow**) elements of a state of the art design system environment. Future requirements for EDA are indicated in **red**.

[3] *Design methodology is developed jointly by designers and design technologists; it is the sequence of steps by which a design process will reliably produce a design "as close as possible" to the design target while maintaining feasibility with respect to constraints. Design methodology is distinct from design techniques, which pertain to implementation of the steps that comprise a methodology and are discussed in the context of their respective DT areas. All known design methodologies combine 1) enforcement of system specifications and constraints via top-down planning and search, with 2) bottom-up propagation of constraints that stem from physical laws, limits of design and manufacturing technology, and system cost limits.*

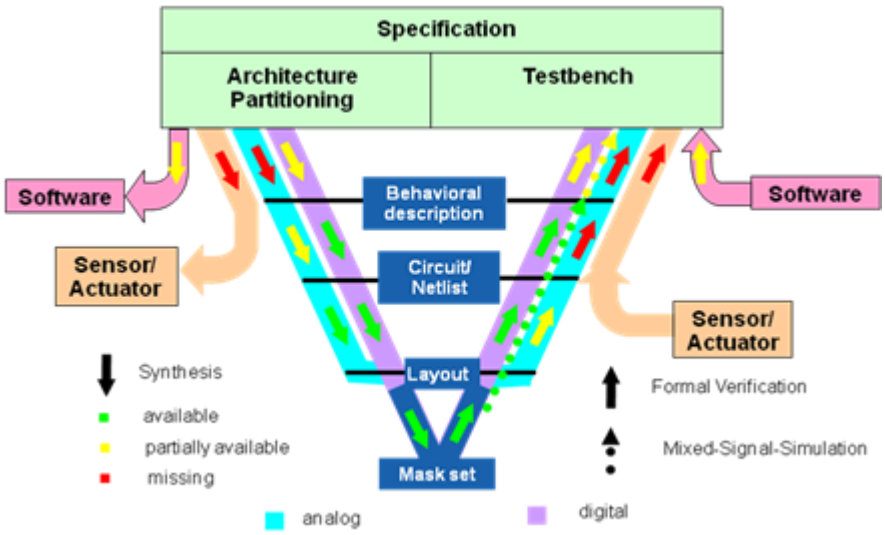

Figure DESN2 The V-Cycle for Design System Architecture[4]

A system specification undergoes stepwise refinement from architecture down to mask set. The idea of SOC design is to explore the system in initial stages; then, while transforming the system-level description to more detailed levels (behavioral, circuit/netlist, layout and mask), a verification methodology must "guard", i.e., prove correctness of, each incremental refinement step. Beyond the SOC's components and behavior, system software and sensors/actuators must also be handled by future implementation and verification procedures (**yellow** and **red**). The digital design path (**purple**) of the design system will require tools and methodologies above the behavioral abstraction level (**yellow**). The analog design path is even more challenging (indicated by several **red** arrows) because complete solutions are rarely found at all level of abstractions. As a result, an automated analog design process (**light blue**) is still an open challenge in today's design environments. To summarize, future requirements are twofold. 1) More than Moore scaling requires the integration of aspects outside the traditional SOC flow, notably software on the one hand and sensors and actuators on the other, all marked in yellow and **red**. 2) The future holds new requirements for the SOC design flow, notably tool support for higher abstraction levels in both digital and analog flows. This will entail behavioral synthesis from higher-level specifications on the left side, as well as the corresponding abstract verification steps on the right hand side, all marked in red. The analog and mixed-signal flow additionally requires tools for analog circuit synthesis and automated analog verification on all design levels.

The transition to nanoscale technologies causes all design steps from specification to fabrication to not only become seriously interdependent, but also closely linked to the IC's eventual yield and reliability. Hence, SOC design productivity cannot straightforwardly follow the pace of nanoelectronics technology innovation as it is characterized by Moore's Law (see Figure DESN3). To drastically increase design productivity, an additional level of abstraction – the so-called System Level – has been introduced, with associated system-level challenges and solutions as given in the next section.

SYSTEM-LEVEL DESIGN

For decades, designers have reasoned about systems at various levels of abstraction (block diagrams, incomplete state charts, program models, etc.) with little support from design automation tools. This situation must change in the near future if necessary advances in productivity are to be achieved. To simplify the specification, verification and implementation of systems including hardware and software, and to enable more efficient design space exploration, a new level of abstraction is needed above the familiar register-transfer level.

[4] *Elements of this discussion were initially developed as part of a recent update of the Strategic Research Agenda (SRA) within the European Nanoelectronics Initiative Advisory Council (ENIAC) Chapter on Design Automation. Figure DESN2 shows status of the EDA design flow from an automotive point of view, but is general enough to demonstrate the future EDA requirements. The figure was developed by Peter van Staa (Bosch).*

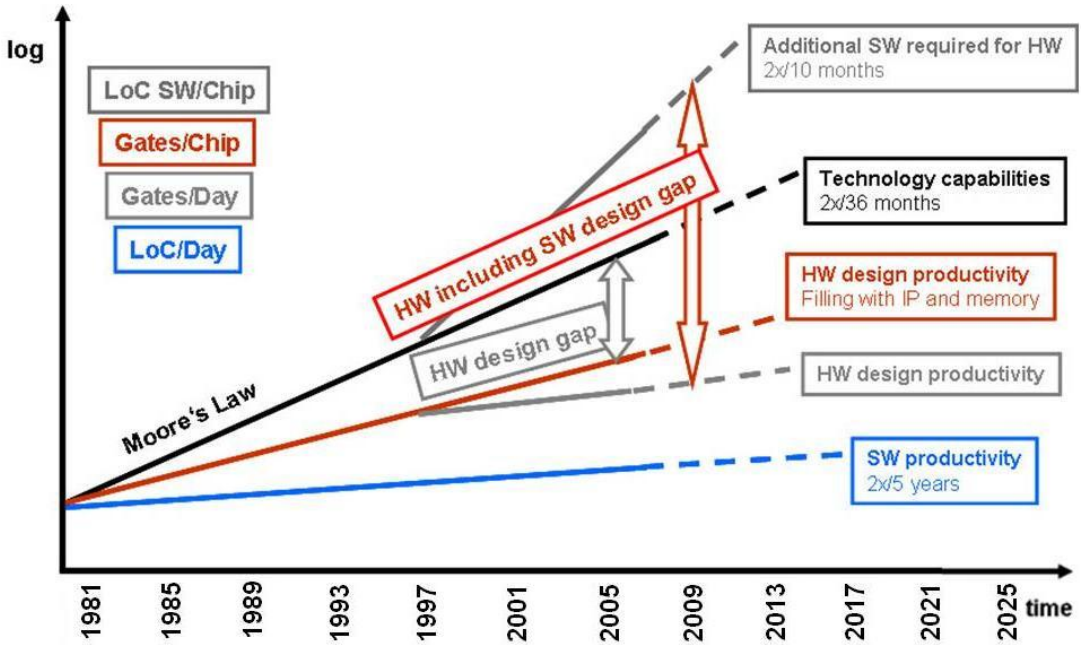

Figure DESN3 Hardware and Software Design Gaps versus Time[5]

In system-level design[6], methodological aspects are rapidly becoming much harder than tools aspects: enormous system complexity can be realized on a single die, but exploiting this potential reliably and cost-effectively will require a roughly 50× increase in design productivity over what is possible today. Silicon complexities imply that reliable and available systems must be built out of heterogeneous, unreliable device and interconnect components. Global synchronization becomes prohibitively costly due to process variability and power dissipation, and cross-chip signaling can no longer be achieved in a single clock cycle. Thus, system design must comprehend networking and distributed computation metaphors (for example, with communication structures designed first, and functional blocks then integrated into the communication backbone), as well as interactions between functional and interconnect pipelining. System complexities dramatically increase with the amount of software in embedded systems and the rapid adoption of multi-core SOC architectures. Not only is software dominating overall design effort as shown in Figure DESN3, but hardware dependent software that is tightly coupled to hardware and required functionality must be eventually handled by an SOC integration and verification process that is still hardware-centric today.

In this 2011 edition of the ITRS roadmap, the challenges in system-level design remain largely the same as in previous editions, this fact itself being evidence of the enormous complexity of these challenges. For instance, although behavioral synthesis is essential to system-level design, efficient behavioral synthesis is not yet realized today, despite having been a research topic for more than a decade, and despite recent advances driven by C- and SystemC-based synthesis and transaction level modeling (TLM) technologies. Also, while the current SOC design process is still largely "best effort" driven, an inevitable new direction is that of post-silicon self-healing, self-configuration and error correction. DT will play an enabling role in linking application requirements derived from societal and market needs to their eventual cost-

[5] *This figure shows the demand for software which is currently doubling every 10 months, the capability of technology which is currently doubling every 36 months, as well as the productivity of hardware and software design. Hardware design productivity has been improved in recent years by filling the silicon with multi-core components and memory, thus providing functionality only with additional software; on the other hand, productivity especially for hardware-dependent software is far behind and doubles only every 5 years. The red arrow summarizes the new design gap including both hardware and software. This ITRS version includes therefore for the first time additional figures for illustrating these new software requirements. This material was introduced in the 2007 ITRS edition.*

[6] *At the system level, silicon resources are defined in terms of abstract functions and blocks; design targets include software (embedded code in high-level and assembly language, configuration data, etc.) and hardware (cores, hardwired circuits, busses, reconfigurable cells). "Hardware" (HW) corresponds to implemented circuit elements, and "software" (SW) corresponds to logical abstractions (instructions) of functions performed by hardware. Behavior and architecture are independent degrees of design freedom, with software and hardware being two components of architecture. The aggregate of behaviors defines the system function, while the aggregate of architecture blocks defines a system platform. Platform mapping from system functionality onto system architecture is at the heart of system-level design, and becomes more difficult with increased system complexity and heterogeneity (whether architectural or functional).*

effective system implementation in the "More Moore" and "More than Moore" domains. Table DESN2 gives quantitative requirements for system-level design in future technology generations, including new requirements for the relative impact on power reduction of system-level design versus lower-level design.

Table DESN2a Near-term System-Level Design Technology Requirements

Table DESN2b Long-term System-Level Design Technology Requirements

Figure DESN4 specifies solutions for system-level design in their corresponding time frames, and Table DESN3 explains how requirements correspond to solutions.

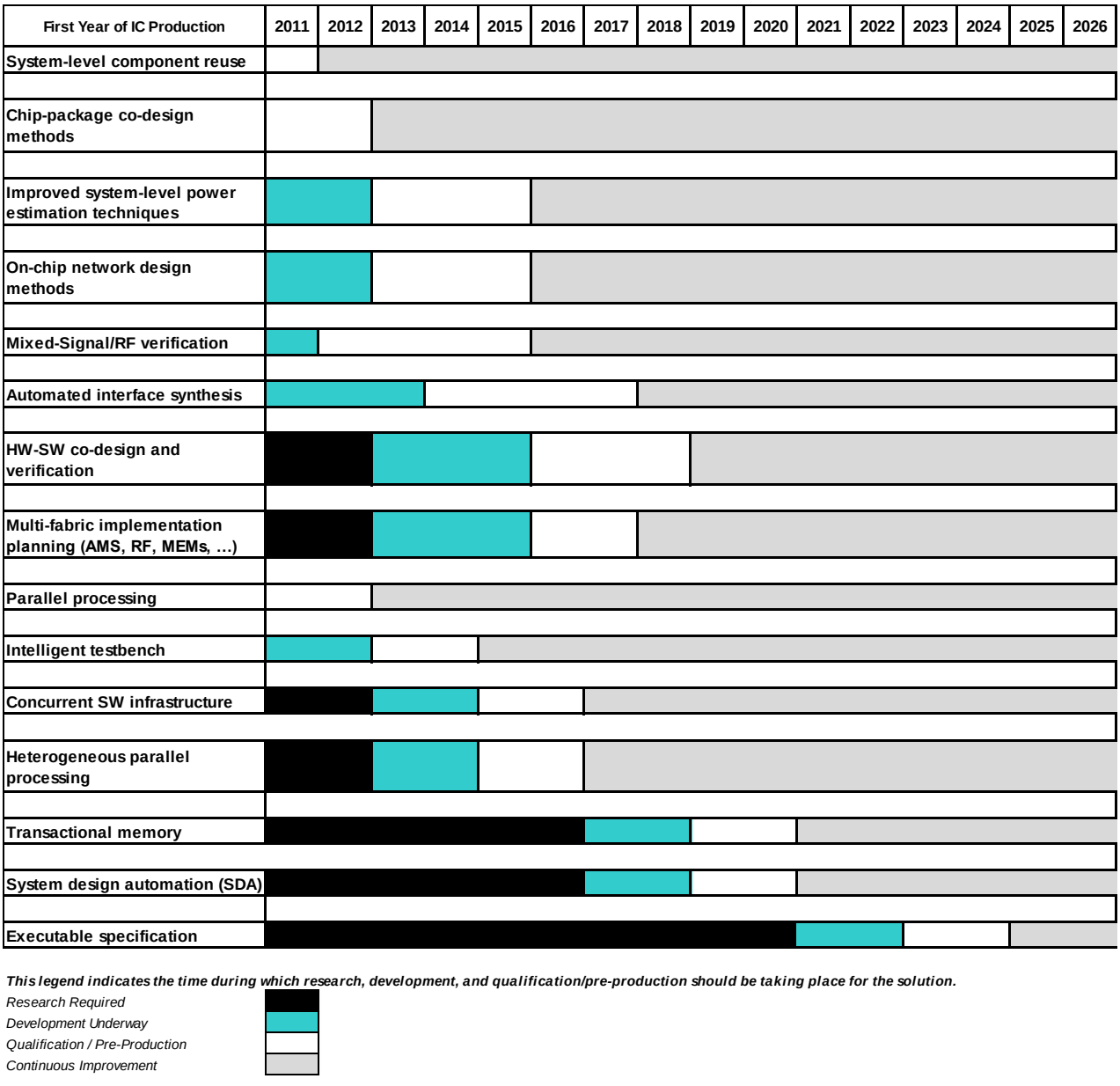

Figure DESN4 System-Level Design Potential Solutions

Table DESN3 Correspondence between System-Level Design Requirements and Solutions

Requirement	*Solution*	*Explanation of the Correspondence*
Design block reuse	System-level component reuse	The larger and more complex the components that can be reused, the greater the expected overall design reuse
	On-chip network design methods	Standardized communication structures and interfaces support reuse: IPs with standardized interfaces can be easily integrated and exchanged, and communication structures reused
Available platforms	Multi-fabric implementation planning (AMS, RF, MEMS, …)	Enables integration of different fabrics on same die or in same package (SIP); hence, enables reduced number of platforms
Platforms supported	Automated interface synthesis	Automated interface synthesis is one building block to an integrated synthesis flow for whole platforms
	Automated HW-SW codesign and verification	Required for integrated, platform-based system development
Accuracy of high level estimates	Improved system-level power estimation techniques	System-level power estimation needs to match progress in high-level area and performance estimation
	Chip-package codesign methods	Packaging effects, e.g., on timing, must be accounted for in higher-level estimations
SOC reconfigurability	On-chip network design methods	To provide flexible, reconfigurable communication structures
Analog automation	Multi-fabric implementation planning (AMS, RF, MEMS, …)	Multi-fabric implementation planning for AMS and RF components are a building block to analog automation
Modeling methodology, description languages, simulation environments	Mixed-Signal/RF verification	As in digital design, verification is an increasingly critical and time-consuming activity in the design flow
HW offers multi-core systems that have to be exploited by SW	Parallel Processing	Due to thermal and power limitations further performance increases have to be realized with multi-core systems.
Reduce SW verification effort	Intelligent Testbench	SW simulation, formal verification and automated testbenches for SW will reduce the verification effort for embedded software and enhance quality
Productivity increase required for SW since SW cost >> 50% of total system cost	Concurrent Software Infrastructure	A set of tools that allow concurrent software development and debug
Increase SW execution performance	Heterogeneous Parallel Processing	Parallel processing using different application-specific processors for each of the separate functions in the system
SW Productivity increase required	Transactional Memory	A concurrency control mechanism analogous to database transactions for controlling access to shared memory in concurrent computing; an alternative to lock-based synchronization
Productivity increase required for HW/SW codesign	System Design Automation (SDA)	True system-level design including electronic hardware and software, mechanical, bio, opto, chemical and fluids domains
Reduce verification effort	Executable Specification	Specifications written in a formal language allow automated verification process starting early in the design process and at high abstraction levels without the need to code several new verification models; this enables an integrated design flow from specification to completed system that can be completely validated at each step

Figure DESN5 gives a roadmap for the increasing role of system-level design in achieving required system power minimization. In the figure, percentage values indicate the fraction of power reduction (measured in watts or microwatts) that must be borne by the given phases of system design in future technology nodes. This trend is also reflected in the new Appendix III: DT-Based Reductions of Power Consumption, where most of the innovations are related to behavioral and architecture levels.

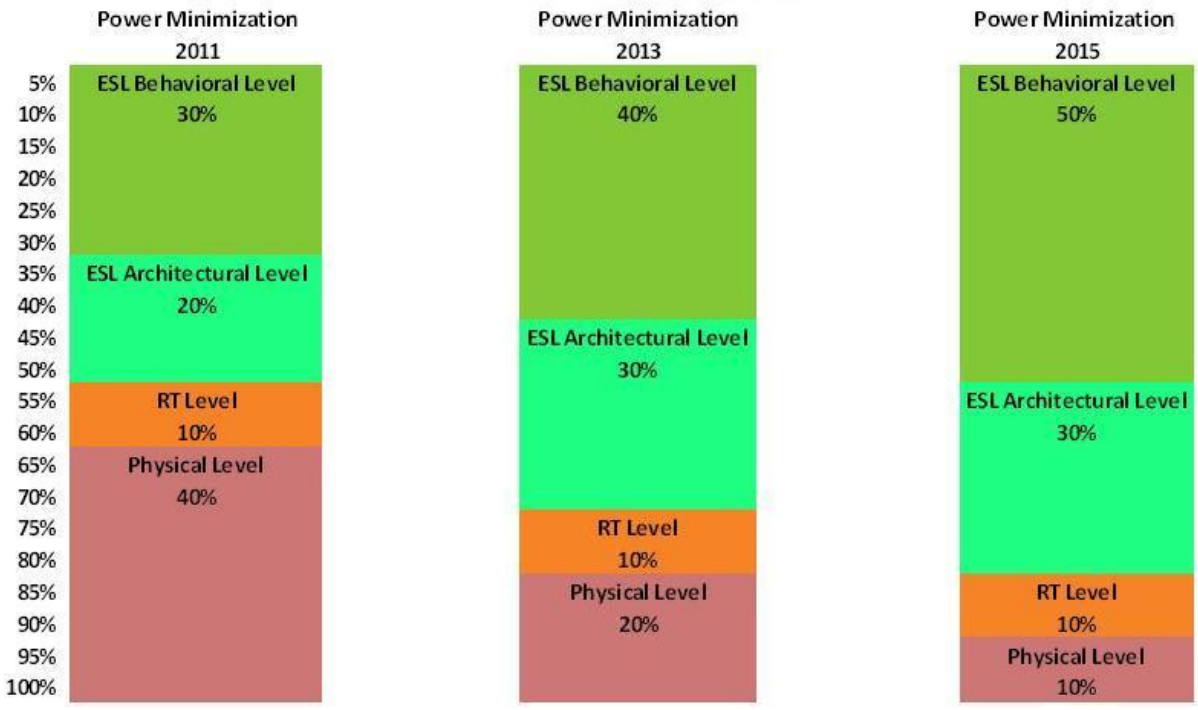

Figure DESN5 Evolving Role of Design Phases in Overall System Power Minimization

LOGICAL, CIRCUIT AND PHYSICAL DESIGN

In the traditional view of IC implementation, *logical design* is the process of mapping from the system-level design handoff (currently at the register-transfer level) to a gate-level representation that is suitable for input to physical design. *Circuit design* addresses creation of device and interconnect topologies (standard cells, full-custom analog, etc.) that achieve prescribed electrical and physical properties while remaining feasible with respect to process- and manufacturability-induced constraints. *Physical design* addresses aspects of chip implementation (floorplanning, placement, routing, extraction, performance analysis) related to the correct spatial embedding of devices and interconnects. The output of physical design is the handoff ("tapeout") to manufacturing (currently centered around a generalized data stream (GDSII) file), along with verifications of correctness (design rules, layout versus schematic, etc.) and constraints (timing, power, reliability, etc.). Together, logical, circuit and physical design comprise the *implementation* layer of DT that supports system-level design.

Design productivity requires system-level signoff into reliable, predictable implementation fabrics. However, silicon complexity makes it difficult to estimate and abstract the effects of physics and embedding on eventual design quality (timing, power, signal integrity, reliability, manufacturability, etc.). To avoid excessive guardbanding due to poor estimates, logical design and eventually system-level design must become more closely linked with physical design. Thus, the recent paradigm of hierarchical, top-down, layout-based implementation planning supported by a tightly integrated, incremental static (power, timing, noise) analysis "backplane" is likely to persist. Future successful implementation DT will depend heavily on methodology choices to juggle process/device abstraction, constraint manipulation, analyses, and optimizations in the face of exploding complexities and emerging concerns such as error-tolerance, variability and cost.

Current hardware design automation practices must evolve to handle the challenges and opportunities of finer-featured fabrication processes. Methodologies premised on the principle of separation of concerns – in which a complex design flow is serialized into a sequence of loosely-coupled, manageable steps – are growing long in the tooth. To the extent that decisions made in early stages of the design flow become binding constraints on later stages, such serialization potentially yields less-optimal designs than a methodology that simultaneously considers and co-optimizes all design aspects. Due to the practical difficulty of concurrent optimization of all design parameters, such serialization is deemed acceptable as

long as the constraints that are fed forward can be met. The methodology breaks down, however, when these constraints become infeasible; the typical action in such cases is to iterate, revisiting earlier design stages to identify and change problematic decisions. Such iteration has become particularly necessary between the logical and physical synthesis steps due to the inability of layout synthesis to satisfy timing requirements, that is, achieve timing closure. (Closure with respect to manufacturability and robustness checks is another looming source of expensive iterations.) Ideally, the time-wasting iteration between logic and layout synthesis (or, between layout synthesis and manufacturability verification) in today's design methodologies could be eliminated by fusing such stages – e.g., to simultaneously optimize the logical and layout structure of a circuit. But, as technology scales and previously ignorable physical or statistical effects become more significant, such optimizations are less effective due to growing inaccuracy of optimization targets or objective functions. Such inaccuracies may lead to timing-related and electrical problems that may not be apparent until masks are produced. To correct such problems, new engineering change order (ECO) techniques that work on the post-synthesis masks may be needed.

LOGICAL, CIRCUIT, AND PHYSICAL DESIGN REQUIREMENTS

Each technology requirement for logical, circuit and physical design is either a workaround that addresses effects of a given technology (i.e., a *control requirement*), or a desired capability for a given time frame and technology (i.e., an *objective requirement*). Table DESN4 gives numerical targets for each requirement.

Table DESN4 Logical/Circuit/Physical Design Technology Requirements

Explanations of the technology requirements are as follows.

Asynchronous global signaling—(Objective Requirement) A key challenge in modern IC design is distribution of a centralized clock signal throughout the chip with acceptably low skew. Combining pre-designed modules on the chip, and providing reliable communication between independent modules, are also challenging. The asynchronous global signaling requirement refers to a design style where synchronization is performed through request and acknowledge lines, rather than through a global clock. The requirement grows with the power, congestion and area costs of traditional repeater insertion in long global lines. Globally asynchronous, locally synchronous (GALS) design develops coarse-grained functional modules using conventional design techniques, then adds local clock generators and self-timed wrappers so that modules can communicate using asynchronous handshake protocols. The number of handshake components determines the system complexity. The number of independent components is not projected to increase dramatically due to the associated coordination overheads, such as clock distribution. By 2012, further progress in asynchronous clocking will depend on commercial tool support. Emerging arbitration schemes (e.g., trees, the crossbar of Fulcrum Microsystems, etc.) are likely to evolve through 2014, improving latency within a design.

Parameter uncertainty—(Control Requirement) EDA tools manage the uncertainty of interconnect delay by creating tighter links between the logical and physical domains. While this works for interconnect, substantial electrical uncertainty in transistors remains unaccounted for. As a result, static timing analysis tools needlessly overestimate operational delay in the circuits, and introduce significant pessimism into the operational frequency of a design. This requirement highlights the need for accurate consideration of process variations and resulting parametric uncertainty in logic and circuit design. In the table, %-effect is predicted as a function of parametric variation in shrinking devices.

Simultaneous analysis objectives—(Objective Requirement) In modern technologies, many objectives and physical effects interact with each other, and must be simultaneously analyzed and optimized. In order, these objectives are: delay, power (active or dynamic), area, signal integrity, delta delay, delta power, printability, and reliability. Current statistical methods integrate yield optimization into existing design flows, and became mainstream in 2010, enabling recovery of up to 25% of development time and yield. Statistical methods are likely to involve simultaneous analysis of delta delay and delta power. Optimizations for reliable computing will extend existing techniques for robust computation (e.g., the Razor technique from the University of Michigan), and may be integrated into the design flow soon. Productivity and cost will be part of the equation by 2013. Since productivity measures are only loosely defined today, and no mature optimization techniques are known, the requirement becomes red in 2013.

Number of circuit families in a single design—(Objective Requirement) High-performance integrated circuits often mix circuit families to achieve better speed, at the possible expense of area, power consumption, design effort, etc. The typical order in which circuit families are integrated is static CMOS, followed by multi-V_{th}, multi-V_{dd}, and dynamic CMOS.

Other circuit types such as asynchronous, or retention FFs, are also making their way into product chips. It is difficult to predict the exact mix of circuit families that will prevail in the future. However, the ability to handle multiple circuit families on a single design will remain an important requirement. At the same time, design complexity and fabrication difficulties that stem from mixing logic families, along with the low dimensionality of design constraints, will likely stabilize the number of different families on a single chip at a relatively small number.

Analog content synthesized—(Objective Requirement) Table DESN4 projects the amount of synthesized analog circuitry, as a percentage of a design's total analog content. See the AMS section of the *System Drivers Chapter* for more details. The trajectory of analog synthesis is reminiscent of logic synthesis 20 years ago: 1) today, it is on the brink of replacing designers; 2) by 2013, the significance of analog synthesis will compare to that of digital synthesis for microprocessors in the 1990s, and will apply to 25% of typical analog content; and 3) by 2020, current analog synthesis techniques will start saturating returns, calling for automation of new techniques just as digital synthesis now looks to automated datapath implementation and other extensions. Three IP reuse challenges that AMS technology must overcome in order to achieve its full benefits are technology node migration, different analog behavior due to foundry changes, and instability of EDA tools that often lack consistency in analyzing circuit behavior.

Adaptive/self-repairing circuits—(Objective Requirement) Nearly all circuits today cease to function properly when they incur any sort of damage. Manufacturing process variations also dramatically reduce reliability and yield of fabricated chips. In many cases, the tradeoffs between system size and complexity will still dictate that it is cheaper to create a compact, non-robust implementation of an electronic component and then replace it when it fails. However, in more and more contexts it is too expensive to access the system, or too expensive to communicate with the system to remotely diagnose its failure state and repair it. Hence, demand will increase for circuits that consume non-traditionally large areas, but are highly adaptive to internal failures. Such designs can help control costs of design verification, manufacturing, and testing. Circuits that remodel and repair themselves are already are already manufactured (e.g., IBM eFuse technology to regulate voltage and frequency of failing parts) and will be more common in new technologies. Autonomic or self-managing computer systems will be vital in the face of continually rising operational and maintenance costs.

Full-chip leakage power—(Control Requirement) Power consumption is now the major technical problem facing the semiconductor industry. Leakage power (subthreshold, gate) increases exponentially as process moves to finer technologies. Techniques such as dynamic V_{th}, clock gating, power domains/voltage islands, dynamic voltage and frequency scaling, multiple V_{th} transistors, and body bias solutions will mitigate leakage until 2012. While current technologies still see dominance of gate leakage over subthreshold leakage, the latter is likely to become performance-limiting as high-k dielectrics bring gate leakage under control. The leakage requirement is normalized to 2011 values, and is based on scaling of inverter leakage across technology nodes.

3D design technology – (Objective Requirement) As roadmapped in the *Interconnect* and *Assembly and Packaging Chapters*, TSV technologies offer the promise of integration across multiple die at very fine levels of granularity and concomitant savings in wiring, delay, power and form factor. As the technology set and infrastructure matures, 3D designs are expected to go through increasingly sophisticated states of development. The stages of development, and the features and challenges of each stage are summarized in Table DESN16 in Appendix IV. The first stage of development (2011-2014) is likely to focus on the use of silicon interposers as fine featured "printed circuit boards" to interconnect multiple chips into a "super-chip" or miniature system. The interposers might be package-level structures or be based on BEOL process technology from silicon interposers. The silicon interposer might be used as a stress relief layer between the chip and the package. The main design challenge in this phase is codesign of interconnect, power-delivery and thermal considerations in a 3D format. To permit codesign, interchange formats will be needed. The second stage is likely to focus on systems that benefit in providing large amounts of bandwidth to modest amounts of memory (typically 2-4 DRAMs). Several 3D design challenges emerge for this stage, including floorplanning to optimize the 3D enhanced performance; thermomechanical analysis and optimization; and codesign for power integrity and signal integrity. Power delivery, with limited TSV resources, is also a challenge. Test and test management present particularly difficult challenges in this stage, notably the reduction of test cost incurred due to the extra steps involved, and design for known good die to ensure high assembled yield. In the third stage, more heterogeneity is expected, together with higher levels of integration. There will be significant opportunities for multiple different layers within the 3D stack. The memory wall will be addressed by the ability to assemble large amounts of memory, e.g., into a co-integrated memory cube that can be tightly integrated with processors. Towards the end of this stage some level of monolithic 3D integration is likely to be available. A particular challenge is system optimization – how to best optimize the system performance and cost when a large number of process flow and assembly options are available. The challenges solved in the second stage will be even

more difficult to deal with in this stage. Particular challenges include power delivery within a stack of many die; thermal and stress management; and optimizing system cost when test and yield are fully taken into account.

LOGICAL, CIRCUIT, AND PHYSICAL DESIGN POTENTIAL SOLUTIONS

Logical, circuit, and physical design potential solutions are given in Figure DESN6. Explanatory comments are as follows.

1. *Automated handshake logic/circuit tools.* Needed for asynchronous and latency-insensitive global communication on chip.
2. *Synthesis and timing accounting for variability.* Synthesis that accounts for parametric variation of a technology; avoids overly pessimistic clock frequencies.
3. *Circuit/layout enhancement accounting for variability.* Power, interference, and error tolerance optimizations which account for parametric variations on a chip.
4. *Macro block/chip leakage analysis.* Accurate full-chip leakage estimation considering environmental variations.
5. *Power management analysis & logic insertion SOI SOC tools.* Techniques for power management at the logic level, e.g., power gating and clock gating – particularly in support of SOI technology.
6. *Analog synthesis (circuit/layout).* ASIC analog physical synthesis requires improved modeling methodology that is suited to structured synthesis flows. Multi-fabric synthesis tools also needed.
7. *Non-static logic implementation.* Logical and physical synthesis for dynamic circuit families, mixed static-dynamic logic.
8. *Cost-driven implementation flow.* Tighter integration of physical domain costs (parametric yield and unit costs, leakage and total power) into analysis and design space exploration flows. Analysis tools to decide between different implementation fabrics. Tools to extrapolate costs into different technology nodes. (Pre-layout) manufacturability analysis.
9. *3D system design space exploration tools.* Such tools will enable determination of whether 3D implementation for a given application is better (cost, performance, power, etc.) than 2D implementation, the appropriate granularity (core, block, gate) of 3D stacking, etc.
10. *Native 3D power/thermal analyses, optimizations.* Physical design for cooling and power delivery will significantly affect module reliability. Analyses and optimizations will be required to tackle manufacturability and cost issues associated with TSVs while optimizing performance and power metrics.

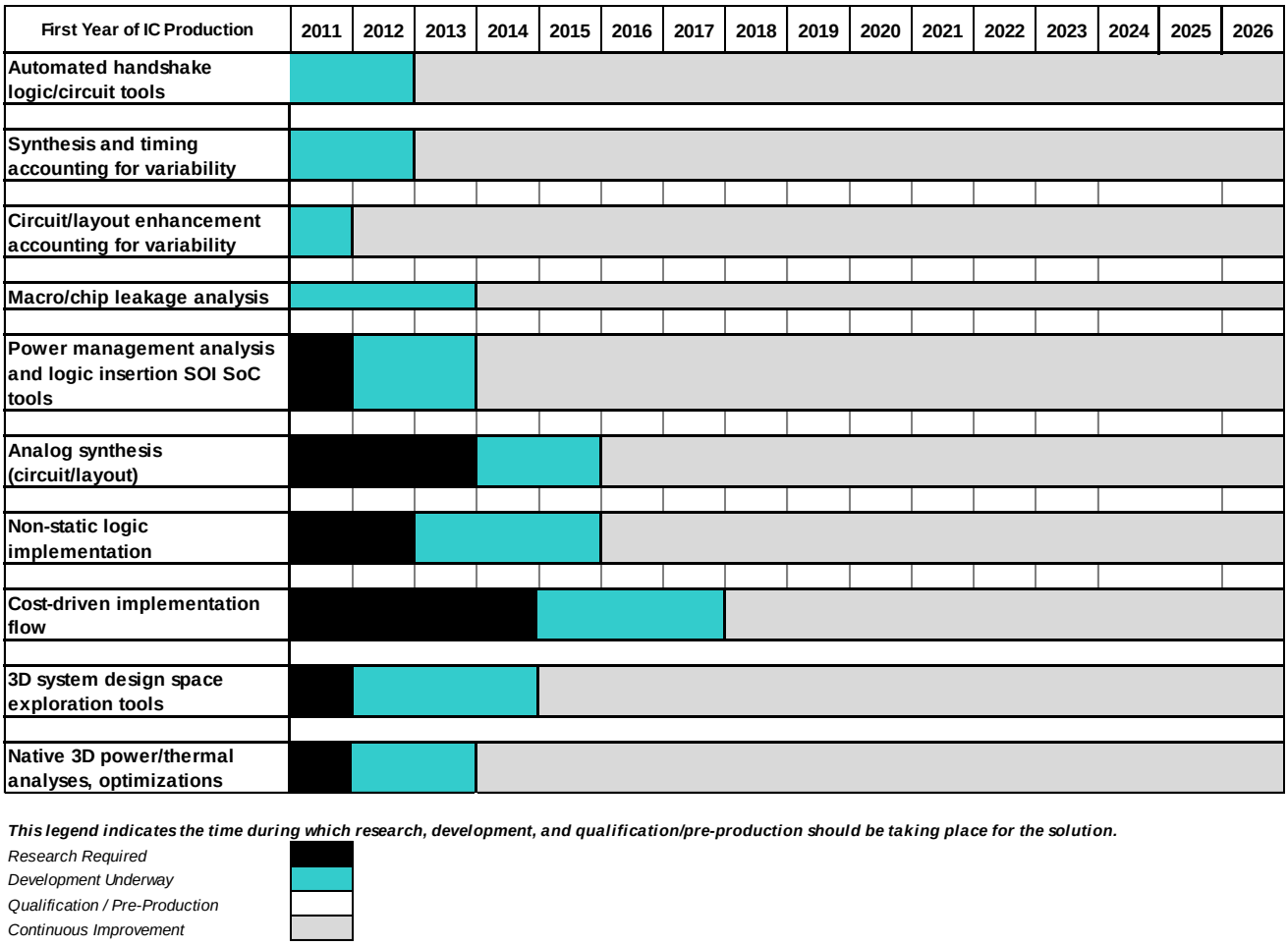

Figure DESN6 Logical/Circuit/Physical Design Potential Solutions

CORRESPONDENCE BETWEEN REQUIREMENTS AND SOLUTIONS

Correspondences between logical/circuit/physical design requirements and potential solutions are given in Table DESN5. For each requirement, potential solutions comprise the tools or methods that enable numerical targets to be met.

Table DESN5 Correspondence between Logical/Circuit/Physical Requirements and Solutions

Requirement	*Solution*	*Explanation of the Correspondence*
Asynchronous global signaling % of a design (SOC)	Automated handshake logic/circuit tools	Departure from fully synchronous design paradigm needed for power reduction, latency insensitivity, variation-tolerance
Parameter uncertainty %-effect (on signoff delay)	Synthesis and timing analysis accounting for variability	Tools that account for process uncertainty, and resulting parametric uncertainty, will reduce guardbanding and increase chip yields
Simultaneous analysis objectives	Circuit/layout enhancement accounting for variability	Optimizations which consider parametric uncertainty
Simultaneous analysis objectives	Power management analysis and logic insertion SOI SOC tools	Requires budgeting of area/power/timing constraints
Simultaneous analysis objectives	Cost-driven implementation flow	Cost is an engineering parameter that affects turnaround times. Silicon cost no longer dominant; test and manufacturing costs increase emphasis on adaptive, self-repairing circuits
Circuit families	Non-static logic implementation	Non-static implementations help improving different chip

Requirement	*Solution*	*Explanation of the Correspondence*
# of circuit families in a single design		parameters
Synthesized analog content	Analog synthesis (circuit/layout)	Allows for larger portions of a chip to be analog
Full-chip leakage	Macro block/chip leakage analysis	Enables accurate leakage predictions
% Native 3D design tech in flow	3D design space exploration tools	Enables correct choice of 2D vs. 3D, and 3D granularity
% Native 3D design tech in flow	Native 3D power/thermal analyses, optimizations	Required for co-optimization of reliability and manufacturability along with performance and power

DESIGN VERIFICATION

The overarching goal of functional verification is to ensure that a system's implementation fully matches its specification, i.e., the planned intended behavior of the device. Unfortunately, due to the growing complexity of silicon designs, functional verification is still an unresolved challenge that has resisted the enormous effort put forth by armies of verification engineers and academic researchers. The current practice in industry is a partial verification process, developed under tight time and cost constraints, where only a few aspects of a design are addressed, checked, and verified. These aspects constitute only a vanishingly small fraction of the entire universe of possible design behaviors. The underlying reasons for this unmanageable complexity lie in the inability of verification to keep pace with highly integrated SOC designs and parallel chip-multiprocessor systems, paired with highly interconnected communication protocols implementing distributed computation strategies. Because of the extremely high costs associated with flawed microchips, industrial development efforts devote a significant fraction of engineering time and resources to achieve functional correctness. Multiple sources report that in current development projects verification engineers outnumber designers, with this ratio reaching two to one for the most complex designs. Part of this high resource allocation is due to verification methodologies that are still largely ad hoc and experimental, and to a lack of robust solutions.

Without major breakthroughs, design verification will be a non-scalable, show-stopping barrier to further progress in the semiconductor industry. There is hope for breakthroughs to emerge via a shift from ad hoc verification methods to more structured, formal processes. The mainstream methodology used in industry today attempts to verify the functionality of a system design by repeatedly building models, simulating them on a few selected vectors, and then patching any bugs that are exposed. To this end, logic simulation techniques predominate, because they can generate simulation vectors at a very high rate. However, the coverage of these tests is usually very low, and when a design error is found, the debugging phase entails manual analysis of very long and complex bug traces to narrow down the source of the problem. Additionally, these methods require substantial engineering effort to direct the verification activity toward specific design areas of critical quality, low coverage, etc. Traditional verification techniques are entrenched because the cost to transition to alternative methodologies is high. Hence, adoption of new and alternative formal and semi-formal techniques, which have started to become available, is progressing slowly.

More structured verification approaches organize the verification effort by generating a "golden model" of the system's intended behavior and comparing simulation outcomes with this model. Coverage metrics are collected to support some degree of confidence in the correctness of the design-under-verification. It is common to organize the verification activities hierarchically, by addressing first individual components (generally developed by a single designer), then the entire chip, and eventually the full system, so that bugs that are contained within a single unit can be addressed and solved earlier and more easily. (Due to the complexity of full system simulation, hardware emulation is sometimes used instead of simulation, particularly for large-market designs, where the additional costs can be easily absorbed. Hardware emulation buys several orders of magnitude of performance improvement in "simulation" speed, and enables an early start to system integration and software development. At the same time, emulation represents only a constant-factor improvement.) The most complex aspect of this methodology is the verification of the communication interface between components; late or escaped bugs are often found in complex unverified interactions between units. Finally, there is a growing interest for formal and semi-formal verification techniques: today, IC development teams are exploring new methodologies where mainstream validation is complemented by semi-formal methods, using a handful of commercially available tools. This is a rapidly evolving arena: several new solutions, often integrating formal methods with simulation-based techniques in new and creative ways, appear each year both in academic environments and in new industrial developments.

Post-silicon validation focuses on the validation of silicon devices after tapeout and before customer shipping. This is an area of verification which has grown rapidly in the past decade, mostly due to the large complexity of silicon-based

systems, which, in turn, leads to many potential flaws that can only be exposed after system integration. The goals of post-silicon validation range from detecting logic and design bugs, which should manifest in each manufactured part, to expose electrical and process-related issues, which usually manifest only in a fraction of the components and, finally, to identify and exclude rare random manufacturing defects. The process involves building full prototype systems with the manufactured components, and running software test programs that range from embedded software to high-level applications. Two key characteristics of post-silicon validations are 1) the high execution speed which enables the verification team to evaluate a system over several orders of magnitude more cycles than is possible in pre-silicon simulation (e.g., a validation suite that requires months of runtime on a simulator or emulator could finish in minutes), and 2) the limited visibility inside the manufactured silicon parts. In fact, internal signals cannot be probed any longer as they could in a pre-silicon simulation environment, making test validation and debugging much more challenging operations. These inspection tasks are usually performed by accessing and probing the device's external signals and register interfaces using specialized tools, such as logic analyzers and oscilloscopes.

DESIGN VERIFICATION REQUIREMENTS

Table DESN6 quantifies design verification technology requirements needed to support design complexities in future technology nodes. The technology metrics evaluate both the ability to guarantee the correctness of a design and the amount of effort spent in verification within a development project. Table values are for a new-development design project, as opposed to a proliferation within an existing family of designs. If verification productivity is to scale in proportion to design complexity, the classification of verification software into formal verification versus simulation will over time evolve into a dichotomy of hybrid versus semi-formal verification solutions. Table DESN6 estimates, within integrated hybrid solutions, the fraction that will be simulation-based as opposed to formal verification-based. The increasing impact of software and mixed-signal components, and the importance of electrical effects due to the reduction in feature sizes, together imply that verification of the interaction between heterogeneous components and the system as a whole will require an increasing fraction of overall verification effort. Finally, the shift toward a more structured approach to verification demands efforts toward formalization of design specifications, which will in turn lead to improved automation of the entire verification process.

We divide verification development into three segments: 1) code newly developed specifically for the design being verified, 2) code acquired from third parties (i.e., verification IP), and 3) code reused from previous designs within the same company. Table DESN6 reports the first two components under "Reuse"; the third can be calculated by subtracting the other two rows from 100%. The last two rows in the table estimate the growing importance of functional coverage. While traditional coverage techniques focus on code and state coverage, functional coverage captures more directly the relevant aspects of a design's functionality that need to be verified, but it requires additional engineering effort because of its domain-specific character. The need to quantify the progress of verification in the near- and long-term future demands improved deployment of this technique. Although expertise within engineering teams in developing and reusing functional coverage is growing, the table indicates that a more pervasive deployment is needed in the future.

Table DESN6 Design Verification Technology Requirements

DESIGN VERIFICATION CHALLENGES

Many of the key challenges for verification are relevant to most or all system drivers. In the near term, the primary issues are centered on making formal and semi-formal verification techniques more reliable and controllable. In particular, major advances in the capacity and robustness of formal verification tools are needed, as well as meaningful metrics for the quality of verification. In the longer term, issues focus mainly on raising the level of abstraction and broadening the scope of formal verification. These longer-term issues are actually relevant now, although they have not reached the same level of crisis as other near-term challenges. In general, all of the verification challenges apply to SOC. MPUs present a distinct set of issues, both because of their leading-edge complexity, and because of the unique economics of an incredibly complex design family that is produced in incomparably high volumes. As a result, different domain-specific verification challenges and opportunities exist, both in the near- and long-term.

Capacity—Today, verification techniques can be grouped in two main families: formal verification and simulation-based. Both have crucial downsides: while formal tools can only handle small to medium size designs, simulation-based tools can simulate designs of almost arbitrary complexity, but they provide a vanishingly small coverage even with extremely long simulation times. Emulation and rapid hardware prototyping perform several orders of magnitude faster than

software logic simulators, thus providing the ability to achieve higher coverage. However, the improvement is only a constant factor and does not scale at the same rate as design complexity. Post-silicon validation is concerned with the overall correct functionality of the system composed of silicon hardware and application software, and as such it operates at the top level of design complexity integrating all components together. The challenge in this space is in creating new solutions, probably integrating the positive aspects of both families of approaches, that can provide high coverage at all hierarchical levels of a design.

Robustness—A crucial aspect of current verification solutions is their reliability. On one hand, simulation-based methods are fairly predictable, because their execution time per simulation vector scales linearly with design complexity. Their performance, thus, will linearly decrease as we move towards the most complex design levels. Emulation techniques present a similar trend, while silicon prototypes are consistently reliable. On the other hand, formal verification techniques depend on highly temperamental heuristics, in order to cope with the complexity of the verification problem. For any given pair of design and verification algorithms, even an expert can be hard-pressed to determine whether the verification algorithm will complete. Common measures of problem size, such as transistor, gate, or latch counts, correlate only vaguely with formal verification complexity; it is fairly easy to find designs with less than one hundred latches that defy all known verification methods, as well as designs with thousands of latches that can be verified easily. Such unpredictability is not acceptable in a production environment. A crucial verification challenge is to make the verification process more robust. This can take the form of either improved heuristics for the verification algorithms, or improved characterization of the difficulty of verifying a given design, leading to methodologies for easy-to-verify designs.

Verification metrics—An important near-term verification challenge is the need to quantify the quality of the verification effort. In particular, a meaningful notion of coverage is needed. Multiple types of coverage metrics are available, each with their own benefits and limitations. *Code coverage* is concerned with measuring which fraction of the design source code has been stimulated during simulation (for example, lines of code, expression coverage). Its main limitation is in its projecting a dynamic simulation execution to a static source code, thus missing many potential problems that could be still present because of all the possible paths of execution available during simulation. *Structural coverage*, such as state or pair arc coverage, focuses on the finite state machines of the design and measures which states have been covered by simulation. However, it is generally not possible to consider the whole design as a single finite state machine, rather only a few machines are considered by coverage. The downside is that while the coverage provides good results for single machines, combinations of states resulting from products of multiple machines are not considered. *Functional coverage* targets each of the design's functionalities. Since these differ for each distinct design, the specific definition needs to be provided by the verification team based on the design's specification. The quality of the result is thus dependent also on the quality of the coverage definition. In the current landscape, there is a lack and a need at the same time for a common foundation metric for functional coverage in a unified environment, so that achieving a certain level of coverage in a design could have meaning beyond the specific coverage used in that particular design. Moreover, there is no general methodology in designing functional coverage units; an abstract fundamental model to use as a guideline during the development is needed and could support the development methodology. Ultimately, there is a need for a defect model for functional bugs, much like there is a defect model for testing, to support the development of coverage and a unified metric for verification.

Software—Verification of complex SOCs entails the verification of hardware components, the hardware/software interface and the application software running on the system. The software components of an SOC can be divided into 1) application software, 2) hardware-independent layer, such as an operating system, which controls the execution of the application, and 3) lower hardware-dependent software, such as drivers, etc. Because in these systems the software layer can provide much of the functionality, a major challenge in SOC verification is how to verify the software and the hardware/software interface. Presently, software development is not as rigorous as hardware development in terms of design reviews, tools for analysis and testing. Software is intrinsically harder to verify: it has more complex, dynamic data and a much larger state space. The most common software verification technique in use today is "on-chip-verification," which entails running the software on a production version of the hardware components. While it allows very fast simulation, as it is required by the intrinsic complexity of software, its downside is that software verification can only start very late in the design cycle. Classical formal techniques for software verification are still too labor-intensive to be widely applicable for SOC, that is, systems with such large state spaces, and require very aggressively abstracted models of the software application. The verification of the hardware/software interface is a challenge on its own, since it requires verifying the two domains together. To make this task more manageable, there is a need for techniques which provide proper abstractions of the interface activity, solutions to check the correctness of the abstracted driver layers, and assertion checking tools for drivers' invariants at the non-abstract level. The near-term challenge will be to develop

techniques that allow verification of even elementary and low-level pieces of software. The longer-term challenge will be to develop robust verification methods for software, and hardware/software interfaces, as well as an understanding of design-for-verifiability as applied to software.

Reuse—Pre-designed IP blocks promise to allow assembling SOCs of unprecedented complexity in a very short time. The major challenge is in developing the corresponding verification methodology to allow rapid verification of a system assembled from pre-designed (and pre-verified) blocks. Key issues are how to rigorously and completely describe the abstract behavior of an IP block, how to describe the environmental constraints assumed by the IP block, and how to exploit the hierarchy to simplify verification. Some IP components have started to ship with associated verification IPs, from standard protocols verification IPs, to abstract models for general IP blocks, protocol checkers to check environmental constraints surrounding the block, to transaction generators. However, these are still preliminary attempts, while the need is for consistent availability of verification IPs associated with any IP block in order to develop a reuse methodology. Besides verification components associated with IP blocks, there is some availability of independent verification IPs, such as environment generators for specific protocols. Near-term progress will most likely occur for standardized IP interconnects, such as on-chip buses, but the general problem for arbitrary IP block interfaces must be eventually solved.

Specialized verification methodology—The biggest issue in design verification is that all currently known algorithmic solutions are running out of capacity with respect to the designs being developed today. The only foreseeable way to overcome this issue in the short term is with an adequate verification methodology. Current trends in this direction include coverage-driven verification, both for simulation-based verification and semi-formal verification, more in use today than in the past; specification documents in formal notation, that make easily available the set of formal properties to be verified in the implementation; coverage model templates. Many challenges are still to be solved to obtain a sufficiently robust and complete methodology: There is a need for ways to obtain consistent abstraction techniques of design components, interfaces, etc. that do not miss key aspects of the design in the abstraction. Formal specifications are starting to be developed; however, a major limitation is the completeness of such specifications, a challenge that becomes even more compelling in a world where different IP components in the same system are developed by completely unrelated design teams. The acceptance of new verification methodologies is progressing slowly; even the basic deployment of formal properties in a design is a challenge since it often requires a global understanding of some specific aspect of a design, which involves additional effort on the development team. Finally, the verification methodology is ultimately an evolving target that can only provide a risk reduction to the development, not a guarantee of correctness through a well-defined recipe.

Soft and permanent failures—There is an acute need for verification techniques that can provide information about a system's reliability in presence of soft errors and permanent transistor failures. Industry experts warn of the decaying reliability of the silicon substrate due to extreme transistor scaling. Initial solutions are available which can harden the storage elements of a design to be resilient to SEU faults. In addition, research-level solutions have proposed protection against permanent transistor failures (whether in the field or at birth) at very low additional area cost. These solutions are also viable to protect manufacturers against lowered yields due to a high incidence of defective manufactured parts.

Design for validation—A few specialized activities in the area of design for validation are already foreseeable, enabling more effective verification. For instance, facilities for software and/or hardware debug can be put in silicon. Debug features can be instrumented into a design by introducing appropriate observability and controllability measures based on an analysis of the design. A major advantage is the ability to correlate simulation of functional scenarios with their execution on the final hardware, which cuts down the debug time drastically. There is noticeable development in this direction in the industry with ongoing commercialization of the concept. A standardization of APIs/databases to connect hardware I/O data collection to simulation-based coverage database will go a long way in making it successful. There is also need for analysis tools to allow for proper placement of monitors. In this direction there is preliminary work being done on self-checking processors, in which a small watchdog processor or a network of distributed hardware checkers verifies the correct execution of the main processor. For mixed-signal designs, the insertion of loop-back modes to bypass the analog portion of the design allows to verify the system as a fully digital design. The use of synchronizers in multithreaded systems forces the tasks to not proceed independently past the synchronization points, creating checkpoints for verification and reducing the searchable state space. In MPU designs synchronizers would reduce the complexity of verifying speculative execution systems. The challenge in this area is the development and the adoption of design-for-verifiability techniques for a few major domains. Efforts are also made in developing design methodologies by incremental refinement to produce systems that are correct-by-construction; however, it is not clear how automatic the

refinement process can be made, while, on the other hand, manual intervention is a potential source of design errors. In the longer term, major changes in methodology may be required, and some performance degradation is likely.

Specification for verifiability—How to specify the desired behavior of a design is a continuing challenge in design verification. Current available notations for specification are not powerful enough to approach this problem in a generalized way. A deeper understanding of what makes a specification clear or opaque, modifiable or intractable will be needed to guide development of languages that are used to specify ever more complex designs. For instance, there is a need for automatic ways to check the self-consistency of a specification document, so that different specifications don't state conflicting requirements. In addition, specific training is needed for designers to use these notations and to be able to develop formal specifications consistently.

New kinds of concurrency—As MPU designs become more complex, new kinds of concurrency become important. Already, many of the bugs that elude verification relate to cache coherence and other concurrency issues. New designs greatly complicate the verification process by increasing the level of concurrency via techniques such as chip-level multiprocessing and on-chip cache coherence protocols, and simultaneous multithreading. In the future, new kinds of concurrency both at the intra-processor level and in multi-processor systems, or in other hardware context will present difficult challenges to verification. There is a need for new models of failure to grasp this additional level of complexity. The solution will probably require a mix of hardware and software techniques to reduce the complexity of the interactions and make the concurrent protocols verifiable.

Higher levels of abstraction—As design moves to a level of abstraction above RTL, verification will have to keep up. The challenges will be to adapt and develop verification methods for the higher levels of abstraction, to cope with the increased system complexity made possible by higher-level design, and to develop means to check the equivalence between the higher-level and lower-level models. This longer-term challenge will be made much more difficult if decisions about the higher-level of abstraction are made without regard for verification (e.g., languages with ill-defined or needlessly complex semantics, or a methodology relying on simulation-only models that have no formal relationship to the RTL model).

Verification in presence of non-digital effects—To date, design verification has mainly focused on the discrete behavior of digital systems. The dual challenges of silicon complexity and system complexity will force future verification efforts to analyze a broader class of aspects. The complexity of silicon integrated circuit systems is making the clean, digital abstraction of a VLSI system increasingly precarious. Analog electrical effects will impact performance and, eventually, functionality. The existing simulation methodology (SPICE) for analyzing these effects is too slow, and may become unreliable as smaller devices become increasingly sensitive to process variations. In this direction there is preliminary work being done on architectural simulation of microprocessors, in which the high-level architectural simulator interacts with a low-level context-specific simulator to acquire metrics such as timing and voltage, which are then fed back for overall system evaluation with minimal performance impact. In the long term, formal techniques will be needed to verify these issues at the boundary of analog and digital, treating them as hybrid systems. (N.B.: Hybrid systems have both complex discrete behavior (e.g., finite-state machines) as well as complex continuous behavior (e.g., differential equation models). The discipline borrows techniques from both discrete formal verification as well as classical control theory.) Similarly, at the highest levels of design, system complexity dictates that future verification tasks will likely require specification and verification of both analog and probabilistic behaviors such as quality of service guarantees in a network processor. This will bring challenges of hybrid systems and probabilistic verification.

Heterogeneous systems—The development of new technologies that are placed side-by-side with a digital design in a silicon die presents a whole set of new challenges. Examples are MEMS, electro-optical devices, and electro-biological devices. These new components will require modeling of both the interface between the digital portion and the non-digital components and a proper abstraction of the non-digital system behavior in order to still be able to verify the digital portion of the system.

Analog/mixed-signal—Today, analog systems are mostly verified through classic systems analysis tools for continuous systems, through system modeling and analysis in the frequency domain. The bulk of verification happens in the post-design phase, by verifying test dies with analog lab equipment. Mixed-signal designs undergo separate verification activities for the digital and analog portions. In the future, mixed-signal systems will become a more relevant fraction of all silicon developments, bringing the development of proper verification methodologies in this arena to a critical level. The challenge in this area is in tying the verification of the analog portion of a system with its digital counterpart. One of the requirements in achieving this goal is in bridging the current performance gap between digital and analog simulation.

Incremental verification—Today, SOC designs predominantly involve reuse of existing IPs with limited re-design. In such cases it is desirable to limit the scope of re-verification to the incremental changes. This eliminates the need to spend valuable verification resources on parts of the design that have already been proven in silicon. Tools and techniques are needed which can determine and eliminate such redundancies. Major EDA vendors are working in this direction, but a concrete solution remains elusive.

Verification strategy planning— Functional verification consumes about 75% of the resources in a typical SOC design cycle. Therefore, verification strategy planning is becoming critical to enhancement of design productivity and verification quality. The following considerations shape a verification strategy which maintains a tradeoff between cost and productivity.

1. Hierarchical division of verification tasks between various scopes: system level, subsystem level, block level.
2. Verification methodology for pre-existing or third-party IP.
3. Identification of appropriate verification techniques, such as model checking, equivalence checking, constrained random simulation, assertion-based verification, and hardware emulation, which are suited for specific verification tasks.
4. Choice of hardware verification languages (e.g., SystemVerilog, Vera, e) and methodologies available from EDA vendors (e.g., VMM, OVM, UVM).
5. Prioritization of verification tasks.
6. Constitution and training of the team of verification engineers such that the team has the diversity of skills required for execution of verification tasks.
7. Development of software for system-level verification such that it can be reused for SOC bring-up.
8. Incorporation of architecture- and performance-oriented verification.

Table DESN7 Verification Strategy Planning

Optimized verification planning	
Current Practice (2011)	1. Verification strategy is decided ad hoc, based on experiences, skills, and risk aversion of verification engineers 2. Increasing shift towards assertions, which enables use of formal verification techniques 3. Virtual Prototyping is increasingly used for system verification, facilitating accelerated software development
Problem Statement	Verification strategy depends on individual experience and skills; there are no criteria or systematic flows to develop it
Difficulties	Dependence on verification engineers' skills and lack of criteria cause variation in design quality
Short Term Solution	Internal standardization as verification methodology is unified to UVM from OVM/VMM
Long Term Solution	1. Develop criteria of verification strategy for QCD 2. Promote platform-based design development and bring up verification methodology with UVM as the mainstream, to create designs of the same quality at high level
Development of expert human resource for verification	
Current Status (2011)	1. Few engineers can develop UVM verification environment 2. Many engineers can use assertions for dynamic simulation, but formal verification is too difficult for most engineers 3. Training of verification engineers is local, and not methodical
Problem Statement	1. Need new skills for new methodologies such as formal verification 2. Few engineers can handle many kinds of verification methodologies
Difficulties	Few skilled persons in verification causes increased TAT and declining design quality
Short Term Solution	Implement human resources programs for verification, e.g., promoting guidelines for IP verification
Long Term Solution	A system for developing verification engineers, understanding how different kinds of skills are learned, and how to measure skills

DESIGN VERIFICATION SOLUTIONS

Figure DESN7 summarizes key avenues by which the above verification challenges may be addressed, along with expected time frames of availability to product development teams.

Figure DESN7 Design Verification Potential Solutions

Notes for Figure DESN7.
[1] This entails verification techniques for high-level integrated HW/SW systems. Current design technologies are starting to automate the process of partitioning the functionalities of a system that should be implemented in hardware and those that should be implemented in software. Typically, the result is a high-level description of the system's components and whether they should be implemented in embedded software or in hardware. The complexity of such high-level descriptions, however, is growing: there is a need for solutions which provide validation of a system at this stage of development and, in particular, of the correctness of the high-level interaction protocols.
[2] In other words, techniques for developing a hardware design so that it is well suited for verification.
[3] In other words, verification solutions which address the correct interaction between the hardware layer and the embedded software layer. Currently available solutions verify these two layers separately: 1) the hardware layer is verified mostly through classic validation techniques, and 2) embedded software is verified by running the software on a simulation model of the hardware (which is usually still under development). Due to the rapidly growing complexity of embedded software applications, this is becoming an increasingly pressing problem that requires specialized solutions.
[4] (Possibly formal) techniques to develop a specification for a design in a way that facilitates verification of the specification itself.

Verification IPs are available today, and are among methodology trends that boost the productivity of the verification process. In future, this approach will become much more widespread, in parallel with the aggressive deployment of third-party IPs in SOC designs. As discussed above, solutions structuring a hierarchical and integrated hardware/software methodology are being researched today and will start to become available in the next few years. The other solutions outlined in the table are still at a conceptual stage today. Formal methodologies to develop easy-to-verify designs and to correct soft errors and permanent transistor defects have seen initial exploration in the MPU domain, as mentioned in the

previous section. Techniques for structural functional coverage and specification are even further from becoming mainstream solutions; however, they will be crucial to solving the design verification productivity gap. Table DESN8 establishes correspondences between design verification requirements and solutions.

Table DESN8 Correspondence between Design Verification Requirements and Solutions

Requirement	*Solution*	*Explanation of the Correspondence*
Productivity of verification tasks	Verification methodology centered on verification IPs and reuse	Verification IPs and reuse reduce the amount of new verification development required in a project
	Hierarchical hardware verification	Structured methodologies improve design team productivity
	Reusable methodologies for functional coverage development	Functional coverage is time-consuming, and specific for each distinct design; development of reusability techniques is critical to boosting productivity
	Concurrent verification of hardware and software components during development	Advancing the verification of hardware in parallel with that of software components can significantly shorten time-to-market of a product, in contrast to methodologies that begin software verification only after the first hardware prototype
Formal and semi-formal verification centered methodology	Hierarchical hardware verification methodology	Enables the decomposition of the system into smaller blocks which are suitable for formal verification
	Design development and structure taking into account verifiability	Design for verifiability organizes a design so as to simplify verification; additional verification-specific hardware structures further simplify design-time verification tasks
Methodologies for system-level verification	Verification methodology centered on verification IPs and reuse	Verification IP components enable an early start on system-level verification
	Integrated verification of hardware and embedded software and their interface	Directly provides solutions for effective system-level verification
Portion of design specification formalized for verifiability	Design specification formalized for verifiability	Formal languages and methodologies to support the formal specification of a design
Escape rate after tapeout	Design structure taking into account verifiability	Development of hardware structures (checker-like) which can be used to detect and correct a system entering an escaped erroneous configuration after customer shipment
System integration bug rate	Analog and mixed-signal verification	Limits the bug rate due to analog effects
	Simulation and verification solutions for the detection and correction of soft failures and manufacturing faults	Manufacturing faults occurring in post-silicon are detected at system-level integration; techniques to detect and correct electrical and transient defects reduce the effort required to expose and correct these problems
	Hierarchical verification methodology	Supports management of complexity through decomposition
Functional coverage	Reusable methodologies for functional coverage development	Reusable functional coverage solutions leverage the coverage development effort and boost quality of results
Incremental verification	Tools and techniques for determination of redundancy in verification	Such tools and techniques will allow for elimination of redundancy from re-verification efforts
Verification strategy planning	Concrete decision making criteria, common platform, specialized training for verification discipline	A clear decision making process within the framework of a common platform with uniformity in skill level of engineers will facilitate successful execution of verification strategy

DESIGN FOR TEST

Nanometer process technology, increasing clock rate, mixed analog-digital-RF circuits, and highly integrated SOC, 3D die stacking, and SIP present severe challenges to DFT. The test industry must cope with an enormous spectrum of problems ranging from high-level test synthesis for component-based design to noise/interference and power dissipation in extremely high-performance analog and RF subsystems. Many test problems can be solved only if proper testability

and DFT are considered and incorporated early in the design phase. Furthermore, there is natural evolution of initial analyses into verifications that ultimately become tests, which then drives the need for deeper unification of design with test. Effort put into and results (flows, vectors, fault models, sensitivities) from analyses in logical-circuit-physical implementation and design verification, must be reused during design and test. Across industry segments as varied as memory, DSP, PE, SOC, analog/mixed-signal/RF, and MPU, high-level test challenges demand significant expansion of on-chip DFT, built-in self-test (BIST), and testability features, as well as tighter integration with a pre-planned manufacturing test equipment set during the chip planning phase of product development.

DFT technology requirements and potential solutions vary widely based on the specific targeted subsystems, yet they must all converge at the system level so that SOC and SIP are testable with lower manufacturing cost and, ideally, no performance degradations. Using system drivers as the common framework, DFT requirements and solutions are described in this section. Table DESN9 summarizes DFT technology requirements for five major classes of drivers based on specific design technologies: analog/mixed-signal/RF driver, MPU/PE/DSP driver, memory driver, generic SOC/SIP driver and drivers for systems with high reliability requirements. Emerging fault models for the structural test of digital logic, such as “small delay fault” and “bridging fault”, are gaining importance for complex digital devices with very high quality and reliability requirements. Coverage goals for these fault models are not currently given in Table DESN9.

Table DESN9 Design for Test Technology Requirements

Analog/Mixed-signal/RF system drivers—Analog, mixed-signal, and RF subsystems, together with component I/O speed, have become as important to system performance as core clock frequency, transistor, and architectural performance. Currently, the industry is facing serious problems regarding RF measurements beyond 5 GHz. It is very difficult to accurately test devices in the GHz at wafer level due to signal degradation as a result of contacting or fixtures. For packaged unit test, high frequency also presents test issues and it is imperative that techniques be developed that allow high-frequency mixed-signal circuitry (10–300 GHz) to be fully tested and characterized using low-cost production test equipment. At the same time, new I/O protocols are being introduced and extended to the multi-GHz range. These I/O schemes are not only faster, but also more complex, with source synchronous, differential, and even simultaneous bidirectional schemes operating at Gbit/s rates with differential voltage swings one-tenth of the supply V_{dd} range. Legacy ATE and component test tools include common clock-based testing and I/O measurements in the sub GHz range. Hence, increased I/O speeds and protocols drive significant instrumentation, materials, and cost challenges to the ATE equipment, interface hardware, and test sockets used for both design verification and manufacturing test. This inflection point demands broad industry development and application of on-die testability capabilities specifically for I/Os.

DFT methods for these analog/mixed-signal/RF subsystems are the subject of intense research and development efforts, but to be successful, these DFT circuits must use digital components as much as possible to reduce the design effort and improve robustness as well as noise immunity. The results from these DFT methods, whether they are Pass/Fail indicators or parametric measurements, must also be correlated to standard specification-based test methods to gain credibility and designers’ acceptance.

Analog/mixed-signal/RF reliability has become more important due to parametric degradation induced over time or by the operating environment. A comprehensive understanding of parametric issues requires well-characterized fault models for these circuits, especially to account for soft faults, noise-induced performance issues (crosstalk, substrate noise), process variations, thermal effects, etc. Fault models, whenever possible, must be physically correlated to defect models so that process quality and control can be improved to reduce and eliminate defects and improve yield. It is crucial to develop meaningful figures of merit for test quality of analog test techniques, including analog self-test. Many analog failures are due to parameters out of specification ranges, measured in continuous variables (time, voltage, phase, etc.), and arising from manufacturing variations or mismatch. Fault models for efficient and effective analog fault grading and test generation will be required. For analog/mixed-signal/RF designs, tools are needed that can minimize the computation complexity of fault simulation while maintaining high simulation accuracy. The requirements of analog fault models and process defect models are quite difficult to meet but they are paramount in future products integrating analog/mixed-signal/RF subsystems into SOC and SIP.

The potential solutions to these requirements are just as numerous as the subsystems. Diverse approaches have been developed by industry and by academia to solve the analog/mixed-signal/RF DFT problems, many of which focus on

specific product requirements (such as phase-locked loop (PLL) jitter BIST, converter BIST, or transceiver DFT). While convergence of analog/mixed-signal/RF DFT methods is not expected or desirable since a DFT method must be selected to optimize the overall electronic system design and test for each product class, several essential features common to all DFT methods are critical to the overall system-level solution. All-digital DFT methods will be the preferred solution due to robustness, ease of design, ease of integration, and ease of developing supporting computer-aided design (CAD) tools. While functional test and parametric test still dominate current DFT approaches, structurally-based DFT methods provide a long-term solution, especially if the results of these methods are demonstrated to correlate well with standard functional and parametric tests. A fundamental advantage of structurally-based DFT methods is the deeper understanding of defects and faults, which leads to quality improvement and cost reduction, two critical figures of merit in any system development effort.

One particular DFT solution, which may be seamlessly merged with current design technologies integrating computing and communication, is the use of radio wrappers for wireless control and communication of on-chip DFT subsystems. The test access problem is reduced for internal subsystems; online monitoring is possible; and communication with external test equipment is standardized, but there are inherent tradeoffs in other design parameters such as chip area, power, noise, and extra loadings on circuit generations. The complex challenges in analog/mixed-signal/RF DFT demand creative solutions and unconventional thinking from both system designers and test architects. These solutions must be developed in a timely manner to avoid the analog/mixed-signal/RF test bottleneck in highly complex system integration efforts.

MPU/PE/DSP system drivers—These system drivers are more mature with respect to DFT, BIST, and alternative test methods such as IDDx testing, even though recent advances in GHz clock speeds have also led to additional test issues not covered by digital DFT and BIST. These additional issues, such as testing high-speed clock for jitter and phase noise, may be considered as part of the overall mixed-signal or RF test requirements discussed above. From the logic design perspective, requirements for the MPU/PE/DSP/Memory system drivers are much better understood, and many challenges were successfully resolved in the past decade by the DFT and ATE community. Looking ahead, a major requirement is improved digital DFT coverage, in the sense that more digital blocks must have built-in DFT or be a part of an overall DFT scheme. These built-in DFT methods can vary widely, depending on specific block functions, design styles, and particular goals in fault detection such as hard fault, soft fault, or parametric fault. Delay and power are two examples of digital parameters where coverage needs to be improved significantly, with more DFT and BIST efforts to be devoted to enhancing the test methods for these parameters. Coverage monitoring, given a wide range of faults, defects, and parameters to be tested, must be incorporated into the overall DFT and BIST schemes, especially to target parametric degradation and reliability requirements. Online or offline coverage monitoring in operating environments is essential to anticipate and isolate possible subsystem failures so that repair can be performed in a timely manner. DFT methods with fault-tolerance and repair capabilities are required for these highly complex digital subsystems to enhance yield in the presence of parametric faults and manufacturing defects. Quality must be an integrated goal in the DFT and BIST frameworks for logic systems.

The emergence of multi-core MPU devices and consumer chips that have large numbers of cores drives a need for DFT that allows concurrent test of similar cores or structures in order to reduce test time and cost. The tradeoff in implementation will involve power considerations, e.g., leading to use of staggered phasing to minimize current surges. Concurrent test methods also must better pinpoint faults for failure analysis, rather than return results that hide specifics and give only a global fault signal as a result of data compression.

Mature solutions exist for many logic systems in this classification. The remaining challenges demand system-level solutions rather than block-level solutions. As pointed out above, several critical parameters in digital system performance still need effective DFT solutions—delay, power, jitter, local operating temperature, etc. Cost-effective DFT and BIST methods are highly desirable to determine these parameters both in characterization test and high-volume manufacturing test. Solutions that provide integrated calibration and repair capabilities will emerge as crucial to overall electronic system design and test architectures. With both current fault-oriented DFT/BIST methods and these new digital parametric-oriented DFT/BIST methods, system test designers must create cohesive system-wide DFT/BIST architectures to make effective use of the various methods at their disposal. A patchwork test architecture or a haphazard test integration will likely create problems in logistics and test execution planning, which in terms of test time and test data volume would reduce the advantages of DFT/BIST.

Memory system drivers—The memory system driver is more mature with respect to DFT, BIST, and alternative test methods, even though recent advances in ultra-large memory blocks have led to additional test issues such as new failure

mechanisms, timing uncertainties in read/write access, etc. These may be considered as part of the overall mixed-signal test requirements discussed above but, given the memory size, they pose separate problems to be solved in the integrated SOC context. From the memory design perspective, the essential requirement is to improve yield. Redundant, fault tolerant and fault correction design methods have helped immensely in this area but as memory blocks increase in size, yield issues encompass both the requirement for good memory cells but also for memory timing performance within specifications. DFT for yield improvement must provide methods to satisfy both of these goals: physical quality and timing performance of memory blocks.

Highly integrated general SOC and SIP designs—Integration of pre-existing design blocks into larger integrated devices produces nonlinear complexity growth for design tools, DFT, and manufacturing test, even when the blocks are homogeneous (such as all logic). Increasingly, a wider variety of circuits are being integrated. Logic, static RAM (SRAM) and DRAM have been commonly integrated in recent years, but now analog, mixed-signal, RF, and non-volatile flash circuits are being combined with logic and RAM. Artificial intelligence (AI) via the use of neural blocks will bring additional challenges to test because of the neural learning process. Silicon complexity and costs are relatively predictable for integrated devices. However, embedded blocks and mixed device types drive highly nonlinear and unpredictable increases in the cost of testability, design verification, and manufacturing test. ASIC or MPU macros wholly embedded within larger logic devices are already seeing this impact which can drive manufacturing test costs to exceed silicon costs. Even with DFT, these costs may be nonlinear. It is emphasized that the DFT/BIST requirements in this section are not just a collection of block-level DFT methods already covered in the three system drivers described above. The system-wide DFT requirements and solutions target the entire system, provide the overall DFT/BIST logistics and scheduling paradigms, and include test capabilities missed or not covered at the lower levels.

Within the system context, DFT must provide methods for on-chip or in-system test generation and test application to reduce the burden in test time and test volume using external ATE. Test generation hardware and embedded algorithms reduce the need for test access while providing an infrastructure for system-wide BIST, which is the preferred DFT approach. As DFT technologies for analog, mixed-signal, and RF subsystems improve, their test generation and application resources must continuously be integrated into this system test infrastructure in a seamless manner. Faults (some classes of hard faults and many classes of parametric faults) detected by DFT should be repairable or made tolerable to improve yield and reduce time to market, thus the DFT and electronic system design methodologies must provide this repair capability to work in concert with DFT. A more global perspective and also an appealing aspect of DFT are the DFT capabilities to perform calibration via *in situ* measurement and feedback. Many calibration methods employed by designers to tune the system performance in the presence of process variations and device imperfections are identical to DFT methods. A consistent usage of DFT as a system calibration methodology would promote designers' acceptance and facilitate tighter integration between design and test. In-service or online measurement, field repair, correction of performance degradation over time, etc. are fundamental performance requirements that can be met by a comprehensive DFT methodology.

Testing of embedded blocks should not entail orders of magnitude longer test time than testing of the non-embedded versions. The test methods, ATEs, and manufacturing integrations for memory, logic, and analog/mixed-signal/RF silicon come from radically different legacies with unique optimizations that can be broken in testing on an integrated logic device. DFT must address the cost of testing in order to prevent the overall cost of an integrated solution from becoming greater than the non-integrated solution.

An impediment to DFT incorporation has been the perception that DFT is costly in terms of system impact: chip area, test-specific I/O allocation, power, bandwidth, signal sensitivity, etc. The integration of analog/mixed-signal/RF subsystems and very high speed digital subsystems into SOC and SIP has led to a stricter examination of additional leakage currents, noise, and loads on sensitive nodes being monitored by DFT circuits. The impact of DFT on system performance, such as bandwidth loss or additional noise, must be quantified and estimated well enough to permit an early benefit versus cost study of possible DFT methods to be chosen for incorporation. DFT methods should also focus on monitoring less sensitive nodes and indirectly estimate test results (such as fault detection and parametric measurement).

The greatest advantage of DFT is in reduction of test volume and test cost. This advantage must be well-demonstrated at the SOC/SIP level, where test time and test cost are most apparent and more easily calculated. The test volume reduction target in Table DESN9 covers the entire system test volume and test time, not the test volume and test time of a specific block within the system. Test time and cost are also strongly correlated to ATE since, even with DFT and BIST, ATE still plays an important role in test. The interface between various DFT methods and any ATE must be well defined to permit flexibility in system implementation of different DFT paradigms and in ATE selection to meet specific product test needs.

While test access port and other test standards, such as IEEE 1149.1 (*IEEE Standard Test Access Port and Boundary-Scan Architecture*), 1149.4, 1450 (*IEEE P1450.6, Draft Standard for Standard Test Interface Language for Digital Test Vector Data – Core Test Language*), and 1500 (*IEEE 1500-2005, IEEE Standard Testability Method for Embedded Core-Based Integrated Circuits*) are available, comprehensive DFT/ATE interface protocols spanning various system layers (behavioral layer, physical layer, data communication layer, etc.) must be established to fully exploit the advantages of the DFT methods and the ATE capabilities.

Solutions to the overall SOC/SIP DFT/BIST problems have been evolving and continue to be created at a very fast pace. The diversity of technologies and subsystem design methods within a SOC or SIP demand solutions for test generation tools and DFT/BIST tools that integrate digital test methods (fault-based, defect-based, and some parametric-based) with analog/mixed-signal/RF test methods (functional-based, parametric-based, and future fault-based and defect-based).

Solutions must be accompanied by design and test planning tools to reduce system complexity, design effort, time-to-market, and overall test cost. The various levels of integration within a SOC or SIP (from highly integrated digital blocks to macro-based RF or analog blocks) lead to solutions that can span the entire system hierarchy and provide appropriate capabilities for the specific subsystem under test: fault tolerance, calibration, tuning, and repair, with the overall goal of yield and quality improvement. At any hierarchy level, an effective system solution must provide a collection of compatible DFT/BIST methods available to the designers and must integrate the various choices made by the designers at different hierarchy levels as the design proceeds. The total solution at the end must be a cohesive integrated solution reflecting both design and test optimization.

Efficient simulation models and tools are inherent in all design and test planning efforts, and must be a part of the test solution, especially in estimating the impact of DFT/BIST on system performance. Simulation tools must provide this impact estimate as the design proceeds across various levels—behavioral, circuit, and physical—to permit the selection of the most suitable DFT/BIST methods. An overall figure of merit is system test coverage (fault or parametric or functional or a combination thereof) that must be computed as part of the DFT/BIST incorporation and monitored both during test and during operations. Capabilities for online coverage monitoring are highly desirable and algorithms must be developed to relate coverage statistics to fault and defect statistics to enable manufacturing quality improvements.

Certainly, the most controversial aspect of system-wide DFT/BIST solutions is the interface between on-chip DFT/BIST capabilities and ATE. Many test protocols exist, at least for digital systems and in recent years for mixed-signal systems, but an overall interface standard for overall SOC/SIP is still lacking. Since test resource partitioning practices vary widely in industry, the interface between DFT/BIST and ATE has not been well-defined but future solutions must contain this definition with at least two critical characteristics: rigorous clear interface definitions to permit the interchangeable and effective use of various ATEs in system testing, and flexible interface definitions to avoid constraining the electronic system design and test architectures.

Drivers for systems with high reliability requirements—The measures described above are no longer sufficient for highly complex devices at leading-edge technology nodes, as system complexity as well as the probability of defects increase. Even if costly methods like burn-in are applied, a latent defect which is not visible during production test may lead to a malfunction later on during the lifetime of the device. Therefore, the test must be extended from production test to a test during the lifetime of the system. This is called *in-system test*.

In-system test may be activated during system start or shutdown, but also during runtime, e.g., when running a test for a few milliseconds every second. Classical methods such as logic/memory BIST or software-based self-test executed by the system CPU may be used, but complex control logic must be added. This BIST circuitry can also be used during normal production test to reduce test cost. Detecting a fault might bring the device to a fail-safe mode, or on-chip repair may even be available using redundant circuitry. Those measures are driven at the moment by automotive applications, but will spread to other application areas in the future.

3DIC—The introduction of 3DIC technologies introduces acute DFT challenges. If chips are not tested and sorted before stacking, the stacked yield is likely to be low for conventionally sized die. Thus, it is expected that some level of sorting to a known good die standard is likely before integration. This introduces additional cost, even before further costs of post-assembly test are incurred. Solutions are needed to reduce the cost of the additional test steps required in 3D assembly. In addition, at the time of writing, TSV yield itself is likely to be low enough that TSVs will need to be tested before integration, and some redundancy might be needed. A low-cost probe or BIST method is likely to be needed.

DESIGN FOR MANUFACTURABILITY (DFM)

Increasing variability, mask cost and data explosion, and lithography hardware limitations are posing significant design challenges for the manufacturability of integrated circuits.

Architecture challenges—Architectural redundancy will be required due to the difficulty in making circuits yield. It will be hard to do much more at this level of abstraction.

Logic and circuit challenges—Adaptive digital and mixed-signal circuits will be increasingly necessary. Statistical design, including power and timing convergence, will be fundamental. This being said, it will be difficult to make designs work without addressing two primary challenges: 1) characterization and modeling inputs to statistical design tools, and 2) the evolution from statistical analysis to optimization with the subsequent increase in computation complexity. Finally, statistical analysis tools and statistical optimization methods must model actual manufacturing and design-induced variations rather than crude and possibly inaccurate abstractions. The composition of variations during statistical analyses and optimizations should match the methods used for initial decomposition via statistical metrology techniques during process characterization. A mismatch between composition and decomposition will introduce unnecessary error and impart dubious value to the computationally costly results.

Layout and physical design challenges—First, the complexity of design rule checking continues its rapid increase. Rules have evolved to a two-tier system (required versus recommended rules) and may need to evolve either to a three-tier system or to a no-tier system (where rules are not pass-fail but provide a Pareto curve of yield benefit versus area cost that the designer can apply as a tapeout criterion). This checking will need to be done without increasing designer-perceived complexity. Second, the limitation in lithography hardware resolution will require design flows to more explicitly account for the impact of reticle enhancement techniques (RET). RET tools, such as OPC and chemical-mechanical planarization (CMP) fill, must become explicitly aware of circuit metrics such as timing and power. Such awareness aligns the tools with overall product goals and enables yield enhancements, manufacturing cost reductions, and improvement in mask data preparation time. As an example, the most aggressive OPC would be applied only to features in critical timing paths. This approach entails a tighter flow integration to communicate circuit intent downstream, and to avoid independent modifications by several tools leading to incorrect results. As a consequence, the tools in the register-transfer level to graphic database system (RTL to GDS) flow will have to properly plan for RET and OPC modifications downstream. For example, global and local layer densities should be taken into account at the global placement level to determine possible locations for dummy fill insertion, thus allowing pre-placement of CMP fills and accounting for their capacitance while at the same time propagating information about critical nets down to layout finishing, mask data preparation (MDP), and OPC steps.

Yield prediction and optimization as design challenge—Layout ground rules are no longer "hard numbers" since such an interpretation tends to drive design *compliance* rather than design *improvement*. To achieve reasonable mature yields and a steep yield ramp capability, a strategy for meaningful design rule relaxation is required: "recommended" rules must be derived from the interaction of the design layout and wafer process requirements, such as alignment tolerances, optical proximity corrections, RET enhancements and many other constraints. During the design process the mutual interaction of yield, area, power and speed must be analyzed, and a commercially useful tradeoff must be found. DFM measures that affect functional and parametric yield must be integrated as a new optimization feature into the design flow and tools up front, rather than as (limited, time-consuming) post-processing. A natural consequence is that yield prediction must be integrated into tools for design planning, synthesis, and place and route – so as to simultaneously optimize for yield, performance, power, signal integrity, and area (with yield as an explicit new design target). Finally, yield is a function of both product-specific design attributes and process-specific failure probabilities. Therefore, a design that has been optimized to tolerate specific fail/marginality patterns of a particular process may actually yield poorly in different process conditions. To enable correct evaluation of the yield cost of different design implementations by, e.g., place and route tools, accurate yield models of the logic library must be precharacterized based on the actual target process data. Library yield models will need to be frequently updated to track process evolution from pre-ramp to a mature stage.

3DIC challenges—3D technologies introduce additional DFM challenges due to the additional stress introduced into the chip stack by CTE mismatch between the silicon and the TSV metals. This could lead to changes in transistor performance, and additional stress issues when the chip-package interaction is accounted for.

Table DESN10 quantifies the key challenges given above as a set of DFM technology requirements.

Table DESN10 Design for Manufacturability Technology Requirements

DFM requirements fall into two basic categories.

Requirements due to fundamental economic limitations—This category includes mask cost, which is reaching multi-million dollar levels, thereby jeopardizing System-On-Chip innovation coming from small companies and emerging-market institutions.

Requirements due to variability and lithography limitations—This category includes limitations at low abstraction levels that are associated directly with devices and wires, as well as limitations at higher levels of abstraction that are associated with the overall circuits being designed. At the lower level, quantified requirements include voltage supply variability (as delivered to on-chip circuits), voltage threshold variability for both minimum-size memory devices and typically-sized logic devices (in terms of both the impact of doping concentration variability and the overall trends), and percentage CD (critical dimension, specifically channel length) variability. At the higher level, requirements include percentage of circuit performance variability (the percentage uncertainty in the speed of the circuit that determines overall chip performance, such as its critical path), and percentage of circuit power variability (the percentage uncertainty for power consumption, including both active and standby power).

Mask cost, a key economic parameter, has historically been difficult to keep in check. Research is needed to achieve breakthrough reductions in mask cost and turnaround time. Absent such breakthroughs, workarounds such as multi-project wafers, configurable logic, and structured ASICs will continue to be found. With respect to variability, some parameters, or at least their targets, will be controlled by design, including "process-level" parameters such as CD control, and "circuit-level" parameters, such as voltage supply. Other parameters, such as threshold voltage variability (including the impact of channel doping, which is known to scale with the inverse of device area) will unavoidably grow, leading to potentially critical red bricks in the coming decade. Due to upward parameter propagation from process level to device level to circuit level, very large overall circuit performance and power consumption variability will need to be addressed, unless radical new solutions are found.

A consequence of the increase in variability predicted in Table DESN10 is the blurring of the boundary between (1) catastrophic (hard) faults traditionally caused by topological changes in circuits due to manufacturing defects and (2) parametric (soft) faults related to device and interconnect variability. In fact, as this variability increases, it can cause circuits to exhibit faulty behavior similar to that observed due to catastrophic defects. As an example, it has been the case for several technology generations that due to increasing threshold voltage variability in small MOSFET devices due primarily to phenomena such as random dopant fluctuations, an SRAM cell can be such that it is capable of storing only one of the two possible Boolean values. This makes the SRAM cell behave as if it is "stuck" at one logic value, and would cause that particular SRAM cell to fail a write test. Forward projections indicate that this type of phenomenon, already pervasive in SRAM at the 65nm node, will become more probable and increase in scope to other critical circuit types like latches and register files. The sources of such failures may be categorized as follows:

- *Process Variations*: Statistical variations of device parameters such as channel length, threshold voltage and carrier mobility. Channel length variability results from line edge roughness (LER) and other patterning fidelity issues, while threshold voltage variations are primarily induced by random dopant fluctuations.
- *Lifetime Variations*: Variations that shift physical parameters over the operating life of a circuit. The most significant are V_{th} shifts due to negative-bias temperature instability (NBTI) and hot-carrier injection (HCI), along with gate current shifts due to time-dependent dielectric breakdown (TDDB).
- *External Noise*: Noise sources that are external to the integrated circuit, such as environmental/power supply noise and energetic particles that cause single-event upsets.
- *Intrinsic Noise*: Noise sources inherent to normal device operation that become significant at small feature sizes. These include shot noise, thermal noise and random telegraph noise.

To make predictions about the impact of these types of phenomena on future technologies, the U.S. Design TWG has studied three commonly used, 'canonical' CMOS logic circuits: an SRAM bitcell, a simple latch, and the common

inverter. This follows the precedent of using design-oriented circuit drivers – e.g., in the analog / mixed-signal domain – to identify future trends. The three circuits selected are good surrogates for major components of a digital CMOS design: storage (SRAM bitcell), synchronization (latch), and logic functions (inverter). It is common to have many millions of SRAM bits, and millions of latches and inverters in current high-performance processors.

Failure probabilities for the three canonical circuits in future high-performance (HP) technology nodes were obtained by simulating their behavior under the influence of manufacturing process variability. The simulations used the Predictive Technology Model (PTM) with variability estimates for both general logic and SRAM circuits for bulk CMOS technologies down to the 16nm node. The respective criteria for failure were as follows.

1. For the SRAM bitcell, two distinct failure modes were explored: (a) the writability fail, where the SRAM is unable to store one of the two Boolean values, and (b) the read disturb fail, where the act of reading an SRAM causes the contents of the cell to reverse polarity.

2. For the latch, the CLK-to-Q (clock to output) delay was measured, with failure corresponding to CLK-to-Q delay 10 times its nominal value. Since the CLK-to-Q delay is an important part of the timing of any digital circuit, such a drastic increase is likely to cause timing failures similar to what would be observed if the latch were to experience a hard fault.

3. For the inverter, the pair delay (the delay of two inverters in series) was measured, with failure corresponding to pair delay 10 times its nominal value. Given the pervasive use of inverters as buffers on long interconnect wires, such a drastic increase in delay is, again, likely to cause significant timing failures.

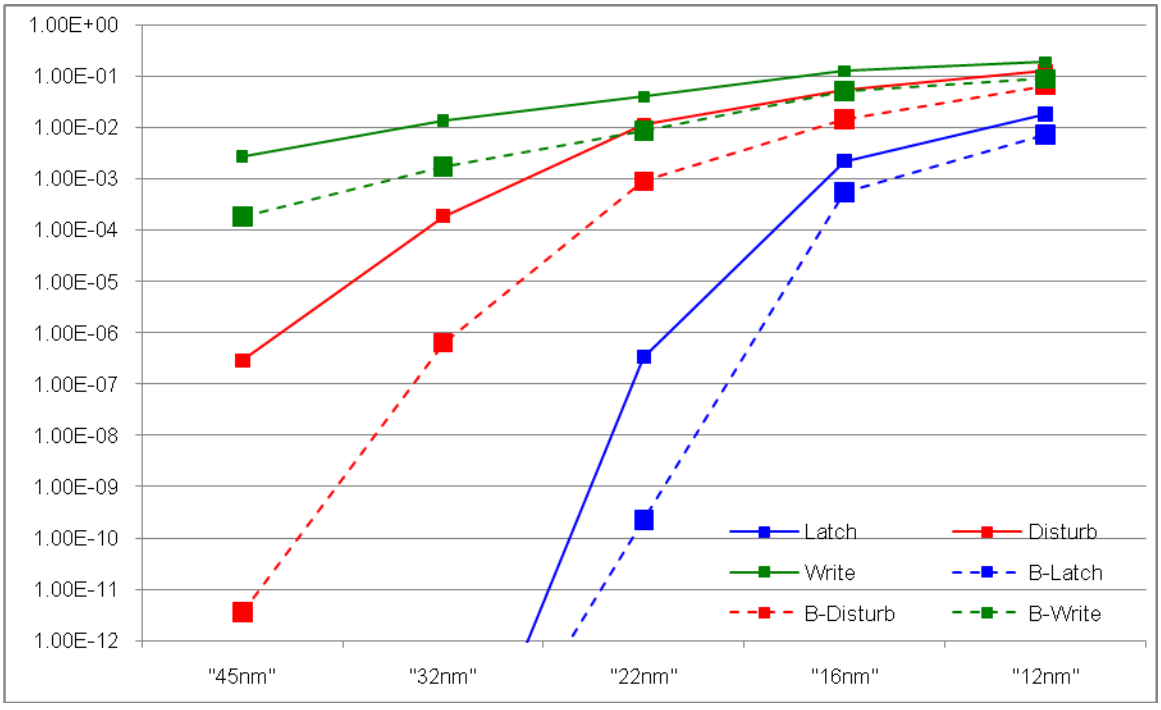

Figure DESN8 Variability-Induced Failure Rates for Three Canonical Circuit Types

The study further assumed that traditional circuit design approaches (circuit topology, P/N ratio, appropriate sizing, etc.) continue unchanged. Circuit-level innovations, such as use of 8-transistor SRAM cells, were explicitly not included so as to make the predictions consistent. In this way, the study projects trends in a future of incremental scaling of circuit and design techniques.

Failure rates for the three canonical circuits are shown in Figure DESN8. The figure shows technologies on the x-axis, and the failure probability on the y-axis. Three families of curves are shown for the latch and the two SRAM failure modes. Inverters do not show failure rates above the minimum value of the plot at one part per billion. Each family has

two curves, one solid and one dashed; the dashed line denotes the performance of the circuit when the device widths are scaled up by a factor of 40%.

1. SRAM failure rates, already a significant problem that requires extensive design intervention such as the introduction of massive redundancy measures, will continue to be a problem and will require even more circuit and architectural innovations to combat increasing manufacturing variability.

2. Latches, which share some broad similarity with SRAM but were more robust at the 65nm node, reach SRAM failure rates at or around the 22nm node. This will necessitate the introduction of new circuit and/or architecture techniques.

3. Enlarging circuits (reverse scaling) is moderately effective at controlling the impact of variability.

An important lever in reducing the impact of variability is the power supply. It is well appreciated that raising the power supply voltage can significantly reduce circuit failure rates due to variability. Of course, this comes at the expense of additional power consumption, which is already a major factor for all types of designs. In order to understand the impact of power supply voltage on failure rates, we expanded the study reported in Figure DESN8 to include the dependence of failure rates on power supply voltage. The range studied was from 10% below the nominal, a common worst-case assumption in digital circuit design, to 20% above the nominal. We show the results in Figure DESN9, where the x-axis is the power supply (as a percentage of the nominal supply for each technology node), and the y-axis is the probability of failure. To simplify the plot, we plot only the results for the Latch and for the SRAM Write failure for the 32nm, 22nm and 16nm nodes. We can see that the power supply has a very strong impact on latch failure rates, and a somewhat more modest impact on SRAM Write failures (still about one order of magnitude for this range of voltage variations).

This observation leads to a clear engineering tradeoff between power and robustness. Absent other sources of innovation at the device and circuit levels, one of the few effective levers for reducing circuit fail rates is power. Given that power itself is now one of the major design drivers, designers and technologists will have to take great care in developing balanced solutions for this problem.

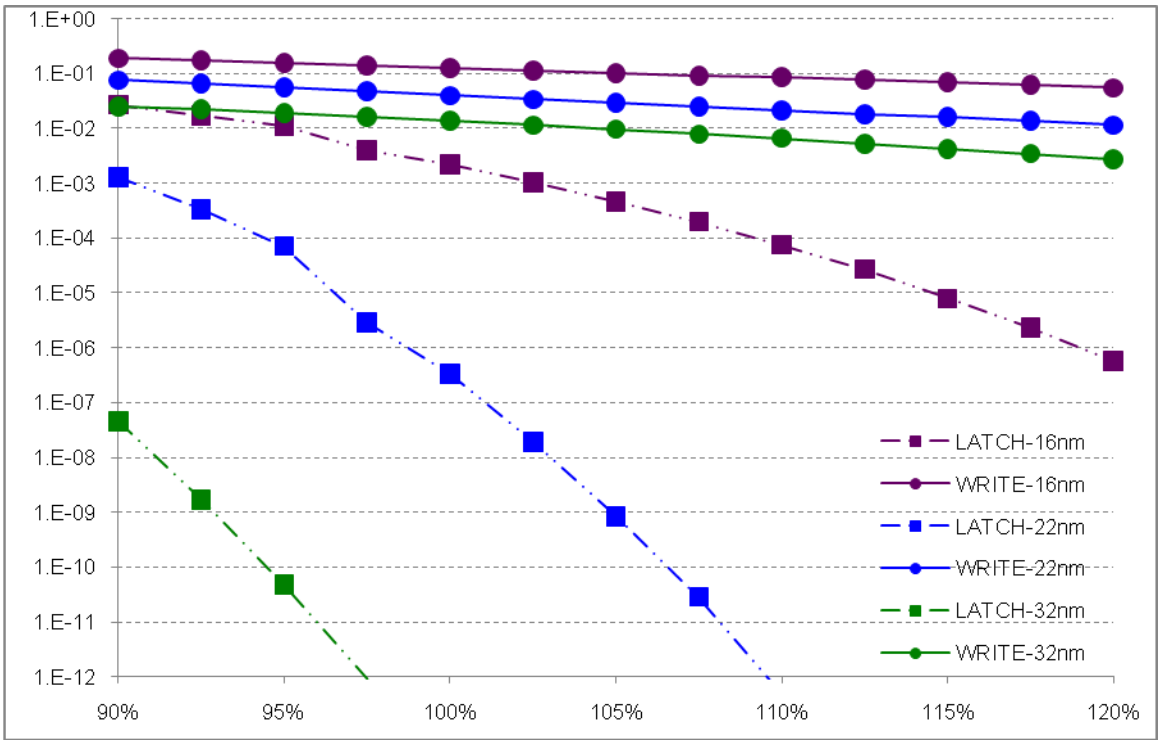

Figure DESN9 Power Supply-Dependent Failure Rates for Three Canonical Circuit Types

Figure DESN10 describes DFM potential solutions.

- *Solutions that address fundamental economic limitations*—Future DFM tools will be required that directly account for economic factors, including but not limited to mask cost, in their main algorithms and interfaces. Teams and their managers will be able to more directly assess the economic value of difficult DFM tradeoffs.
- *Solutions that address the impact of variability*—Future DFM will have to directly address and tolerate various dimension of variability. First, they will need to address variability in both performance and power consumption—thereby leading to a surge in the importance of statistical performance analysis, active power, and leakage power analysis tools. Second, they will need to address and distinguish between two distinctive types of statistical yield-loss behavior: systematic and random. Third, they will need to optimize across the various sources of environmental and process-induced variability, including voltage supply, temperature, and threshold voltage variability. Finally, tool-based solutions will not be sufficient—design techniques that help compensate for variability will be needed, including 1) sophisticated adaptive circuits that can sense and differentiate between sources of variability and adjust circuit activity, supply, clocking and/or inputs, and 2) architectures that are fundamentally resistant to variability, including locally asynchronous designs, redundancy, and error correcting signal encodings (ECC).
- *Solutions that address the impact of lithography limitations*—due to the prominent role played by lithography, the ITRS addresses lithography-related DFM problems and areas requiring solutions. Future design flows will need to include a radically increased effort specifically addressing lithography limitations. These techniques will likely include both rule-based (which does not change tools and/or flows) and model-based (directly changing tools and/or flows) layout correction. First, RET, which are applied post-layout today, will need to increasingly interact with traditional design steps, such as synthesis, timing, placement and routing, and they will need to more explicitly include performance and power consumption metrics in their functionality. This interaction may be direct or indirect, in the form of "model-based" design steps, such as physical verification and synthesis. Second, conventional rules and designs will increasingly become "manufacturing friendly," such as fundamentally manufacturable by design. Rules will be a key piece of these solutions, including manufacturing-friendly rules (strict "hard" rules that follow basic yet effective manufacturability principles), radically restricted rules (RDRs, simple rules that ensure manufacturability at little area or performance cost, such as grid-like layouts with no diagonals), and litho-router-friendly standard cells and cores.

Early solutions that directly handle variability (e.g., in timing analysis) have emerged as previously predicted. The embedding of statistical methods throughout the design flow, while slower to occur than previously predicted, is still viewed as inevitable. In the meantime, such methods will be selectively applied as they mature, or will be part of premium design technologies used for high-value or high-volume designs. DFM techniques that directly account for lithography have become very popular, but will take even longer to become qualified in production flows, primarily due to their even tighter link to manufacturing models and data. The next decade will feature chips that will be swarmed with manufacturability issues—fortunately, however, the design flow will have been completely overhauled by then with production-level DFM techniques that will be transparent to the designer except for the information that can effectively be used to address manufacturability directly on the design.

Table DESN11 establishes correspondences between the DFM requirements and the DFM potential solutions.

First Year of IC Production	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026
Tools that account for mask cost in their algorithms																
RET tools that are aware of circuit metrics (timing, power)																
Radically-restricted rules (grid-like layouts, no diagonals, etc.)																
Statistical analysis and optimization tools and flows (Vdd, T, Vth)																
Statistical leakage analysis and optimization tools																
Architectures resistant to variability (redundancy, ECC)																
Adaptable and redundant circuits																
Post-tapeout RET interacting with synthesis, timing, P&R																
Model-based physical verification																
Model-based physical synthesis																
Manufacturing-friendly design rules, including Radically Restricted Rules																
Litho-friendly (router-friendly) standard cells																
Tools that consider both systematic and random yield loss																
3D and integration of high-quality regulators and inductors																
Fin-fets and thin-body-soi-fets																
Extreme ultraviolet lithography																

This legend indicates the time during which research, development, and qualification/pre-production should be taking place for the solution.

Research Required

Development Underway

Qualification / Pre-Production

Continuous Improvement

Figure DESN10 Design for Manufacturability Potential Solutions

Table DESN11 Correspondence between Design for Manufacturability Requirements and Solutions

	Solution	*Explanation of the Correspondence*
Mask cost	Tools that account for mask cost in their algorithms	Obvious
	RDRs (grid-like layouts, no diagonals, etc.)	Better manufacturability and yield, less mask complexity
	RET tools aware of circuit metrics (timing, power)	More effective optimization, fewer design iterations
	Statistical leakage analysis and optimization tools	Estimation and control of soaring leakage variability
	Post-tapeout RET interacting with synthesis, timing, P&R	By interacting with earlier-in-the-flow EDA tools, can more effectively address litho issues
	Model-based physical verification	Can address litho issues with precision
	Model-based physical synthesis	Explicit litho model-based approach moves into the physical synthesis toolset
	Manufacturing-friendly design rules (hard rules)	Reduces mask, manufacturing cost; addresses printability
% V_{dd} variability seen in on-chip circuits	Tools that account for mask cost in their algorithms	Obvious
	RDRs (grid-like layouts, no diagonals, etc.)	Better manufacturability and yield, less mask complexity
	RET tools aware of circuit metrics (timing, power)	More effective optimization, fewer design iterations
	Statistical leakage analysis and optimization tools	Estimation and control of soaring leakage variability
	Post-tapeout RET interacting with synthesis, timing, P&R	By interacting with earlier-in-the-flow EDA tools, can more effectively address litho issues
	Model-based physical verification	Can address litho issues with precision
	Model-based physical synthesis	Explicit litho model-based approach moves into the physical synthesis toolset
	3D integration of high-quality regulators, insulators	Improves power delivery problem
% V_{th} variability (doping variability impact)	Statistical analysis, opt tools and flows (V_{dd}, T, V_{th})	Better estimate of variability impact reduces overdesign
	FinFETs and ultra thin-body SOI FETs	Lowers outlook of % V_{th} variability by a factor of 2 to 3
% V_{th} variability Includes all sources	Statistical analysis, opt tools and flows (V_{dd}, T, V_{th})	Better estimate of variability impact reduces overdesign
	Adaptable and redundant circuits	Inherent circuit robustness to variability
	Statistical leakage analysis and optimization tools	Estimation and control of soaring leakage variability.
	FinFETs and ultra thin-body SOI FETs	Lowers outlook of % V_{th} variability by a factor of 2 to 3
% CD variability	RET tools aware of circuit metrics (timing, power)	More effective optimization, fewer design iterations
	RDRs (grid-like layouts, no diagonals, etc.)	Better manufacturability and yield, less mask complexity
	Adaptable and redundant circuits	Inherent circuit robustness to variability
	Statistical leakage analysis and optimization tools	Leakage power variability will soar. Statistical leakage tools are critical to estimate and control it.
	Post-tapeout RET interacting with synthesis, timing, P&R	By interacting with earlier-in-the-flow EDA tools, can more effectively address litho issues
	Model-based physical verification	Can address litho issues with precision
	Model-based physical synthesis	Explicit litho model-based approach moves into the physical synthesis toolset
	Manufacturing-friendly design rules (hard rules)	Reduces mask, manufacturing cost; addresses printability
	Extreme ultraviolet (EUV) lithography	Reduces critical dimension variability
Circuit performance variability (gates and wires)	Router-friendly standard cells	Routing-friendly rules reduce design, mask, and manufacturing complexity
	Adaptable and redundant circuits	Inherent circuit robustness to variability
Circuit power variability (gates and wires)	Adaptable and redundant circuits	Inherent circuit robustness to variability
	Statistical leakage analysis and optimization tools	Estimation and control of soaring leakage variability.
	Post-tapeout RET interacting with synthesis, timing, P&R	By interacting with earlier-in-the-flow EDA tools, can more effectively address litho issues
	Model-based physical verification	Can address litho issues with precision
	Model-based physical synthesis	Explicit litho model-based approach moves into the physical synthesis toolset
	Manufacturing-friendly design rules (hard rules)	Reduces mask, manufacturing cost; addresses printability
	Router-friendly standard cells	Routing-friendly rules reduce design, mask, and manufacturing complexity
	3D integration of high-quality regulators, insulators	Improves power delivery
	FinFETs and ultra thin-body SOI FETs	Lowers outlook of % V_{th} variability by a factor of 2 to 3
Circuit leakage power variability (gates and wires)	FinFETs and ultra thin-body SOI FETs	Lowers outlook of % V_{th} variability by a factor of 2 to 3

MORE THAN MOORE ANALYSIS

To understand the relative impact of Moore-based and non-Moore industry trends on the requirements for design technology improvement, we have classified all solutions in the five main sections in this chapter into the categories of 'geometric scaling', 'equivalent scaling', and 'functional diversification'. The results are shown in Figure DESN11.

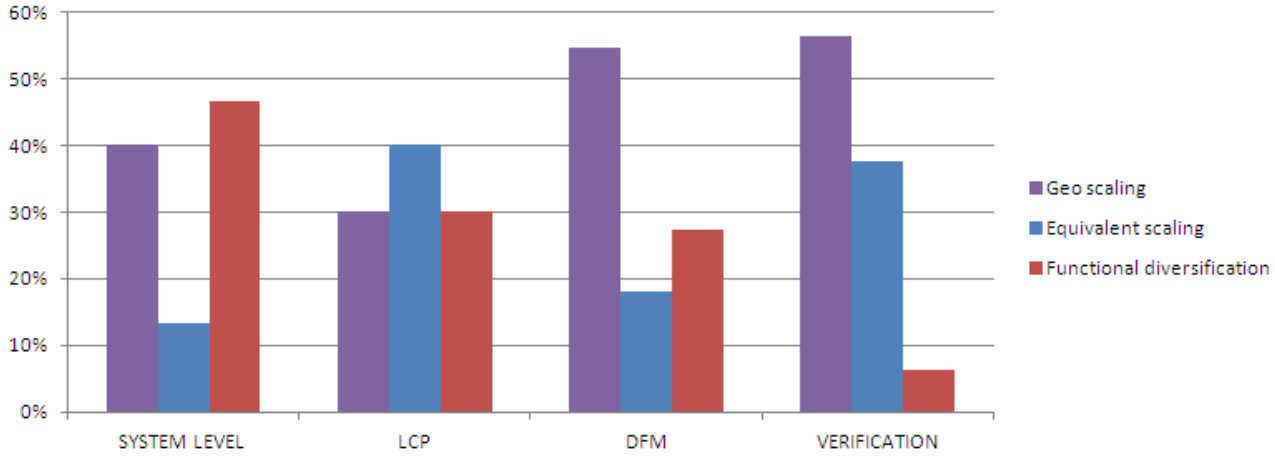

Figure DESN11 Moore and Non-Moore Design Technology Improvements

As the figure shows, it is early in the design process, during system-level design tasks, that functional diversification has greatest impact in terms of needs for solutions to the industry's product requirements. As the design process progresses to lower levels of abstraction, equivalent scaling and especially geometrical scaling techniques dominate. This is consistent with the way modern design works: 1) initially, the system level of abstraction mixes multiple cores, implementation fabrics, and circuit types, while 2) at physical realization levels of abstraction every block is synthesized into a type of layout and circuit for which well-defined design tools and methodology can be applied.

ANALOG, MIXED-SIGNAL AND RF SPECIFIC DT TRENDS AND CHALLENGES

Analog/mixed-signal and RF circuits (AMSRF circuits) are different compared to digital circuits. Digital circuits contain coarsely quantized information that is interpreted in a defined range of voltage levels only. The two states of a bit ("high" (V_{dd} – noise tolerance) and "low" (V_{ss} + noise tolerance)) are widely separated by a forbidden band. Moreover, in synchronous circuits the signal levels are only interpreted at discrete time instances (such as clock edges). The signals processed in AMSRF circuits on the other hand are continuous in time and amplitude up to a much higher degree of precision. Therefore non-idealities such as nonlinearity, noise, parasitic elements, and any kind of parametric variation of the devices of the circuit directly cause distortion and noise in the analog or RF signals. Digital circuits have the built-in ability to suppress a high level of these noise sources due to a significant gain in the transition point of each logic gate and synchronous operating paradigm. This simple signal-recovery principle cannot be used in analog and RF signal processing due to continuous operation and the much higher dynamic range of the involved signals. Speed issues, or simply the fact that a signal-recovery circuit may produce more noise and distortion than it prevents, make these issues much more challenging and less straightforward in the analog domain.

AMSRF design inherently deals with a huge number of specific classes of design problems, each of which requires a customized design approach. An operational amplifier, a sense amplifier of a RAM, a phase-locked loop, an A/D converter, an analog receiver front end – each requires its own design flow and optimization constraints. Therefore, "THE" AMSRF design flow does not exist; AMSRF design is full-custom and will never reach the same degree of automation as digital design. It should also be accepted that attempts to transfer design technology from the digital domain to the analog domain must fail due to the full-custom nature of AMSRF design. But as long as the fraction of AMSRF parts of a system is small enough that the effort for their design is small compared to the digital parts, and as long as every competitor has to deal with the same tasks this is not really a problem. However, recent advances in digital

DT have led to the situation where AMSRF components start to lie on the critical path of the chip design flow. In this situation, relief is afforded by two types of design technology enhancement: further AMSRF design automation, and substitution of analog components by digital circuit techniques.

FURTHER AMSRF DESIGN AUTOMATION

Interactive and semiautomatic AMSRF design—Despite continuing advances in academic algorithms for automated analog design, practical AMSRF design in industry is still manual if it goes beyond simulation. This is for two reasons. First, analog design is strongly interrelated to system design. For instance, the decision where to place gain elements, mixers, ADCs or DACs in the signal chain is a difficult one, as it depends on a huge number of system tradeoffs and constraints that often cannot be translated into a goal-function. Even in the digital domain, system and architecture decisions are not taken by synthesis tools but rather by experienced human system architects. Second, the design of building blocks is hard to automate, as goal and cost functions are usually multidimensional and the weighting between different performance figures is often based on experience and knowledge of other system components and constraints. For example, timing closure in digital design is a clear and hard criterion that is not subject to discussion. On the other hand, the decision to trade power for noise and accuracy is a continuous soft constraint that depends on overall system specification and other building blocks in the signal chain.

Simulation based on more or less abstract models at behavioral or transistor level is virtually the sole area where EDA tools are frequently applied in practice. Synthesis and optimization, on the other hand, remain manual despite the available methods in literature and tools. Even the synthesis and optimization tools that find their way into the tool suites of large EDA vendors vanish from the scene after a while. As a roadmap to analog synthesis has been unsuccessfully placed on the agenda for decades, we should simply let go of the goal of a digital domain-like synthesis process and set more realistic goals. Obviously, it is more desirable to support the manual AMSRF design process with computer aids and make it an interactive process between designer and computer tool. In the following, some major challenges in AMSRF design technology are described, which arise in the context of this interactive approach.

Parametric models and simulation of complex AMSRF blocks—As simulation is the acknowledged EDA method for design as well as for verification of AMSRF circuits, it must be enhanced to provide simulation of complex blocks that include both analog and (small) digital parts as well as both transistor- and behavioral-level models (mixed-mode simulation). The designer must be enabled to simulate, e.g., a phase-locked loop, an A/D converter or a receiver front end over a representative time interval. This can happen either as a whole, or in a two-step approach, where sub-blocks are simulated on transistor level and the results are transferred to the block simulation at the behavioral level. If a two-step approach is taken, it should work automatically without any requirement of designer interaction during simulation. The industrial state of the art is that transistor parameter extraction and transistor models allow the designer to automatically simulate simple AMSRF blocks such as operational amplifiers. An extensive simulation of complex blocks such as those mentioned above does not yet exist. New technologies supplementing CMOS (e.g., carbon nanotubes or MEMS), or simply passives such as coils and capacitors, lead to new types of sub-blocks that must be modeled and included in the simulation capability as well. The task is to provide complete models that can be used by the designer, rather than provide only modeling methods while leaving the modeling to the designer. The solution of this task requires the cooperation of EDA engineers and design engineers. EDA engineers contribute, e.g., automatic modeling methods such as response surface modeling or symbolic analysis; design engineers contribute, e.g., knowledge about the physical behavior to be modeled, or system requirements. Following the above-mentioned approach of interactive design support, the established sub-block and block models should not be buried in a synthesis flow or tool, but should be made available to the design engineer in the sense of library elements. Therewith, the designer gains understanding of the challenges and tradeoffs needed to make adequate system decisions – e.g., selection of a more appropriate architecture for a given building block.

Table DESN12 Required Simulation Models for AMSRF Design

Available	Required Immediately	Required As Soon As Possible
Transistor models Transistor-level models of op-amps, oscillators, mixers, low-noise amplifiers	Parameterized behavorial models of oscillators, mixers, low-noise amplifiers, etc, which are pin-compatible with corresponding transistor-level models	Parameterized behavioral models of phase-locked loops, A/D and D/A converters, receiver front ends, etc.

Standardized modeling and analysis of physical effects and complex performance features—Besides the basic transient or frequency domain behavior that must be captured by the above-mentioned simulation models, it is necessary to provide a

library of potentially relevant performance features and circuit parameters for each sub-block and block of the AMSRF circuit classes under consideration, as well as the necessary items to automatically simulate these features. The simulation models of a circuit block or sub-block should therefore include

- the set of all potential performance features (e.g., gain, bandwidth, phase margin, noise figure, input/output voltage range, PSRR and CMRR for an op-amp),
- the simulation bench (extra circuitry and input stimuli) to simulate these features, and
- the extraction routines to compute these features from the output waveforms

in such a way that the designer can select what he/she wants to simulate by simply pressing a button.

Appreciation of simulation as THE abstraction from the physical layer in AMSRF design—AMSRF design, despite the physical foundation of its finely discretized signals, provides a means to abstract from the physical layer: simulation. While simulation itself still has a physical character due to the underlying differential equations, the subsequent extraction of well-defined performance features provides for the establishment of a mathematical function whose values depend on circuit parameters included in the simulation models. This mathematical function represents the abstraction from the physical layer of an AMSRF circuit to its mathematically abstract description y=f(x). It is the task of the simulation models to provide not only the performance features y, but also any relevant parameter x, be it a design parameter such as transistor width, a parameter from the manufacturing process such as V_{th}, an operational parameter such as temperature, or a parameter that describes the aging of a component. Simulation must be prepared in such a way that the mathematical model y=f(x) is available point-wise. This means that a set of parameter values can be given, a simulation is activated, and a set of performance values results without any user interaction. This automatic simulation is the crucial interface to any interactive, semi-automatic or fully-automatic design. Electronic circuit design usually follows a hierarchical approach. This must reflect also in the simulation tools – e.g., a small transistor-level building block needs accurate modeling and simulation. After the block specification is met, most of the internal performances figures may be dropped and only some relevant parameters must be passed to the next level of hierarchy. However, this level has to handle many blocks, i.e., a much higher complexity. An appropriate simulator may handle this increased complexity at reduced accuracy. According to state-of-the-art system design SPICE simulators are used for critical blocks, fast-MOS simulators for the next level, followed by VHDL-AMS, MATLAB/Simulink, and SystemC (or equivalent tools by other vendors). So far it is the task of the designer to choose and determine the relevant parameters and to pass them to the next level of hierarchy. Together with the automatic simulation approach proposed in the previous section it would be desirable to have a predefined interface to other simulators including automatic model and parameter generation. However, it must be understood that this has nothing to do with synthesis.

Appreciation of simulation as THE interface for design automation in AMSRF design—There are several reasons why simulation should be an open interface from any EDA design tool or algorithm. The first reason is the mentioned abstraction from physics that simulation provides. An EDA design tool that allows interaction with arbitrary simulators will be applicable to a large class of design problems. A second reason is the prevalence of customized simulation environments in industry. If an EDA design tool comes with its own simulator, it will encounter resistance in a designer community that is used to its own simulators. A third reason is that the transistor and other models involved in simulation are usually specific to design classes and manufacturing technologies. Taking what is there, rather than bringing in new models, will increase the acceptance of an EDA design tool. This holds as well for response surface modeling techniques: while they may reduce overall simulation cost of an EDA design tool, they add another source of modeling error which reduces acceptance on the designer side.

Provide simulators with sensitivity analysis capabilities—It has been fashionable, necessary and advantageous to provide simulators with extensive capabilities so that designers can write their own models. However, this comes at the cost of missing capability of simulators to perform a sensitivity analysis, e.g., using the adjoint method. Today, no available simulator, with the exception of some in-house simulators, can provide the sensitivity of performance features with regard to circuit parameters at reasonable cost, e.g., 10% CPU overhead per parameter. Rather, costly finite-difference approaches are used, involving a 100% CPU overhead per parameter. The reason lies in the missing sensitivity models that allow for an adjoint-based sensitivity analysis. A big win would be for simulators to afford an inexpensive sensitivity analysis which gives a first (linear) insight into how a multi-objective, multivariate design problem behaves in detail.

Education of AMSRF design engineers and EDA tool developers—Having accepted that AMSRF design is a process where EDA tools are applied in a customized way immediately leads to the requirement of an extended education of both AMSRF designers and their EDA tool developer counterparts. In order to be able to select the right tools for his design problem and to be able to effectively apply these tools, a design engineer must know about the algorithms inside those

tools. AMSRF design curricula should therefore be supplemented by topics such as statistics and numerical optimization. On the other hand, developers of AMSRF design tools must know more about specific circuit classes and the actual design steps performed by design engineers. EDA courses should therefore be supplemented with design courses. In addition, EDA tool developers should sit close to the designers, and engineers should sometimes work "on the other side", i.e., an AMSRF designer should also develop some tool sometime, and an EDA developer should design some circuit sometime.

AMSRF design tools are not for a flow but for a class of design problems—In trying to automate parts of AMSRF design tasks, there is no virtue in talking about general levels of abstraction and where synthesis switches between them, as with digital design. More germane is the class of circuits that is targeted and that can be handled, e.g., as in Table DESN12. In AMSRF design, we basically have the (nonlinear) transistor level, the hardware description level with some language (e.g., Verilog, VHDL) and the (linear) architectural level (e.g., Matlab/Simulink). Depending on the available models, we can include a transfer from one level to the other, but the synthesis and optimization process is based on whatever model is present. With this in mind, a number of problems are still not satisfactorily solved:

- Interactive design aids for the generation and selection of circuit structures
- Interactive design aids for placement and routing
- Interactive design for yield and reliability
- Discrete optimization (e.g., layout fingers, manufacturing grid)
- Closer interaction between structural synthesis and layout synthesis
- Design space exploration
- Enhanced simulation speed

Research on these tasks must carefully consider how self-explanatory are the developed solutions, and how much "consulting overhead" they involve.

SUBSTITUTE ANALOG COMPONENTS BY DIGITAL CIRCUIT TECHNIQUES

Analog, mixed-signal, and RF circuits have always been susceptible to parameter and environmental variations. Indeed, even as variability and circuit sensitivity increase, the basic and underlying problems are not new. Countermeasures in terms of trimming, calibration loops, digital calibration and error correction are well-known. However, their realization has often not been feasible due to area and power penalties. In nanoscale CMOS technologies, power and area consumption per basic digital function is incredibly low, so that complex calibration and correction engines become affordable. If adding some thousands of digital gates can reduce the amount of analog and RF components, or their accuracy requirements, then these digital gates should be deployed. The same holds if digital assist techniques can improve the analog performance by foreground/background calibration and/or redundancy, and/or error correction. Even if the digital approach does not pay off immediately, it should be taken into account as the digital shrink factor assures a benefit over the ensuing product generation(s).

Ongoing device scaling will further degrade the analog device performance, especially the intrinsic gain of a single transistor. Circuit techniques for gain and accuracy improvement such as cascodes become impractical due to the small voltage headroom of around 1V. As noise does not scale, the decreasing signal levels lead to diminishing signal-to-noise ratios. Digital delays, on the other hand, scale continuously. The same holds for area and power of basic digital gates. This leads to an emerging class of mixed-signal circuits, namely time-to-digital converters. Time-to-digital converters are data converters that quantize a continuous input signal. While ADCs work on continuous voltages, TDCs work on continuous time intervals. The first and most prominent example for a successful application of time-to-digital converters is the all-digital phase-locked-loop.

It is to be understood that it is neither the goal, nor it is possible, to fully replace analog functionality by digital realizations. Often, analog realizations are not only straightforward but also cheaper and better-performing than digital ones. However, digital assist techniques improve the overall system performance, improve reproducibility, and provide a simple and powerful interface. Obviously, these advantages do not come for free: extensive digital circuitry near analog/RF building blocks not only causes additional power consumption but may also pollute substrate and power supply, which may lead to increased noise levels.

CROSS-CUT TWG ISSUES

MODELING AND SIMULATION

[N.B.: This text is shared with the Modeling and Simulation ITWG.] One of the key problems that designers face due to further shrinking of feature sizes is the increasing variability of design-related parameters, resulting either from variations of fabrication parameters or from the intrinsic atomistic nature which affects, e.g., channel doping. This problem is discussed in detail throughout this Design Chapter, especially in the part on Design For Manufacturability. Modeling and simulation can and must help to ease this problem by assessing the quantitative impact of such variabilities on the relevant design parameters. Statistical variations, as well as drifts of fabrication parameters, must be translated via appropriate equipment, process, and device simulation as well as parameter extraction into the resulting distributions of design parameters, such as size and spacings of active and passive devices, transistor characteristics, and coupling of interconnects leading to signal delay and distortion. Increasingly important is the atomistic nature of dopants which in some cases results in an average of just one or several dopant atoms present in the channel region, giving rise to enormous relative fluctuations of doping and, in turn, electrical device parameters. Especially important are the interactions between different subsequent process steps, such as lithography and etching, which may either amplify or smooth such process fluctuations. Simulation should further contribute to the assessment of the impact of parasitics, delay variations, noise, and reliability issues, including thermal effects during operation. The treatment of such "second-order" effects is especially important for analog design where, for example, matching is a key issue. The overall target is to link design parameters more closely to the technology and device architectures used, especially including their process-induced variations, in order to help designers to select appropriate safety margins, which may vary within the layout. This should also help design to avoid the overcompensation of variations in design resulting from the use of corner models, which increasingly tend to be over-pessimistic. Analog and RF designs have driven the need for high-precision compact modeling as well as for characterization and extraction of all kinds of device non-idealities. However, extraction of simple rules for higher levels of abstraction in analog and RF circuit design (or even layout) is made very difficult by variations and non-idealities such as noise and parasitic elements. The added value which only simulation can provide is that a wide set of variations may be investigated largely automatically, within relatively short runtimes, and at relatively small costs. In this way, Modeling and Simulation must contribute to the solution of the problem that historical process tolerances can in future no longer be met, and that therefore realistic estimates of the new tolerances and their implications must be provided. To achieve this goal, appropriate methodologies must be developed to extract relevant information from the microscopic TCAD simulations (which are mostly carried out at the device or cell level) in a format which allows further processing with design tools—e.g., via SPICE parameters and their statistical distributions.

Particularly in the near term, issues related to mask making, e.g., the efficient definition and assessment of assist features needed to transfer features from layout into the photoresist, and resulting variations of electrical parameters such as threshold voltage, are especially important. Simulation must not only contribute to the correction and adaptation of masks to make sure that the feature printed on the wafer approximates well enough the "ideal" structure intended by the designer, but also help to avoid costly overcorrection of the mask features and to select the most cost-efficient mask structure for the printing of the required features on the wafer. Besides this, a long-term challenge will be the uncontrollable CD and dopant variability. In the end, Modeling and Simulation should contribute to solving the design challenge of yield prediction and optimization by providing information on the impact of process variations and the printability of defects. This would allow designers to adapt their designs to be less sensitive to these non-ideal effects, and thereby increase yield.

APPENDIX I: VARIABILITY MODELING AND ROADMAP

Since variability is expected to be the source of multiple critical DFM and resilience challenges, a systematic method to roadmap required and/or desired variability trends will become a crucial component of the overall Design roadmap. It may also allow for design-manufacturing "co-roadmapping," which may help gather indications of whether our industry should invest in variability reduction or in design robustness/productivity improvements. The need for such a framework is further underscored by the sensitivity of variability-related information for industry and foundry participants.

The design community's view of technology is centered on the device model parameters used in circuit simulation (such as the commonly used BSIM model). These models participate in the design process by enabling a hierarchy of derived models, starting at the device level, then the gate and other component levels, and terminating at models for chip-wide

performance quantities such as power, frequency, leakage, lifetime etc. There are two main *modeling stacks*, corresponding to the two major *design fabrics* used in digital circuits.

1. The first stack is for logic circuits implemented using simple primitive gates (such as a NAND gate). This model stack includes models for gate-level timing and power, and relies on static timing analysis as the signoff engine.

2. The second stack is for memory circuits, which are implemented as complex arrays of memory bit cells with interface circuitry. This model stack requires models of inherent device variability and relies on statistical analysis methods (such as Monte-Carlo) for signoff.

A variability roadmap framework (VRF) is being developed by the Design TWG and is illustrated in Figure DESN12. In this framework ,the following four levels of abstraction have initially been considered:

- *Circuit/chip level*—this is the abstraction level most relevant to designers. Ideally, timing and power consumption variability for a given circuit would be roadmapped at this level, based on lower-level parametric variations.
- *Gate/component level*—for digital systems, current design methodologies operate at this level via standard synthesis/placement/routing/timing abstractions and tools. These components are often stored in *libraries*, and such libraries are not a standard part of the design/technology interface.
- *Device level*—since circuits are composed of devices, at this level device-related parameters such as threshold voltage V_{th} or off-current I_{off} are roadmapped.
- *Physical level*—this is the level closest to the design-manufacturing interface, where parameters such as CD or effective device length (L_e), and actual doping level (N_A) are roadmapped. These parameters' variability stems from some of challenging issues enumerated above, including limitations of lithography hardware and the inability to control the exact number of dopants in a channel.

In this framework, the modeling process can be symbolically described as

$$\Delta \text{ outputs} = \text{model } (\Delta \text{ inputs}) \qquad [1]$$

Based on published approximated models, this model uses a simplified gate + wire circuit slice, whereby parameters are decomposed into gate- and wire-related parameters. Performance is analytically modeled in the current gate-based model as total delay, which in turn is decomposed into gate delay and wire delay. Delay variability is modeled as a delta that is statistically computed from a statistical distribution of individual delay values that stems from "simulating" distributions of the input parameters in a Monte Carlo style. Model inputs include gate channel length and width, oxide thickness; wire width, length, and thickness; ILD thickness, and wire sheet resistance.

Similar models for SRAM memory arrays need to be developed and integrated with the logic circuit models in order to make holistic predictions for modern chips. Such models may integrate first order statistical bitcell models into arrays with performances such as parametric yield, leakage power, access time, and so on.

Among other applications, this framework will be used for trend analysis of circuit performance variability. For example, estimated delay variability will be estimated over time for two possible scenarios: 10% required CD (actual channel feature length) variation, and 20% required CD variation. In this particular illustration, if the model indicates that circuit performance variability does not vary significantly with the CD tolerance requirement, this might suggest that the requirement may be relaxed.

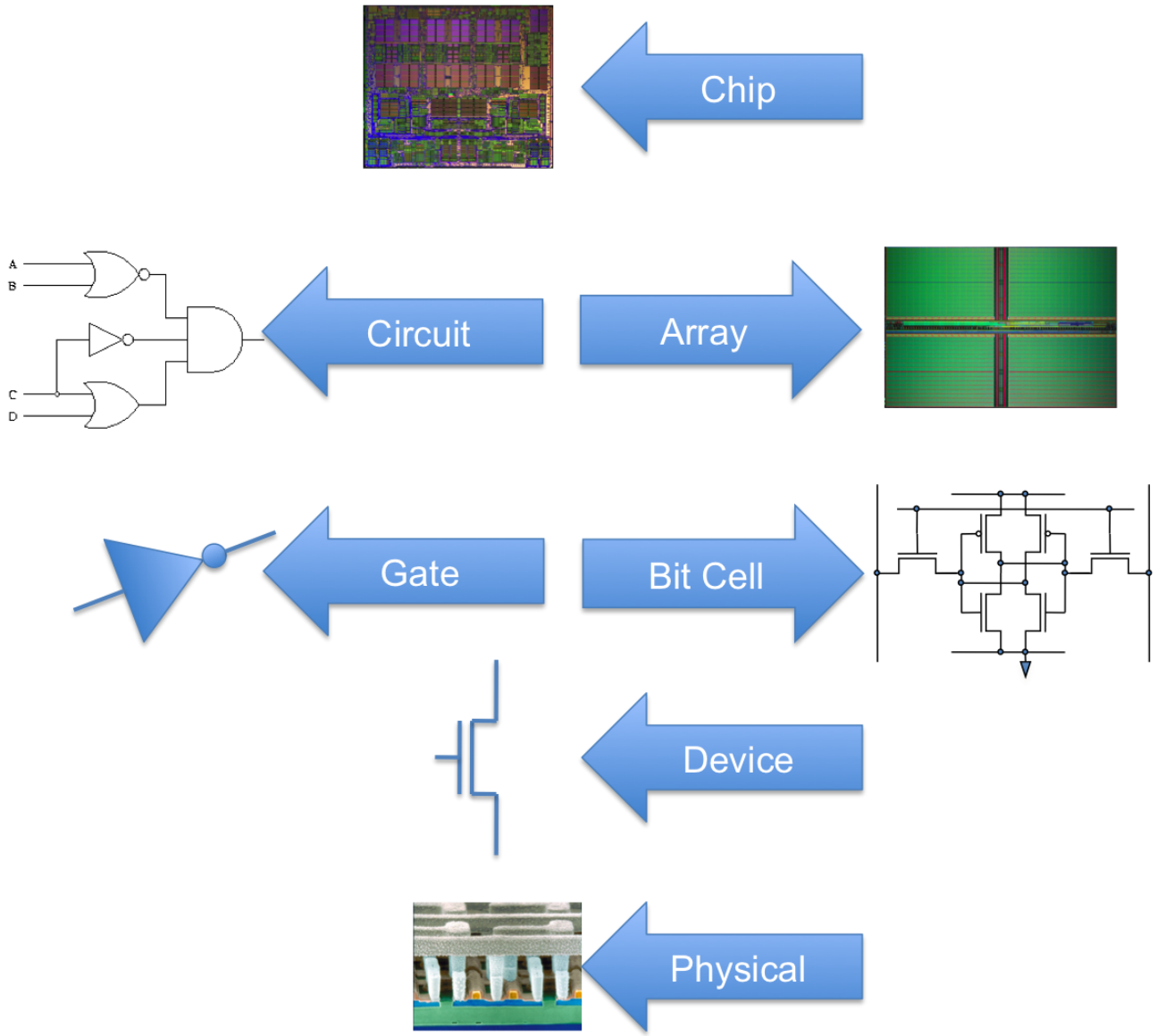

Figure DESN12 Possible Variability Abstraction Levels

APPENDIX II: DT COST AND VALUE

Figure DESN13 shows how many different factors affect the cost of developing and marketing a complex integrated circuit. Each factor represents either a fixed or a variable cost component. While fixed costs do not depend on the number of units being sold, variable costs grow with the number of units sold. Product development is a fundamental part of the electronic product value chain, and can generally be seen as a fixed cost factor that is spread across the number of units sold. For the purpose of this discussion, design cost is defined as the direct product development R&D cost plus its associated overhead. Unfortunately, the ever-increasing complexity of systems-on-a-chip makes design costs, and thus unit costs, difficult to control. Rising costs, combined with an increasingly competitive environment, can make profitability ever more difficult to achieve. This is exacerbated by the fact that product development costs come upfront in the life cycle, whereas substantial revenue often comes years later (discounted-cash-flow effect). The following

analysis suggests that without a continued design technology innovation pipeline, design cost (and thus product development cost) would quickly become prohibitive, or else designs will be forced to have less valuable content.

In Figure DESN13, virtually all items under the Development R&D header can be seen as design costs (opportunity and lost-revenue costs are not included). The figure shows that product development cost can be roughly decomposed into direct labor costs and infrastructure costs. Labor costs include chip, circuit, and layout/physical design; chip integration; verification and test; software development; EDA integration; and software and technology support. Infrastructure costs include design software licenses (including software development environments), test chip infrastructure, and depreciation. These costs are often expressed as direct costs plus an allocated "overhead" component, including general and administrative expenses. The vital contribution of DT to semiconductor product profitability can be understood by enumerating and analyzing the impact of DT innovations on each of these cost components.

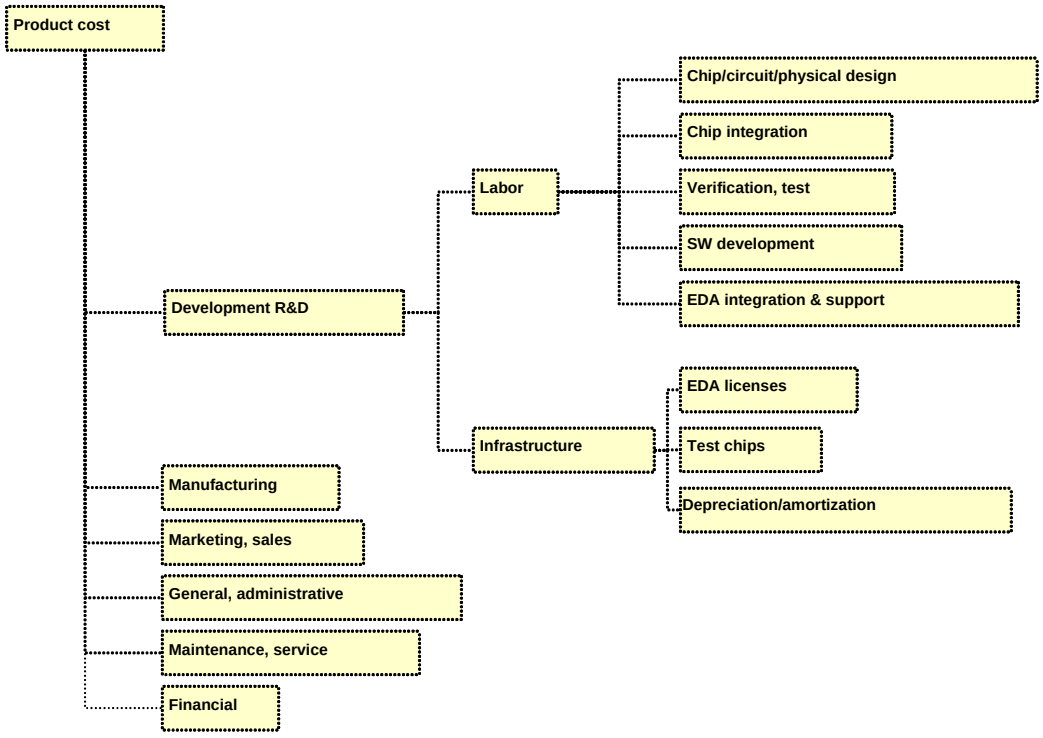

Figure DESN13 Simplified Electronic Product Development Cost Model

LABOR COSTS

The labor cost component is approximately proportional to *labor unit costs* (in terms of compensation dollars per engineer-year), *design complexity* (in terms of the amount of behavior or devices in a chip) and *designer productivity* (in terms of the design complexity that an average engineer can fully design in a year):

$$DesignLaborCost = \frac{LaborUnitCost \times DesignComplexity}{Designer\ Pr oductivity}$$

Since DT innovations have increased designer productivity, their strongest effect is on the labor cost component. To measure the labor cost impact of DT innovation, Gartner/Dataquest was requested by the ITRS Design ITWG to measure designer productivity and to calibrate productivity improvements from major DT innovations. Designer productivity was measured at 4K gates (=16K transistors) per year in 1990, the year in which the so-called "RTL methodology" originated. Succeeding improvements are described in Table DESN13, where the gray items denote ongoing or future DT innovations. The table shows that designer productivity (measured as number of logic gates per designer-year) has grown

an average of 31.2% per year from 1990 to 2011. Specifically, the number of designers per million gates (inverse of productivity) went from 250 in the year 1990 to 1 in 2011. Labor unit costs, however, have not remained constant since 1990. According to the ITRS design cost model, the historical rate of increase in engineer labor cost is 5% per year (assuming salary and overheads starting at $181,568 in 1990). While growth slowed to 2% during 2002 thru 2005, it returned to the typical 5% per year in 2007 and 2008. The global recession caused salaries to go flat in 2009; however, they rebounded to the 5% growth level in 2010 and beyond. With IC design engineering shortages being reported from India and China, the cost of a high-level SOC design engineer worldwide is being driven up to the North American level, which we have been using in the design cost model.

INFRASTRUCTURE COSTS

The rate of increase in EDA tool cost *per engineer* is estimated at 3.9% per year (starting at $99,301 per engineer in 1990). But again in 2002 that growth stopped and has been below historical norms for the last eight years. Tool average selling price growth is expected to be at its typical rate of 3.9% going forward. The total implied infrastructure cost from this item is given by the product of EDA tool cost times the number of engineer-years:

$$EDAInfrastructureCost = \frac{EDAUnitCost \times DesignComplexity}{Designer\ Productivity}$$

which relates this cost to labor cost. Other infrastructure costs are assumed to be included as overhead in the current model. Since average labor unit costs have grown faster than EDA infrastructure costs, the proportion of labor in the product development costs is increasing.

TOTAL DESIGN COST

To bring this chapter full circle, we refer again to Figure DESN1, which quantifies the impact of the DT innovations on design cost for the SOC-CP (Consumer Portable) driver defined in the *System Drivers Chapter*. The SOC-CP has 48.9M logic gates in 2011, implying a typical consumer/portable SOC hardware design cost (designers + tools) of almost $25.7M, plus $14.1M in software design cost; software design cost exceeded hardware design cost between 2007 and 2010. Without the seven major DT innovations that occurred between 1993 and 2009, the hardware design cost alone for this SOC would have been approximately $7,708M in 2011. This gap becomes larger if we use an alternative estimate of current designer productivity developed by the Japanese Semiconductor Technology Roadmap Working Group 1 (STRJ-WG1) and cited in the 2001 ITRS *System Drivers Chapter*; that estimate sets new (reuse) logic productivity to be 360K (720K) gates/designer-year in 1999, a 6× (12×) factor higher than the Gartner/Dataquest estimate for the same year.

Table DESN13 Design Technology Improvements and Impact on Designer Productivity

DT Improvement	*Year*	*Productivity Delta*	*Productivity (Gates/Year/Designer)*	*Cost Component Affected*	*Description of Improvement*
None	1990		4K HW		
In-House Place and Route	1993	38.90%	5.55K HW	PD Integration	Automated block placement and routing, transferred from the semiconductor house to the design team
Tall-Thin Engineer	1995	63.60%	9.09K HW	Chip/circuit/PD verification	Engineer capable of pursuing all required tasks to complete a design block, from RTL to GDSII
Reuse—Small Blocks	1997	340%	40K HW	Circuit/PD verification	Blocks from 2,500–74,999 gates
Reuse—Large Blocks	1999	38.90%	56K HW	Chip/circuit/PD integration verification	Blocks from 75,000–1M gates
IC Implementation Suite	2001	63.60%	91K HW, 87K SW	Chip/circuit/PD integration EDA support	Tightly integrated tool set that goes from RTL synthesis to GDSII through IC place and route
RTL Functional Verification Tool Suite	2003	37.50%	125K HW, 87K SW	SW development verification	Tightly integrated RTL verification tool suite including all simulators and formal tools needed to complete the verification process
Transactional Modeling	2005	60%	200K HW, 250K SW	SW development verification	Level above RTL, including both HW and SW design and consisting of behavioral (where the system function has not been partitioned) and architectural (where HW and SW are identified and handed off to design teams) levels
Very Large Block Reuse	2007	200%	600K HW, 323K SW	Chip/circuit/PD verification	Blocks >1M gates; intellectual-property cores
Homogeneous Parallel Processing	2009	100% HW, 100% SW	1200K HW, 646K SW	Chip/circuit/PD design and verification	Many identical cores provide specialized processing around a main processor, enabling performance, power efficiency, and high reuse
Software Virtual Prototype	2011	300% SW	1200K HW, 2584K SW	SW development	Virtualization tools used to allow development prior to completed silicon
Intelligent Testbench	2012	37.5% HW	1650K HW, 2584K SW	System design and verification	Like RTL verification tool suite, but also with automation of the verification partitioning step
Reusable Platform Blocks	2013	200% HW, 100% SW	4949K HW, 5168K SW	Chip/circuit/PD verification	Fully functional platforms used as a block in larger platform design (e.g., ARM in OMAP)
Silicon Virtual Prototype	2015	100% HW	9897K HW, 5168K SW	System design and verification	A hardware virtualization platform that enables an RTL handoff of a SOC
Heterogeneous (AMP) Parallel Processing	2017	100% HW, 100% SW	19794K HW, 10336K SW	SW development verification	Many specialized cores provide processing around a main processor, which allows for performance, power efficiency, and high reuse
Many-Core SW Development Tools	2019	60% SW	19794K HW, 16537K SW	SW development	Enables compilation and SW development in highly parallel processing SOCs
Concurrent Memory	2021	100% SW	19794K HW, 33074K SW	SW development	Memories capable of on-chip memory management
System-Level Design Automation (SDA)	2023	60% HW, 37.5% SW	31671K HW, 45476K SW	System design and verification	Automates true system design on- and off-chip for the first time, including electronic, mechanical and other heterogeneous technologies
Executable Specification	2025	200% HW, 200% SW	95013K HW, 136429K SW	System design and verification	Describes the system specification in a manner that allows automated design and validation
Total		7920% HW, 21119% SW			

APPENDIX III: DT-BASED REDUCTIONS OF POWER CONSUMPTION

Table DESN14 shows eight major low-power DT innovations for the upcoming 15 years (2011~2026). Each has impacts on static and dynamic power reduction. Table DESN14 also lists ten DT innovations which occurred before 2011. We assume that three of these previous innovations (adaptive body biasing, power gating and dynamic voltage/frequency scaling) will have some amount of remaining impact (1.19x, 4.64x and 1.00x for static power, 1.05x, 0.93x and 1.11x for dynamic power) on SOC-CP power reduction over the next 5, 7, and 10 years, respectively. Device-level innovations (e.g., ultra thin-body SOI, FinFET, etc.) also have impacts on power reduction, but these are roadmapped in the *PIDS Chapter* and their improvements are hence already integrated into the SOC-CP modeling. As a result, we do not list them here.

The "Power Chart" in Figure DESN14 shows the impact of DT innovations for low-power design on the SOC-CP power given in Figure SYSD6. Figure DESN14 quantifies the impact of previous and future DT innovations on the power consumption of the SOC-CP design. The SOC-CP starts with 48.8M logic gates in 2011 and ends with 1995.5M logic gates in 2026, with corresponding power dissipation values of 3.5 watts and 43.9 watts, respectively. With the integration of low-power impacts from future DT innovations, SOC-CP will instead consume only 8.22 watts in the year 2026.

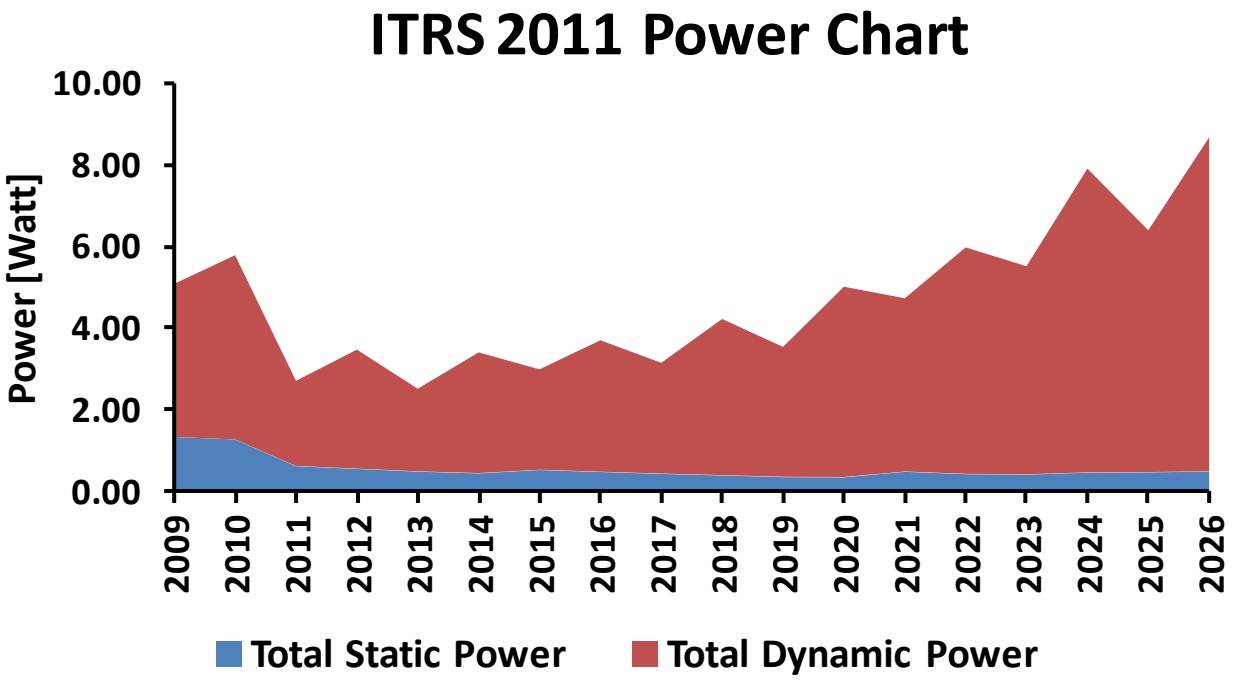

Figure DESN14 Impact of Low-Power Design Technology on SOC Consumer Portable Power Consumption

Table DESN14 Low-Power Design Technology Improvements and Impact on Dynamic and Static Power

DT Improvement	*Year*	*Dynamic Power Improvement (x)*	*Static Power Improvement (x)*	*Description of Improvements*
Low Power Physical Libraries	Before 2011	1.50	1.50	Optimizing transistor size, layout style and cell topology for the standard-cell library
Back Biasing		1.00	1.35	Biasing wells of devices independently of the sources to shift the threshold voltage
Adaptive Body Biasing (ABB)		1.20	2.00	Delivering a positive or negative voltage below a transistor to reduce leakage
Power Gating		0.90	10.00	Turning off the power supplies to idle blocks for leakage reduction
Dynamic Voltage/Frequency Scaling (DVFS)		1.50	1.00	Dynamic management of supply voltage and operating frequency for power reduction
Multilevel Cache Architecture		1.00	1.20	Reduce amount of off-chip memory accesses for performance improvement and power reduction
Hardware Multithreading		1.00	1.30	Using multithreads to improve hardware utilization with leakage reduction
Hardware Virtualization		1.00	1.20	Using one physical server to support multiple guest operating systems simultaneously
Superscalar Architecture		1.00	2.00	Parallel instruction issue and execution for performance improvement and power reduction
Symmetric Multiple Processing (SMP)		1.50	1.00	Lowering the frequency by using multiple processors and parallel programing
Software Virtual Prototype	2011	1.23	1.20	Virtualization tools to allow the programmer to develop software prior to silicon
Frequency Islands	2013	1.26	1.00	Designing blocks that operate at different frequencies
Near-Threshold Computing	2015	1.23	0.80	Lowering Vdd to 400 - 500 mV
Hardware/Software Co-Partitioning	2017	1.18	1.00	Hardware/software partitioning at the behavioral level based on power
Heterogeneous Parallel Processing (AMP)	2019	1.18	1.00	Using multiple types of processors in a parallel computing architecture
Many Core Software Development Tools	2021	1.20	1.00	Using multiple types of processors in a parallel computing architecture
Power-Aware Software	2023	1.21	1.00	Developing software using power consumption as a parameter
Asynchronous Design	2025	1.21	1.00	Non-clock driven design
Total		4.66	0.96	

APPENDIX IV: DESIGN CHALLENGES FOR 3DIC

The objective of this appendix is to identify design challenges related to deployment of 3DIC chip technologies, with a specific focus on 3DIC structures relying on chip or wafer bonding and Through-Silicon Vias (TSVs).

Advanced highly integrated 3D stacked systems will be enabled by a number of technologies that are fast becoming mainstream. These include chip-to-chip, chip-to-wafer and wafer-to-wafer bonding, high-density microbump technologies (typically copper and solder), and TSVs that enable connection through to the backside of a chip or wafer. These technologies enable the intimate integration of CMOS silicon chips, passive silicon structures, and semiconductors from other families, including III-V's. Beyond 3D enabled by these technologies there is the exciting option of monolithic 3D integration, with multiple high quality transistor layers integrated onto one substrate.

These technologies enable a broad range of design solutions that are likely to more aggressively exploit the technology as the technology matures. Three phases of design product maturity, together with design related challenges are summarized in Table DESN15.

The first phase of 3D deployment, occurring now, will focus on DRAM stacks and interposer-enabled products. Design will be conducted largely using incremental extensions of current techniques. However, there are specific challenges whose solution is useful. In general, codesign with interposers introduces several challenges, including deciding on the appropriate interposer stack-up, power delivery with thin and thick metals, and chip-package codesign for high-density I/O with high-density interposers. Details of the interposer technology have to be properly modeled to permit thermal, power and signal integrity analysis (though this is little different than current chip-package codesign practices). A specific design challenge that applies to all generations of design, but must be addressed early, is how to manage TSV induced stress in pre-strained transistor processes – the stress will change transistor mobility in the vicinity of the TSV. Will this be solved by new keep-out design rules, or are more sophisticated design solutions likely to be available? Test and yield management is a challenge to all generations. In this first generation a specific problem is how to test a high density interposer cost-effectively, especially if it has fine-pitch I/O. Another problem is how to optimize the total test and yield cost for a DRAM stack.

The flagship product in the second generation is likely to be a "wide I/O" DRAM stack integrated with a mobile processor. Several new design challenges arise. Even though the total heat flux is low, thermal-driven floorplanning will be needed, including consideration of transient heat spikes. Thermal verification will need commensurate accuracy. Since power is being delivered through the TSV stack, power integrity and IR drop control analysis becomes essential. In this generation, chips are being designed by different vendors and being integrated by the system integrator. Thus design interchange formats (and supporting extraction methods) are needed to permit thermal, power and signal integrity analysis to be performed with the needed accuracy. Test becomes more complex due to the need to separately test the logic and DRAM die, then perform an integrated test. The correct balance between fault coverage, test complexity and test escape needs to be determined. New test techniques that reduce overall test cost will be valuable.

The third generation will be characterized by increasing system density and increased heterogeneity. Server class computing is likely to require high-bandwidth integration of large amounts of memory with high-power processors. Sensors are likely to require highly heterogeneous chip stacks. Sometimes heterogeneity might mean that some of the technologies in the stack are only slightly different than others (e.g. just different metal options). Monolithic 3D stacking is likely to become available in some form. The design challenges become increasingly complex. How to decide on the optimum technology mix simultaneously with the best system architecture? How to manage power delivery and thermal dissipation cost-effectively? With the anticipated low V_{dd} levels, in-the-stack DC-DC conversion becomes valuable. Analysis tools will have to operate on more complex problems and with higher fidelity results, as dictated by growing system complexity and high power fluxes. Test and yield management will require a system-level optimization.

Table DESN15 Three Phases of Design Product Maturity and Design Challenges in 3DIC

	2011 - 2013	2013 - 2017	2017-2020
3D Technologies	Interposers	Memory tightly integrated with Logic	Multiscale heterogeneous 3D
	Homogeneous silicon stack		Optimized 3D subsystems
			Monolithic 3DICs
Exemplar Product(s)	DRAM stack	Mobile memory-on-logic	Exascale compute node
Product Advantages	Yield enhancement	Significant power savings	Highly integrated systems
	Miniaturization	Bandwidth enhancement, especially to 2-4 DRAMs	Solve the memory wall
			Cost-optimized systems
Design Challenges			
Early Design (Pathfinding)	Chip-package codesign with interposers	Thermal-driven floorplanning, including transients	Heterogeneous system optimization – system planning; cost; thermal and power delivery co-optimization
Thermal & Stress	TSV stress design rules	Transient thermal prediction to 5 deg C accuracy. Total chip-package stress control	Transient thermal prediction to 1 deg C accuracy
Power Delivery	Power integrity with interposers	Power integrity and IR drop with interposers and TSVs to 10 mV accuracy	Low-overhead power delivery of 100+ Amps with 10 mV accuracy
Standards and Formats	Chip-package design interchange for power and signal integrity and thermal design	Chip-package and chip-chip design interchange to permit 5 deg C and 10 mV accuracy	Interchange of complex tradeoffs including heterogeneous die and multiple package solutions
Test and Yield Management	Optimized test flow and yield management for ~4 chip stacks. Cost-effective interposer testing at < 10 micron pitch	Optimized test flow and yield management for logic on memory ~ 4 chip stacks	Optimized test flow and yield management heterogeneous systems of > 10 die