INTERNATIONAL TECHNOLOGY ROADMAP FOR SEMICONDUCTORS 2005 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 2005 ITRS, the Design International Technology Working Group (ITWG) 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 2005 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. Without foundry amortization and return-on-investment (ROI) for supplier industries, the semiconductor investment cycle stalls. ITRS editions prior to 2003 have documented a *design productivity gap*—the number of available transistors grows faster than the ability to meaningfully design them. Yet, investment in process technology has by far dominated investment in design technology. The good news is that enabling progress in DT continues. Figure 16 shows that estimated design cost of the power-efficient system-on-chip (SOC-PE) defined in the *System Drivers chapter* is near $20M in 2005, versus around $900M had DT innovations between 1993 and 2005 not occurred (analysis details are given in the Appendix). The bad news is that software can account for 80% of embedded-systems development cost; test cost has grown exponentially relative to manufacturing cost; verification engineers outnumber design engineers on microprocessor project teams; etc. *Today, many design technology gaps are crises.*

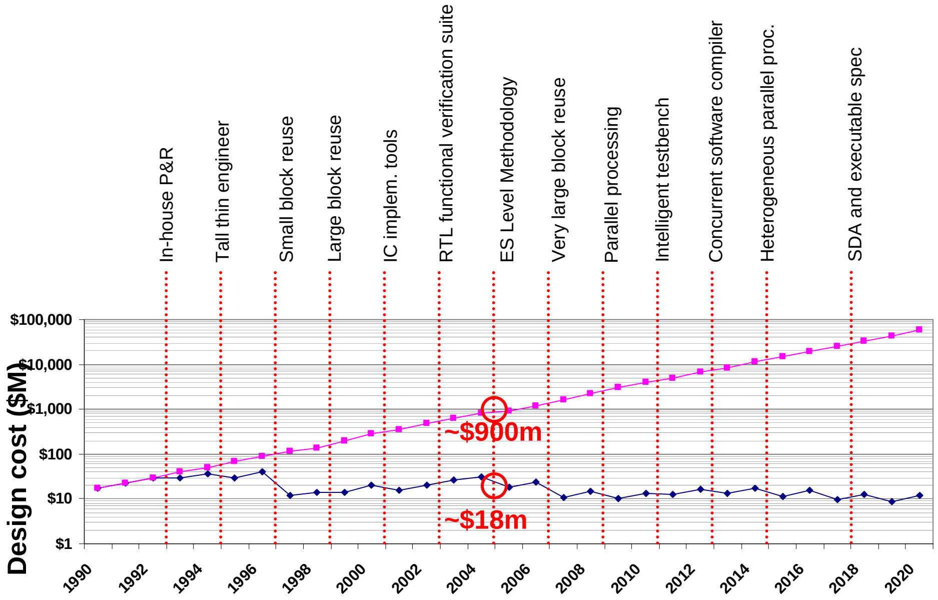


Figure 16 Impact of Design Technology on SOC PE Implementation Cost

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, thereby forming the first worldwide quantitative design technology roadmap. The organization follows a traditional landscape of DT areas (see Figure 18): design process; system-level design; logical, circuit and physical design; design verification; design test, and a new area—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, such as the electronic design automation (EDA) industry. As appropriate, the detailed challenges are mapped to the microprocesser (MPU), system on chip (SOC), 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 given. The overall approach reflects the rise of application- and driver-specific DT.

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 a design that will meet those 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 trade-offs 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, alignment of technology advances with the ITRS generations is less strict for DT. While ITRS technology generations occur discretely when all needed technology elements are in place, DT improvements can generally improve productivity or quality even "in isolation," and are thus deployable when developed.

[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).*

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)
- *Complexity of manufacturing handoff* (reticle enhancement and mask writing/inspection flow, NRE cost)
- *Process variability* (library characterization, analog and digital circuit performance, error-tolerant design, layout reuse, reliable and predictable implementation platforms)
- *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)

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 (like direct-mapped custom through general-purpose software programmable) easily span four orders of magnitude in performance (such as GOps/mW), and there is added opportunity to leave value on the table via ill-advised guard bands, 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. Design specification and validation become extremely challenging, particularly with respect to complex operating contexts. Trade-offs must be made between all aspects of value or quality, and all aspects of cost. (A simplistic example: "Moore's Law" for clock frequency might suggest trade-off of design time (= time-to-market) for speed 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.

[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). Similarly diminishing returns in the MPU space (Pollack's Rule) are described in the System Drivers chapter.*

- *Embedded software design*—predictable platform-based electronic system design methodologies, codesign with hardware and for networked system environments, software verification/analysis
- *Reliable implementation platforms*—predictable chip implementation onto multiple circuit fabrics, higher-level handoff to implementation and
- *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 in general provide concurrent optimization and analysis of more complex objectives and constraints, acknowledge additional considerations such as design reuse and manufactured system cost in the design optimization, and encompass additional scope such as embedded software design and interfaces to manufacturing. The tremendous scope of silicon and system complexities is in itself also a challenge to the roadmapping of DT and the EDA industry. Five *crosscutting* challenges—1) productivity, 2) power, 3) manufacturing integration, 4) interference, and 5) error-tolerance—are given for which potential solutions are distributed across all areas of DT, underlying the overall design cost "meta-challenge."

Design productivity, closely linked to system and design process complexity, and of course affecting design cost, is the most massive and critical, both in the short and long term, and is affected by the other four. The second through fifth crosscut challenges are narrower in scope, and mostly address silicon complexity issues. From those challenges, the most critical are power consumption and manufacturability.

- Power consumption is an urgent, short-term challenge, quickly shifting from a performance-driven active power crisis to a variability-driven leakage power crisis in the long term. Power consumption is classified as an "enhancing performance" type challenge in the ITRS Executive summary.
- Manufacturability, that is, the ability to produce a chip in large quantities at acceptable cost and according to an economically feasible schedule, has been affecting the industry primarily due to lithography hardware limitations but will become a major crisis in the long term as variability in its multiple forms invades all aspects of a design. Manufacturability is classified as a "cost-effective manufacturing" type challenge in the ITRS Executive summary.

Table 12 Overall Design Technology Challenges

Challenges ≥32 nm	*Summary of Issues*
Design productivity	System level: high level of abstraction (HW/SW) functionality spec, platform based design, multi-processor programmability, system integration, AMS co-design and automation Verification: executable specification, ESL formal verification, intelligent testbench, coverage-based verification Logic/circuit/layout: analog circuit synthesis, multi-objective optimization
Power consumption	Logic/circuit/layout: dynamic and static (leakage), system and circuit, 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
Reliability	Logic/circuit/layout: MTTF-aware design, BISR, soft-error correction
Interference	Logic/circuit/layout: signal integrity analysis, EMI analysis, thermal analysis
Challenges <32 nm	*Summary of Issues*
Design productivity	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 Heterogeneous component integration (optical, mechanical, chemical, bio, etc.)
Power consumption	SOI power management
Manufacturability	Uncontrollable threshold voltage variability Advanced analog/mixed signal DFT (digital, structural, radio), "statistical" and yield-improvement DFT Thermal BIST, system-level BIST
Reliability	Autonomic computing, robust design, SW reliability
Interference	Interactions between heterogeneous components (optical, mechanical, chemical, bio, etc.)

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 Design HW/SW—hardware/software MTTF—mean time to failure SOI—silicon on insulator

CHALLENGE 1—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; 5) automated methods for AMS design and test, which are required by the SOC and AMS system drivers. These improvements will 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 developed and improved as well. Time-to-market of new design technology must be reduced, such as via standards and platforms for interoperability and DT reuse.

CHALLENGE 2—POWER

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. These challenges are as follows:

1) Extrapolation from the *Overall Roadmap Technology Characteristics* and *System Drivers chapter* shows that high performance (HP) MPU power consumption significantly exceeds the high-performance single-chip

package power limits established in the *Assembly and Packaging chapter*, even with allowed power densities in excess of 250 W/cm^2. The SOC-low-power personal digital assistant (SOC-LP PDA) driver requires flat average and standby power, even as logic content and throughput continue to grow exponentially. DT must address the resulting *power management gap* that is shown in the System Drivers chapter.

2) Increasing power densities worsen thermal impact on reliability and performance, while decreasing supply voltages worsen switching currents and noise. These trends stress on-chip interconnect resources (such as to control infrared (IR) drop in light of the Assembly and Packaging roadmap for bump count and passivation opening size), ATE equipment limits, and burn-in paradigms.

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_t, 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 in light of both scaling and variability.

CHALLENGE 3—MANUFACTURABILITY

"Red bricks," technology requirements for which no known solutions exist, are increasingly common throughout the ITRS. On the other hand, challenges that are impossible to solve within a single technology area of the ITRS may be solvable (more cost-effectively) with appropriate intervention from, or partnership with, DT. Feasibility of future technology generations will come to depend on such "sharing of red bricks." 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 NRE cost can be reduced through more intelligent interfaces to mask production and inspection flows.

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, while apparently scalable through the 100 nm generation, 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.

CHALLENGE 5—RELIABILITY

Relaxing the requirement of 100% correctness for devices and interconnects may dramatically reduce costs of manufacturing, verification, and test. Such a paradigm shift is likely 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) Beyond 90 nm, single-event upsets (soft errors) severely impact field-level product reliability, not only for embedded memory, but for logic and latches as well.

2) Current 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. In general, automatic insertion of robustness into the design will become a priority as systems become too large to functionally test at manufacturing exit. Potential measures include automatic introduction of redundant logic and on-chip reconfigurability for fault tolerance, development of adaptive and self-correcting or self-repairing circuits, and software-based fault-tolerance.

DESIGN TECHNOLOGY CHALLENGES

The remainder of this chapter details quantified challenges and potential solutions in the five traditional areas of DT, preceded by an overview of the design methodology. 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. Even more difficult is determining the most effective sequence in which issues should be considered, and design decisions made, in order to minimize iterations. Four major trends govern future leading-edge chip design processes and their supporting design system structures.

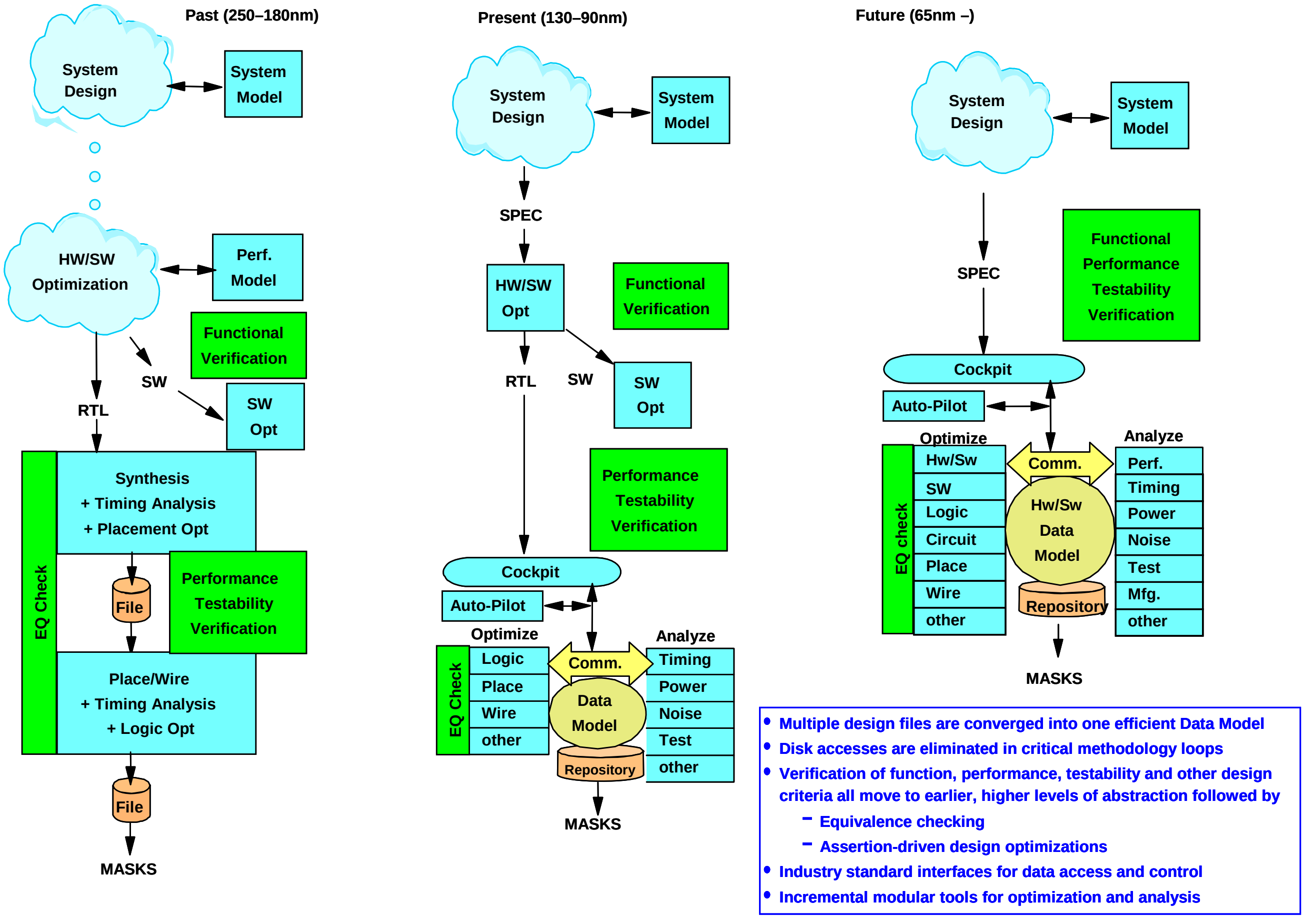
Trend 1: Tight coupling—Design processes that formerly comprised series of batch tools operating from files are evolving into collections of modular applications that operate concurrently and share design data in memory. In modern methodologies, optimization loops can no longer contain slow file accesses, and the plethora of design issues requires simultaneous optimization of multiple criteria. This trend is visible today in most commercial design systems, where logical optimizations are working together with placement, global routing and timing analysis to close timing on advanced chip designs. Further advances will be needed to avoid noise problems, minimize power dissipation and ensure manufacturability. Figure 17 illustrates this trend. The left column shows a hardware design process, where synthesis, timing and some placement are combined to handle the impact of placement on wire delays and synthesis results. Some place and route systems of that era used limited logic changes to reduce routing congestion. The middle column illustrates today's design process, in which suites of analysis and optimization modules cooperate to produce a chip with acceptable performance, power, noise, and area while maintaining testability and manufacturability. The right column illustrates the required design system for the future, in which hardware and software are co-analyzed and co-optimized to achieve an acceptable system implementation.

Trend 2: Design for manufacture—Preparation of mask data for manufacturing is an increasingly critical part of the design process. Past "data prep" applications converted design data into information for mask making.

[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 the implementation of the steps that comprise a methodology and are discussed below 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.*

Today, basic shapes that describe a design, along with added shapes that correct process distortions and enhance printability, are handed off to mask making via standard file formats. However, flaws in the current paradigm have caused exponential growth of manufacturing NRE cost. First, corrective shapes (reticle enhancement technology (RET), metal fill, etc.) are inserted without complete understanding of effects on the printed wafer or on mask cost; designs are hence "over-corrected." New characterizations of manufacturing process and cost trade-offs are needed to enable more intelligent data preparation. Second, mask inspection and repair is the largest component of cost and delay in mask making, yet is also performed without insight into design intent. Effort is wasted in satisfying identical tolerances for every shape. A standard framework is needed to communicate criticality of design data to the mask making process, and feed manufacturing complexity back to the design process. Such bidirectional communication between design and manufacturing can help contain future costs of chip design and manufacture. Given the growth of mask and foundry outsourcing, this communication must be via an industry standard interface. Finally, new design and analysis tools that use manufacturing characteristics to optimize the design across all phases of the design cycle must work with enhanced manufacturing software to achieve design intent with minimum total cost. Figure 17 shows the influence of manufacturability, for example, previous wiring tools were successful in spacing apart wires to improve critical-area yield. Today there is greater focus on yield-driven layout, and manufacturability is a standard design criterion. In the 65 nm generation and beyond, design and manufacturing data must be unified in a single database, so that designers can understand early on the impact on mask cost when making design trade-offs, and so that manufacturing flows can understand design intent when applying yield- or cost-driven optimizations. In general, manufacturability will join power, performance and signal integrity as a first-class goal of future multi-objective design optimization.

Trend 3: Increasing level of abstraction—While some critical components are still crafted at the device level today, most design is at the gate-level for greater productivity, and register-transfer level (RTL) is used to specify design in a modern flow. Each advance in the level of abstraction requires tremendous innovation to discover the correct primitive concepts that form the basis of the abstraction, and to create the tools that allow trade-offs to be considered at this level, and map results to the next lower level. For continued improvements in designer productivity, an emerging system-level of design, well above RTL, is required.



- Multiple design files are converged into one efficient Data Model
- Disk accesses are eliminated in critical methodology loops
- Verification of function, performance, testability and other design criteria all move to earlier, higher levels of abstraction followed by
  - Equivalence checking
  - Assertion-driven design optimizations
- Industry standard interfaces for data access and control
- Incremental modular tools for optimization and analysis

Figure 17 Required Evolution of Design System Architecture

Higher levels of abstraction allow many forms of verification to be performed much earlier in the design process, reducing time to market and lowering cost by discovering problems earlier. Figure 17 illustrates this trend, with green verification boxes occurring earlier in the design process as DT advances. A finer-grain breakdown of this trend is as follows. 1) *Functional verification* used to begin when a gate-level implementation was available for simulation; early models were rarely complete or accurate. Today, functional verification can begin at the RT-level with Boolean equivalency checking of the later gate-level implementation. The result is accurate and provides more efficient, earlier functional verification. For future system-level abstraction, “transaction-level” modeling[4] is emerging as a strong possibility, with equivalence checking via lower-level RTL descriptions still required. 2) *Performance and timing verification* confirm product performance as early as possible, allowing time for redesign. Early performance verification techniques based on mathematical or simulation models yielded early estimates, but required confirmation via later gate-level simulation. RTL modeling now provides earlier estimates, and system-level modeling will provide even earlier feedback (transaction-level models can provide early cycle-accurate performance estimates). While detailed timing analysis is always done at gate or transistor levels where device models are best understood, emergent RT- and system-level timing estimators promise valuable early feedback to designers, earlier in the design process. Tight coupling within design tool suites helps achieve the necessary predictability in meeting design constraints and gives designers more confidence that the RTL implementation process will meet early performance estimates. 3) *Testability verification* is yet another vital checkpoint in the design process. While this historically required a gate-level implementation, more sophisticated tools and self-test methods now permit a high level of confidence in a design’s testability at the

[4] *In a transaction-level model, both data and time are abstracted. Atomic actions may take multiple cycles and complex data transfers may be represented with simple read and write commands.*

RT-level. Similar advances are required to provide confirmation at the system-level of abstraction in very early design stages.

Trend 4: Increasing level of automation—Historically, new levels of abstraction have been established primarily for more efficient simulation and verification of larger designs. However, once designers start to use the new models to specify design intent, opportunities arise for other tools such as synthesis. An important aspect of this trend is the replacement of designer guidance by constraint-driven optimization, so as to reduce the number of iterations in later process steps. Today's RTL design process emerged in this way, and a similar advance is needed at the system level. This trend is also visible in Figure 17. At the left, more comprehensive performance models emerge from what had been a very informal and indirect connection between initial system specification and automated RTL implementation. Executable system-level specification has matured in the current technology generation, and in future generations the system-level specification must include both software and hardware, and become the controlling representation for constraint-driven implementation. More detailed requirements for system-level DT, along with concrete flows that realize the progression of Figure 19, are given in the System-Level Design section, below.

DESIGN TECHNOLOGY BREAKOUT

In system-level design[5], 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. The context is daunting. Silicon complexities such as variability and reliability mean that highly 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, electronic system design must comprehend networking and distributed computation metaphors (for example, with communications structures designed first and functional blocks then integrated into the communications backbone), as well as interactions between functional and interconnect pipelining. Furthermore, the portion of SW in embedded systems dramatically increases, such that HW/SW co-design is one the major challenges in system level design at present time.

With respect to the 2003 edition of the ITRS roadmap, the challenges in system level design remain widely the same, which proves their enormous complexity. 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. This will require the following advances along the above-noted trends of increased abstraction and increased automation.

- *Reuse-based design in both HW and SW domains*—Reusable, high-level functional blocks ("cores" or intellectual-property (IP) blocks) offer the potential for productivity gains that are estimated to be at least 200%. Pre-verification and reusable tests reduce design complexity, and libraries of reusable software modules can speed embedded software development. Ideally, an SOC designer can rapidly assemble a set of cores into a complex, application-oriented architecture as easily as drawing a block diagram. In practice, some amount of new cores and software will typically be required in the system, somewhat slowing implementation. The reuse of IP blocks, such as, for instance, processor cores and multimedia codecs, in the HW domain, has made some progress during the past years. Nevertheless, the

[5] *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" corresponds to implemented circuit elements, and "software" 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).*

basis of reusable, and, in particular qualified and customizable cores for various application domains yet has to be significantly extended in order to fully permeate the electronic system design process.

- *Platform-based design*—An extension of core-based design creates highly reusable *groups* of cores to form a complete hardware "platform," further simplifying the SOC design process. With highly programmable platforms that include one or more programmable processor(s) and/or reconfigurable logic, derivative designs may be created without fabricating a new SOC. Platform customization for a particular SOC derivative then becomes a constrained form of design space exploration: the basic communications architecture and platform processor choices are fixed, and the design team is restricted to choosing certain customization parameters and optional IP from a library. Platform-based design also entails HW–SW partitioning, which decides the mapping of key processing tasks into either HW or SW, and which has major impact on system performance, energy consumption, on-chip communications bandwidth consumption, and other system figures of merit. Multi-processor systems require "SW–SW" partitioning and co-design, i.e., assignment of SW tasks to various processor options. While perhaps 80–95% of these decisions can be made *a priori*, particularly with platform-based or derivative SOCs, such co-design decisions usually are made for a small number of functions that have critical impact. Platform-based design is increasingly permeating the design process. However, while the number of available platforms is growing, the critical point is the lack of sufficient tool support for supporting partitioning decisions, design space exploration, and automated mapping to different target platforms on system level.
- *System-level verification*—Fundamental to raising the abstraction level is a single notation for system-level design. Several years of experimentation with C, C++, and Java variants has led to recent emergence of SystemC as a reasonable form for building interoperable system models of hardware and software for simulation. As noted in the Design Process section, transaction-level modeling shows promise for high-performance system simulation. On the backside, the application of SW languages to the HW domain also introduces additional sources for errors and design flaws. For that reason, standardized and generally accepted functional verification technologies, which pay respect to the requirements in the HW domain, too, are required for the near future, which are, for instance, currently under development for SystemC. In addition, formal methods may be able to exploit this higher level of abstraction, allowing application to larger problems.
- *Micro-architecture synthesis*—As a standard form for system-level specification is adopted for simulation and verification, other tools will emerge. Although system synthesis is an extremely difficult task, progress can be envisioned according to a sequence of innovations. The first step will likely be automatic creation of an effective RTL specification from a slightly higher-level representation of the hardware in the form of a micro-architecture specification. Figures 18 and 19 illustrate this advance. Figure 18 shows a typical modern design flow (a realization of the left portion of Figure 18) with a mixture of manual steps above RTL and an automated process for RTL implementation. Figure 19 depicts a flow that is required in the near future (~2007); the manual process of mapping micro-architectural design decisions into RTL is going to be replaced with an automated step.

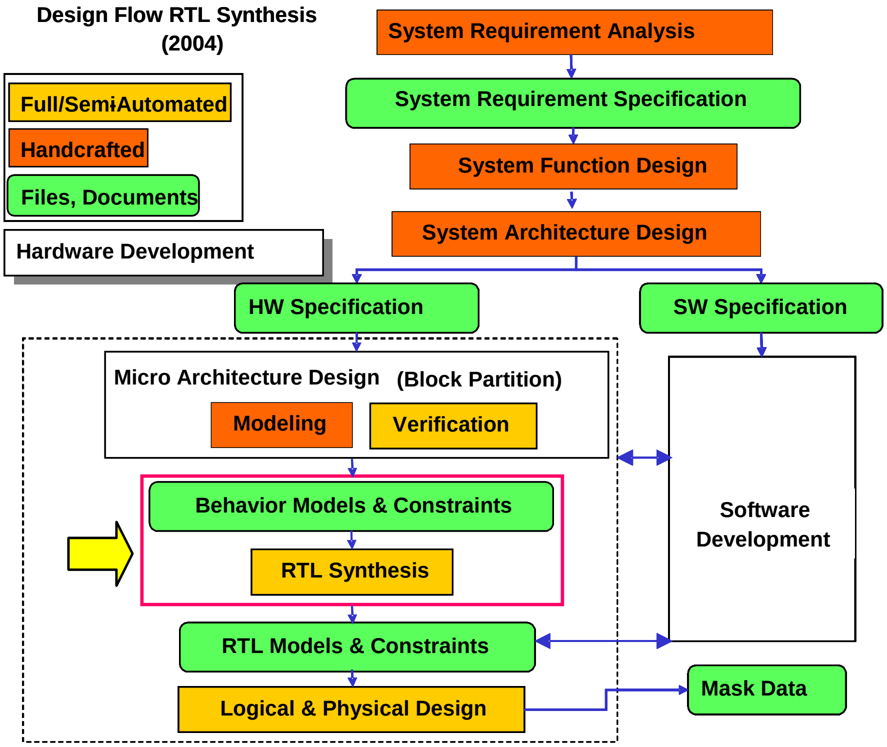

Design Flow RTL Synthesis (2004)
Full/Semi-Automated
Handcrafted
Files, Documents
Hardware Development
System Requirement Analysis
System Requirement Specification
System Function Design
System Architecture Design
HW Specification
SW Specification
Micro Architecture Design (Block Partition)
Modeling
Verification
Behavior Models & Constraints
RTL Synthesis
RTL Models & Constraints
Logical & Physical Design
Software Development
Mask Data

Figure 18 RTL Synthesis for Design Flow in Year 2004

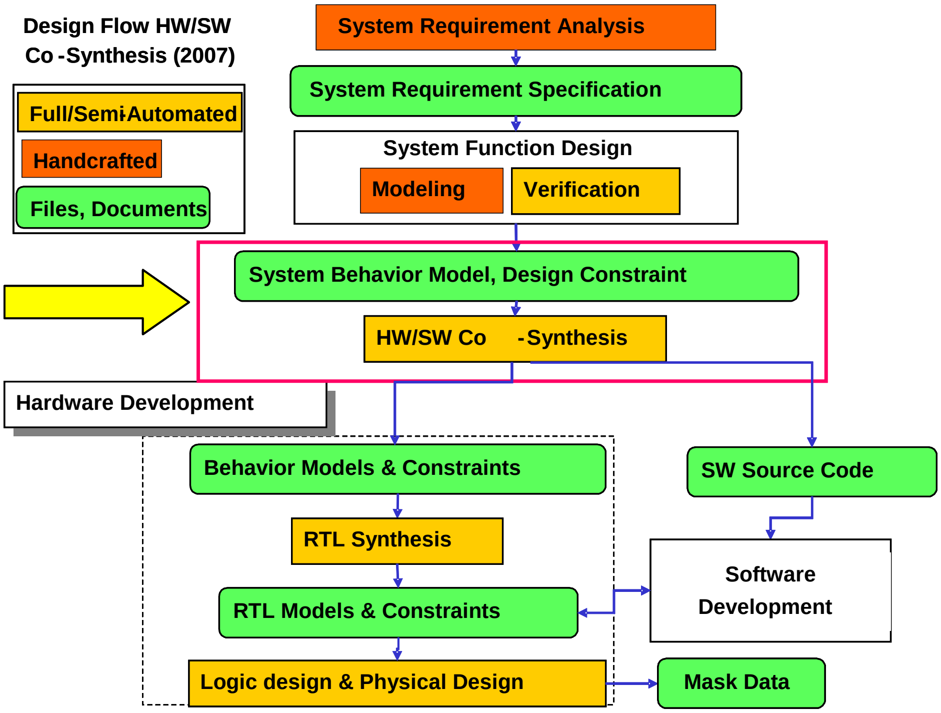


Figure 19 Design Flow in Year 2007

- *HW–SW co-synthesis*—The next required advance is the ability to concurrently synthesize a hardware and software implementation to achieve the best overall solution. Figure 19 shows the corresponding design flow, which is required circa 2007; the manual process of mapping a behavioral specification to a software program and a hardware micro-architecture has been replaced with an automated step. A form of HW–SW co-synthesis is "co-processor synthesis," in which software descriptions of algorithms are analyzed and automatically, or semi-automatically, divided into two parts: 1) a control structure which remains as SW running on a standard (on-chip) RISC processor, handling most of the control branching but only a relatively small amount of the computation, and 2) a data-processing/dataflow portion which is implemented in HW as co-processor(s) to complement the control SW. Due to the HW implementation, the latter part can reduce execution time of the overall application up to 90–95%, depending on the amount of control versus dataflow processing and the nature of the hardware fabric to which the data processing is mapped. Co-processor synthesis has already begun to appear commercially with some success, and remains the most likely form for HW–SW co-synthesis in the next several years. A key to the possibilities in this space is a unified (and software-based) representation of an application's implied processing, which can then be partitioned into HW and SW forms without undue manual effort.
- *High level analog specification and synthesis*—The more complex systems are going to be, the more likely is that they do not only comprise HW and SW components, but also analog components, which is essential for many application domains, for instance, wireless applications, automotive electronics, et cetera. A system level specification language or methodology which claims to be suitable to describe such systems entirely must, hence, not only comprise the capability to describe HW and SW in a holistic way, but also analog components. Automated or semi-automated synthesis of these high level specifications of analog components is then the logical next step that has to be reached. In the future also the necessity to integrate further components from new domains, such as, for instance, micromechanics, in the same way will likely arise.

The following table (Table 13) lists quantitative requirements for system level design for the next generations of technology generations:

Table 13a System Level Design Requirements—Near-term Years

Year of Production	*2005*	*2006*	*2007*	*2008*	*2009*	*2010*	*2011*	*2012*	*2013*
DRAM ½ Pitch (nm) (contacted)	*80*	*70*	*65*	*57*	*50*	*45*	*40*	*36*	*32*
Design Reuse									
Design block reuse [1] % to all logic size	32%	33%	35%	36%	38%	40%	41%	42%	44%
Platform Based Design									
Available platforms [2] Normalized to 100% in the start year [3]	96%	88%	83%	83%	75%	67%	60%	55%	50%
Platforms supported [4] % of platforms fully supported by tools [5]	3%	6%	10%	25%	35%	50%	57%	64%	75%
High Level Synthesis									
Accuracy of high level estimates (performance, area, power, costs) [6] % versus measurements	53%	56%	60%	63%	66%	70%	73%	76%	80%
Reconfigurability									
SOC reconfigurability [7] % of SOC functionality reconfigurable	23%	26%	28%	28%	30%	35%	38%	40%	42%
Analog/Mixed Signal									
Analog automation [8] % versus digital automation [9]	12%	14%	17%	17%	24%	24%	27%	30%	32%
Modeling methodology, description languages, and simulation environments [10] % versus digital methodology [11] [12]	53%	55%	58%	60%	62%	65%	67%	70%	73%

Table 13b System Level Design Requirements—Long-term Years

Year of Production	*2014*	*2015*	*2016*	*2017*	*2018*	*2019*	*2020*
DRAM ½ Pitch (nm)(contacted)	*28*	*25*	*22*	*20*	*18*	*16*	*14*
Design Reuse							
Design block reuse [1] % to all logic size	**46%**	**48%**	**49%**	**51%**	**52%**	**54%**	**55%**
Platform Based Design							
Available platforms [2] Normalized to 100% in the start year [3]	**46%**	**43%**	**42%**	**39%**	**36%**	**33%**	**32%**
Platforms supported [4] % of platforms fully supported by tools [5]	**80%**	**85%**	**90%**	**92%**	**94%**	**95%**	**97%**
High Level Synthesis							
Accuracy of high level estimates (performance, area, power, costs) [6] % versus measurements	**83%**	**86%**	**90%**	**92%**	**94%**	**95%**	**97%**
Reconfigurability							
SOC reconfigurability [7] % of SOC functionality reconfigurable	**45%**	**48%**	**50%**	**53%**	**56%**	**60%**	**62%**
Analog/Mixed Signal							
Analog automation [8] % versus digital automation [9]	**35%**	**38%**	**40%**	**43%**	**46%**	**50%**	**52%**
Modeling methodology, description languages, and simulation environments [10] % versus digital methodology [11] [12]	**76%**	**78%**	**80%**	**83%**	**86%**	**90%**	**92%**

Manufacturable solutions exist, and are being optimized	□
Manufacturable solutions are known	🟨
Interim solutions are known	◆
Manufacturable solutions are NOT known	🟥

Notes for Table 13a and b

[1] This requirement is not unique to system level design, but it is also a requirement to design, in general (See the SOC-PE Productivity Trends table in the System Drivers chapter).

DEFINITION

The portion of a design, which is not newly developed but which is composed of pre-existing components.

RATIONALE

Reuse is one of the main factors which drive design productivity, and it is one of the key concepts behind system level design.

The reuse in year n from a particular reference year can be calculated according to the following formula

reuse(n) = 1 - (1 – reuse) • ((1 + pgrowth)^n / (1 + cgrowth)^n)

with

reuse: reuse in reference year

pgrowth: (expected) mean annual productivity growth rate, excluding the effect of reuse

cgrowth: (expected) mean annual growth rate of design complexity

and assuming, that the size of the design staff as well as the design cycle time stay constant during the considered period of time.

The rationale for the formula is that the gap between the productivity growth (without effect of reuse) and the complexity growth has to be filled by reuse, if the technological progress shall be fully exploited in SOC design.

[2] DEFINITION

A platform is a specific combination of system components that support specific application areas (e.g. wireless, automotive, consumer electronics/multimedia, Small Office Home Office (SOHO) networks, etc.). System components are one or more processors, (real time) operating system, communication infrastructure, memory, customizable analog and digital logic, and virtual sockets for new logic. Basic functionality for the application area is provided by a number of already integrated components. System differentiation is achieved by integration of few new components either in hardware or software.

RATIONALE

Platform based design is an important driver for design productivity, since it highly promotes reuse. In addition, system level specifications require platforms to which they can be mapped.

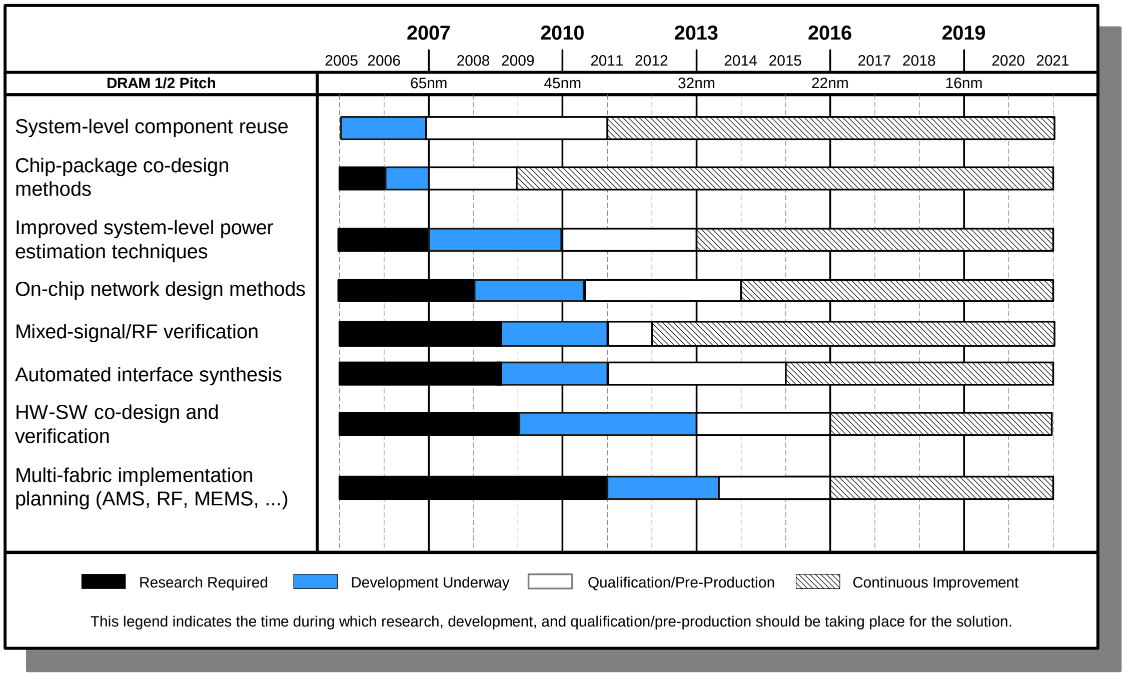
[3] Different platforms are expected to converge in the future, owing to advances in manufacturing technology and higher integration densities, wherefore the total number of platforms is expected to decrease.

[4] DEFINITION

(Full) Support for a particular platform means an integrated development environment that supports and automates architectural exploration, HW/SW partitioning, architectural/platform mapping, HW/SW co-verification, performance/area/power/costs trade-offs, HW and SW synthesis and HW/SW interface synthesis for that platform.

RATIONALE
A high degree of automation is a key to the success of system level design.
[5] Although there already exist some solutions today for some aspects of platform based modeling, a full integration has not yet been achieved.
[6] DEFINITION
The degree to which the estimated results match the measurements on the fabricated IC.
RATIONALE
For high-level synthesis techniques a high accuracy of estimations is essential in order to be able to deliver high quality synthesis results that meet user constraints, such as min. performance, max. area, et cetera.
Take into account the quality is much different for the different aspects, keep one row for now
[7] DEFINITION
The portion of a SOC, or a design, respectively, in terms of functionality implemented either in SW or HW that is reconfigurable.
RATIONALE
The growing system complexity will make it impossible to ship designs without errors in the future. Hence, it is essential to be able to fix errors after fabrication. In addition, reconfigurability increases reuse, since existing devices can be reprogrammed to fulfill new tasks.
[8] DEFINITION
Automation grade in analog design.
RATIONALE
Analog components are to be found in most electronic systems today and analog/mixed-signal design is an essential and important part of electronic design. Hence, as for digital design a high automation grade in analog design on as many levels as possible is required in the future, if the design productivity growth shall be maintained or increased.
[9] To which degree does the degree of automation in analog design match the degree of automation in digital design.
[10] DEFINITION
To which degree do analog methodology, description languages, and simulation environments match the maturity of their digital counterparts.
RATIONALE
As digital and analog design become nearly equally important on system level, analog design and modeling methodologies are required to reach a similar maturity as their digital counterparts in order to be able to keep up the productivity growth in system level design.
[11] To which degree do analog methodology, description languages, and simulation environments match the maturity of their digital counterparts?
[12] This number is based on a survey carried out in 2004.

The following Figure 20 defines solutions for system level design in their corresponding time frame:



MEMS—micro-electronmechanical systems

Figure 20 System Level Design Potential Solutions

Finally the following Table 14 explains how requirements from Table 13 correspond to the solutions of Figure 20 for system level design.

Table 14 Correspondence between Requirements and Solutions

Design block reuse	System-level component reuse	The larger and more complex the components that can be reused, the larger the overall design reuse expected to be.
	On-chip network design methods	Standardized communication structures and interfaces support reuse, since IPs with standardized interfaces can be easily integrated and exchanged, and also the communication structure itself is reused.
Available platforms	Multi-fabric implementation planning (AMS, RF, MEMS, …)	Enables to integrate different fabrics on the same die or in the same package (SIP), and, hence, should enable to reduce the absolute number of platforms, since what needed different platforms before could now be integrated on one.
Platforms supported	Automated interface synthesis	Automated interface synthesis is one building block to an integrated synthesis flow for whole platforms.
	Automated HW-SW co-design and verification	An obvious requirement for an integrated, platform-based system development approach
Accuracy of high level estimates	Improved system-level power estimation techniques	While area and performance estimation on higher levels has already been a topic for several years and made some progress, system-level power estimation is a rather innovative but important topic and has to catch up.
	Chip-package co-design methods	Allows taking effects of packaging, for instance impact on the timing, into account for estimations on higher levels.
SOC reconfigurability	On-chip network design methods	On-chip networks are flexible and reconfigurable communication structures.
Analog automation	Multi-fabric implementation planning (AMS, RF, MEMS, …)	Multi-fabric implementation planning for AMS and RF components seems to be one building block to analog automation.
Modelling methodology, description languages, and simulation environments	Mixed-signal/RF verification	As for digital design, verification is of major importance and verification is increasingly the most important and most time-consuming activity in the design flow.

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 RT-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 (GDS) II stream 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. The methodologies premised on the principle of separation of concerns in which a complex design flow is serialized into a sequence of manageable steps that are loosely coupled are becoming increasingly outdated. In those scenarios, decisions made in the early stages of design flow become binding constraints on later stages. Such serialization potentially yields less optimal designs than a methodology that simultaneously considers all design aspects. This situation is unavoidable, however, due to the practical difficulty of concurrent optimization of all design parameters, and is deemed acceptable as long as the constraints that are fed forward can be met. The methodology breaks down, however, when these constraints become unsatisfactory; the typical action in such cases is an iteration that revisits earlier design stages to change suspected 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. Ideally, the time-wasting iteration between logic and layout synthesis in today's design methodologies could be eliminated by fusing these stages to simultaneously optimize the logical structure as well the layout of a circuit.

Requirements to address challenges of logic, circuit and physical design are stated in Table 15. The key emphasis is on the design styles used and the effect of evolving technologies on the design parameters. This section also presents targeted potential solutions, which are also presented in Figure 21. The solutions are focused on the tools capability to address stated requirements. The explanatory comments for these requirements and potential solutions are provided below.

LOGICAL, CIRCUIT, AND PHYSICAL DESIGN REQUIREMENTS

1. *Asynchronous global signaling*—One of the main challenges of modern IC design is to distribute a centralized clock signal throughout the chip with an acceptable low skew. A similarly daunting task is to combine several pre-designed modules on the chip and to provide reliable communication between independent modules. Globally asynchronous locally synchronous (GALS) electronic system design is a methodology that addresses these problems. In GALS rather coarse-grained functional modules area designed using conventional design techniques. Each module is complemented by its own local clock generator and a self-timed wrapper that enables the modules to communicate using asynchronous handshake protocols. The number of handshake components determines the complexity of a system that operates in this way. The projected number of the independent components is not projected to increase dramatically due to the associated overhead, such as clock distribution, in coordinating them.

2. *Parameter uncertainty*—EDA tools attempt to eliminate uncertainty caused by data storage management (DSM) interconnect delay by creating tighter links between the logical and physical domains, thus reducing the number of times the application-specific integrated circuit (ASIC) vendor must pass the design back to the design team. While this works for interconnect, a great deal of electrical uncertainty in transistor remains unaccounted. As a result, static timing analysis tools needlessly overestimate operational delay in the circuits, introducing significant pessimist into operational frequency of a design. Table 15 lists projected effect of parametric uncertainty that is due to process variations. The increasing uncertainty emphasizes the need for accurate consideration of process variations in logic and circuit design. Dealing with variability is an increasingly important aspect of high-performance digital integrated circuit design, and indispensable for first-time-right hardware and cutting-edge performance.

3. *Simultaneous analysis objectives*—Most current very large scale integration (VLSI) tools could not handle objectives such as accurate RC extraction, transmission line effect and coupling effect simultaneously. Even more traditional models for route-ability, area, and power dissipation are often considered independently from each other. The advent DSM technologies that require many objects to be integrated into on flow whose affects are considered at the same time.

4. *Manufacturability for fault tolerance*—There is a direct correlation between a vendor's share of market and the quality of its products. With the advent of technologies it is therefore increasingly more important to have manufacturability for fault tolerance (MFFT). In the table MFFT requirements are measured in terms of the reliability factor. The reliability of a device may come from the design decisions, such as lowering design temperature. The metric used in the table is based on reliability factor that is normalized against state-of-the art yield parameter.

5. *Number of circuit families in a single design*—High-performance integrated circuits often mix circuit families to achieve better speed, at the possible expense of area, power consumption, design effort, etc. The ability to handle different circuit families on a single design would therefore remain an important requirement. However, the associated design complexity and fabrication difficulties arising with mixing logic families is likely to keep the number of different families on a single chip stabilize at a relatively small number in the next decade.

6. *Analog content synthesized*—The table summarizes projected amount of synthesized analog circuitry, as a percentage of a design. See the AMS section of the *Systems Driver chapter* for more details.

7. *Design on predictable platforms.*

8. *Adaptive/self-repairing circuits*—Nearly all circuits in use cease to function properly when they incur any sort of damage. The increasing variations in manufacturing process also dramatically reduce yield of fabricated chips. In many cases the trade-offs between system size and complexity will continue to be such that it is cheaper to create a compact, non-robust implementation of an electronic component and then replace it when it fails; still, there are cases where it is too expensive to access the system, as well as cases where it is too expensive to communicate with the system to remotely diagnose its failure state and repair it. For this reason ITRS predicts an increased demand on circuits that could consume non-traditionally large areas, but exhibit highly adaptive to internal failures.

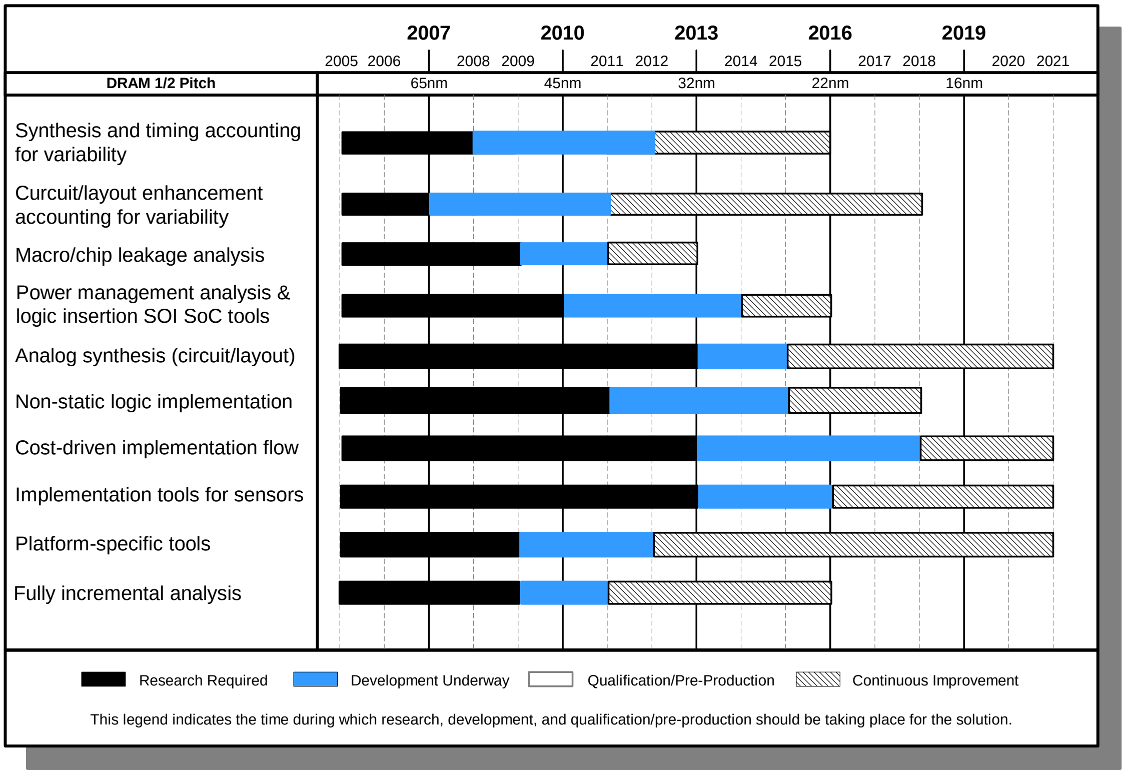
9. *Power leakage*—Power consumption is now the major technical problem facing the semiconductor industry. As feature sizes shrink below 0.1 micron, static power is posing new low-power design challenges. ITRS predicts a decrease in dynamic power per device over time. However, the doubling of on-chip devices every two years would force an increase of current leakage on a per-chip basis.

Table 15a Logic/Circuit/Physical Design Technology Requirements—Near-term Years

Year of Production	*2005*	*2006*	*2007*	*2008*	*2009*	*2010*	*2011*	*2012*	*2013*
DRAM ½ Pitch (nm) (contacted)	*80*	*70*	*65*	*57*	*50*	*45*	*40*	*36*	*32*
Asynchronous global signaling % of a design driven by handshake clocking	**5%**	**5%**	**7%**	**11%**	**15%**	**17%**	**19%**	**20%**	**22%**
Parameter uncertainty %-effect (on sign-off delay)	**5%**	**6%**	**6%**	**8%**	**10%**	**11%**	**11%**	**12%**	**14%**
Simultaneous analysis objectives # of objectives during optimization	**4**	**4**	**4**	**5**	**6**	**6**	**6**	**6**	**7**
MTTF contribution reliability factor	**1**	**1.1**	**1.2**	**1.3**	**1.4**	**1.6**	**1.7**	**1.8**	**1.9**
Circuit families # of circuit families in a single design	**2**	**2**	**3**	**3**	**4**	**4**	**4**	**4**	**4**
Analog content synthesized % of a design	**10%**	**13%**	**15%**	**16%**	**17%**	**18%**	**19%**	**20%**	**23%**
Leakage # times per device	**2**	**3**	**4**	**6**	**8**	**9.5**	**11**	**12**	**14**

Table 15b Logic/Circuit/Physical Design Technology Requirements—Long-term Years

Year of Production	*2014*	*2015*	*2016*	*2017*	*2018*	*2019*	*2020*
DRAM ½ Pitch (nm)(contacted)	*28*	*25*	*22*	*20*	*18*	*16*	*14*
Asynchronous global signaling % of a design driven by handshake clocking	**20%**	**25%**	**30%**	**30%**	**30%**	**35%**	**40%**
Parameter uncertainty %-effect (on sign-off delay)	**15%**	**18%**	**20%**	**20%**	**20%**	**22%**	**25%**
Simultaneous analysis objectives # of objectives during optimization	**8**	**8**	**8**	**8**	**8**	**8**	**8**
MTTF contribution reliability factor	**2**	**2.1**	**2.2**	**2.3**	**2.5**	**2.6**	**2.7**
Circuit families # of circuit families in a single design	**4**	**4**	**4**	**4**	**4**	**4**	**4**
Analog content synthesized % of a design	**25%**	**28%**	**30%**	**35%**	**40%**	**45%**	**50%**
Leakage # times per device	**16**	**24**	**32**	**32**	**32**	**32**	**32**



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

Figure 21 Logical/Circuit/Physical Design Potential Solutions

Design Verification

Design verification is the task of establishing that a given design accurately implements the intended behavior. Today, the verification of modern computing systems has grown to dominate the cost of electronic system design, often with limited success, as designs continue to be released with latent bugs. In fact, in many application domains, the verification portion has become the predominant component of a project development, in terms of time, cost and human resources dedicated to it. In current projects verification engineers outnumber designers, with this ratio reaching two or three to one for the most complex designs. Design conception and implementation are becoming mere preludes to the main activity of verification.

This situation is the result of two processes. First, the functional complexity of modern designs is increasing at a breathtaking pace. Design size is growing exponentially with Moore's Law. In the worst case, functional complexity, as measured by the number of distinct system configurations that must be verified, may be subjected to a doubly exponential blow-up.[6] Second, the historically greater emphasis on other aspects of the design process has produced enormous progress (automated tools for logic synthesis, place-and-route, and test pattern generation, etc.), leaving verification as the bottleneck. *Without major breakthroughs, verification will be a non-scalable, show-stopping barrier to further progress in the semiconductor industry.*

The overall trend from which these breakthroughs will emerge is the 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 electronic system design by repeatedly building models, simulating them on an ad hoc selection of vectors, and then patching any bugs that happen to be triggered. To this end, logic simulation techniques are predominant in the industry, because they can generate simulation vectors at a very high rate. However, the coverage of the tests generated is usually very low, with the result that even months of

[6] *There are many ways of reasoning to arrive to this conclusion. For example, a new design that requires doubling the number of transistors on a chip is likely to also double the number of latches on the chip, which likely means roughly squaring the number of reachable states of the design. This analysis assumes that the correct behavior can be verified by examining the set of reachable states of the system, or by similar computation. If verifying correct behavior requires reasoning over sequences of states, the computational challenge will be even worse.*

simulation provide little confidence in the correctness of the design and, when design errors are found, the error analysis phase is daunted by very long and complex bug traces. Additionally, these traditional techniques require a lot of effort from the engineering team to direct the verification activity towards specific design areas of critical quality, or with low-coverage, etc. Nonetheless, these techniques have such high inertia in existing development processes that the cost to transition to alternative methodologies is very high. Because of this, the recent availability of formal and semi-formal techniques has seen only limited adoption in the industry at large.

The more structured verification approaches organize the verification effort by generating a model of the system's intended behavior[7] (commonly called a "golden model") and comparing the outcomes of the design's simulation with this model. In addition, coverage metrics are collected to attempt to evaluate the level of confidence in the correctness of the design-under-verification. It is common to organize the verification activities hierarchically, by addressing first individual components within a chip, then the entire chip, and eventually the full system[8], so that bugs contained within a single unit can be addressed and solved earlier and more easily. Finally, there is a growing interest for formal and semi-formal verification techniques, to the point that teams are starting to experiment with the commercial tools available, and are exploring ways to complement their mainstream verification methodology with some of this technology. Formal methods span multiple aspects of verification: from a formalized verification process, to a formal notation for specification and verification and formal solvers and proof techniques.

In tackling the complexity of very complex systems, and in particular, systems-on-a-chip, a "hierarchical" verification methodology seems to be necessary, along with the development of specific methods and tools associated with each level of the hierarchy. In this context, the design and implementation pyramid can be summarized roughly as follows:

1) *Module level*—(structural, dataflow and behavioral module description and synthesis) where the verification tools must prove the correctness of systems in the range of 10K to 500K logic gates

2) Sub-System level—(for the most part, this is the result of structural composition of modules) producing systems with 100k to 10M logic gates

3) System/Full chip level—(obtained by composing sub-systems) producing designs of 1M to more than 100M logic gates.

In a hierarchical verification approach, different verification techniques would be used at each design level: At the module level, the goal is to produce high-quality modules through the simulation of handcrafted tests and the formal verification of properties and assertions. For sub-systems, verification would entail a mix of property verification for the interface and simulation with constrained random pattern generation. Property verification at the sub-system level could be achieved in the future, for instance, through the definition of aggressive automatic design-abstraction techniques, and of automatic techniques for the refinement of such abstractions, driven by counter-examples, which would allow proving sub-system properties while posing a minimum demand on the verification engineering resources. At the system level, simulation-based assertion checking paired with coverage metrics seems the most appropriate among the techniques available. Technological progress depends on developing rigorous and efficient methods to achieve high-quality verification results through the development of appropriate coverage metrics, verification tools with a high ratio of coverage over resources required, and techniques to ease to the burden of debugging a design, once a bug is found.

Table 16 below presents a set of key quantitative requirements in design verification, necessary to support the trends of design complexity in the next technology generations. The requirements evaluate both the

[7] *Golden models are commonly described using high-level languages, such as SystemC or ANSI C.*

[8] *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, providing an invaluable aid to verification. It also enables an early start to system integration and software development. An emulation system, however, still simulates vectors one-at-a-time, so it can only deliver a constant-factor improvement, and cannot provide a scalable, long-term solution to the verification problem.*

ability to guarantee the correctness of a design and the amount of effort spent in verification within a development project.

Table 16a Design Verification Requirements—Near-term

Year of Production	*2005*	*2006*	*2007*	*2008*	*2009*	*2010*	*2011*	*2012*	*2013*
DRAM ½ Pitch (nm) (contacted)	*80*	*70*	*65*	*57*	*50*	*45*	*40*	*36*	*32*
Productivity									
SOC logic size verified Mtx* per engineer-year (based on a ten-engineer team)	**4.7**	**6.1**	**8.0**	**10.5**	**13.8**	**18.1**	**23.8**	**31.2**	**41.1**
Methodology									
% bugs found by formal and semi-formal verification technology (versus simulation)	**0.0**	**2.7**	**5.3**	**8.0**	**10.7**	**13.3**	**16.0**	**18.7**	**21.3**
% verification effort spent on verification of integrated SW / HW / electrical effects	**0.0**	**0.0**	**0.1**	**0.1**	**0.1**	**0.1**	**0.2**	**0.2**	**0.2**
% design specification formalized for verifiability	**0.1**	**0.1**	**0.2**	**0.2**	**0.3**	**0.3**	**0.3**	**0.4**	**0.4**
Reuse									
% new verification development (versus IP and reuse)	**0.8**	**0.8**	**0.7**	**0.7**	**0.7**	**0.6**	**0.6**	**0.6**	**0.5**
% verification infrastructure acquired from third parties (verification IP)	**0.1**	**0.1**	**0.2**	**0.2**	**0.3**	**0.3**	**0.3**	**0.4**	**0.4**
Functional coverage									
% project verified through functional coverage	**40.0**	**43.7**	**47.3**	**51.0**	**54.7**	**58.3**	**62.0**	**65.7**	**69.3**
Coverage goal density (goals / Mtx* of logic)	**667**	**1022**	**1378**	**1733**	**2089**	**2444**	**2800**	**3156**	**3511**

** Mtx = millions of transistors*

Manufacturable solutions exist, and are being optimized	⬜
Manufacturable solutions are known	🟨
Interim solutions are known	◆ ▤
Manufacturable solutions are NOT known	🟥

Table 16b Design Verification Requirements—Long-term

Year of Production	*2014*	*2015*	*2016*	*2017*	*2018*	*2019*	*2020*
DRAM ½ Pitch (nm) (contacted)	*28*	*25*	*22*	*20*	*18*	*16*	*14*
Productivity							
SOC logic size verified Mtx* per engineer-year (based on a ten-engineer team)	54.1	71.3	94.0	124	163	216	286
Methodology							
% bugs found by formal and semi-formal verification technology (versus simulation)	24.0	26.7	29.3	32.0	34.7	37.3	40.0
% verification effort spent on verification of integrated SW / HW / electrical effects	0.2	0.3	0.3	0.3	0.3	0.4	0.4
% design specification formalized for verifiability	0.5	0.5	0.5	0.6	0.6	0.7	0.7
Reuse							
% new verification development (versus IP and reuse)	0.5	0.5	0.4	0.4	0.4	0.3	0.3
% verification infrastructure acquired from third parties (verification IP)	0.5	0.5	0.5	0.6	0.6	0.7	0.7
Functional coverage							
% project verified through functional coverage	73.0	76.7	80.3	84.0	87.7	91.3	95.0
Coverage goal density (goals / Mtx* of logic)	3867	4222	4578	4933	5289	5644	6000

** Mtx = millions of transistors*

Manufacturable solutions exist, and are being optimized	⬜
Manufacturable solutions are known	🟨
Interim solutions are known	▨ ◆
Manufacturable solutions are NOT known	🟥

The values in the table refer to a new development design project, in contrast with a proliferation within a family of designs. The productivity of verification must scale proportionally with the complexity growth of a design, in order to sustain similar time-to-markets. The classification of verification software between formal verification and simulation in time is going to morph into hybrid and semi-formal verification solutions, particularly in the long-term horizon. The estimation is, within those integrated hybrid solutions, which fraction will resemble a simulation-based approach versus a formal verification technique. To support this productivity growth, the methodology of verification must increasingly integrate more formal and semi-formal engines to complement simulation-based techniques. In addition, due to the increasing impact of software and mixed-signal components, and the importance of electrical effects due to the reduction in feature sizes, the verification of the interaction between heterogeneous components and the system as a whole will require an increasing fraction of the overall verification effort. Finally the shift towards a more structures approach to verification demands an effort towards the formalization of a design specification, which in turns leads to improved automation of the entire verification process.

The verification development is divided into three segments: 1) code newly developed specifically for the design being verified; 2) code acquired from third party (also known as “verification IP”); and 3) code reused from previous designs within the same company. The table presents the first two components under “Reuse,” the third can be computed by the difference from the other two rows. The last two columns 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 functionality that needs to be verified but requires more engineering effort because of its domain specific quality. The need to quantify the progress of verification in the near- and long-term future requires improved deployment of this technique. Although the expertise among engineering teams in developing and reuse functional coverage is growing, the table indicates that a pervasive deployment is needed in the future.

DESIGN VERIFICATION—CHALLENGES

Near Term

Many of the most important 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 of the quality of verification. In the longer term, issues focus on raising the level of abstraction and broadening the scope of formal verification. These longer-term issues are actually relevant now, but the near-term challenges are already crises. In general, all of the verification challenges apply to SOC. MPUs present a distinct verification challenge, both because of their leading-edge complexity, and because of the unique economics of an incredibly complex design that is produced in incomparably high volumes. As a result, different, domain-specific verification challenges and opportunities exist, both near- and long-term.

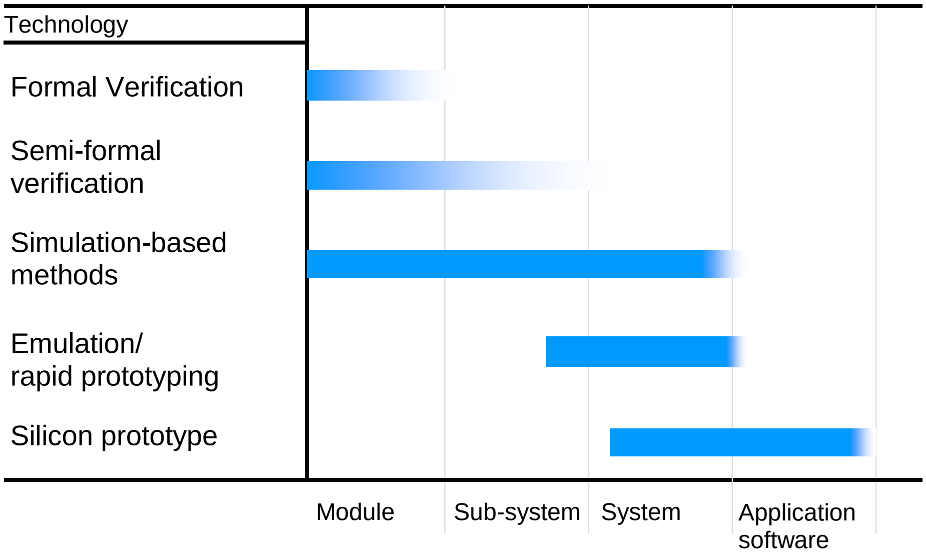


Figure 22 Current Verification Tool Landscape

Capacity—Figure 22 relates the current landscape of available verification tools and design granularity. The horizontal axis presents increasing design complexity from module components to sub-systems and multiple modules connected together to provide some high-level functionality including systems, complete systems-on-a-chip integrating multiple functionalities on a single die, and to the application software running on the system. The vertical axis indicates the verification techniques available.

Conventional simulation-based methods[9] span a large range; they are practically used for anything from module to full-system verification. However, the quality of the results varies: For small modules it is possible to run many simulation vectors in a short time and thus achieve a fairly good coverage of the design under

[9] *"Conventional simulation" refers to verification techniques based on simulating possible behaviors of the system one-at-a-time. "Formal verification" refers to any techniques, such as symbolic simulation, symbolic trajectory evaluation, model checking, and theorem proving that provide the effect of an exhaustive analysis of all possible behaviors of the system. An occasionally useful distinction can be drawn between theorem-proving approaches, which tend to provide the greatest expressive and analytical power at the expense of requiring substantial human expertise, versus the other approaches, which tend to trade-off theoretical power for greater automation. "Semi-formal" refers to a broad range of techniques that attempt to handle larger designs by sacrificing complete, formal coverage, usually by blending techniques from formal verification and conventional simulation.*

verification. At the system level the performance of simulation is very low, since the complexity of the algorithms involved is linear on the design size and the number of simulation vectors. Thus, for very complex systems, only a few simulation vectors can be run within a reasonable amount of time, and the fraction of design coverage they provide is even much smaller considering the complexity of the system that is being simulated. Formal verification can achieve very high coverage, granted that a broad set of properties are in place targeting the various aspects of the system. The downside of these techniques is their limited scalability that usually cannot go above the module level. Semi-formal verification attempts to blend formal and simulation-based techniques. As the figure shows, they present generally better capacity, sometimes at the cost of lower design coverage. For very complex systems and when the project involves software development, software tools are insufficient to provide any confidence in the correctness in the system. Solutions used at this level include field-programmable gate array (FPGA)-based rapid prototyping and/or a silicon production of the hardware. The challenge is to create new solutions that together provide high coverage at all levels of complexity. The two orthogonal methodologies of formal verification and simulation in use today have both important 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 achiever higher coverage. However, the improvement is only a constant factor and does not scale at the same rate as design complexity.

Robustness—A crucial aspect of current verification solutions that is not shown in Figure 22 is their reliability. On one hand, simulation-based methods are fairly predictable, because their execution time per simulation vector scales linearly with design complexity. Thus, their performance will linearly decrease as the industry moves 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, 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 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 robustness 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 design.

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). Variants of line coverage are concerned with covering all components of expressions in the code, or all cases in switch statements, etc…The limitation of line coverage 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. Thus, the quality of the result is dependent also on the quality of the coverage definition.

Code and structural coverage metrics quantify aspects of the designs that do not necessarily imply its correctness. Functional coverage potentially could achieve this goal of measuring the fraction of a design's functionality that has been verified. However, the challenge is in the definition of a good functional coverage.

In the current landscape, there is a lack and a need 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 a meaning beyond the specific functional coverage used in that particular design. Moreover, there is no general methodology in designing the 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—The verification of complex systems, such as SOC, entails the verification of the hardware components, of the hardware/software interface and of the application software running on the system. The software components of an SOC can be divided into application software running over an operating system or, in general, a layer that is hardware-independent and a lower-level hardware-dependent software, such as drivers, etc. In these systems the software layer could provide much of the functionality, so a major challenge in SOC verification is how to verify software and the hardware/software interface. Presently, software development is not as rigorous as hardware development in terms of design reviews, analysis tools, and testing. Software is intrinsically harder to verify—it has more complex, dynamic data and an enormous state space. Techniques used today include “on-chip-verification,” that is, running the software on a production version of the hardware components to verify the software. While this technique is required by the intrinsic complexity of software, and allows very fast simulation, its downside is that software verification can only happen very late in the design cycle. Classical formal techniques for software verification are still too labor-intensive to be widely applicable for SOC, to attack systems with such large state spaces, and require very aggressive abstraction techniques. The verification of the hardware/ software interface is a challenge on its own, since it requires verifying the two domains together, a task of enormous complexity. To make this task manageable, there is a need for techniques that provide a proper abstraction of the interface activity, verification techniques to check the correctness of the abstracted driver layers, and assertion checking 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 to develop 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 there is a need 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 1) coverage-driven verification, both for simulation-based verification and semi-formal verification, (more in use today than in the past); 2) specification documents in formal notation, that make easily available the set of formal properties to be verified in the implementation; and 3) coverage model templates. Many challenges are still to be solved to obtain a sufficiently robust and complete methodology. For instance, there is a need for ways to obtain consistent abstraction techniques of design components, interfaces, etc. that do not drop key aspects of the design in the abstraction. These aspects depend on the context where the abstraction is used. Formal specifications are starting to be developed; however, a major challenge is the completeness of such specifications. This challenge 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, 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.

Specialized design-for-verifiability—To enable more effective verification, a few specialized activities in design for verifiability are already foreseeable. For instance, facilities for software and/or hardware debug can be put in silicon. In this direction there is preliminary work on self-checking processors, in which a small watchdog processor 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 particular at the software level forces tasks not to 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 challenges in this area will be 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.

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.

Long Term

In the long term, the issues will be centered on the need of improved verification productivity, verification of extremely complex designs, and the need to address verification of heterogeneous systems. Below some of the most important challenges are presented.

Design for verifiability—As the solutions to the near-term challenges produce understandings of what is easy or hard to verify and how design errors occur, the longer-term challenge arises of how to codify that understanding into producing easy-to-verify designs. Without design-for-verifiability, it is unlikely that verification will be tractable for the designs envisioned beyond 2007. Major changes in methodology may be required, and some performance degradation is likely. A useful analogy is to sequential testability, where the computational intractability of sequential ATPG has resulted in near-universal adoption of scan-based testing.

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 more difficult if decisions about the higher-level of abstraction are made without regard for verification (for example, 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).

Specification for verifiability—A continuing challenge for design verification is how to specify the desired behavior of a design. 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 do not state conflicting requirements. In addition, specific training is needed for designers to use these specification notations and to be able to develop formal specifications consistently.

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.[10] Similarly, at the highest-levels of design, system complexity dictates that future verification tasks will likely require specifying and verifying analog and probable behaviors (such as quality of service guarantees in a network processor). Thus, there will be the challenges of hybrid systems and probabilistic verification.

Heterogeneous systems—The development of new technologies that are placed side-by-side to 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 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.

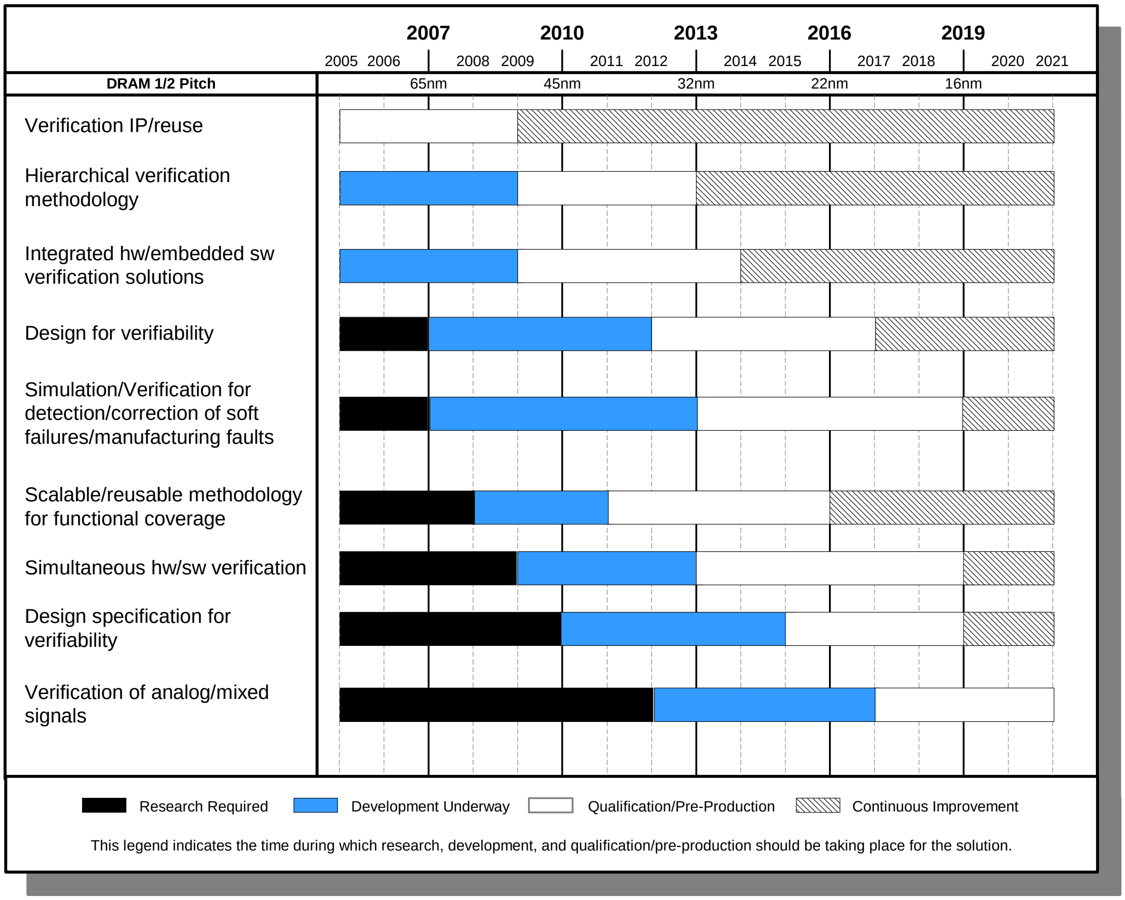
Soft failures—In the long term there is a need to develop verification techniques that can provide information on a system's reliability in presence of soft errors.

Verification of redundancy—The quantifiable evaluation of the reliability of a redundant system, and the estimation of the level of redundancy needed to achieve a certain level of reliability is particularly important for safety-critical applications. In the future, the increased complexity of electrical effects in a system performance and correctness will make these evaluations even more important, even for non safety-critical designs.

DESIGN VERIFICATION—SOLUTIONS

Solutions that are available or are being developed to address the challenges are presented in Figure 23. This figure summarizes some of the key directions to address the verification crisis along with their expected availability to development teams.

[10] *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.*



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

Figure 23 Design Verification Potential Solutions

Verification IPs are already available today, and they are part of a growing methodology trends to boost the productivity of the process of verification. In the future this approach will be much more widespread, in parallel with the aggressive deployment of third party IPs in SOC designs. As discussed in the previous section, 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 figure are still at a conceptual stage today. Formal methodologies to develop easy-to-verify designs and to correct soft errors and defects have seen an initial exploration in the MPU domain, as mentioned in the previous section. Techniques for structural functional coverage and specification are even further from being mainstream, however they are crucial in solving the design-verification productivity gap.

DESIGN FOR TEST

Nanometer process technology, increasing clock rate, mixed analog-digital-RF circuits, and highly integrated SOC 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 problems can be solved only if proper testability and DFT are considered and incorporated early in the design phase. Furthermore, the methodology precepts above note a natural evolution of analyses into verifications into tests, and the need for deeper unification of design with test. Effort and results (flows, vectors, fault models, sensitivities) from analyses in logical-circuit-physical implementation, and from design verification, must be reused during design and test. Across industry segments as varied as memory, digital signal processing (DSP), PE, SOC, analog/mixed-signal/RF, and MPU, high-level test challenges demand significant expansion of on-chip DFT, BIST, and testability features, and better integration to a pre-planned manufacturing test equipment set during the chip planning phase of development.

The DFT technology requirements and potential solutions vary widely based on the specific subsystems and yet they all have to converge at the system level so that SOC and SIP are testable with lower manufacturing cost and little performance degradations. Using system drivers as the common framework, DFT requirements and solutions are described in this section.

Table 17 summarizes the DFT technology requirements and Figure 24 summarizes the DFT potential solution for four major classes of system drivers based on specific design technologies: analog/mixed-signal/RF system driver, MPU/PE/DSP system driver, memory system driver, and general SOC/SIP system driver.

1. *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 or transistor and architectural performance. Currently, the industry is facing serious problems regarding RF measurements beyond 5 GHz. It is very expensive to test parts in excess of 5 GHz at wafer level or package tests—special testers or added hardware modules are needed to perform such tests, with added cost and test time. Designs at 10 GHz and beyond are being generated but there are no production testers capable of testing at those frequencies. This problem is expected to get worst in the future. It is imperative that new techniques be developed that allow high-frequency mixed-signal circuitry (10–30 GHz) to be fully tested and characterized for complex test specifications using low-cost testers. 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. By contrast, ATE and component test legacies include common clock-based testing and I/O measurements in the MHz range. Hence, 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 under 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. This correlation is even more critical before the complexity and performance of analog/mixed-signal/RF subsystems increase to the point where they will no longer be testable using any external ATE or instrumentation.

Table 17a Design for Test Technology Requirements—Near-term Years

Year of Production	*2005*	*2006*	*2007*	*2008*	*2009*	*2010*	*2011*	*2012*	*2013*
DRAM ½ Pitch (nm) (contacted)	*80*	*70*	*65*	*57*	*50*	*45*	*40*	*36*	*32*
System Driver: Analog/Mixed-signal/RF									
1. All-digital DFT for analog/mixed-signal/RF circuits and systems % digital circuits in DFT implementations	30	35	40	45	50	55	60	60	60
2. Correlation of DFT results with existing specification-based test methods. % results correlated	30	35	40	45	50	55	60	60	60
3. Availability of fault/defect models for DFT-oriented test methods. % AMS/RF blocks with accepted fault models	20	20	25	30	35	40	45	50	55
System Drivers: MPU/PE/DSP/									
1. DFT coverage of digital blocks or subsystems. % blocks with DFT	60	60	70	70	70	75	75	75	80
2. DFT for delay test of critical paths. % paths covered	50	50	55	55	60	60	60	60	70
3. DFT for fault tolerance in logic blocks. % blocks with fault tolerance	40	40	40	40	45	45	50	50	55
System Drivers: Memories									
1. DFT for yield improvement.	85	85	85	90	90	90	90	95	95
General SOC/SIP requirements									
1. DFT-support for logic and other circuit repair (except memory). % blocks with repair	50	50	50	60	60	60	70	70	70
2. DFT re-use for performance calibration, and measurement purposes. % DFT circuits re-used	30	30	35	35	40	40	40	45	45
3. DFT impact on system performance (noise, power, sensitivity, bandwidth, etc.). % performance impact (aggregate figure of merit)	15	15	15	15	10	10	10	10	10
4. DFT efficacy in test volume reduction. reduction magnitude	2×	2×	5×	5×	5×	10×	10×	10×	20×
5. DFT / ATE interface standard, including DFT control via standard test access protocols. % of test interface standardized	40	40	45	45	50	50	60	60	70

Manufacturable solutions exist, and are being optimized	□
Manufacturable solutions are known	🟨
Interim solutions are known	◆
Manufacturable solutions are NOT known	🟥

Table 17b Design for Test Technology Requirements—Long-term Years

Year of Production	*2014*	*2015*	*2016*	*2017*	*2018*	*2019*	*2020*
DRAM ½ Pitch (nm)(contacted)	*28*	*25*	*22*	*20*	*18*	*16*	*14*
System Driver: Analog/Mixed-signal/RF							
1. All-digital DFT for analog/mixed-signal/RF circuits and systems. % digital circuits in DFT implementations	60	80	85	90	90	100	100
2. Correlation of DFT results with existing specification-based test methods. % results correlated	60	80	85	90	90	100	100
3. Availability of fault/defect models for DFT-oriented test methods. % % AMS/RF blocks with accepted fault models	60	65	70	75	80	85	90
System Drivers: MPU/PE/DSP/							
1. DFT coverage of digital blocks or subsystems. % blocks with DFT	80	85	85	90	90	95	95
2. DFT for delay test of critical paths. % paths covered	70	70	80	80	90	90	100
3. DFT for fault tolerance in logic blocks. % blocks with fault tolerance	55	60	65	70	80	90	100
System Drivers: Memory							
1. DFT for yield improvement.	95	95	98	98	98	100	100
General SOC/SIP requirements							
1. DFT-support for logic and other circuit repair (except memory). % blocks with repair	80	80	80	90	90	100	100
2. DFT re-use for performance calibration, and measurement purposes. % DFT circuits re-used	50	50	60	60	70	70	70
3. DFT impact on system performance (noise, power, sensitivity, bandwidth, etc.). % performance impact (aggregate figure of merit)	10	10	5	5	5	5	5
4. DFT efficacy in test volume reduction. reduction magnitude	20×	20×	20×	50×	50×	50×	50×
5. DFT / ATE interface standard, including DFT control via standard test access protocols. % of test interface standardized	70	75	90	80	90	100	100

Manufacturable solutions exist, and are being optimized	□
Manufacturable solutions are known	🟨
Interim solutions are known	◆
Manufacturable solutions are NOT known	🟥

Analog/mixed-signal/RF reliability issues 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 subsystem. 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, 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 the characteristics mentioned above: 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.

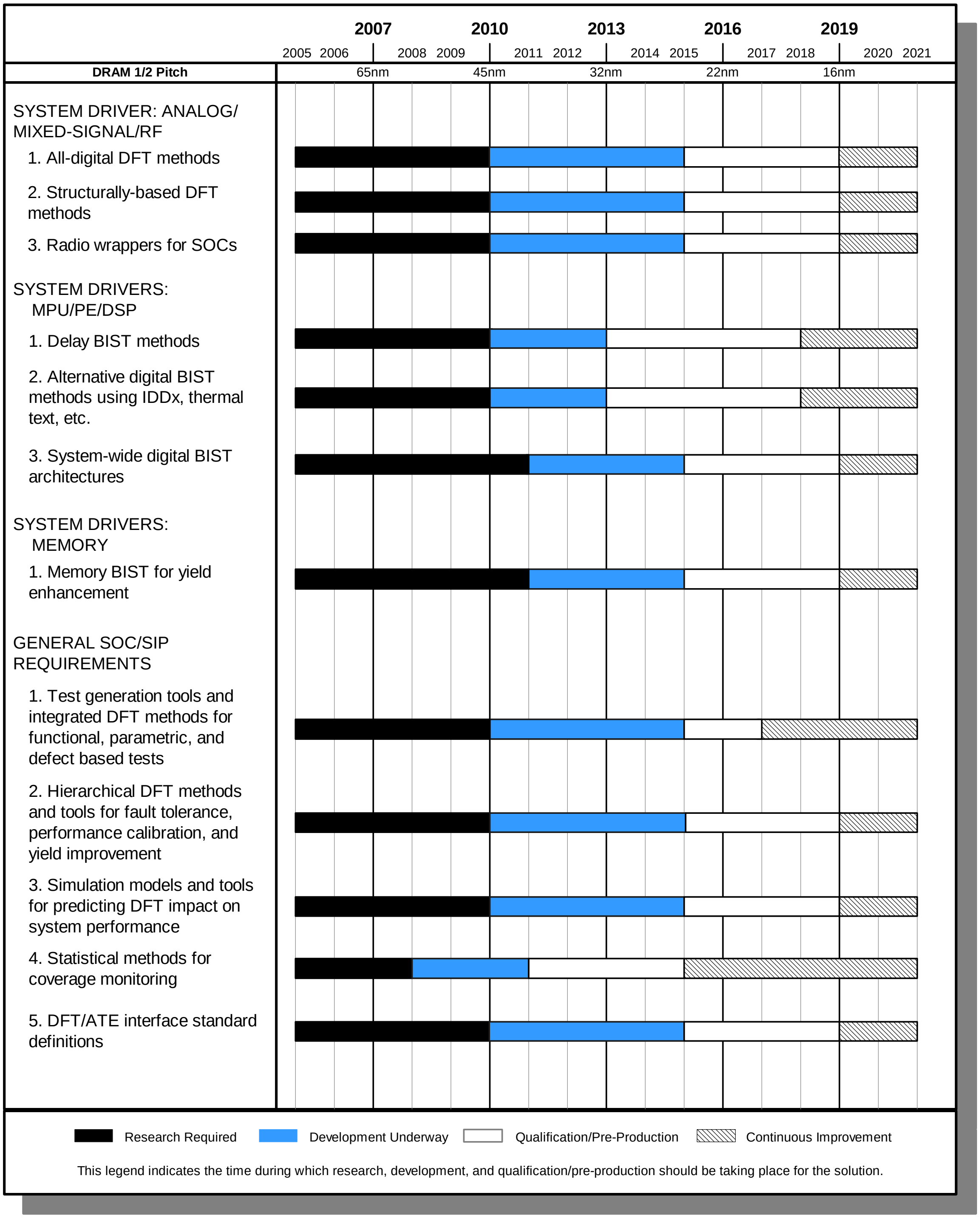


Figure 24 Design for Test Potential Solutions

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 trade-offs 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.

2. *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 perspectives, the 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, the major requirements include better 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, 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.

   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 likely create problems in logistics and in test execution planning, which in terms of test time and test data volume, would reduce the advantages of DFT/BIST. Given the diversity of digital DFT/BIST methods, a system test designer should be able to create a unified test architecture for a particular digital system. Unity in diversity is a critical feature in any system-wide digital test architecture solution.

3. *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 also led to additional test issues. These additional issues, e.g. new failure mechanisms, timing uncertainties in read/write access, etc., 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 perspectives, the essential requirement is to improve yield. Redundant 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.

   Mature solutions exist for memory systems in this classification with respect to physical quality enhancement. As mentioned above, redundant design methods and better layout styles, coupled with better understanding of memory fault models, have gone a long way toward improving yield. The remaining challenges in timing performance demand DFT / BIST methodologies that optimize read/write access time and improve reliability, especially for large and / or programmable memory blocks.

4. *Highly integrated general SOC and SIP designs*—Integration of pre-existing design blocks into larger integrated devices produces non-linear 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. 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; manufacturing test costs are exceeding silicon costs. Even with DFT, these costs may be non-linear. 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. The test generation hardware and embedded algorithms reduce the need for test access while providing an infrastructure for system-wide built-in self-test (BIST), which would be the ultimate system DFT requirements. 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 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 may also entail orders of magnitude longer test time than testing of the non-embedded versions. The test methods, ATEs, and manufacturing integrations for SRAM/DRAM, logic, flash, and analog/mixed-signal/RF silicon come from radically different legacies with unique optimizations that are typically broken in testing on an integrated logic device. Extremely long test times for standalone analog or RAM devices are offset by different ATE or test equipment costs and the high degree of parallelism (SRAM/DRAM) that is unavailable to SOC or integrated devices. Again, the embedded nature of the integration may preclude or hamper access to block I/Os that would be available on standalone devices. Not only expanded DFT techniques and protocols, but significant advances and use of BIST and/or embedded software-based self-testing for larger portions of test, are required. DFT and BIST for embedded mixed-signal/analog/RF blocks—long a research area—will become an industrial reality, driven by the dominant use of these types of circuits in integrated devices and SOCs.

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, parametric measurement).

The greatest advantage of DFT is test volume and test cost reduction. 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 the table covers the entire system test volume and test time, not the test volume and test time of a specific block within the system. For example, a 2× reduction requirement for 2006 seems to be quite conservative for the digital subsystems but considering the entire system (where analog/RF tests consume most of the test time), the 2× reduction is a reasonable requirement. 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, 1149.4, 1450, 1500)[11, 12, 13] are available, comprehensive DFT/ATE interface protocols spanning various system layers

[11] *IEEE Std 1149.1, IEEE Standard Test Access Port and Boundary-Scan Architecture.*

(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 developed in the past five years 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, 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.

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:

1. *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.
2. *Logic and circuit challenges*—digital and mixed-signal adaptive circuits will be increasingly necessary. Statistical design, including power and timing convergence will be fundamental. But it will be hard 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.
3. *Layout and physical design challenges*—First, the complexity of design rules checking is increasing (include graph). Rules have evolved to a two-tier system (required versus suggested rules) and may need

[12] *IEEE P1450.6, Draft Standard for Standard Test Interface Language (STIL) for Digital Test Vector Data--Core Test Language (CTL).*

[13] *IEEE Std* ***1500-2005, IEEE Standard Testability Method for Embedded Core-based Integrated Circuits.***

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 criteria for tape-out). 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, 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 general data stream (RTL2GDS) 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 global placement level to determine possible locations for dummy fill insertion, thus allowing to pre-place CMP fills and account for their capacitance while at the same time propagating the information about critical nets down to layout finishing, mask data preparation (MDP), and OPC levels.

4. *Yield prediction and optimization as design challenge*—The ground rules as specified are not longer "hard numbers" since such a scenario 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 has to be used. These "recommended" rules have been 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. DFM measures act differently for each design because they impact power, area, and speed as the design is tuned for yield. During the design process the mutual interaction of yield, area, power and speed must be analyzed and a commercially useful trade-off must be found. DFM measures that effect the functional and parametric yield must be integrated as a new optimization feature into the design flow and tools rather than as post processing that allows only limited results and is typically time consuming. That means that yield prediction has to be integrated into tools for design planning synthesis and place and route to consider all design targets to optimize simultaneously for yield, performance, power, signal integrity, and area (with yield as the 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 the correct evaluation of the yield cost of different design implementations in the place and route (P&R) tools it is therefore necessary to pre-characterize accurate yield models of the logic library that are based on the actual target process data. Library yield models need to be frequently updated to track process evolution from pre-ramp to a mature stage.

Table 18 quantifies the key challenges above into a set of DFM requirements. Figure 25 defines a set of DFM solutions that will be needed to address these requirements over time.

Table 18a Design-for-Manufacturability—Near-term Years

Year of Production	*2005*	*2006*	*2007*	*2008*	*2009*	*2010*	*2011*	*2012*	*2013*	*Driver*
DRAM ½ Pitch (nm) (contacted)	*80*	*70*	*65*	*57*	*50*	*45*	*40*	*36*	*32*	
Mask cost ($m) from publicly available data	**1.5**	**2.2**	**3.0**	**4.5**	**6.0**	**9.0**	**12.0**	**18.0**	**24.0**	*SOC*
% V_{dd} Variability % variability seen at on-chip circuits	**10%**	**10%**	**10%**	**10%**	**10%**	**10%**	**10%**	**10%**	**10%**	*SOC*
% V_{th} variability Doping Variability impact on VTH	**24%**	**29%**	**31%**	**35%**	**40%**	**40%**	**40%**	**58%**	**58%**	*SOC*
% V_{th} variability Includes all sources	**26%**	**29%**	**33%**	**37%**	**42%**	**42%**	**42%**	**58%**	**58%**	*SOC*
% CD variability CD for now, might add doping later	**10%**	**10%**	**10%**	**10%**	**10%**	**10%**	**10%**	**10%**	**10%**	*SOC*
% circuit performance variability circuit comprising gates and wires	**41%**	**42%**	**45%**	**46%**	**49%**	**50%**	**53%**	**54%**	**57%**	*SOC*
% circuit power variability circuit comprising gates and wires	**55%**	**55%**	**56%**	**57%**	**57%**	**58%**	**58%**	**59%**	**59%**	*SOC*

Table 18b Design-for-Manufacturability—Long-term Years

Year of Production	*2014*	*2015*	*2016*	*2017*	*2018*	*2019*	*2020*	*Driver*
DRAM ½ Pitch (nm)(contacted)	*28*	*25*	*22*	*20*	*18*	*16*	*14*	
Mask cost ($m) from publicly available data	**36.0**	**48.0**	**72.0**	**96.0**	**144.0**	**192.0**	**288.0**	*SOC*
% V_{dd} Variability % variability seen at on-chip circuits	**10%**	**10%**	**10%**	**10%**	**10%**	**10%**	**10%**	*SOC*
% V_{th} variability Doping Variability impact on VTH	**81%**	**81%**	**81%**	**81%**	**112%**	**112%**	**112%**	*SOC*
% V_{th} variability Includes all sources	**81%**	**81%**	**81%**	**81%**	**112%**	**112%**	**112%**	*SOC*
% CD variability CD for now, might add doping later	**10%**	**10%**	**10%**	**10%**	**10%**	**10%**	**10%**	*SOC*
% circuit performance variability circuit comprising gates and wires	**58%**	**61%**	**62%**	**65%**	**66%**	**69%**	**69%**	*SOC*
% circuit power variability circuit comprising gates and wires	**59%**	**60%**	**60%**	**61%**	**61%**	**62%**	**62%**	*SOC*

Manufacturable solutions exist, and are being optimized	□
Manufacturable solutions are known	🟨
Interim solutions are known	◆
Manufacturable solutions are NOT known	🟥

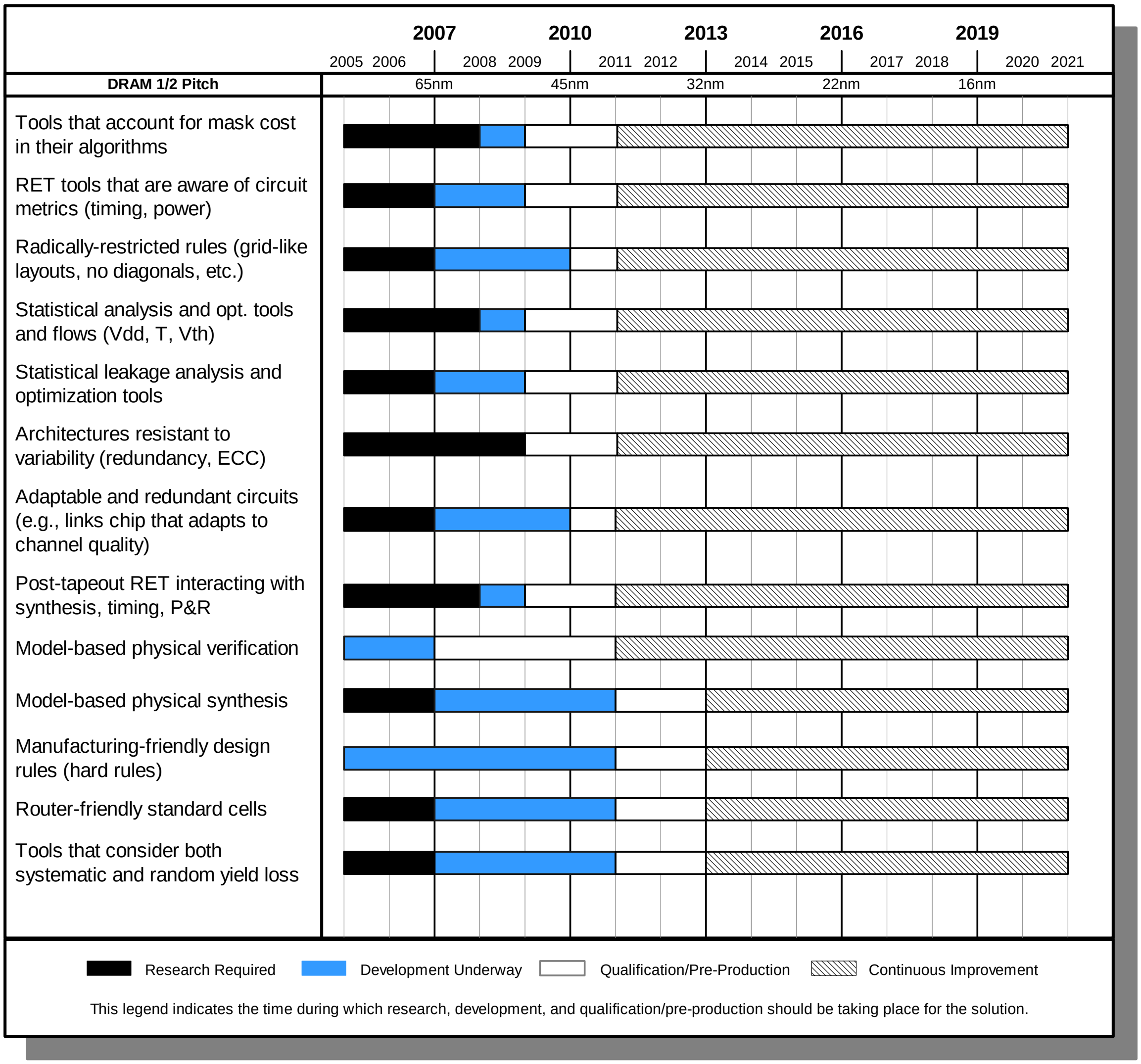
DFM requirements fall into the following categories:

- *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 abstractions levels; those that are directly associated with devices and wires; and at higher levels of abstraction, those 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 (in terms of both the impact of doping concentration variability and the overall trends), and percentage CD (critical dimension) 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).

Economic parameters such as mask cost will be difficult to control directly and thus increases are expected. Instead, workarounds will be found, such as multi-project wafers, configurable logic, and structured ASICs. 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 next decade or earlier. 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.

To address DFM technology requirements, the DFM solutions described in Figure 25 will be needed:

- *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 trade-offs.
- *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, redundance, and error correction coding (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 (simple rules that ensure manufacturability at little area or performance cost, such as grid-like layouts with no diagonals), and router-friendly standard cells and cores.



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

Figure 25 Design for Manufacturability Potential Solutions

Early solutions that directly handle variability (such as timing analysis) are emerging. In less than five years (between 2008 and 2010) statistical methods will be completely embedded in the design flow. In the meantime, they will be selectively applied as they mature, or will be part of premium design technologies used in high value or high volume designs.

DFM techniques that directly account for lithography have become very popular but will take even longer to become production-level across the board, 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.

Analog, Mixed-signal and RF Specific DT Trends and Challenges

Analog/mixed-signal and RF circuits are different compared to digital circuits. They do not contain quantized information that is represented in a defined range of voltage levels for the two states of a bit (“high” (V_{dd} – noise tolerance) and “low” (V_{ss} + noise tolerance)) and during a defined discrete time interval (such as clock signal high). Rather the signals processed in analog and RF circuits are continuous in time and amplitude up to a much higher degree of precision (or to a smaller tolerance both in time and amplitude). Therefore non-idealities like linearity, noise, parasitic elements, and electrical non-uniformity of the devices used in a circuit directly cause distortion and noise in the analog or RF signals processed in the circuit. 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. This simple signal-recovery principle cannot be used in analog and RF signal processing due to the much higher dynamic range of the involved signals. Speed issues, or simply the fact that a signal-recovery circuit produces more noise and distortion than it prevents the signal from being susceptible to, make these issues much more challenging and less straightforward in the analog domain.

Analog and RF designs have driven the needs for high precision compact modeling as well as for characterization and extraction of all kinds of device non-idealities. However, the above-mentioned problems make extraction of simple rules for higher levels of abstraction in analog and RF circuit design (or even layout) very difficult. Tools used for digital circuits have been in a wrong style but also too inaccurate to be used in analog design. Since the number of analog or even RF transistors per chip increases at a much slower rate compared to digital transistors per chip, historically all these boundary conditions have kept analog and RF designers away from system-level design tools. Circuit and electronic system design has remained mainly at the level covered by the section on Logical, Circuit and Physical Design in this chapter. Today similar problems to those commonly known for analog and RF designs start to arise in digital designs as well (IR-drop, cross-talk, etc.). SOC digital designs share chip area with analog and RF circuits. The ever-shortened time to market and the need for higher productivity have changed historic paradigms in recent years, and new analog and RF specific challenges for the EDA industry are arising today.

As noted in the *System Drivers chapter*, there are many challenges to scaling and migration of AMS designs. These challenges include decreasing supply voltages, increasing relative parametric variations, increasing numbers of analog transistors per chip, increasing signal, clock and intrinsic device speeds, increasing leakage and crosstalk in SOC integration, and a shortage of design skills and automation. Particular challenges include 1) deep submicron effects and higher signal and clock frequencies: even "digital goes to analog," increasing role of parasitics, transmission line effects, and signal integrity (SI); 2) analog synthesis tools to reduce effort spent on analog circuit design by a limited number of analog designers; 3) yield enhancement that requires "design for manufacturing," 4) close integration of signal processing systems (mostly digital) and analog RF front ends in almost all mobile communication devices; 5) common consideration of chip, bond wires and package with respect to electrical parasitics modeling, EMI/SI as well as thermal effects and 6) tight integration of electrical and non-electrical components (such as MEMS). These challenges may be elaborated in the context of traditional DT areas, as follows:

Accurate calculation of distributed analog components (inductors and coupled inductors, interconnects and transmission lines, electronically tunable capacitors) is necessary for higher frequencies (> 1 GHz). These calculations have to take into account skin effect and dispersion (that is, the frequency-dependent phase velocity that leads to waveform distortion) and to introduce these effects into macromodels. The impact of such calculations and some interesting results are reported in the literature (2005) in the context of the design of a 43 GHz voltage-controlled oscillator with today's technology (130 nm CMOS standard process). An inductor has been realized as a single turn of the top copper metal layer of length 2×100 µm, width of the wire 5 µm, and distance between the centers of the two wires of 15 µm. This very simple, almost rectangular layout resulted in an effective inductance of 83 pH and a Q of 35. 2.5D electromagnetic simulation has been used for accurate modeling the magnetic field and the skin effect. Capacitances have been realized with an area of about 100×100 µm each.

Antenna effects are important for frequencies > 10 GHz (the wavelength λ in vacuum is 3 mm at f = 100 GHz!). It is clear that antennas-on-chip have to be designed with sophisticated numerical methods, for example, finite difference time domain (FDTD), but the same effects have to be captured by simpler methods (perhaps with reduces accuracy) if they should be considered as parasitics in the context of interconnection networks. Similar high-frequency problems exist and have been solved if advanced analog circuits are used for digital clock generation (> 1 GHz) with standing-wave and traveling-wave oscillators.

Almost all of the above mentioned effects have influence on the construction of macromodels. Electrical capacitive and magnetic coupling in interconnection networks, thermal properties of the chip embedded in the package, far-field and near-field effects of antennas as well as the behavior of MEMS can be analyzed with 2D/3D field calculation methods and lead to systems of 104 … 108 equations. Without serious loss of accuracy, the number of equations can be reduced to 102 … 104 by various mathematical methods, known as the “order reduction methods.” The large scientific progress in this area should result in powerful CAD tools in the next few years. This new kind of automatically generated macromodel may be used in analog circuit design as well as in the investigation of timing behavior and signal integrity of digital circuits.

System Level Design for Analog, Mixed-signal and RF

In System-Level Design, the critical AMS challenges are non-scalability of analog circuits and analog behavioral modeling and synthesis. Automated analog circuit synthesis and optimization is needed, along with language-level modeling methodologies that permit analysis of overall system function and interfaces with other integrated technologies (digital, software). Issues include coping with vastly varying time scales (analog frequencies up to 100s of GHz versus digital frequencies up to 1s of GHz) in simulation; creating heterogeneous test benches and ensuring coverage; achieving systematic top-down constraint propagation; and mixing of functional and structural representations.

Logical, Physical and Circuit Design for Analog, Mixed-signal and RF

In Logical, Physical and Circuit Design, the key challenge is analog synthesis. Scalable SOC design requires elimination of the analog design bottleneck. A technology need shared with system-level design is for reusable, retargetable analog IP generators. Today’s specialized automatic circuit syntheses for particular classes of circuits (PLL, op-amp, power amplifier, etc.) must be augmented by more general techniques. Automatic layout syntheses must be able to handle the needs for high-performance analog designs (for example, cross-coupled layout for mismatch sensitive transistors). Analog post-layout simulation must handle increased opportunities for distortion and nonlinearity due to impact ionization, thermal nonlinearity, body contacts acting as low-pass filters. Syntheses must also handle future regimes of increased manufacturing variability, such as by hybrid analog-digital compensation for device mismatch. In the near term, new synthesis tools for optical interface circuits and high-Q CMOS on-chip inductors and tunable resonators are needed. Circuit types of interest in the long term include extremely low-power sensing and sensor interface circuits, as well as micro-optical (beam-steering) devices.

Design Verification for Analog, Mixed-signal and RF

In Design Verification, AMS circuits require checking “to specification” rather than with respect to structure. While ever-faster simulation has been the historical solution, new verification solutions must include statistical techniques, better compact models that speed up simulation even increasing accuracy, and new acceptance criteria. AMS designs also force the long-term issue of hybrid-systems verification—a field still in its infancy—into the near term. Thus, an immediate challenge is to provide any support possible to improve the current ad hoc approaches and find a path toward more powerful techniques. As MEMS, electro-optic, and electro-biological devices become more than simple transducers, a further challenge is to model, analyze and verify integrated systems having such heterogeneous parts. The same is true for pure electronic circuits with new device types (such as carbon nanotube transistors, single electron transistors, and resonant tunneling diodes) that use sophisticated physical effects and lead to non-classical device models.

Table 19 summarizes near-term AMS design technology breakthroughs expected through 2009. The reader is also referred to the excellent discussion of AMS DT requirements in the *March 2005 MEDEA+ Design Automation Roadmap.*

Table 19 Near-term Breakthroughs in Design Technology for AMS

Field of Breakthrough	*2005 State-of-the-Art*	*2006/07*	*2008/09*
Specification, validation, verification	Established AMS Hardware Description Languages	Multi-language support, AMS extension of HW/SW description languages for full system simulation	Complete specification-driven design flow; some specialized formal verification methods
Architectural design	Algorithm-oriented design (e.g., with Matlab/Simulink)	Language-based performance evaluation; closer coupling of architectural, block, and circuit level	Synthesizeable AMS description; power-aware HW/SW partitioning extended to AMS systems
Physical mixed A/D and RF design	Procedural layout generation, module generators for a few block types	Module generators for often re-used blocks, design centering, performance estimation	Synthesis: behavior to layout (at least for the most important building blocks)
Parasitics extraction, automated modeling, accelerated simulation	Electromagnetic immunity simulation works but is too complicated for broad usage	2D/3D model-based order reduction for interconnect systems and substrate effects on chip, thermal package modeling	New fault-tolerant circuit architectures, robustness against technology parameter variations; order reduction for all kinds of parasitics and antennas

With respect to Design Test, analog circuitry dominates production test cost despite being a small fraction of the total area of mixed-signal SOCs. The ratio of analog testing cost to total mixed-signal product cost will continue to increase unless changes to analog testing are made. The near-term requirement is for analog/mixed-signal DFT/BIST, especially at higher resolution and/or higher frequencies beyond baseband. Test techniques for on-chip, high-resolution (>14–16 bits) ADC and high-speed (>1–5 GHz) RF components must be not only cost-effective, but also nonintrusive—they must not degrade performance of embedded analog blocks. Since high-resolution ADCs are usually constructed with multiple stages connected in serial or mash configuration, one possible direction is to utilize this structural knowledge in developing a DFT or self-test strategy.

Although the *Process Integration, Devices, and Structures chapter* specifies an analog CMOS transistor that has a higher analog supply voltage and is unscaled across 2–3 technology generations, this does not solve critical cost issues of power, process compatibility, area efficiency, design complexity, or verification and test. Furthermore, AMS design productivity remains a key challenge to development of new mixed-signal parts. A near-term roadmap for AMS DT includes new description languages as well as tools for:

- System exploration, considering the chip and the package
- Circuit synthesis and sizing
- Schematic validation
- Design for manufacturing
- Analog/RF layout synthesis
- Parasitics extraction (capacitive, inductive, thermal), automated modeling and fast simulation
- Analog IP encapsulation and reuse

Additional Design Technology Requirements

Table 20 Additional Design Technology Requirements

Year of Production	*2005*	*2006*	*2007*	*2008*	*2009*	*2012*	*2015*	*2018*	*Driver*
DRAM ½ Pitch (nm)	*80*	*70*	*65*	*57*	*50*	*36*	*25*	*18*	
SOC new design cycle (months)	**12**	**12**	**12**	**12**	**11**	**11**	**10**	**9**	SOC
SOC logic Mtx per designer-year (10-person team)	**3.3**	**4.3**	**5.4**	**7.4**	**10.6**	**24.6**	**73.4**	**113**	SOC
SOC dynamic power reduction beyond scaling (X)	**0.1**	**0.2**	**0.2**	**0.2**	**0.2**	**6**	**4.7**	**8.1**	SOC
SOC standby power reduction beyond scaling (X)	**2.4**	**3.4**	**5.1**	**6.4**	**8.73**	**18.8**	**44.4**	**232**	SOC
% Test covered by BIST	**25**	**30**	**35**	**40**	**45**	**60**	**75**	**90**	MPU, SOC

Mtx—Million transistors

Manufacturable solutions exist, and are being optimized	□
Manufacturable solutions are known	🟨
Interim solutions are known	◆
Manufacturable solutions are NOT known	🟥

Cross-cut TWG Issues

Modeling and Simulation

One of the key problems that challenge design in connection with further shrinking feature sizes is the increasing variability of design-related parameters, resulting either from fluctuations of fabrication parameters or from the intrinsic atomistic nature affecting, for example, channel doping. 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 distribution 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 just one or a few dopant atoms being at average 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 smoothen out 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 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. The added value that only simulation can provide is that a wide set of variations may be investigated largely automatically, within relatively small time and at relatively small costs.

Appendix

Variability Modeling and Roadmap

Since variability is expected to be the source of multiple critical DFM 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 productivity improvements. The requirement for such a framework is underlined by the sensitivity of variability-related information for industry participants. Since the design community needs to view variability from their parameter-based perspective, such a variability framework needs to be multi-level, i.e., it needs to cover multiple levels of design abstraction.

A variability roadmap framework (VRF) is being developed by the Design TWG as illustrated in Figure 26. In this Framework the following three 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 road-mapped at this level, based on lower-level parametric variations.
- *Device level*—since circuits are composed of devices, at this level device-related parameters are roadmapped, such as the threshold voltage V_t, or the off-current I_{off}.
- *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 road-mapped. These parameters' variability stems from some of challenging issues enumerated above, including lithography hardware resolution limitations, and the inability to control the exact number of dopants in a channel.

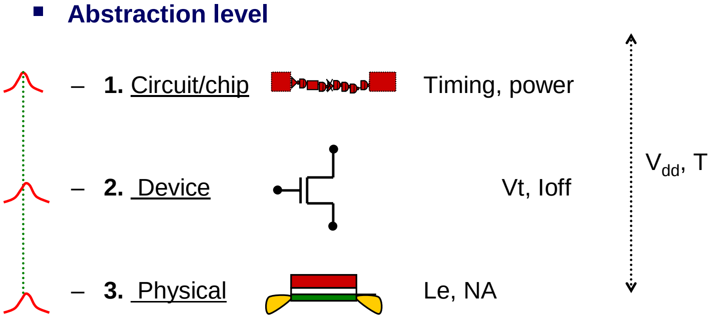


Figure 26 Possible Variability Abstraction Levels

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

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

Based on published approximated models, this model is based on a simplified gate + wire circuit slice where parameters are decomposed into gate-related and wire-related parameters. Performance is analytically modeled in the current 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, and oxide thickness; wire width, length, and thickness; ILD thickness, and wire sheet resistance.

Among other applications, this framework will be used for circuit performance variability trending. 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, the model could indicate that circuit performance variability does not seem to vary very significantly with the CD tolerance requirement, which might suggest the possibility of a relaxation of the requirement.

DT Cost and Value

As Figure 27 shows, the cost of developing and marketing a complex integrated circuit is affected by many different factors. 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 (compare to Figure 27). 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.

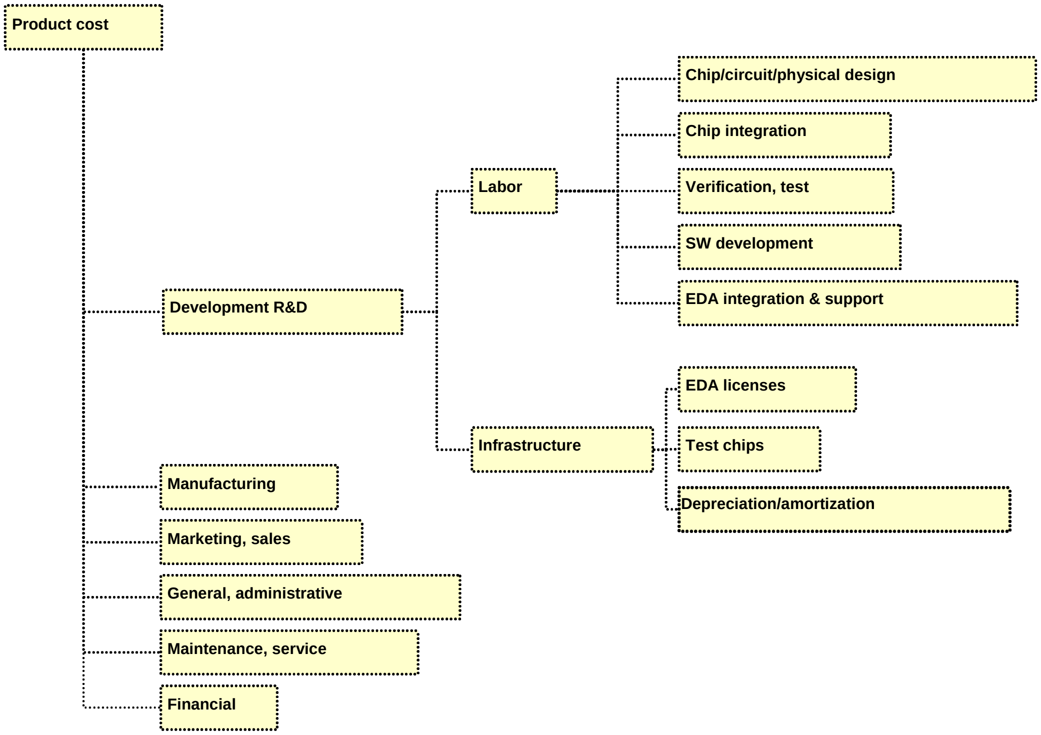


Figure 27 Simplified Electronic Product Development Cost Model

In Figure 27, items in bold 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.

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 21, 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 39.6% per year from 1990 to 2005. Specifically, the number of designers per million gates (inverse of productivity) went from 250 in the year 1990 to 8 in 2005. Labor unit costs, however, have not remained constant since 1990. According to the GTX model, the historical rate of increase in engineer labor cost is 5% per year (assuming salary and overheads starting at $181,568 in 1990). This assumption may be reviewed in upcoming revisions in light of design globalization trends. The *GTX model* is available as part of the electronic Design chapter.

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). The total implied infrastructure cost from this item is given by the product of EDA tool cost times the number of engineer-years:

$$EDAInfrastructureCost = \frac{EDAUnitCost \times DesignComplexity}{Designer\ Pr\ oductivity},$$

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.

Table 21 Design Technology Improvements and Impact on Designer Productivity

DT Improvement	*Year*	*Productivity Delta*	*Productivity (Gates/Design-Year)*	*Cost of Component Affected*	*Description of Improvement*
None	1990		4K		
In-house place and route	1993	+38.9%	5.55K	PD Integration	Automated block placement and routing.
Engineer	1995	+63.6%	9.09K	Chip/circuit/PD Verification	Engineer than can pursue all required tasks to complete a design block, from RTL to GDSII.
Reuse—small blocks	1997	+340%	40K	Circuit/PD Verification	Blocks from 2,500–74,999 gates.
Reuse—large blocks	1999	+38.9%	56K	Chip/circuit/PD Integration Verification	Blocks from 75,000–1M gates.
IC implementation suite	2001	+63.6%	91K	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	+37.5%	125K	SW development Verification	RTL verification tool ("cockpit") that takes an ES-level description and partitions it into verifiable blocks, then executes verification tools on the blocks, while tracking and reporting code coverage.
Electronic system-level (ES-level) methodology	2005	+60%	200K	SW development Verification	Level above RTL, including both HW and SW design. It consists of a behavioral (where the system function has not been partitioned) and an architectural level (where HW and SW are identified and handed off to design teams).
Very large block reuse	2007	+200%	600K	Chip/circuit/PD Verification	Blocks >1M gates; intellectual-property cores
Homogeneous parallel processing	2009	+100–200%	1200K	Chip/circuit/PD Design and Verification	Many identical cores provide specialized processing around a main processor, which allows for performance, power efficiency, and high reuse
Intelligent test bench	2011	37.5%	2400K	Chip/circuit/PD Verification	Like RTL verification tool suite, but also with automation of the Verification Partitioning step.
Concurrent software compiler	2013	60%	3300K	Chip and Electronic System Design and Verification	Enables compilation and SW development in highly parallel processing SOCs
Heterogeneous massive parallel processing	2015	+100–200%	5278K	System Electronic Design and Verification	Each of the specialized cores around the main processor is not identical from the programming and implementation standpoint
System-level DA and executable specification	2017-19	+100–200%	10557K	System Electronic Design and Verification	Automates true electronic system design on- and off-chip for the first time, including heterogenous technologies.
Total		+264,000%			

APPROXIMATED TOTAL DESIGN COST

Figure 16 quantifies the impact of the DT innovations on design cost for the power-efficient SOC-LP driver defined in the System Drivers chapter. The SOC-PE has 6.5M logic gates in 2005, implying a typical consumer/portable SOC design cost (designers + tools) of almost $20M. Without the six major DT innovations that occurred between 1993 and 2005, the design cost for the same SOC in 2005 would be approximately $900M. Furthermore, 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.[14]

[14] *The difference is thought to be due to disparities in market segments, and resulting differences in levels of circuit customization, datapath percentages, and methodology.*