INTERNATIONAL TECHNOLOGY ROADMAP FOR SEMICONDUCTORS

2003 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 2003 ITRS, the Design 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 2003 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 one million 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 ROI for supplier industries, the semiconductor investment cycle stalls. Previous ITRS editions 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 13 shows that estimated design cost of the system-on-chip low-power (SOC-LP) PDA defined in the System Drivers Chapter is $20M in 2003, versus $630M had DT innovations between 1993 and 2003 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.*

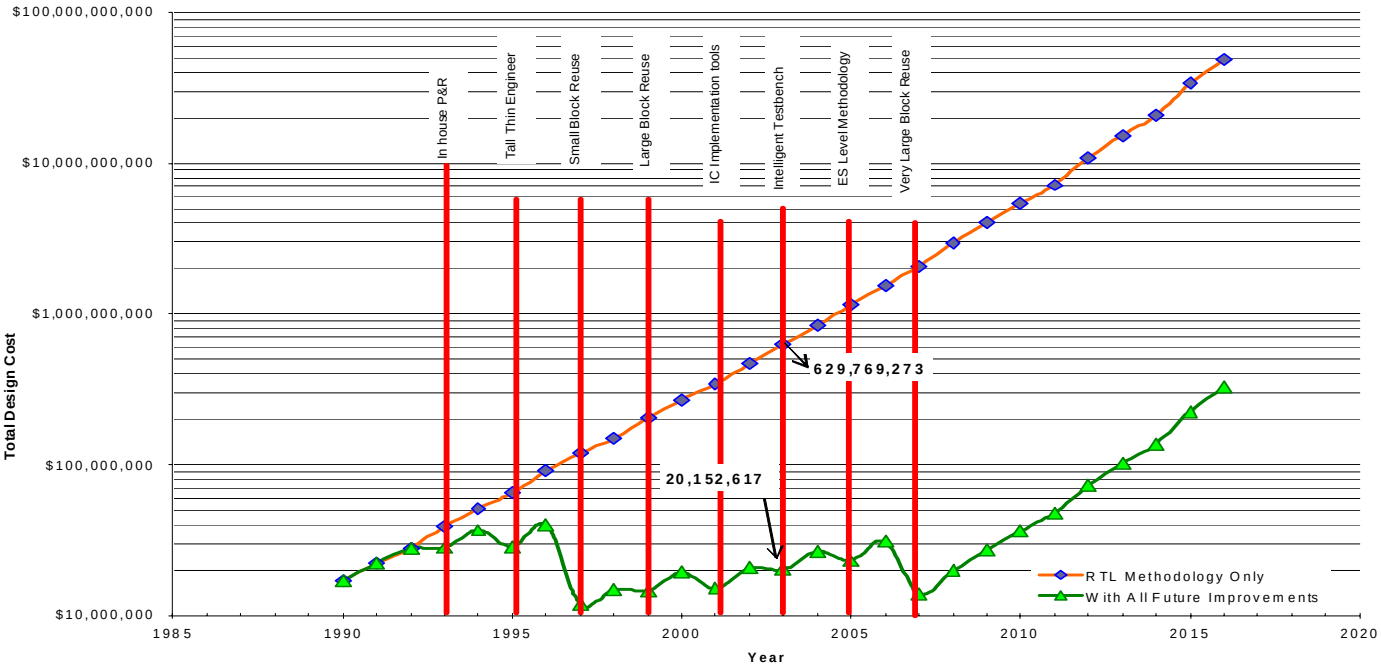


Figure 13 Impact of Design Technology on SOC LP-PDA Implementation Cost

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 according to a traditional landscape of DT areas (see Figure 14): design process; system-level design; logical, circuit and physical design; design verification; and design test.[1] These challenges are discussed at a level of detail that is actionable by management, R&D, and academia in the target supplier community, i.e., the electronic design automation (EDA) industry. As appropriate, the detailed challenges are mapped to the MPU, SOC, 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 to reflect the rise of application- and driver-specific DT, and the likely organization of future ITRS Design Chapter editions according to system drivers, rather than traditional areas of DT.

[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).*

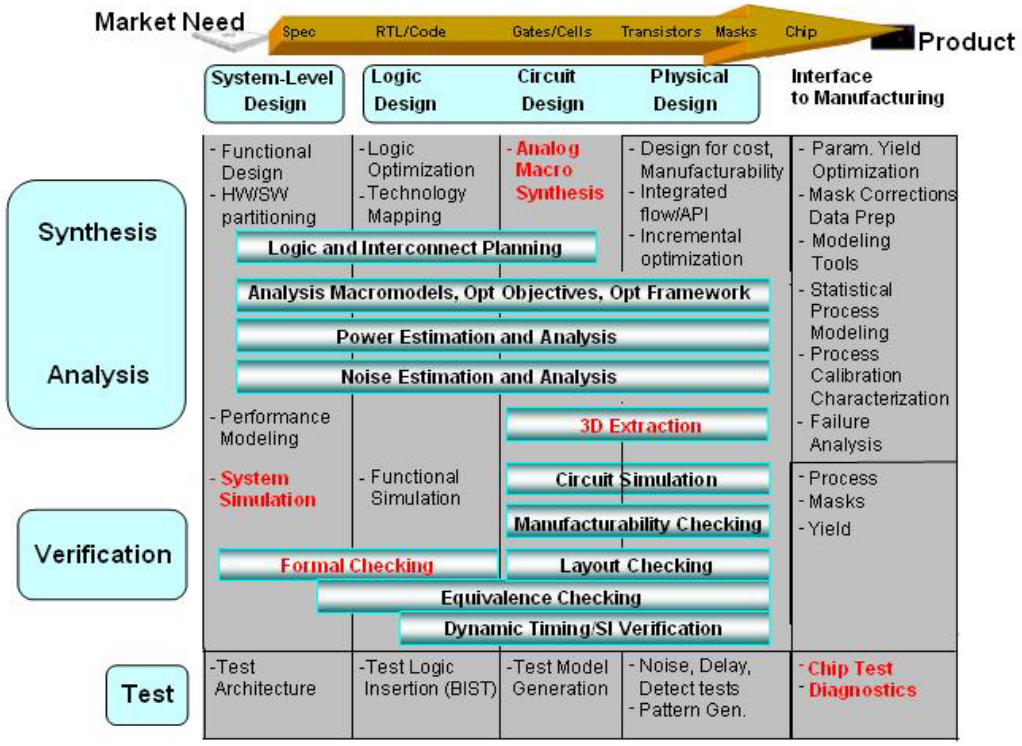


Figure 14 Landscape of Design Technology.

Roadmapping of DT is different from roadmapping of manufacturing technology. Manufacturing technology seeks to realize (for example, a target tolerance) and faces limits imposed by physical laws and material properties. In contrast, DT seeks to optimize (for example, total wirelength) 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 ITRS nodes is less strict for DT. While ITRS nodes 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.

COMPLEXITY AND CROSSCUTTING 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)
- *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)
- *complexity of manufacturing handoff* (reticle enhancement and mask writing/inspection flow, NRE cost) and

- *process variability* (library characterization, analog and digital circuit performance, error-tolerant design, layout reuse, reliable and predictable implementation platforms)

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.)
- *embedded software design* (predictable platform-based 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 1) provide concurrent optimization and analysis of more complex objectives and constraints, 2) acknowledge additional considerations such as design reuse and manufactured system cost in the design optimization, and 3) 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—Productivity, Power, Manufacturing Integration, Interference, and Error-Tolerance—are given for which potential solutions are distributed across all areas of DT. The first crosscut challenge, closely linked to system and design process complexity, is by far the most massive and critical. The second through fifth crosscut challenges are narrower in scope, and mostly address silicon complexity issues.

CROSSCUTTING CHALLENGE 1—PRODUCTIVITY

To avoid exponentially increasing design cost, overall productivity of designed functions on chip must scale at > 2× per node. Reuse productivity (including migration and AMSRF core reuse) of design, verification and test must also scale at > 2× per node. 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 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 analog and mixed-signal (AMS) design and test, which are required by the SOC and AMS

[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.*

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.

CROSSCUTTING 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. 1) Extrapolation from the Overall Roadmap Technology Characteristics and System Drivers Chapter shows that 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 250W/cm^2. The 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 Table 12 of 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 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, and low standby power 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.

CROSSCUTTING CHALLENGE 3—MANUFACTURING INTEGRATION

"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 nodes 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 built-in self-test (BIST) techniques. 2) System implementation cost, performance verification, and overall design 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, PIDS, Front-End Processing, 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.

CROSSCUTTING CHALLENGE 4—INTERFERENCE

Resource-efficient communication and synchronization, already challenged by global interconnect scaling trends, are increasingly hampered by noise and interference. Prevailing signal integrity methodologies in logical, circuit and physical design, while apparently scalable through the 100 nm node, 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 RF components will require more flexible and powerful methodologies. Issues include noise headroom (especially in low-power devices and dynamic circuits); large numbers of capacitively and inductively coupled interconnects; supply voltage IR drop and ground bounce; thermal impact on device off-currents and interconnect resistivities; and substrate coupling. A basic DT challenge is to improve characterization, modeling, analysis and estimation of noise and interference at all levels of design.

CROSSCUTTING CHALLENGE 5—ERROR TOLERANCE

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 challenges and potential solutions in the five traditional areas of DT. As noted above, most challenges map to MPU and SOC, reflecting today's EDA technology and market segmentation. Nevertheless, we begin with an overview of AMS-specific DT trends and challenges in order to emphasize 1) the trend toward application- and driver-specific DT; 2) the criticality of AMS-specific DT in scaling design productivity and quality; and 3) the likely organization of future ITRS Design Chapter editions according to system drivers. These AMS-specific challenges are also included in the summary Tables 14 through 18.

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

Analog 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 system design has remained mainly at the level covered by the section on Logical, Circuit and Physical Design later 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; 2) analog synthesis tools to reduce effort spent on analog circuit design by a limited number of analog designers; 3) yield enhancement which requires "design for manufacturing"; 4) close integration of signal processing systems (mostly digital) and analog RF front ends in almost all mobile communication devices; and 5) tight integration of electrical and non-electrical components (such as micro-electro-mechanical systems (MEMS)). These challenges may be elaborated in the context of traditional DT areas, as follows.

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.

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 (e.g., 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, etc. 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.

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.

Finally, 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 (>1-5GHz) RF components must be not only cost-effective, but also nonintrusive, i.e., 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 PIDS Chapter specifies an analog CMOS transistor that has a higher analog supply voltage and is unscaled across 2–3 technology nodes, 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
- circuit synthesis and sizing
- schematic validation
- design for manufacturing
- analog/RF layout synthesis
- parasitics extraction, modeling, and simulation and
- analog IP encapsulation and reuse

The following table summarizes near-term AMS design technology breakthroughs expected through 2007. The reader is also referred to the excellent and detailed discussion of AMS DT requirements in the March 2002 *MEDEA+ Design Automation Roadmap*.

Table 13 Near-term Breakthroughs in Design Technology for AMS

Field Of Breakthrough	*2003*	*2004–2005*	*2006–2007*
Specification, validation	**Mixed-signal description languages**	**Multi-language support**	**Complete specification-driven design flow**
Architectural design	**Algorithm-oriented design**	**Language-based**	**Synthesizable AMS**

		performance evaluation	description
Mixed A/D and RF physical design	Procedural layout generation	Design centering, performance estimation	Constraint-driven synthesis: behavior to layout
Parasitic extraction, modeling, simulation	EMI simulation	2D/3D modeling, order reduction	Fault-tolerant circuit architectures, robustness

DESIGN PROCESS

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 node 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 was described in the 1999 ITRS and 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 15 illustrates this trend. The left column shows a hardware design process, typical of the mid-1990s, 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. 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 (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 15 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 node 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

[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.*

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.

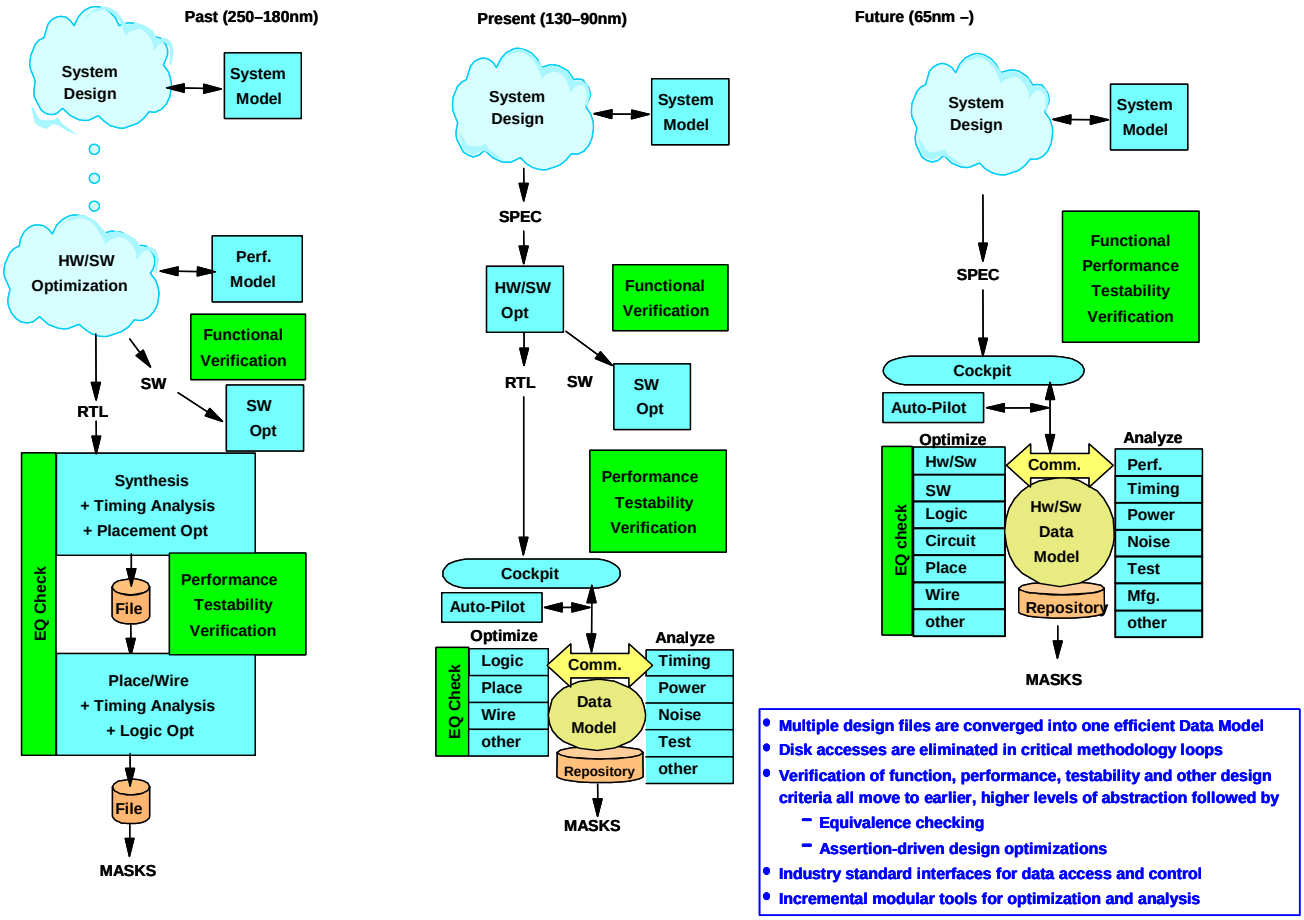


Figure 15 Required Evolution of Design System Architecture

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 15 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. 4) *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 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 15. 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 node, and in future nodes 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 15, are given in the System-Level Design section, below.

[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.*

Table 14 Design Process Challenges

Challenges 50 nm/Through 2009	*Summary Of Issues*
Silicon Complexity: devices and interconnects	All—Exponential increase in leakage power S, P, A—Power density and distribution All—Technology and library characterization S, P, A—High-frequency noise analysis S, P, A—Transmission-line interconnects All—eDRAM, eFPGA, SiGe, optical, MEMS
System Complexity: number of states, design diversity	S, P—Verifying systems with exploding number of states S, P, A—Concurrent multi-factor analysis and optimization S, P—Scalable algorithms S, A—Design and test of mixed analog and digital designs All—Complex package analysis S, A—Integrating A/D tools
Design Productivity	S—Integrating third party components S, P—Design tool interoperability S, P—Early analysis and verification methods
Time-to-Market	S, A—Support for platform-based design S, P, A—Exploiting parallel processing
Challenges <50 nm/Beyond 2009	
Manufacturability	All—Cross-chip variability All—Sub-wavelength mask correction (AltPSM, OPC, RET) All—Design for yield and manufacturability All—Standards for sharing design and manufacturing data
System-level Design	S—Common hardware/software (HW/SW) representation S—SW synthesis, HW/SW optimization

This table summarizes challenges to the design process advances implied by the above four trends. Each challenge is labeled with a list of the most relevant system drivers (S—system on chip, P—microprocessor, A—analog/mixed-signal, M—memory).

Table 14 summarizes challenges to the design process advances implied by the above four trends. Each challenge is labeled with a list of the most relevant system drivers (S = SOC, P = MPU, A = AMS, M = Memory). The remainder of this section gives explanatory comments.

DESIGN PROCESS CHALLENGES—NEAR TERM (>50 nm)

1. Silicon Complexity—This refers to the "silicon complexity" challenges noted above, as well as the second through fifth crosscutting challenges. Potential solution approaches will entail restrictions on design rules (such as layout ground rules) as well as continued improvement in analyses (substrate coupling, crosstalk-induced timing, and IR drop, etc.).

2. System Complexity—Potential solution approaches seek increased capacity, use of hierarchical methods, and a higher abstraction level of design. Tool integrations must minimize unproductive data translation time and data redundancy, and support synthesis-analysis tool architectures that execute concurrently on distributed or SMP platforms (for die-package-board codesign, manufacturability optimization, and power/noise management). Fundamental algorithms must be scalable, so that exponentially growing instance sizes remain effectively solvable on current computing hardware. Within a "construct by correction," successive approximation framework, incremental analyses and optimizations are needed with runtimes proportional to only the amount of design that changes between iterations. At the same time, shrinking feature size and pitch, coupled with increasing frequencies, demand increased levels of detail and accuracy within analysis models for delay, noise/interference, power, and reliability; hence, a broad spectrum of optimizations and analyses require increased awareness of the manufacturing process character. Such characterization must be provided via standard device technology and process specific models that will likely remain under control of that technology or process owner. Consistent interpretation of these models, such as via execution engines, is needed across the DT that uses these models. Finally, a complementary requirement at all levels of the design process is the identification of analysis abstractions and

macromodels that can serve as optimization objectives for upstream steps in the design process. A common data model and database must serve as much of the design flow as possible (certainly, from RTL to manufacturing data generation).

3. Productivity—Designer productivity will be improved through multi-level automation, increased reuse, and "freedom from choice." To systematically advance DT in this direction, the notion of design as "art" rather than "science" must be dispelled, and a mindset of "measure to improve" adopted. To develop methods for continuous design process and design productivity improvement, near-term technology needs begin with metrics of the design process with respect to quality and predictability. These in turn depend on metrics of design *quality* and design process *predictability*. Quality metrics assess the extent to which available manufacturing capability is fully exploited by DT.[5] Predictability metrics enable quantified progress toward *noise-free* algorithms, tools and flows that are predictable and stable with respect to accepted solution metrics.[6] Related needs include infrastructure for calibration and benchmarking. In addition, top-down design and rapid design space exploration require estimators of tool sweet spots ("field of use"), instance-specific tool runtime parameters, and tool output solution quality based on parameterized models of both tools and design instances. Eventually, predictive models of application- or driver-specific silicon implementation are needed. Finally, complementary innovations should measure and improve design *technology* productivity.[7]

Today, the spectrum of design flows ranges from loosely coupled tools integrated via industry-standard sequential files to tightly integrated systems with closed/proprietary interfaces. In high-end design projects, no single DT vendor, nor the collection of all DT vendors together, provides the full scope of required DT. Hence, internally developed DT must augment multi-vendor design flows. High-end designs can also involve large, geographically distributed design teams from different companies. This mandates the need to share designs and reuse them across DT solutions. To facilitate this, needs include 1) an industry standard method to share design descriptions in real-time across disparate DT's and between design and manufacturing, and 2) industry adoption of this method across all DT solutions. There is a great need to reduce integration costs, currently estimated (Dataquest, 2001)[8] to be 30% of total design costs, and keep them from exceeding raw tool and machine costs while at the same time providing real-time DT interoperability in place of sequential DT data exchange. Improved tool reusability and interoperability, with reduced integration costs, will in the long term enable high-quality implementation flows for multiple platforms and applications (wireless, internetworking, etc.).

4. Time to Market—Obvious approaches to reduce time to market include use of reprogrammable and structured-ASIC fabrics, reuse of predesigned cores and "platform" architectures, and "pervasive automation." In many fast-growing semiconductor markets, the greatest demand is for low-cost, relatively low-performance, and fast time-to-market designs. Products in these markets are typically SOCs with heavy reuse. Addressing this demand requires common information models to describe the characteristics of reusable cores at levels of abstraction that permit efficient design optimization. New tools will be required to support a find-and-try style of reuse-centric design space exploration and design optimization. Development and formalization of design rules that, if followed, assure reusability will require associated design and analysis software to support and enforce such rules. As increasingly heterogeneous fabrics and technologies are integrated in a single SOC or SIP, new design flows must emerge that enable effective integration of hardware, software, digital and possibly analog, MEMS, and memory on a single chip. As already noted, AMS design is a productivity bottleneck that mandates improved ability to synthesize analog (and, in the long-term, mixed-technology) designs with

---

[5] *Obvious metrics include speed, power and density relative to process geometry and supply voltage. Reusability, testability, verification coverage, etc. affect ROI of design and manufacturing technology investment, and should also be considered in formulating design quality metrics.*

[6] *Inherent tool noise is the variation in solution quality that results from non-functional changes to the tool input (e.g., renaming variables or instances, permuting lines of a gate-level netlist, etc.). Recent literature measures inherent tool noise of 30% in commercial placement and routing tools; such levels block development of accurate estimation capability and must be reduced.*

[7] *Several recent studies estimate the worldwide population of design technology researchers and developers (including academics) to be approximately 6,000. Not only must this resource be carefully targeted for maximum impact, but DT productivity (i.e., technology delivery) must be improved. Example measures toward this end include understanding of DT impact on design productivity, and greater reuse of DT intellectual property.*

[8] *Total design NRE cost is comprised of (1) designer and CAD support engineer salaries per unit time, multiplied by design time; (2) tool license cost per unit time, multiplied by design time; (3) machine and other infrastructure cost per unit time, multiplied by design time. "Interoperability cost" is reflected in increased design time (extra translation steps within the flow, and extra design iterations due to errors caused by lack of interoperability), increased CAD support salaries, and increased infrastructure cost (data storage and management). Design NRE* cost, *which is a function of design turnaround time (TAT) and per-unit time resource costs, should not be confused with design* value, *which is a function of time-to-market (and hence design TAT) and design quality. Only from knowledge of* both *design cost and design value can the ROI of design technology be determined. See the Design Cost analysis in the Appendix.*

efficiency and quality comparable to digital RTL-based synthesis. Longer-term needs also include higher-level verification of function, performance and manufacturability, as well as supply-chain management and design processes that holistically minimize system implementation cost. The model for EDA may change so that specific platforms, application domains and even individual designs may require the building of customized, design-specific design flows out of interoperable components. This would change the focus of EDA as an industry from building complete flows to building composable parts with standardized interfaces.[9]

DESIGN PROCESS CHALLENGES—LONG TERM (<50 nm)

1. Manufacturability—This refers to issues noted above with respect to the Manufacturing Integration cross-cutting challenge and the trend toward Design for Manufacturability. Potential solution approaches include restricted design rules as well as integrated design and manufacturability optimizations.

2. System-Level Design—At 65 nm and below, the transition to system-level design, along with unification of HW–SW design, cannot be avoided. The next section discusses system-level design technology requirements in detail.

SYSTEM-LEVEL DESIGN

In system-level design[10], 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, 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, as emerging hardware platforms become increasingly fixed, embedded software becomes the main focus for product differentiation.

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% (compare to "Very Large Block Reuse" in the Design Cost Model appendix). 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. While reuse has been a requirement for several editions of the ITRS, it has not yet permeated the system design process. At the 90 nm to 65 nm transition, this requirement is critical.

---

[9] *Programmable platforms and embedded software-centered system-level design flows (see the System-Level Design section, below) may require highly specialized tools yet generate too few design starts to sustain today's EDA business model. In such a regime, a services-oriented business model for complex system design could emerge, wherein EDA companies (a) build tool and infrastructure pieces from which design systems can be created, then (b) in partnership with system design teams, create design-specific design systems in parallel with the system designs themselves.*

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

- *Platform-based design*—An extension of core-based design creates highly reusable *groups* of cores to form a complete hardware "platforms," 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 codesign, 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 codesign decisions usually made for a small number of functions that have critical impact
- *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. 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 16 and 17 illustrate this advance. Figure 16 shows a typical modern design flow (a realization of the left portion of Figure 15) with a mixture of manual steps above RTL and an automated process for RTL implementation. Figure 17 depicts a flow that is required in the near future (~2004); the manual process of mapping micro-architectural design decisions into RTL has been replaced with an automated step.

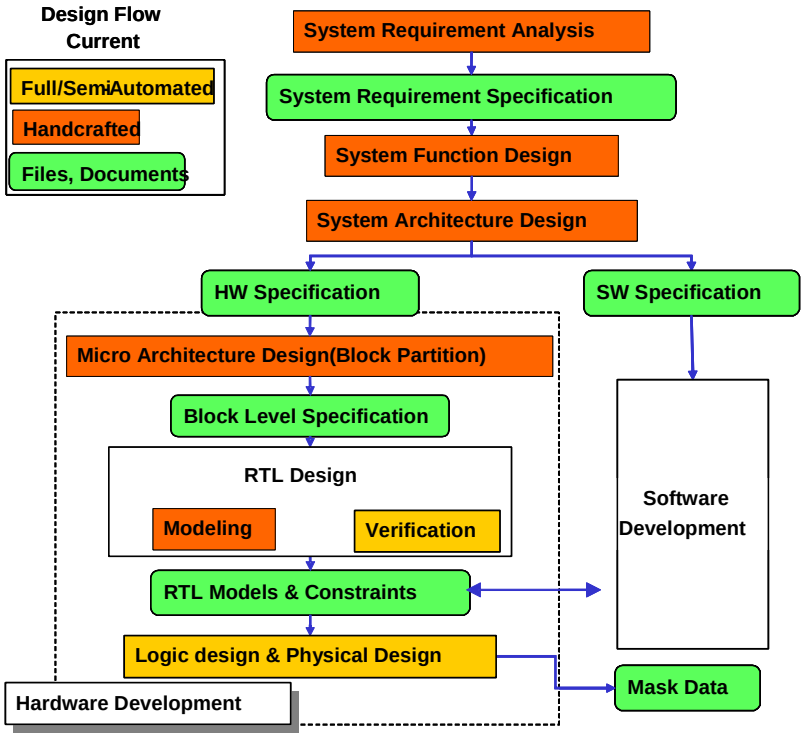


Figure 16 Current Design Flow

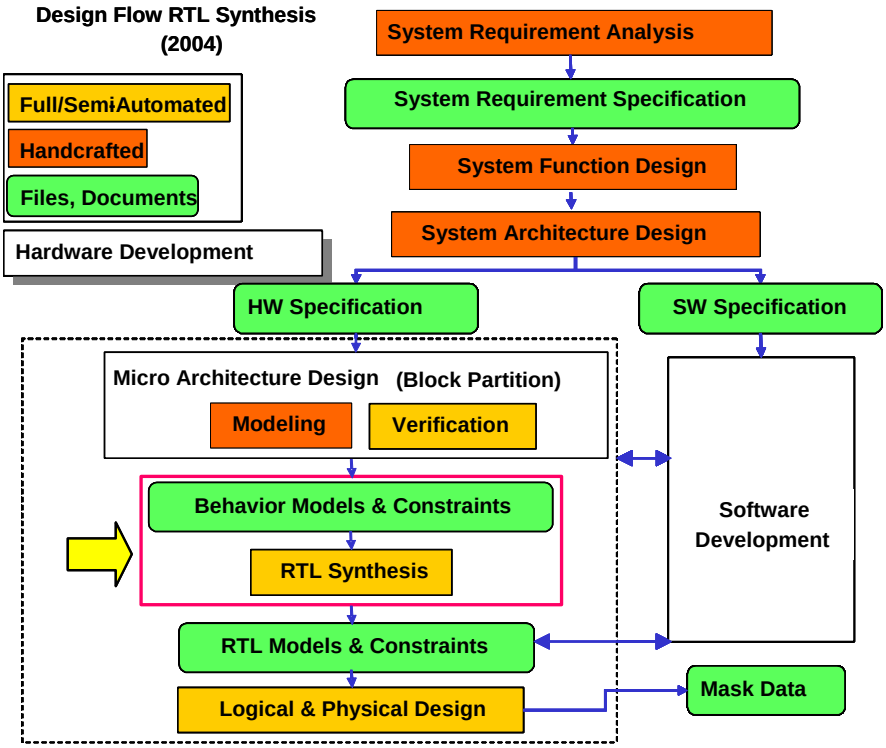


Figure 17 RTL Synthesis for Design Flow in year 2004

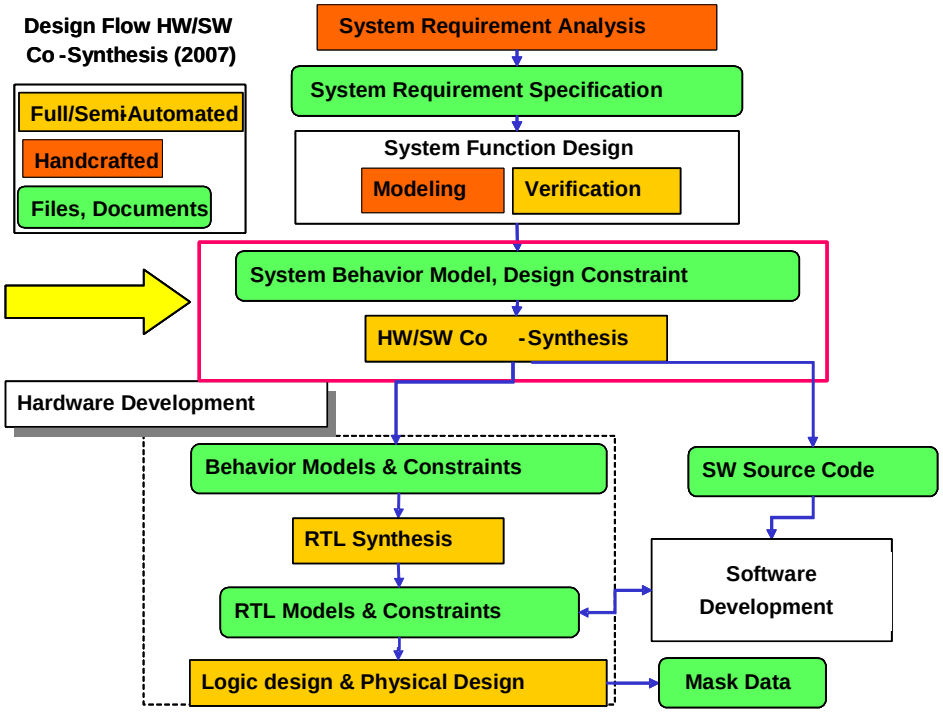


Figure 18 Design Flow in Year 2007

- *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 18 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.

Table 15 lists system-level design challenges and issues. The remainder of this section provides explanatory comments.

Table 15 System-level Design Challenges

Challenges 50 nm/Through 2009	*Summary Of Issues*
System complexity	S—Higher-level abstraction and specification S—Dynamism and softness S, A—System-level reuse S, A, P—Design space exploration and system-level estimation S—Efficiency of behavioral synthesis and software compilation S—Automatic interface synthesis
System power consumption	P, S—Energy-performance-flexibility trade-offs P, S—Novel data transfer and storage techniques
Integration of heterogeneous technologies	S, P, A—Codesign (HW–SW, chip-package-board, fixed-reprogrammable) A, S—Non-scalability of analog circuits; analog behavioral modeling and synthesis P, S, A—Top-down implementation planning with diverse fabrics (digital, AMS, RF, MEMS, EO, SW)
Embedded software	S—SW–SW codesign onto highly programmable platforms S—System capture and abstraction S—New automation from high level description to HW–SW implementations, including SW synthesis S—Formal verification for SW S—HW–SW coverification
Links to verification, test and culture	S—Integration-oriented verification and test architectures S, P, A—Divergent design practices and cultures
Additional Challenges <50 nm/Beyond 2009	
System complexity	S, P—Communication-centric design and network-based communications on chip All—Design robustness
System power consumption	P—Non-scaling of centrally organized architectures S—Building large systems from heterogeneous SOCs
Integration of heterogeneous technologies	S—Total system integration including new integrated technologies (MEMS, electro-optical, electro-chemical, electro-biological or organic)

This table summarizes challenges to the design process advances implied by the above four trends. Each challenge is labeled with a list of the most relevant system drivers (S—system on chip, P—microprocessor, A—analog/mixed-signal, M—memory).

SYSTEM-LEVEL DESIGN CHALLENGES—NEAR TERM (>50 nm)

1. System complexity—System complexity and design productivity challenges induce the following technology needs.

- *Higher-level abstraction and specification*—Design abstraction levels must be raised above the present-day RT-level for HW and (often processor-dependent) C code level for embedded SW. Models of intended system function in the operating environment, and of both processing and communicating architectural resources, are necessary for design space exploration for complex, multi-function systems. Emerging design language infrastructure for complex models (e.g., C++ derivatives such as SystemC) must be leveraged by industry consensus on use-methodology and abstraction levels.
- *Dynamism and softness—Dynamism* is a system property that enables the system to adapt at runtime under the influence of use requirements (such as multimedia quality of service). New abstractions are required for such runtime modification of function and architecture. *Softness* is a system property that enables a system to be modified or reprogrammed. While softness will apply at multiple abstraction levels, such as process, ISA, microarchitecture, and implementation, no methodology yet exists that enables designers to identify and select programmable features of a chip. Architectural innovation is needed to meet SOC opportunities and constraints, and to support such dynamic behavior/architecture.
- *System-level reuse*—While some progress has been made in RT- and layout-level design reuse, reuse methodologies and associated design tools (for example, for evaluation of IP quality and suitability) are still in their infancy.[11] Design productivity impact requires the reuse of complex HW–SW architectures via methods such as platform-based design. Software reuse can require design methodologies for processors that run "standard" software but are specialized to different cost-performance trade-offs or computational domains.
- *Design space exploration and system-level estimation*—The increasing range of integratable implementation fabrics requires new tools for optimization of the function-architecture mapping. Criteria for such optimizations include power, area, throughput, etc. *Estimation* technology is a fundamental and yet-undeveloped adjunct of top-down design space exploration. Estimators allow accurate prediction of the eventual values of the design criteria without going through all details of the design process. Such a capability must increasingly span multiple levels of abstraction (timing estimation at physical, RTL, and system levels) as well as statistical modeling of specific application spaces.
- *Efficient behavioral synthesis and SW compilation*—To complement eventual reuse solutions, new synthesis methods are required. Automated mappings from function to architecture—behavioral synthesis for hardware, and compilation for software—have limited application today due to poor quality of results (such as inadequate exploitation of processor features and parallelism by compilers). Needs include acceptable synthesis quality for mixed control and dataflow, and retargetable SW compilers that automatically achieve high-quality results on a wide variety of instruction set architectures (control processing and DSP) and design domains.
- *Automatic interface synthesis*—To better match system constraints, interfaces between HW–HW, HW–SW and SW–SW must be synthesized (or standardized) instead of hand-designed or drawn from parameterized libraries. This is a natural adjunct to emerging communication-centric design approaches.

2. System power consumption—Several current processor architecture paradigms (always-on, monolithic processor; high-power VLIW and speculative techniques) will give way in the near term to more massive, heterogeneous parallelism. Effective trade-offs of energy, performance and flexibility must be predictable at the system-level, and will require such techniques as dynamic power control; flexible block shut-down and restart; dynamically variable on-chip voltage supplies; standards-controlled power management; and power-efficient use of reconfigurable logic. The main near-term levers for power consumption at the system level are data transfer, and memory architecture and hierarchy.

3. Integration of heterogeneous technologies—Making effective design choices across the range of available component technologies requires new approaches to heterogeneous modeling and codesign.

- *Codesign*—DT for partitioning and codesign, whether of HW–SW, analog-digital, fixed-reprogrammable, or die-package (-board), must be increasingly aware of cost and risk factors across the entire system development process. Trade-offs and partitioning across analog-digital boundaries require compatible specification and functional modeling abstractions, as well as the ability to simulate all parts of a system within a single execution and analysis

[11] *Supporting infrastructure for IP reuse (IP certification and validation services, as well as IP protection mechanisms) is required.*

environment. More generally, design optimization must increasingly consider a device in its system context rather than as a standalone entity.

- *Top-down implementation planning with diverse fabrics*—With more integratable fabrics, there is increased need for a single hierarchical mixed-technology planning environment that links system-level objectives and constraints with fabric details (circuit topology, interfaces, and spatial embedding).

4. Embedded software (ESW)—Reuse- and SW-centric SOC design is increasingly partitioned into 1) the creation of programmable platforms by a team of hardware-centric and hardware-dependent software-centric designers, and 2) use of the platforms in derivative products by a team of applications and software-centric designers. The system platform (one can think of this as an application programmer's interface, or API) is the intermediate vehicle between the two design teams, and between application domain requirements and underlying silicon capability. For this context, key issues include the following:

- *SW–SW codesign onto highly programmable platforms*—A highly programmable, application domain-specific platform offers multiple implementation fabrics (SW running on possibly configurable control processors or DSPs, HW implemented with multi-mode dedicated function blocks, reconfigurable logic, etc.) and programmable interfacing resources (SW–SW, HW–HW and HW–SW) to a derivative design. With such softness at multiple levels (software, assembly code, bit streams, etc.) there is a need for "SW–SW codesign"—rapid, reliable and predictable mapping of functions onto programmable processing and communications resources to meet system timing and power constraints.
- *System capture and abstraction*—Systems are denoted within a variety of models of computation (MOCs), such as discrete event, static and dynamic dataflow, or continuous-time. Designers must be able to model system functional and communication requirements using implementation-independent executable notations that are "natural" for each domain or MOC. These models must be composable into correct, executable system specifications.

Additional technology needs for the SOC context include new automation (including SW synthesis) from high-level description to HW–SW implementations, formal verification for SW, and HW–SW coverification.

5. Links to verification, test and culture—Shifting the design focus from block creation to block reuse on SOC platforms has the following implications for verification, test and "cultural" aspects of DT.

- *Integration-oriented verification and test architectures*—Design integration must be paralleled by verification integration via hierarchical testbench integration and migration. System-level block models form the basis of abstract transaction-level verification models that, upon integration, can be used within implementation synthesis and verification. Formally specified abstract specification notations will enable application to SW of model and property checking notions from formal HW verification; research in formal SW verification can also be exploited. Integration-oriented architectures are similarly required for test, so that reusable test elements for specific blocks can be consolidated into an overall system test. The cost of test application (tester time, and ATE speed limitations) will increasingly motivate SW-driven, on-chip self-test technologies; dependencies between system design and test architecture design must be planned from initial system conception.
- *Divergent design practices and cultures*—The trajectory of system-level design, toward ever-greater abstraction and application specificity, diverges from the trajectory of IC implementation, which must be increasingly aware of detailed physical and manufacturing cost realities. This *implementation gap* and its solution have several facets, as follow: 1) The associated culture gap must be addressed by careful and rigorous definition of design abstraction levels that closely track the path from system designer to IC implementation designer.[12] 2) IC "silicon implementation platforms" must emerge that afford automated mapping, predictable implementation flows, and accurate cost estimation to the system designer. Without DT support of such a connection point (such as a central design environment or "cockpit"), links between system-level design and IC implementation will remain ad hoc and a source of design inefficiency.

SYSTEM-LEVEL DESIGN CHALLENGES—LONG TERM (<50 nm)

1. System complexity—In the long term, system complexity leads to new focus on communication over computation, and on overall system robustness.

[12] *System design has different engineering foundations than IC design implementation. The effect of the implementation gap is that designers who are proficient at the implementation level will grow unaware of the concerns of system level designers, and vice versa—i.e., a culture gap.*

- *Communications-centric design*—At 65 nm and below, communication architectures and protocols for on-chip functional processing units will require significant change from today's approaches. As it becomes impossible to move signals across a large die within one clock cycle or in a power-effective manner, or to run control and dataflow processes at the same clock rate, the likely result is a shift to asynchronous (or, globally asynchronous and locally synchronous (GALS)) design style. In such a regime, islands of self-timed functionality communicate via network-oriented protocols. This matches requirements of system-level power optimization schemes, like the avoidance of switching large clock nets at the maximum chip frequency. The communication-centric perspective implies that 1) algorithms must be redesigned as concurrent collaborating data transformations controlled by high-level occasional control events, and 2) the functional partitioning task, and communications between functions, will remain at the center of the design process.
- *Design robustness*—Yield issues, asynchronous design styles and communications-centric design following network-oriented paradigms together imply that SOC design will more resemble the creation of a large-scale communications network than traditional IC design practice. In such a network, communications can be assumed to be potentially lossy, and nodes may fail. Yet, the network must still achieve its system requirements for processing correctness and throughput. Such a mapping of a completely specified function to an inherently imperfect set of implementation fabric(s) will demand whole new ways of designing, mapping and controlling at run-time a fault-tolerant processing and communications fabric.

2. System power consumption—Below 65 nm, MPU designs hit fundamental walls of performance, power consumption and heat dissipation. The magnitude and difficulty are such that radical rethinking of high-performance systems becomes necessary. Power consumption can be managed only by careful application of on-chip processing parallelism. This will result in a different approach to system level design. Instead of mapping a selected function to a predefined architecture, the future goal of system-level design is to map a maximally parallel function to a maximally parallel implementation. Methodologically, this defines a new design domain that emphasizes distributed implementation over centralized implementation; this reprises the need for communications-centric design stated above. For well-understood applications, direct mapping to optimized HW implementations may be viable; for applications that are dependent on multiple and variable standards, reconfigurable logic may be viable; when lower throughput is needed, all unnecessary parts of the system will be powered down. Given such trends, standalone MPU design style will likely evolve into a sea-of-processing elements design style.

3. Integration of heterogeneous technologies—Future SOC and system-in-package technologies, along with their associated integration capacities, may bring about the end of analog and mixed-signal design as a separate discipline. System elements from virtually all design domains will be codesigned for integration at either the package or the substrate level. New technologies such as electrochemical ("labs on a chip") or electro-biological (with organic materials for sensing, actuating and possibly computation) will require new modeling approaches, new types of creation and integration guidelines and rules, and, depending on the numbers of such design starts, may foster whole new toolsets. Nascent research in this area still focuses on the new technologies themselves, and not yet on the codesign, modeling, integration and tool issues.

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 GDSII 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.

Challenges for logical, circuit and physical design are summarized in Table 16, and the remainder of this section provides explanatory comments. Because near-term challenges continue into the long-term except as noted, we combine the near- and long-term discussions for each challenge. The key emphases are 1) predictability, and 2) improved capability to model, analyze and leverage nanometer-scale circuit phenomena. Point 1 encompasses such concepts as platform-based design and reliable implementation platforms, reuse, and process variability mitigation (e.g., reticle enhancement techniques); these correspond to the first two challenges given in the table. Point 2 encompasses such phenomena as increased leakage, soft errors, and atomic-scale effects, as well as the increased significance of SOI, mixed-signal implementations and circuit innovation; these correspond to the second two challenges in Table 16. Note that some analog/mixed-signal discussions have been consolidated within a separate section in this chapter.

Table 16 Logical, Circuit, and Physical Design Challenges

Challenges 50 nm/Through 2009	*Summary Of Issues*
Efficient and predictable implementation	P—Scalable, incremental analyses and optimizations
	P, S—Unified implementation/interconnect planning and estimation/prediction
	P, S—Synchronization and global signaling
	S, A, P—Heterogeneous system composition
	P, S—Links to verification and test
Variability and design-manufacturing interface	P, S, A—Uncertainty of fundamental chip parameters (timing, skew, matching) due to manufacturing and dynamic variability sources
	All—Process modeling and characterization
	P, S—Cost-effective circuit, layout and reticle enhancement to manage manufacturing variability
Silicon complexity, non-ideal device scaling and power management	P, S—Leakage and power management
	All—Reliability and fault tolerance, soft error
	P, S—Analysis complexity and consistent analyses / synthesis objectives
Circuit design to fully exploit device technology innovation	P—Support for new circuit families that address power and performance challenges
	P, S—Implementation tools for SOI
	A, S—Analog synthesis
Additional Challenges <50 nm/Beyond 2009	
Efficient and predictable implementation	S—Reliable, predictable fabric- and application-specific silicon implementation platforms
	S—Cost-driven implementation flows
Variability and design-manufacturing interface	P, S, A—Increasing atomic-scale variability effects
Silicon complexity, non-ideal device scaling and power management	P—Recapture of reliability lost in manufacturing test
Circuit design to fully exploit device technology innovation	P, A—Increasing atomic-scale effects
	P, S, A—Adaptive and self-repairing circuits
	A, S—Low-power sensing and sensor interface circuits; micro-optical devices

This table summarizes challenges to the design process advances implied by the above four trends. Each challenge is labeled with a list of the most relevant system drivers (S—system on chip, P—microprocessor, A—analog/mixed-signal, M—memory).

LOGICAL, CIRCUIT, AND PHYSICAL DESIGN CHALLENGES

1. Efficient and predictable implementation

- *Scalable, incremental analyses and optimizations*—Since logical and physical design tools operate at the lowest levels of abstraction, their core algorithms face instance complexities that grow by at least 2× per technology node. Scalability requires new ways to manage data, traverse solution spaces, and map optimizations onto distributed/parallel computational resources. For construct-by-correction methodologies as well as for reuse

productivity, incremental specification, synthesis and analysis/verification is needed with runtimes proportional to the amount of change made to the input. Furthermore, chip implementation will increasingly entail large-scale, interacting, multi-level and multi-objective global optimizations; tools will need to generate families of solutions that capture complex trade-offs among different objectives. Techniques for constraint-dominated global optimization, resource-bounded optimization, and optimization with partial or probabilistic design information are needed. Such improvements in core algorithm technology should occur across the range of available implementation fabrics (full-custom to fully-programmable). As noted in the Design Process discussion, metrics are needed to confirm productivity improvements.

- *Unified implementation/interconnect planning and estimation/prediction*—Today's implementation planning tools create logic and timing structure concurrently with constraint budgets and spatial embedding. An established mechanism combines RT-level floorplanning and global interconnect planning to define repeater insertion and pipelining as well as detailed layout of global signals; the result is passed as a constraint to logic synthesis and optimization, placement and routing. Associated system architecture optimizations can reduce global wires; interconnect architecture optimizations can match the number and dimensions of wiring layers to the given system. Future productivity improvements will require continued logical-physical synthesis, layout-clock-test synthesis, etc. unifications, with a near-term goal (compare to Figure 15) being co-optimization of timing structure, signaling strategies, logic optimization, placement and routing in a single environment. Note that a top-down design process will improve design space exploration and productivity by improving estimation and prediction. For chip implementation, layout-aware estimation is needed due to 1) topology-sensitive global interconnect delays that do not scale with device switching times, 2) capacitive and inductive coupling interactions among devices and interconnects that dominate timing/noise validations, and 3) "lumpy" performance metrics in reprogrammable or IP block-dominated fabrics. DT must actively pursue a variety of methodologies for achieving predictable design processes.[13] Long-term, as the handoff from system-level design evolves upward from RTL and toward greater platform or fabric specificity, implementation tools and flows will need to adapt similarly, eventually affording platform-specific implementation flows. Cost optimization is another long-term goal.
- *Synchronization and global signaling*—Minimum clock periods scale as roughly 12 FO4 inverter delays (approximately 170× the CV/I metric in the PIDS Chapter) for high-speed designs while die sizes remain constant; across-chip communication hence requires increasingly many clock cycles. As noted in the System Drivers chapter, MPU global interconnects are already pipelined and are not a limiting factor for clock frequency. However, clock distribution in purely synchronous designs can account for over 40% of dynamic power and is subject to increasing stress (for example, shielding resource requirements, limits of edge rates and jitter, and parametric yield loss due to variability-induced skew). As a result, there is a clear need for more robust and power-efficient hybridization of synchronous and asynchronous designs. For example, the communication-based design paradigm entails globally asynchronous coupling of locally synchronous, high-performance blocks. Globally synchronous, locally asynchronous paradigms are also viable; in multi-core MPUs. Clock rates on chip will vary by two orders of magnitude or more, and multi-cycle interconnect paths will be commonplace. Implementation DT must support such synchronization paradigms, as well as timing structures wherein "more paths are critical" in the aftermath of timing and power optimizations. The latter phenomenon is accompanied by quadratically increasing delay sensitivities to process variation; the need for greater margins ultimately limits the return on traditional delay slack optimization. On-chip global signaling increasingly relies on low-swing differential and multi-phase clocking techniques, as well as digital PLL/DLL synchronization, following off-chip signaling trends. Improved efficiency and signal reliability of traditional buffered global interconnect must be accompanied by new syntheses and analyses for boosters, state- and transition-awareness, multilevel encoding, time-sharing of interconnect resources, and other emerging signaling paradigms.
- *Heterogeneous system composition*—Heterogeneity affects implementation in many forms. 1) Heterogeneous single-die integration of analog, mixed-signal, and RF (AMSRF) with digital logic presents new challenges from planning through layout. Tools must handle greater sensitivities of, and interactions between, AMSRF circuits with respect to noise and interference, along with constraint-dominated formulations (matching, symmetry, and electrical rules in

[13] *Past methodologies rely on simple statistical models (e.g., "wireload models") and "limited-loops" iteration (e.g., one implementation pass to estimate layout, and a second pass that begins by assuming the layout estimate). Current and future methodologies entail (a) restricted circuit and layout styles (e.g., two-level programmable logic fabric implementation, or doubly-shielded signal wires) to improve or even guarantee timing and noise correctness; (b) use of "enforceable assumptions" in the absence of good predictions (e.g., constant-delay methodology variants); and (c) removing the requirement of predictability in the first place (e.g., latency-insensitive synchronization protocols that guarantee correct behavior no matter how many clock cycles separate components). As with all methodologies, these entail trade-offs among various metrics such as design TAT, area, timing, power, etc.*

layout, etc.). Long-term, the integration of MEMS or other technologies will require novel fault avoidance and fault tolerance methods. 2) Driven by cost factors (manufacturing cost, and the cost of communication), integration choices must transparently encompass multi-die (system in package) and stacked-die options. This requires optimization of system-level interconnect from die through package and board. The *Assembly and Packaging Chapter* names tool support for die-package analysis and co-optimization as a key challenge that is shared with DT. 3) Again driven by cost considerations (mask NRE, design TAT), reprogrammable blocks will be increasingly present in SOC designs. Mapping and layout tools must handle more physical effects, constraint types, and "lumpiness" of device/interconnect sizes and delays, as well as a richer palette (electrical, metal-only, via-only or hybrid) of reprogrammability. 4) New tools are needed to support hard or semi-hard IP *reuse*, including block characterization and abstraction, constraints management, and flexible functional/physical hierarchy management. With IP block integration, all phases of implementation become more difficult; "mixed-mode" placement of blocks and cells remains challenging today. Process technology abstraction and "projection" will be needed for reuse (migration) productivity across process generations. Another facet of reuse is *redesign*: long-term, incremental syntheses should merge new design features into an existing design while preserving as much of the existing design realization as possible. (New designs must therefore be made amenable to participation in redesign.)

- *Links to verification and test*—As traditional post silicon testing methodologies become too costly and complex, more systematic test and verification support will be needed during chip implementation. Future logic and layout syntheses must automatically insert test structures while respecting timing, power, density, reliability, and signal integrity constraints. Approaches will span self-test circuitry to use of on-chip processors (such as via supplemental firmware code). The trend to greater integration of dynamic, asynchronous and AMS/RF circuits increases vulnerability to noise, crosstalk and soft error. This will likely lead to increased reliance on defect-based testing: for example, physical design tools will need to extract fault locations for new types of defects. At the same time, rapid production ramps will require physical-level tools that support rapid diagnosis of failures on this widening variety of circuit types.

2. Variability and design-manufacturing interface—The inability to continue historical process tolerances (refer to the *PIDS*, *Interconnect* and *Lithography* Chapters) leads to significantly greater variability challenges for DT. As 3-sigma variation of fundamental parameters such as L_{gate}, T_{ox}, and interconnect dimensions goes well beyond 15%, new circuit topologies and logic/layout optimizations are needed to cope with variability. Truly atomic-scale effects govern the statistics of many process steps (gate oxide planarization, ion implant, etc.).

- *Statistical timing analysis (STA) and performance verification tools*—STA and performance verification must comprehend parasitics, delays and geometries that are parameterized by distributions. In general, design centering must optimize for parametric yield and revenue per wafer, rather than for traditional performance metrics. Manufacturing variability (and the proliferation of new materials and processes) also requires a more extensive design-manufacturing interface that supplies design rules and process abstractions to layout. Richer statistical and electrical/geometric characterization of manufacturing variability sources is needed. A key form of dynamic variability is due to thermal effects during operation; this variation is on time scales of billions of clock cycles[14], and can strongly affect timing and noise phenomena. As power densities continue to rise, naive guardbanding against thermally induced variability will be costly. More accurate analyses and bounds for local thermal variation are needed to reduce overdesign. Another requirement is for statistical power analysis, especially with respect to leakage variability: due to exponential dependence of leakage on process parameters (gate length, oxide thickness and threshold voltage), high-end parts today exhibit 20× spreads of leakage power, versus performance spreads of ~35%. As this trend worsens, careful analysis and correlation to variability sources will be essential.
- *Reticle enhancement technology (RET)*—RET encompasses planarization of multilayer interconnect processes (necessitating layout density control with dummy fill) and deep-subwavelength optical lithography (necessitating optical proximity corrections (OPC) and layout of alternating-aperture phase-shifting masks (PSM)). RET is projected to be a growing source of manufacturing complexity, reflecting the increased difficulty of continuing the process roadmap. RET places a growing burden on DT with respect to layout design complexity, manufacturing handoff complexity, and manufacturing (mask) NRE cost. 1) With OPC and PSM, layout synthesis productivity is challenged by extremely complex, context-dependent design rules. Layout verification must adapt to regimes where "local design rules" no longer exist. Physical verification must accurately understand and model, for example, the RLC extraction impact of downstream dummy metal insertion in the post-tapeout layout database. 2) Handoff to

[14] *Other reliability-related forms of dynamic variability (threshold shift due to hot carrier-induced oxide breakdown, resistivity change due to electromigration, etc.) occur on time scales that are a million times larger.*

manufacturing becomes intolerable with indiscriminate application of RET[15], which explodes data volumes and mask write/inspection costs. RET insertion (and mask inspection) must understand that only certain critical device or interconnect dimensions are worth the expense of careful enforcement, and that some enforcement mechanisms are costlier to implement and verify than others. Hence, future design flows must pass detailed functional intent and simulation results forward into the physical verification and mask flows. Long-term supporting technology includes replacement of the outdated and inefficient GDSII Stream and MEBES data standards: a more fully bidirectional pipeline between DT and manufacturing must emerge, possibly in the context of electronic system "supply chain" infrastructure.

3. Silicon complexity, non-ideal device scaling, and power management—Operation at the lowest levels of abstraction requires detailed modeling, analysis and optimization with respect to many challenging physical effects.

- *Leakage and power management*—Leakage power is an extremely severe problem for both low-power and high-performance designs, since it is not only "useless" power consumption, but also unavoidable as device engineers continue to scale transistor performance. Buffering paradigms for critical global interconnects result in substantial leakage, since there is no stack effect in buffers, high total device width, and low-threshold devices (for high interconnect bandwidth). Gate leakage will likely continue to be problematic for circuit designers, as material fixes are historically slower to integrate versus aggressive scaling of existing techniques. Power-related noise issues include supply rail inductive noise, which results from lower supply voltages, along with larger currents stemming from increased power densities. This is exacerbated by synchronous operation and power-up / reset conditions, along with less aggressive scaling of bump counts and pitches. Supply rail design to manage voltage IR drop and current surges is required early in the design process, as is planning of inserted decoupling capacitance. Below 90 nm, even carefully designed supply rails may be overwhelmed by large switching currents. On the package side, reliability requires management of peak power so as not to continuously exceed package ratings for more than ~100 ms. Package and performance reliability both require control of temperature variation across the die. Hence, tools spanning algorithm development, logic synthesis, and timing/layout optimization must cooperate to manage both instantaneous and average power. As the System Drivers chapter points out, dynamic and standby power management for SOC-LP design faces gaps of 8× and 230×, respectively, by the end of the ITRS. Steadily increasing power budgets for MPUs demand new die-package thermal analyses, along with improved supporting analyses (parasitic extraction, logic activity, current flows). Library characterization, synthesis, and layout (including power distribution design) require substantial advances to deliver the roughly 5× power reduction available from fine-grain use of multiple thresholds and supplies in the same core. Tools must automatically synthesize structures that enable active thermal management via OS-mediated dynamic frequency and supply scaling, as well as potential longer-term techniques such as Peltier-type thermoelectric cooling.
- *Reliability and fault-tolerance*—Reliability criteria (hot-carrier effect, electromigration, joule self-heating, etc.) have been integrated into implementation flows via simple and transparent abstractions (e.g., upper bounds on gate load capacitance vs. output slew time). Currently, such "methodological" abstractions permit correctness by construction with little disruption of traditional tool flows. However, improved abstractions and analyses that reduce guardbanding (thus increasing extracted value per wafer) will be needed. 1) A near- and long-term issue is the loss of 100% correctness in manufactured or operating devices and interconnects. For example, the Design Test discussion below notes that traditional accelerated lifetime testing ("burn-in") paradigms become infeasible as decreasing supply voltages require exponentially increasing burn-in times, and higher power dissipation increases the cost of supplying power to burn-in ovens. Automatic logical and physical syntheses will be increasingly called upon to integrate fault-tolerance techniques (including hardware redundancy, reprogrammable interconnect, and reconfigurable control logic) operating under the control of appropriately synthesized and integrated on-chip self-test logic (compare to the discussion of self-checking cores in the Design Verification discussion). 2) New defect models are needed due to atomic-scale effects. For example, gate oxides are now too thin (with too few trap states) to exhibit catastrophic oxide breakdown; rather, currents simply increase over time and with oxide damage. Not only are novel screening criteria needed for what constitutes a "defect," but the dependence of acceptable defect density on particular circuit topologies must be understood.
- *Soft-error reliability*—Soft errors induced by extraterrestrial radiations (cosmic rays) or by materials (due to alpha particles) are no longer confined to SRAM bit cell upsets, and are a near-term discontinuity for reliability-driven DT. Beyond the 90 nm node, logic circuits and latches become increasingly vulnerable due to reduced Q_{crit} (critical charge). This vulnerability increases with scaling due to lower storage node capacitance and scaling of V_{dd}.

[15] *Up to 200+ GB uncompressed MEBES file sizes are anticipated for a single mask layer at the 90 nm node.*

Increasing soft error rate of latch circuits due to direct fails has a significant impact on system reliability. Additionally, with deeper pipelining in the processor microarchitecture, the number of logic stages between latches becomes smaller, increasing the probability of a single event upset making its way into a latch. An upset in static logic must propagate to the latch to cause an error; other circuit topologies such as dynamic logic can originate their own fails. Either scenario requires analysis solutions (performed as a separate characterization step) to accurately predict soft error rate for each cell, device type, and placement. In general, automated methods are needed to modify logical, circuit and physical design (by automatic introduction of error correction, sizing, etc.) to prevent or manage SEU without violating design constraints. 1) At the packaging level, system reliability and mitigation of soft errors will require lead-free C4s (area-array I/O) in future technology generations. In technologies with lead C4s, placement of critical circuits outside of the C4 influence region will help mitigate single event upsets due to alpha particles. 2) At the logic and circuit level, redundancy and parity could be introduced to protect the machine state, but this must be traded off against performance and power constraints. Techniques to increase Q_{crit}, such as selective increase of latch storage node capacitance, must also trade off against performance. 3) At the microarchitectural level, space and time redundancy may be applied to reduce soft-error impact and increase system reliability.

- *Modeling and analysis*—Increased silicon complexity and guardbanding lead to modeling and analysis challenges with respect to power/current and timing/noise. The main challenge is to support interconnect optimizations (signal, clock, power/ground) that mitigate coupling noise, delay uncertainty, etc. with minimum resource and performance cost. 1) To support power distribution design and current management, more accurate analyses of IR drop and active power dissipation are needed, supported by characterization tools and richer library modeling standards. Fast detailed thermal analyses may be required, e.g., as joule heating of interconnects combines with reduced thermal conductivity of low-permittivity dielectrics to create electromigration reliability risks.[16] 2) *Timing and signal integrity closure* depends on accurate analyses of parasitic capacitance and inductance in on-chip and package interconnects, interconnect crosstalk noise, coupling-induced delay uncertainty, and substrate noise injection. Estimations that drive future interconnect optimizations must account for signal and supply noise, thermal gradients and—in sub-100 nm nodes—reflections, EMI and substrate coupling. In this context, today's timing/noise library characterizations have big holes (such as in capturing process variation) with respect to ability to drive synthesis and support required analysis accuracy in nanometer processes. 3) More generally, nanometer implementation flows require "monotonically pessimistic" *families of analyses* that trade off CPU versus accuracy. For example, a known critical net may require slower and more accurate analysis in timing signoff, while faster and less accurate analysis is sufficient for filtering and prevention in the early stages of logic/layout synthesis. In the limit, analyses are approximated by efficiently evaluated and well-behaved "macromodels" that serve as synthesis optimization objectives.

4. Circuit design to fully exploit device technology innovation—As described in the PIDS roadmap, non-ideal scaling (notably the scaling of supplies faster than thresholds) results in higher gate and drain leakage currents, body effect (making pass gate logic less attractive), and loss of overdrive. In light of daunting power management challenges, past trade-offs of higher power, noise susceptibility, and unavailability of automated tools in return for speed become less attractive. DT must enable deployment of new alternatives to static CMOS that permit overall improvement in speed/power performance. Layout automation and physical verification (automated extraction of novel active and passive structures from layout) capabilities must be developed in step with circuit innovations. Such DT advances must be complementary to advances in device technology, as follows: 1) mitigation of gate insulator tunneling and breakdown integrity (which affects floating body devices, dynamic circuits, etc.) requires a high-permittivity gate insulator, and 2) use of substrate biasing (essential for dynamic V_t adjustment to address the power management gap) requires improved body contact technology.[17]

- *Support for new circuit families that address power and performance challenges*—As power management becomes centered at architecture, OS and application levels, several circuit trends require DT support. 1) Use of locally asynchronous, globally synchronous architectures will make *self-sufficient circuits* such as clock-delayed domino or delayed-reset domino more popular (although static CMOS will remain prevalent). 2) Operation at extremely low voltage, possibly entirely subthreshold, will become more common as device performance requirements are alleviated by use of parallelism. 3) Increased compiler- and OS-based control of such parameters as body bias, clock,

[16] *Similar thermal challenges arise with silicon-on-insulator processes; these constitute sources of dynamic variability that affect timing and noise analyses, as discussed with the next challenge.*

[17] *Expectations for advances in device technology are given in the PIDS Chapter. Note that such advances are not panaceas: all of the DT requirements presented here still remain.*

and supply rails means that physical responses must be increasingly modeled at the behavioral level in order to support architecture-level management.

- *Implementation tools for SOI*—Silicon-on-insulator (SOI) technology offers faster switching due to lower junction capacitances, and improved isolation that eliminates digital-to-analog substrate noise but poses severe challenges for analog and mixed-signal circuits. Many DT issues must be solved before designers can switch transparently from bulk MOS devices. The floating body effect and self-heating requires history-aware analyses: worst-case guardbanding would otherwise leave much of the SOI advantage on the table. These analyses include timing (leading to more complex libraries) and static power (V_{th}, and hence leakage, fluctuates with capacitive coupling and impact ionization).[18] Reliability mechanisms must be carefully elucidated, as alpha particle SEU is reduced and latchup is eliminated, but self-heating and ESD protection becomes more problematic due to increased thermal resistance of the buried oxide layer. Lowered device junction capacitance in SOI, while aiding device switching speed, reduces built-in V_{dd}-GND decoupling capacitance; hence, maintaining power supply integrity (on-chip decoupling roughly 10× switching capacitance) may require management of explicit decoupling capacitance starting from early floorplanning stages. Fully depleted SOI (FD-SOI) has manufacturability concerns but may receive increased attention if solutions to the above problems are not found.

DESIGN VERIFICATION

Design verification is the task of establishing that a given design accurately implements the intended behavior. The prevailing view by much of the semiconductor industry is that design verification plays a relatively minor supporting role to the design process. (Possibly, such a view is reflected in the very structure of this document.) Current reality, however, is markedly different. Verification has become the dominant cost in the design process. On 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 unfortunate 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 states of the system that must be verified, can grow exponentially in the size of the design, producing a doubly exponential blow-up.[19] 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 formal ones. Whether any particular formal verification technique will succeed is debatable, but the overall shift is unavoidable. One should not attempt to verify the functionality of a system design by repeatedly building models, simulating them on an ad hoc selection of vectors, and then patching any bugs that happen to be noticed—but this is exactly the methodology used today. A trial-and-error verification methodology based on simulation is inherently slow and unscalable, because of the aforementioned doubly exponential growth in functional complexity.[20] Technological progress depends on developing rigorous and efficient methods for analyzing a design (the bulk of formal and semi-formal verification research today), and eventually codifying reliable, predictable engineering practices that simplify or even obviate (via forms of "correctness by construction") many verification challenges. Table 17 summarizes the main challenges in design verification. The remainder of this section provides explanatory comments.

---

18 Floating body effects can be eliminated via dedicated body contacts in partially depleted SOI (PD-SOI) at the expense of area (libraries may proliferate if a particular cell is needed both with and without body contacts, and/or with a variety of types of contacts), or by using fully-depleted SOI. Body-contacted PD-SOI has implications for modeling and simulation infrastructure, since a 4-terminal MOSFET model is needed.

[19] There are many variations that arrive at 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 a similar computation. If verifying correct behavior requires reasoning over sequences of states, the computational challenge can be even worse.

[20] This argument applies to hardware emulation as well as to conventional simulation. Hardware emulation buys several orders of magnitude improvement in "simulation" speed, providing an invaluable aid to verification. It also often allows an earlier start to system integration and software development. An emulation system, however, still runs vectors one-at-a-time, so it cannot possibly provide a scalable, long-term solution to the verification problem.

Table 17 Design Verification Challenges

Challenges 50 nm/Through 2009	*Summary Of Issues*
Increased verification capacity	S, P, A—Verification complexity is double-exponential in design size S, P, A—Need high coverage as well as capacity to handle large designs S, P, A—Semi-formal techniques
Robust verification tools	S, P, A—Highly unpredictable verification algorithms demand improved heuristics and characterizations of problem difficulty
Verification metrics	S, P, A—Behavior coverage S, P, A—Realistic bug models are needed, along with algorithms to determine bug coverage
Software verification	S—Software intrinsically more difficult to verify S—Traditional software verification techniques inapplicable S—Integrated hardware/software systems S—Design for verifiability
Verification reuse	S—Must allow reuse of verification of IP blocks S—Specify abstract behavior of IP blocks S—Specify environmental constraints of IP blocks S—Hierarchical verification algorithms
MPU verification methodology	P—Different cost-benefit trade-off (higher cost acceptable) P—Need exceptionally high capacity P—Must be very predictable due to long design cycle and pipelined development teams
MPU design-for-verifiability	P—Will be necessary sooner than for other system drivers; specialized techniques are likely
Greater concurrency	P—Far more concurrency in new processors greatly increases verification complexity
Anything is progress for AMS	A—Extremely primitive state-of-the-art forces difficult hybrid-systems issues into near term

This table summarizes challenges to the design process advances implied by the above four trends. Each challenge is labeled with a list of the most relevant system drivers (S—system on chip, P—microprocessor, A—analog/mixed-signal, M—memory).

Table 17 Design Verification Challenges (continued)

Additional Challenges <50 nm/Beyond 2009	
Design for verifiability	S, P, A—New methodology needed
	S, P, A—Characterize and minimize performance and area impact
Higher levels of abstraction	S, P, A—New algorithms needed
	S, P, A—Complexity of designs enabled by higher-level design
	S, P, A—Equivalence checking vs. RTL
Human factors in specification	S, P, A—Specifications of correctness will become unmanageable
	S, P, A—Need to understand what kinds of specifications are most understandable
	S, P, A—Need to consider how to make specifications modular and modifiable
Verification of non-digital systems	S, P, A—Hybrid systems verification for analog effects
	S, P, A—Hybrid systems verification for analog properties
	S, P, A—Verification of probabilistic systems
Heterogeneous systems	A, S—How to model, analyze, and verify MEMS, EO devices, and electro-biological devices

This table summarizes challenges to the design process advances implied by the above four trends. Each challenge is labeled with a list of the most relevant system drivers (S—system on chip, P—microprocessor, A—analog/mixed-signal, M—memory).

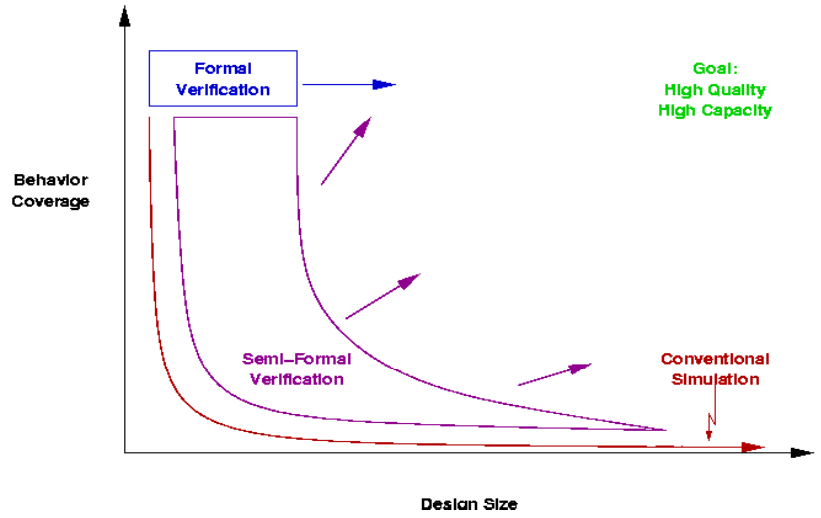
DESIGN VERIFICATION CHALLENGES—NEAR TERM (>50 nm)

Many of the most important challenges for verification are relevant to most or all system drivers. In the near-term, eight primary issues (1–8 below) center around 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, four primary issues (9–12 below) center around 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. In addition, the following near term challenges are especially important for 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.

1. Capacity—Current formal verification tools cannot reliably handle large, complex designs. Simulation-based tools can simulate arbitrarily complex designs, but provide vanishingly small coverage despite exploding simulation times. The single most important challenge for verification is how to provide high quality verification coverage for large, complex designs. Figure 19 illustrates the current tool landscape.[21] Conventional simulation can provide high coverage of a small design, or extremely poor coverage of a large design, but not high coverage of a large design. Formal verification provides complete coverage of the possible behaviors of the design being verified, but cannot currently handle large designs. Semi-formal verification attempts to blend formal and simulation-based techniques, sacrificing coverage to gain capacity. The challenge is to move toward the upper right corner of the diagram.

[21] *"Conventional simulation" here 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.*

Figure 19 Current Verification Tool Landscape



2. Robustness—Figure 19 gives the impression that certain verification methods work reliably for designs of a certain size. This impression is erroneous. In order to cope with the complexity of the verification problem, verification algorithms depend on highly temperamental heuristics. For any given pairing of design and verification algorithm, 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 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 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.

3. Verification metrics—A related near-term verification challenge is the need to quantify the quality of a verification effort. In particular, a meaningful notion of coverage is needed. Two kinds of coverage must be distinguished: coverage of possible behaviors of the design and coverage of possible bugs.

- Behavior coverage is not well defined, and in any case applies only to simulation-based and semi-formal verification. (Any formal technique explores 100% of the possible behaviors of the system, yielding 100% behavior coverage.). Behavior coverage metrics generally indicate that conventional simulation can evaluate only a miniscule fraction of the possible behaviors of the design. In practice, simulation is far more effective at catching design errors than these metrics would indicate.
- Bug coverage is the important measure of verification coverage, but it is currently poorly understood. It is roughly analogous to fault coverage in testing, but unfortunately, there is no evidence that design errors in any way resemble manufacturing defects. The problem is to determine what fraction of possible bugs (faults) have been checked based on some combination of behavior coverage and the specification being verified. If the specification being verified is incomplete, the design under verification might pass even formal verification, yet still fail to behave as desired. The challenge is to create a meaningful bug-model (analogous to fault model, but requiring a deep understanding of the design errors that arise in practice) and the algorithms to compute bug-coverage for a given (design, specification, verification-run) triple.

4. Software—Because much of the functionality of SOCs will be defined by software, a major SOC verification challenge is how to verify software and hardware/software systems. 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 (generally modeled as being infinite, making verification undecidable). Classical formal techniques for software verification are too labor-intensive to be widely applicable for SOC. 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, as well as an understanding of design-for-verifiability as applied to software.

5. Reuse—Predesigned 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. 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.

6. Specialized verification methodology—MPU designs will always strain the capacity of verification tools. However, the high economic return of a successful MPU allows different, more labor-intensive methodologies. For example, semi-automated theorem-proving techniques will likely remain too slow and too expensive in the foreseeable future for widespread use on quick-time-to-market SOC designs; they are, however, being used effectively for certain difficult verification tasks on high-volume MPU designs. The key challenges are capacity and predictability. Predictability is more important for MPU verification than for other system drivers because the design cycle is very long and because multiple design teams for successive product generations are often pipelined. The lengthy design cycle, planned far in advance, provides time for a lengthy verification process, but only if the verification process does not balloon unexpectedly. How can one best deploy additional resources to handle the complexity of MPU verification? How can one deploy additional resources to reduce the variance in verification time?

7. Specialized design-for-verifiability—Similarly, the complexity of MPU designs will likely require design-for-verifiability sooner than other system drivers, and the specialized nature of MPU designs means that domain-specific design-for-verifiability techniques should be possible. For example, there is preliminary work being done on self-checking processors, in which a small watchdog processor verifies the correct execution of the main processor. The challenge will be to develop design-for-verifiability techniques for MPU designs that are effective in reducing verification cost, while imposing minimal area and performance penalties.

8. 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. The challenge is to develop techniques to handle these new forms of concurrency.

DESIGN VERIFICATION CHALLENGES—LONG TERM (<50 nm)

1. 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.

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

3. Human factors in specification—A continuing challenge for design verification is how to specify the desired behavior of a design. 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. In addition, designers will have to be trained to use these languages and to become more disciplined about writing specifications.

4. Broadening the scope of formal verification—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 systems. At the low-level, silicon complexity dictates that the clean, digital abstraction of a VLSI system will become 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 the long term, formal techniques will be needed to verify these issues at the boundary of analog and digital, treating them as hybrid systems.[22] Similarly, at the highest-levels of design, system complexity dictates that future verification tasks will likely require specifying and verifying analog and probabilistic behaviors (such as quality of service guarantees in a network processor). Thus, there will be the challenges of hybrid systems and probabilistic verification.

DESIGN TEST

Nanometer process technology, increasing clock rate, and increasing SOC and SIP integrations present severe challenges to design for test. 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) pin electronics. 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, sensitivities) from analyses in logical-circuit-physical implementation, and from design verification, must be reused during design test. Across industry segments as varied as memory, SOC, AMS and MPU, three high-level test challenges demand significant expansion of on-die DFT, BIST and testability features, and better integration to a pre-planned manufacturing test equipment set during the chip planning phase of development.

1. High speed device interfaces—Component I/O speed has become as important to system performance as core clock frequency or transistor and architectural performance. 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. I/O DFT and BIST methods such as loopback, jitter measurement, edge detection, pseudo-random pattern generation, etc. will become standard techniques that are required for verification and manufacturing test of these new I/O speeds and architectures.

2. Highly integrated SOC and SIP designs and components—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, SRAM and DRAM have been commonly integrated in recent years, but now analog, mixed signal, and non-volatile flash 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. Direct access DFT (DAT) to access embedded block or macro I/Os enables test database reuse, but can require additional test insertions due to test database size. Testing of embedded blocks with DAT may also entail an order 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 AMS 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 (IEEE P1500), 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 and analog 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.

[22] *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.*

In many cases, the high cost of integrating different circuit types into the same process technology (notably through additional lithography steps) can be and are being circumvented with SIP or "system in package" integrations such as stacked dice. SIPs allow quick integration of a wider range of silicon technologies than SOCs and also can be done with shorter process and product development cycles, sometimes re-using existing die designs not originally planned for SIP. Testing end form factor SIPs present the same technical and economic challenges of SOCs, i.e. access to internal ports and integration of economically feasible test equipment set and manufacturing flow to achieve adequate component quality. SIP product cycles are typically shorter and more restrictive in terms of DFT and product "interceptability" for testability needs.

3. Reliability screens running out of gas—Manufacturing test has historically not only measured device performance and functionality, but also performed the required business task of identification and segregation of latent reliability defects. More specifically, these are the defect driven infant mortality reliability failures, not intrinsic device or thin film reliabilities. Dynamic burn-in, I_{ddq}, and above V_{dd} voltage stress during test are three essential methodologies that are all severely limited by rapidly increasing leakage currents of advanced silicon technologies. At the same time, the ability to use above-nominal V_{dd} is being reduced. As the ratio of stress to nominal V_{dd} decreases with each technology node, the acceleration, identification, and screening capabilities of both burn-in and "on ATE" voltage stressing become extremely limited. At 130 nm and 90 nm, thermal runaway limits the use of, and drives non-linear cost increases for, burn-in of high-end products such as microprocessors. At the same time, V_{dd} voltage stress and even advanced I_{ddq} techniques such as I_{ddq} delta are extremely limited. In the near term, significant cost increases could result from the overkill (signal to noise) yield impact of extending our current approaches just to keep pace with market reliability requirements. Fundamentally new long-term solutions are required and may include significant on-die hardware for stressing or special reliability measurements. Eventually, a broader development and deployment of on-die self-test, self-diagnosis, self-correction, and self-repair will be required to meet cost and device reliability goals. Continued growth of the fabless device industry adds another dimension of orthogonal business variables to the space of potential challenges and integrated solutions for this area—for example, to the extent that foundries can inform and support integrated test strategies, they will become preferred vendors. The resulting Design Test Challenges are summarized in Table 18. The remainder of this section provides explanatory comments.

Table 18 Design Test Challenges

Challenges 50 nm/Through 2009	*Summary Of Issues*
Effective speed test with increasing core frequencies and widespread proliferation of multi-GHz serial I/O protocols	P, S—Continuation (avoidance) of at-speed functional test with increased clock frequencies P, S—At-speed structure test with increased clock frequencies P, S, A—DFT, test and on-chip measurement techniques for multi-gigahertz serial I/Os and non-deterministic interfaces
Capacity gap between DFT/test generation/fault grading tools and design complexity	P, S—Better EDA tools for advanced (open, delay, etc.) fault models P, S—DFT to enable low-cost ATE P, S—Non-intrusive logic BIST (including advanced fault models) A—AMS DFT/BIST, especially at beyond-baseband frequencies
Quality and yield impact due to test process and diagnostic limitations	P, S—Power and thermal management during test P, S—Fault diagnosis and design for diagnosability S—Yield improvement and failure analysis tools and methods All—Increasing difficulty to fault isolate and root cause yield limiting defects
Signal integrity testability and new fault models	P, S—Signal integrity (noise, interference, capacitive/inductive coupling, etc.) testability A—Fault models for analog (parametric) failures
SOC and SIP test	S—Integration of SOC test methods into chip-level DFT S—Integration of multiple fabric-specific test methodologies in cost-effective manufacturing flows A—DFT, BIST and test methods compatible with core-based SOC environment and constraints M—Embedded memory (DRAM, SRAM, Flash) built-in self-diagnosis and self-repair All—Test reuse in context of higher integration
Additional Challenges <50 nm/Beyond 2009	
Integrated self-testing for heterogeneous SOCs and SIPs	A—Test of multi-gigahertz RF front ends on chip S—Use of on-chip programmable resources for SOC and SIP self-test S, A—Dependence on self-test solutions for SOC (including RF and analog) A— (Analog) signal integrity test issues caused by interference from digital to analog circuitry S—Test methods for heterogeneous SOC and SIP including MEMS and EO components
Diagnosis, reliability screens, and yield improvement	A—Diagnosis and failure analysis for AMS parts P, S—Electrical automated fault isolation techniques below gate level P, S—Design for efficient and effective burn-in to screen out latent defects P, S—Quality and yield impact due to test equipment limits P, S—New timing-related fault models for defects/noise
Fault tolerance and on-line testing	P, S—DFT and fault tolerant design for logic soft errors S—Logic self-repair using on-chip reconfigurability S—System-level on-line testing

This table summarizes challenges to the design process advances implied by the above four trends. Each challenge is labeled with a list of the most relevant system drivers (S—system on chip, P—microprocessor, A—analog/mixed-signal, M—memory).

DESIGN TEST CHALLENGES—NEAR TERM (>50 nm)

1. Effective speed test with increasing core frequencies and widespread proliferations of GHz rate serial I/O protocols

- *Continuation (avoidance) of at-speed functional test with increased clock frequencies for MPU (ASIC/SOC)*—For MPU, DFT, and BIST are needed for the required at-speed functional testing that is performed on ATEs with data rates and edge placement accuracies less than MPU specs. By contrast, ASIC/SOC must continue to avoid the high

human resource and capital costs of at speed functional testing through a combination of additional fault types, improved fault coverage (tools), DFT, and increasing BIST.

- *At-speed structure test with increased clock frequencies.* DFT must continue to enable transition and path delay scan testing on slower-than-device ATE and within the boundary conditions of increasingly complex clocking schemes.
- *Test and on-chip measurement techniques for multi-gigahertz serial I/O protocols and non-deterministic interfaces*—Protocols such as Gigabit Ethernet, Firewire, SATA, etc. are exploding in product plans from ASICs to SOCs to MPUs as system performance improvements are being delivered by making the "pipes" go faster rather than by moving forward with major new computing architectural improvements. Such applications require very expensive test systems capable of generating and receiving differential signals up to several GHz, with voltage swings as low as 100 mV. The excessive cost makes existing test solutions impossible for volume production, and technically challenging even for critical and expensive engineering design verification ATEs. Today's component ATEs are typically common-clock machines unsuited for high-volume manufacturing test of rapidly evolving device I/O speeds and signaling schemes: interactive, packet based protocols directly conflict with the deterministic digital ATE architecture paradigm of the last 20 years. New I/O test methodologies, DFT, and on-chip measurement techniques for serial communications buses and components are needed. Tester interface hardware will include broad custom silicon applications to interface numerous interactive protocols per DUT to deterministic ATEs. For very high-speed serial ports, a variety of new parameters including transmitter jitter, crosstalk, signal loss, ability of receiver's PLL to lock onto signal streams, receiver jitter tolerance, etc. increase the test challenges and must be comprehended and addressed with numerous on die testability features and DFT schemes. Loopback, compensation, and other on die instrumentation techniques become mandatory.

2. Capacity gap between DFT/Test generation/fault grading tools and design complexity—Design databases of current microprocessors, SOCs, and ASICs exceed data size limits of industry EDA tools. This forces complex patch schemes and partitioning, and makes full chip fault simulation and scan ATPG difficult to impossible, even for just stuck faults. Simulation of full chip test stimulus/response data sets prior to tapeout is a fundamental requirement in the design flow, and slows the increase in DFT and device bug rates, as well as revision cycles. The alternative is vector validation on actual silicon, which adds silicon-processing time to the design iteration loop, and dramatically increases the engineering resource needed to get the test content correctly out of the design process (furthermore, the proportion of test content that goes through rework increases by tens of percent).

- *Better EDA tools for advanced fault models (open, delay, etc. faults)*—Pragmatically usable fault simulation and ATPG for realistic fault types (> stuck faults) are needed as complex chips push design and transistor performance at sub-100 nm nodes.
- *DFT to enable low-cost ATE*—DFT is needed to enable testing of all parametric aspects and logic on ATEs with reduced signal pin counts, limited pin electronics tolerances, reduced test data rates, and fewer features (for example, no APG). This requirement includes I/O, cache, and logic DFT.
- *Non-intrusive logic BIST (including advanced fault models)*—New BIST schemes are needed with better tolerances and contingencies for non-fully decoded logic, limited contention tri-state buses, and other design elements unique to the MPU context. To expand test coverage and content without significant increase in development effort and manufacturing cost, logic BIST must be extended to cover realistic fault types, just as fault modeling, fault simulation, and ATPG extended to cover beyond stuck at and transition faults (to directly cover modeled opens and bridges and other fault types).

3. Quality and yield impact due to test process and diagnostic limitations

- *Power and thermal management during test*—Power and thermal control during manufacturing test may be more constrained than end use system needs or lack end use integrated controls and management (heat sinks, air flow, feedback bios/system integration, etc.). Required levels of control demand new techniques such as active feedback and design for di/dt mitigation.
- *Fault diagnosis and design for diagnosability*—Tools, schemes, and rules are needed to enable automatic fault diagnosis (isolation and identification of failing gates or circuit) that is usable for all fault types (not just stuck and transition faults).
- *Yield improvement and failure analysis tools and methods*—High costs of DSM processes combined with short lifetime and short time to market severely challenge the failure analysis process. These trends demand that process random abnormalities (disturbances/defects) be quickly located and characterized to efficiently understand the failure mechanisms, provide corrective actions and improve yield. Automated fault isolation to lower than gate level abstraction (i.e., to failing node or interconnect) is needed for logic and memory circuits as continued yield

improvement learning curve at each technology node is faced with lower root cause results per attempted failure analysis as each technology node further beyond the optical wavelength.

4. Signal integrity testability and new fault models

- *Signal integrity testability (noise/interference, capacitive/inductive coupling, etc)*—While geometries are shrinking, defects are not shrinking in proportion. Increased wiring levels and wire delays demand new fault models, as signal integrity verification problems become test problems (e.g., distributed delay variations, crosstalk-induced delay and logic errors, excessive voltage drop and/or swing on power nets, and substrate and thermal noise). Testing must target such parametric performance failures. New classes of noise faults caused by parametric variations must be properly modeled to levels of abstraction higher than the electrical, circuit, and transistor levels for purposes of fault simulation, ATPG and BIST.
- *Fault models for analog (parametric) failures*—Test evaluation, design-for-test, and self-test solutions for analog blocks and converters remain very limited. Most existing solutions are primarily based on functional testing, and have frequency limitations. There are no standard figures of merit to evaluate a given analog test program. 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 mixed-signal design, tools are needed that can minimize the computation complexity of fault simulation while maintaining high simulation accuracy.

5. SOC and SIP test

- *Integration of SOC test methods into chip-level DFT*—Multiple insertions using a cheaper, dedicated tester per integrated block or circuit type (analog, RAM, flash, logic, MEMS, etc.) can be traded off against integration of all measurement hardware and software onto a single platform and using a single insertion. Additional DFT beyond the previous standalone blocks or pre-existing designs is essential to provide access to and across embedded blocks, or for particular block and circuit types. The impact of maintaining a longer and more complex manufacturing test flow (with multiple platforms and steps) on inventory and wafers in process must also be considered. Historically, integration of additional hardware complexity has resulted in non-linear increase of overall ATE development and capital cost. Amortizing the additional capital cost of a "do all" machine is even more problematic in that different blocks are not tested simultaneously. Thus, circuit-specific test hardware is often idle during the majority of the manufacturing use duty cycle, waiting for other tests to be executed on other parts of the ATE hardware. At a minimum, consistent block level access for embedded blocks (IEEE P1500 or equivalent) is required. In the long run, to avoid losing economies of SOC, SIP, and silicon integration, non-linear manufacturing test costs will drive as many blocks as possible to standardized DFT and more extensive BIST solutions. A design engineering focus in this space will provide significant ROI in terms of SOC/SIP product cost reduction, but must be addressed during the earliest (architectural) phases of product planning.
- *Integration of multiple fabric-specific test methodologies*—SOC and SIP integration of blocks from different design/tool flows and legacies will severely challenge DFT, DFT integration, ATPG, test database verification, and back-end vector processing tools and flows. As in the MPU context, the ability to simulate full chip test stimulus/response data sets prior to tapeout is a fundamental requirement.
- *DFT, BIST and test methods compatible with core-based SOC environment and constraints*—Self-test techniques for complementary components must be developed for analog/RF DFT. Within the context of transceivers (RF, optical, ADC/DAC, I/O, etc.) the sender and receiver can often be used to test each other. For example, a DSP-based (digital) technique for both stimulus generation and response analysis uses on-chip DAC and ADC along with embedded processor or DSP cores to generate signals and analyze responses. Potential advantages include low hardware overhead, low performance intrusion, higher flexibility and use of digital ATE. This technology must be advanced to handle devices at higher frequencies.
- *Embedded memory (DRAM, SDRAM, Flash, etc.) built-in self-diagnosis and self-repair*—This capability for embedded memory requires significant on-die features and logic, including self-test capabilities, redundant elements (extra columns, rows, or blocks), redundancy analysis, and replacement logic and hardware.
- *Test reuse*—SOC design productivity requires test IP or test data set reuse on integrated products, within ATE boundary conditions (constraints) on test data size and expected equivalent test time. For example, test data sets of an integrated product may drive additional insertions because they do not fit on existing ATE. DFT modes for embedded blocks should provide equivalent or faster test times than those for standalone versions of such blocks.

DESIGN TEST CHALLENGES—LONG TERM (<50 nm)

1. Integrated self-testing for heterogeneous SOCs and SIPs

- *Test of multi-gigahertz RF front ends on chip*—The manufacturing test process of RF components requires significant instrumentation and human effort. Breakthroughs are needed to improve test repeatability, reduce test time, increase the accuracy of the measurement, and lower the cost of troubleshooting.
- *Use of on-chip programmable resources for SOC self-test*—A possible means of reducing test cost is the extension of self–test capabilities beyond STUMPS style logic BIST, for example, by using on die MPU, processing, or other programmable resources. This is a long-term means of maintaining adequate quality levels while addressing increased need to test for process and parametric variations, as well as delay and timing specific failure modes.
- *Dependence on self-test solutions for SOC (including RF and analog)*—BIST or ATE-assisted self-test must become pervasive.
- *(Analog) signal integrity test issues caused by interference from digital to analog circuitry*—Interference generated by digital circuitry on the same chip can significantly disrupt the performance of sensitive analog circuitry. In order to analyze such effects, it is necessary to test the analog circuitry when the digital circuitry is running at full speed. Methodologies must be developed to avoid crosstalk in design, and to support crosstalk testing during operation at 4 Gbit/sec rates and beyond.
- *Test methods for heterogeneous SOC, including MEMS and electro-optical components*—All new circuit types integrated into SOCs must fit within existing design and DFT frameworks, and eventually expand self-test capabilities. For the same (mostly economic) reasons, analogous test methods are needed in the near term for integrated RF, mixed-signal, flash, etc.

2. Diagnosis, reliability screens, and yield improvement

- *Diagnosis and failure analysis for analog/mixed-signal parts*—These are largely achieved by manual effort today. Reduction of process debugging time and cost demands new fault diagnosis and failure analysis techniques, especially for analog/mixed signal components. New tools and techniques are needed to correlate data from several sources (including design data, process data and test data) with appropriate fault models and metrics. The diagnostic methods should also be compatible with emerging DFT/BIST methods.
- *Design for efficient and effective burn-in to screen out latent defects*—For acceptable control of latent defects, burn-in thermal runaway should be mitigated via design-for-burn-in or new alternate schemes (such as self-repair).
- *Quality and yield impact due to test equipment limits*—ATE and test-related metrology instrumentation will have tolerances in the range of the actual measured parameters. Without fundamentally new approaches, signal to noise and yield overkill will reach unacceptable levels.
- *New timing-related fault models for defects/noise in nanometer technologies*—Parametric variation and defect fault behaviors become more uniquely observable at the timing level of abstraction. These must be modeled and built into fault simulation, ATPG, DFT, and BIST tools and methodologies.

3. Fault tolerance and on-line testing

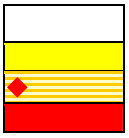
- *DFT and fault tolerant design for logic soft errors*—Circuit and architectural solutions are needed for soft error mitigation, particularly for logic, where no or limited solutions are currently deployed or planned.
- *Logic self-repair using on-chip reconfigurability*—Higher system-level reliability and mitigation of increased effects of soft errors require significant development effort (both new methods and integration of known techniques) in self test, self-diagnosis, self-correction, and self repair, along with on-die logic and circuits.
- *System-level on-line testing*—As SOC sub-blocks move to structural test with circuit-specific self-test methods, overall functionality testing will likely shift to holistic applications-based testers and testing, as opposed to dedicated functional testing on ATEs. Two factors will drive this. First, there is the already prohibitive engineering development cost of developing deterministic and exponentially growing functional test data sets. Second, system quality and reliability expectations for ever more complex system platforms will be most effectively met through applications based testing of the SOCs *in situ,* i.e., in the same systems and ways used by customers in applications.

ADDITIONAL DESIGN TECHNOLOGY REQUIREMENTS

Table 19 Additional Design Technology Requirements

Year of Production	*2003*	*2004*	*2005*	*2006*	*2007*	*2008*	*2009*	*2012*	*2015*	*2018*	*Driver*
Technology Node		*hp90*			*hp65*						
DRAM ½ Pitch (nm)	*100*	*90*	*80*	*70*	*65*	*57*	*50*	*35*	*25*	*18*	
MPU/ASIC ½ Pitch (nm)	*107*	*90*	*80*	*70*	*65*	*57*	*50*	*35*	*25*	*18*	
MPU Printed Gate Length (nm)	*65*	*53*	*45*	*40*	*35*	*32*	*28*	*20*	*15*	*10*	
MPU Physical Gate Length (nm)	*45*	*37*	*32*	*28*	*25*	*22*	*20*	*14*	*10*	*7*	
SOC new design cycle (months)	**12**	**12**	**12**	**12**	**12**	**12**	**11**	**11**	**10**	**9**	SOC
SOC logic Mtx per designer-year (10-person team)	**1.9**	**2.5**	**3.3**	**4.3**	**5.4**	**7.4**	**10.6**	**24.6**	**73.4**	**113**	SOC
SOC dynamic power reduction beyond scaling (X)	**0**	**0.1**	**0.1**	**0.2**	**0.2**	**0.2**	**0.2**	**6**	**4.7**	**8.1**	SOC
SOC standby power reduction beyond scaling (X)	**0.37**	**1.4**	**2.4**	**3.4**	**5.1**	**6.4**	**8.73**	**18.8**	**44.4**	**232**	SOC
%Test covered by BIST	**20**	**20**	**25**	**30**	**35**	**40**	**45**	**60**	**75**	**90**	MPU, SOC

Mtx—Million transistors



CROSS-CUT TWG ISSUES

MODELING AND SIMULATION

One of the key problems which 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: DT COST AND VALUE

As Figure 20 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 20). 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.

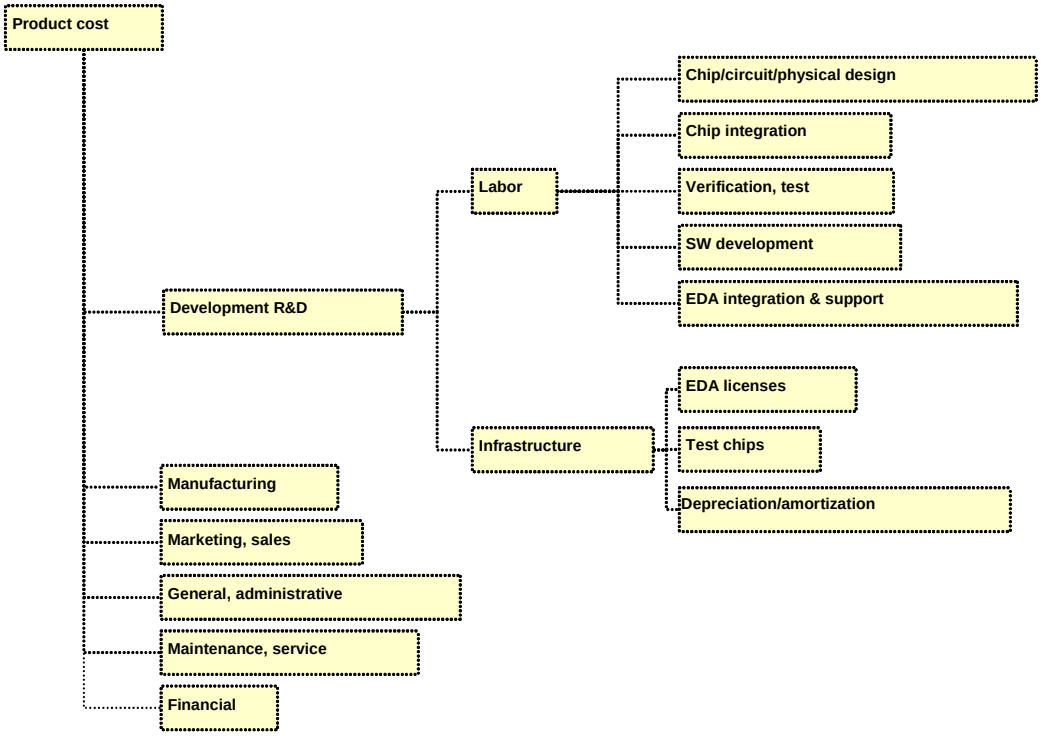


Figure 20 Simplified Electronic Product Development Cost Model

In Figure 20, 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 20, 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 2003. Specifically, the number of designers per million gates (inverse of productivity) went from 250 in the year 1990 to 8 in 2003. 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).

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 20 Design Technology Improvements and Impact on Designer Productivity

DT Improvement	*Year*	*Productivity Delta*	*Productivity (Gates/Desn-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 GDS II through IC place and route.**
Intelligent testbench	**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**
TOTAL		**+15,000%**			

APPROXIMATED TOTAL DESIGN COST

Figure 13 quantifies the impact of the DT innovations on design cost for the low-power System-on-Chip (SOC-LP) PDA driver defined in the System Drivers chapter. The SOC-LP has 4.9M logic gates in 2003, implying an SOC PDA design

cost (designers + tools) of $20.1M. Without the six major DT innovations that occurred between 1993 and 2003, the design cost for the same SOC in 2003 would be approximately $629.8M. 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 Dataquest estimate for the same year.[23]

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