INTERNATIONAL TECHNOLOGY ROADMAP FOR SEMICONDUCTORS

2001 EDITION

DESIGN

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 is the link that 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.

For the 2001 ITRS, the Design ITWG has developed the new System Drivers Chapter, along with models for clock frequency, layout density, power dissipation, etc. in support of the Overall Roadmap Technology Characteristics. New analyses of design cost and design productivity have been introduced into this Design Chapter. DT challenges and needs have been mapped, where possible, to the 2001 ITRS 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 2001 is this: *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 (NRE) costs are just reaching 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.

It is understood that without foundry amortization, the semiconductor investment cycle is at risk. Indeed, “fill the fab” has been the rallying cry for DT. It is also understood from previous ITRS editions that there is a *design productivity gap*: the number of available transistors grows faster than the ability to design them meaningfully. Yet, investment in process technology has by far dominated investment in design technology. The good news is that enabling progress in DT continues to be made. Figure 15 shows that estimated design cost of the low-power SOC PDA defined in the System Drivers Chapter is approximately $15M in 2001, versus $342M had DT innovations between 1993 and 2001 not occurred (details of the analysis are given in the Appendix). The bad news: software now routinely accounts for 80% of embedded-systems development cost; test cost has grown exponentially relative to manufacturing cost; verification engineers are twice as numerous as design engineers on microprocessor project teams; etc. *In 2001, many design technology gaps are crises.*

This Chapter first discusses the DT *complexity challenge*, which is comprised of *silicon complexity* and *system complexity*. We then describe how specific *methodology precepts* lead to an overarching vision of DT’s future design system architecture and functionality. The bulk of the Chapter then sets out detailed challenges for a traditional landscape of DT areas: Design Process, System-Level Design, Logical/Circuit/Physical Design, Design Verification, and Design Test. As appropriate, detailed challenges are mapped to the System Drivers (MPU, AMS, SOC) defined in the preceding chapter. The detailed challenges are prefaced by five crosscutting challenges (alternatively, “motifs”) that permeate all DT areas: Productivity, Power, Manufacturing Integration, Interference, and Error-Tolerance. Particularly for the detailed challenges, the discussion is 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.

Before continuing, we observe that roadmapping of DT is different from roadmapping of manufacturing technology. Manufacturing technology seeks to *realize* (e.g., a target tolerance) and faces *limits* imposed by physical laws and material properties. In contrast, DT seeks to *optimize* (e.g., 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 tradeoffs among multiple criteria such as density, speed, power, testability, or turnaround time. Evaluation of DT quality is thus context-sensitive, and dependent on particular methodologies or design instances. Furthermore, 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.

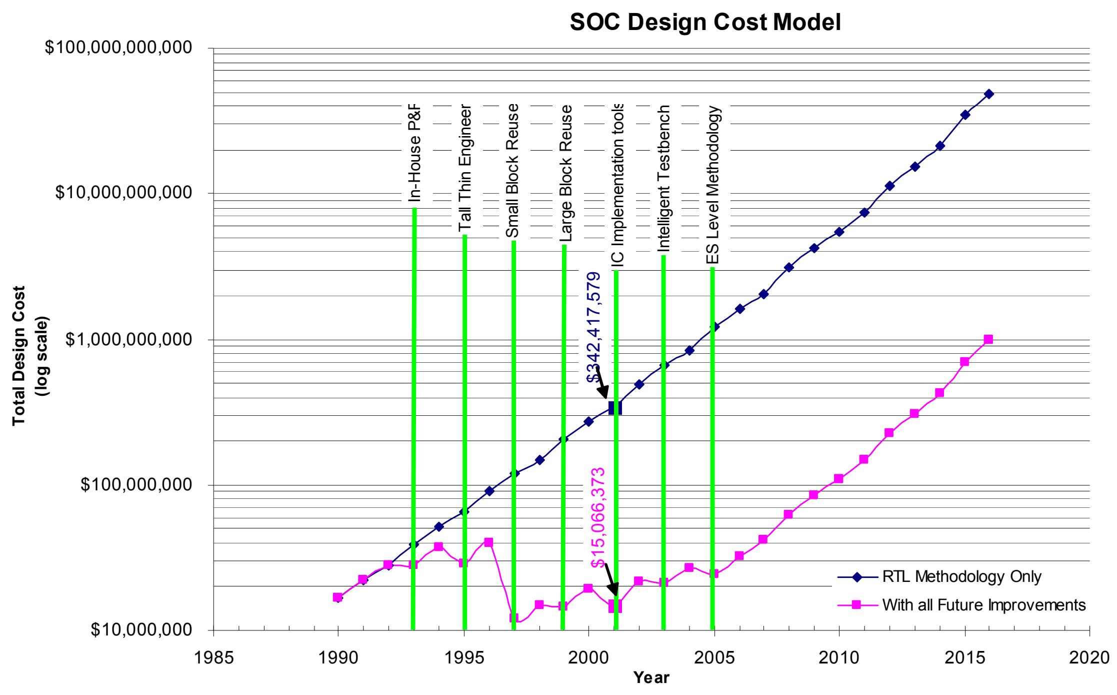

Figure 15 Impact of Design Technology on System Implementation Cost.

THE COMPLEXITY CHALLENGE

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. Previously ignorable phenomena (implied challenges) have greater 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 equipment limits* (statistical process modeling, library characterization);
- *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: (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 (e.g., direct-mapped custom through general-purpose software programmable) easily span four orders of magnitude in performance (e.g., GOps/mW), and there is added opportunity to leave value on the table via ill-advised guardbands, abstractions, or other methodological choices.

Such trends 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.[1] 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. Tradeoffs 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 tradeoff 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).

METHODOLOGY PRECEPTS

Together, silicon and system complexity trends lead to *superexponentially increasing complexity* of the design process. A number of near-term goals for DT readily come to mind, e.g.: (1) provide concurrent optimization and analysis of more complicated 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. A more general and long-term set of guidelines may be stated as *methodology precepts*.

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. Future design methodologies and component tools *must* acknowledge the following precepts.

1. *Exploit reuse.* Design productivity depends on a framework for easy incorporation of previously designed subcircuits (intellectual property (IP) cores such as bus controllers, CPUs, memory subsystems, etc.). Standard core interfaces, communication protocol syntheses, verification and test collars, etc. are aspects of such reuse. Reuse of DT IP is another aspect of reuse that is enabling to "mix-and-match" platform- or application domain-specific implementation flows.
2. *Evolve rapidly.* DT must follow natural evolutionary trajectories: (a) analyses evolve into verifications, which evolve into tests, and (b) analyses and simulations evolve into models and verifications, which evolve into either objectives or constraints for synthesis and optimization. A related trend is "bottom-up commoditization" of DT, a

[1] *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.*

phenomenon that allows available R&D and designer resources to focus on ever-higher levels of design abstraction, and sets the stage for standardization and interoperability within commoditized areas.

3. *Avoid iteration.* Iteration between levels of design incurs repeated translation and other interfacing costs, and is less conducive to predictability and reliability of the design process. When used, iteration should be at higher levels for wider solution space exploration.
4. *Replace verification by prevention.* Lower-level problems (e.g., crosstalk delay uncertainty) are more cheaply addressed by higher-level prevention (e.g., repeater insertion and slew rate control rules, static verification methodology), given sufficient predictability and fidelity of understanding and representations.
5. *Improve predictability.* Constructive estimation ("estimation by doing") does not afford the necessary productivity leverage in the design flow; it is *prediction-based estimation* that enables efficient search for good design solutions. (see also Footnote 13). Tightly bounded estimates enable design space exploration – the enumeration of different design solutions before committing to one – at higher levels. Associated frameworks permit *incremental optimization* and *successive approximation.*
6. *Orthogonalize concerns.* Notably at the system-level, where the concept of a *system platform* results from separating behavior from architecture, and computation from communication, orthogonal and unrelated issues should remain separate whenever possible.
7. *Expand scope.* DT methodologies and tool architectures, driven by heterogeneous SOC integration and shrinking design margins, must expand across domains in such ways as: (a) integration of software and analog/mixed-signal/RF with digital flows; (b) extension to interfaces with humans (user interface (UI)), the physical world (photonic and sensors), and other operational context; and (c) modeling, analysis and simulation at multiple levels of the design hierarchy (up to package and board, and down to mask and process).
8. *Unify.* Silicon complexity promotes the binding together of previously disparate DT areas, notably (a) synthesis and analysis (e.g., via a static performance analysis backplane), (b) logical, physical and timing design (e.g., via RTL-to-GDSII IC implementation tools), and (c) design and test (e.g., BIST for analog or high-speed links, and signal integrity test). Increased linkages within the flow permit intentions and assumptions to be passed downstream, while estimation/prediction models are passed upstream. Consistent metrics across different stages of the design process become possible. *Interoperability* is a corollary of this trend. It should be noted that this unification is not strictly between steps in the traditional design flow. In particular, such unification must soon expand to join design with manufacturing, as characteristics of the wafer recipe, device architectures, and process equipment increasingly affect design decisions and tradeoffs. As manufacturing technology advances, so does the importance and complexity of such characterizations. Methods are needed to package manufacturing technology characterizations for "immutable interpretation" by all interested DT tools. The unification of design and test is also much more far-reaching than "between steps" of a traditional flow.

Precepts 1 and 2 allow DT to advance generically through improved focus on larger systems and higher-value technology. Precepts 3 through 6 are aspects of a "correct by construction" approach, typically reflecting a top-down, iteration-free, decomposition-oriented perspective.[2] By contrast, precepts 7 and 8 are more suited to a "construct by correction" approach, where iterations are expected but made less painful via frameworks for "co-optimization". Both of these perspectives are reflected in Figure 16, which illustrates the transition of design system architecture from the traditional "waterfall" chart in which design proceeds independently at discrete levels, to an integrated system in which logical, physical, layout, and other tools can operate together. Realization of the Figure 16 vision is almost certainly a requirement for the DT industry as a whole.

[2] *Several instances of the correct-by-construction approach have already been realized for clock and power distribution, global signaling, and block-level timing-driven synthesis-place-route. However, substantial design quality (density, speed, power) is left on the table by these early efforts. The long-term challenge is to achieve electrically- and manufacturing-correct by construction design at the behavior level.*

- **Verification moves to higher levels, followed in lower levels by equivalence checking and assertion driven optimizations**
- **Design optimized over many constraints with tightly integrated analyses and syntheses (optimizations)**
- **Integration through modular open architecture with industry standard interface for data control**
- **Shared data in memory to eliminate disk accesses in critical loops with common data for cooperating applications**
- **Incremental specification, synthesis (optimization) and analysis**

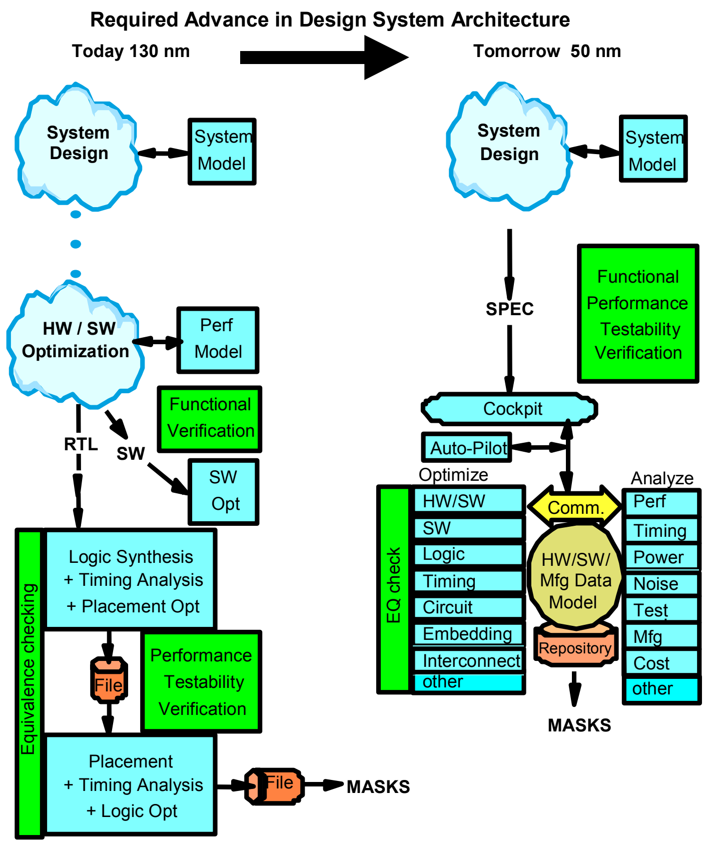

Figure 16 Required Evolution of Design System Architecture.

DESIGN TECHNOLOGY CHALLENGES

The remainder of this Chapter discusses detailed DT challenges and potential solutions in five traditional areas of DT. These areas are:[3]

- Design Process;
- System-Level Design;
- Logical, Circuit, and Physical Design;
- Design Verification; and
- Design Test.

Figure 17 depicts these areas and their scope in the form of a "landscape" for DT. Before progressing to the detailed challenges, however, we briefly call out five crosscutting challenges (or, "motifs") – Productivity, Power, Manufacturing Integration, Interference, and Error-Tolerance – that will reappear throughout the various discussions. These crosscut challenges are in some sense precursors of the detailed challenges; potential solutions are distributed across all areas of DT. The first crosscut challenge, being synonymous with system 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.

[3] *Some 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).*

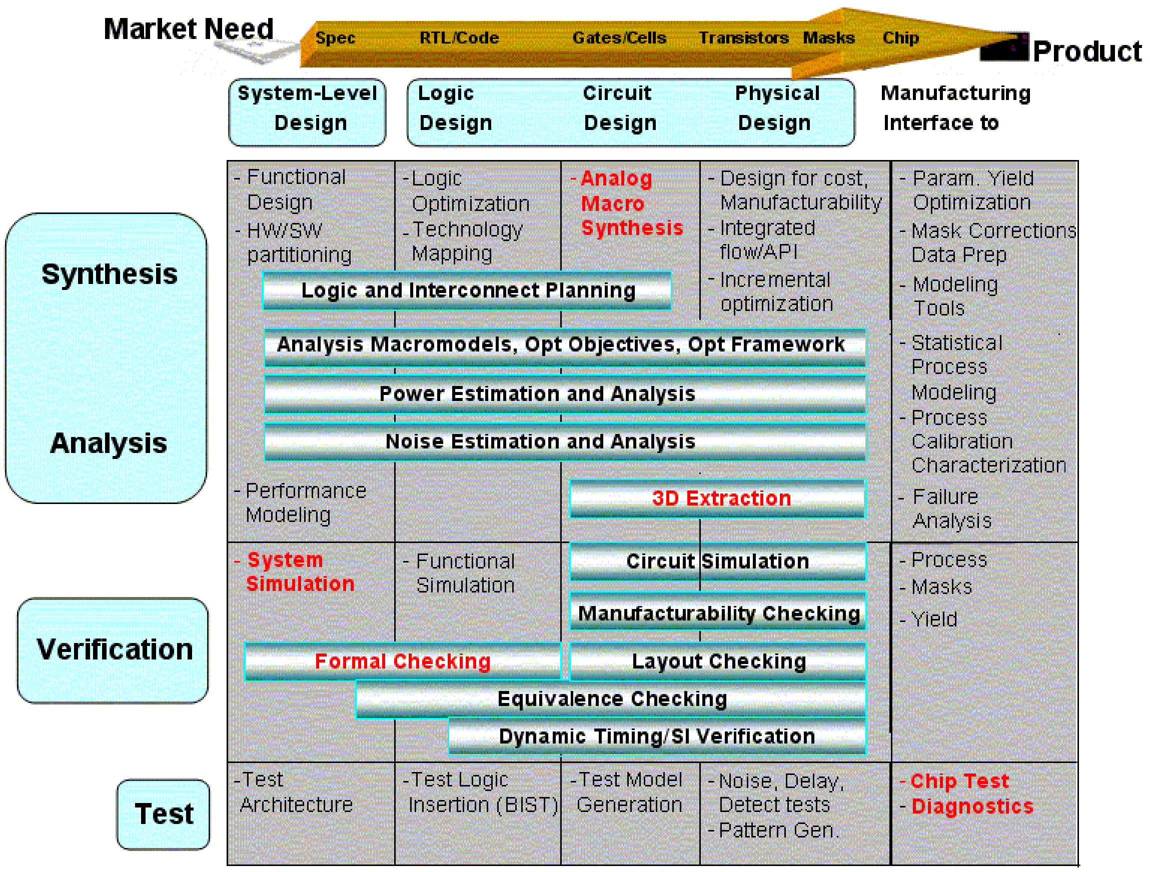

Figure 17 Landscape of Design Technology.

- *Crosscutting Challenge 1—Productivity* To avoid exponentially increasing design cost, overall productivity of designed functions on chip must scale at > 2X per node. Reuse productivity (including migration and AMSRF core reuse) of design, verification and test must also scale at > 2X 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) synthesis, verification and test, which are required by the SOC and AMS system drivers. These improvements will require metrics of normalized design quality as a function of design quality, design NRE cost, manufacturing NRE cost, manufacturing variable cost, and semiconductor product value. Metrics of design technology quality such as stability, predictability, and interoperability must be developed and improved as well. Time-to-market of new design technology must be reduced, e.g., 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) Simple extrapolation of HP MPU power dissipation, based on logic/memory content projections in the Overall Roadmap Technology Characteristics and System Drivers Chapters, exceeds by 25X the high-performance single-chip package power limits established in the Assembly and Packaging Chapter. 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 (e.g., to control IR drop in light of the roadmap for bump count and passivation opening size in the A&P Chapter), ATE equipment limits, and even current 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.

- *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 (e.g., 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, design for regularity, timing structure optimization, and static performance verification).[4] (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 100nm 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) Below 100 nm, single-event upsets (soft errors) severely impact field-level product reliability, not only for memory, but for logic 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 PROCESS

Table 13 summarizes Design Process Difficult Challenges, including associated technology needs and specific system drivers as relevant. These are *generic* requirements for design methodologies, tools, and design system architectures. All challenges, except as noted, apply to both near- and long-term years.

DESIGN PROCESS CHALLENGES

(1) Design sharing and reuse. 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 (a) an industry standard method to share design descriptions in real-time across disparate DT's and between design and manufacturing, and (b) industry adoption of this method across all

[4] The historical 10% 3-sigma gate CD variability limit in Lithography corresponds to a tolerance of less than an atomic monolayer at the end of the ITRS.

DT solutions. There is a great need to reduce integration costs, currently estimated (Dataquest, 2001)[5] 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. Long-term, improved tool reusability and interoperability, and reduction of integration costs, will be enabling to the availability of high-quality implementation flows for multiple platforms (reprogrammable, ASIC, etc.) and applications (wireless, internetworking, etc.).

(2) Increased system and silicon complexities. To address system complexity, tool integrations must minimize unproductive data translation time and data redundancy (associated also with the previous challenge), and support synthesis-analysis tool architectures that execute concurrently on distributed or SMP platforms (e.g., 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, e.g., 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.

(3) Time-to-market for cost-driven SOC. In many of the fastest-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 find/try style 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. More specifically, several areas of DT in this chapter cite analog and mixed-signal design as a productivity bottleneck. Thus, cost-driven SOC requires the ability to synthesize analog (and, in the long-term, mixed-technology) designs with efficiency and quality comparable to digital RTL-based synthesis. Long-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.[6]

[5] *Total design NRE cost is comprised of (i) designer and CAD support engineer salaries per unit time, multiplied by design time; (ii) tool license cost per unit time, multiplied by design time; (iii) 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.*

[6] *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.*

Table 13 Design Process Difficult Challenges

DIFFICULT CHALLENGES ≥ 65 nm / THROUGH 2007	*SUMMARY OF ISSUES*
Design sharing and reuse	Geographically distributed and multi-company design projects Integration of multi-vendor and internal design technology (MPU, SOC) Standard information model for IC design data, with standard interface (access mechanism) adopted across tools, databases (MPU, SOC) Tool interoperability that minimizes data translation time and redundancy to reduce design cycle times (MPU, SOC) Reduction of integration cost
Increased system and silicon complexities	Device count, scaling, operating frequency, power and noise management (MPU, SOC) Incremental analysis and optimization capability for constraint-dominated design, with runtimes proportional to amount of design changed (MPU, SOC) Scalable design optimization algorithms Concurrent execution of design and analysis tools with appropriate objectives and abstractions (e.g., for power, noise/interference, package-die, and manufacturability optimizations) (MPU, SOC) Common device, wafer recipe and equipment characterizations, controlled by process owner and packaged for "immutable interpretation" by design and analysis tools
Time-to-market for cost-driven SOC	Common information models to support reuse and (cost-driven) design space exploration (SOC) Design rules and information models (e.g., abstracts) that assure reusability; design and validation tools to assure these rules for reusable design IP (SOC) DT integration for hardware/software, digital/analog, MEMS, memory, design tools (AMS, SOC) Synthesis of analog designs comparable to digital RTL-based synthesis (AMS, SOC)
Systematic improvement of design process and design productivity	Standard design process metrics, calibration and benchmarking
ADDITIONAL DIFFICULT CHALLENGES < 65 nm / BEYOND 2007	
Time-to-market for cost-driven SOC	Platform- and application- (and even design-) specific design flows via reusable, interoperable tools (SOC) Synthesis of mixed-technology designs (including analog) comparable to digital RTL-based synthesis (AMS, SOC) System cost minimization tools spanning from standardized process description to supply chain management (SOC) Higher-level verification of function, performance and manufacturability (SOC, MPU)
Systematic improvement of design process and design productivity	Design technology productivity analysis and optimization tools Predictable physical implementation flows, along with predictive models for such flows

(4) Systematic improvement of design process and design productivity. Electronic system design will always be to some extent an art rather than a science. However, the precept of "measure, then improve" must be applied if DT is to advance systematically and rapidly. 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*. Design quality metrics help assess the extent to which available manufacturing capability is fully exploited by DT.[7] Predictability metrics enable quantified progress

[7] *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.*

toward *noise-free* algorithms, tools and flows that are predictable and stable with respect to accepted solution metrics.[8] Related needs include infrastructure for calibration and benchmarking. Support of top-down design and rapid design space exploration requires 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. Long-term, predictive models of platform- or fabric-specific silicon implementation flows are needed. Finally, complementary innovations should measure and improve design *technology* productivity.[9]

SYSTEM-LEVEL DESIGN

The electronic system design process enables designers to allocate and exploit silicon resources in a top-down and structured fashion. 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).[10] 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). It is believed that successful approaches to large-scale system design must rely on extensive reuse of predesigned intellectual property (IP) blocks and functions. However, reuse has not yet permeated the system design process: design with IP blocks is not yet fully understood, and mapping remains a mostly manual task.

Design technology has both a methodological aspect and a design automation aspect. The former covers elements such as design space exploration and model refinement; the latter defines tools and algorithms needed to implement design methods. In system-level design, the methodological aspect is rapidly becoming much harder than the tools aspect. Future process technology will allow enormous system complexity to be realized on a single die. Exploiting this potential while managing design costs will require a huge increase in design productivity – roughly 50X what is possible today. Silicon complexities such as variability and reliability mean that this productivity improvement must be achieved in a context where highly reliable and available systems built out of highly unreliable device and interconnect components. Major resulting trends are:

- *Reuse- and platform-based design in both HW and SW domains.* Reuse of pre-designed, pre-verified, pre-characterized IP blocks organized into complex application-oriented architectures still offers the highest potential for improved design productivity.
- *Increasingly prohibitive cost of communication and synchronization.* Process variability and power dissipation make global synchronization impractical for large ICs. Signaling across chip is no longer achieved in one clock cycle. System design styles must increasingly acknowledge networking and distributed computation metaphors, e.g., with communications structures designed first and functional blocks then integrated into the communications backbone. Interactions between functional and interconnect pipelining, and their integration, must be comprehended.
- *Heterogeneous integration.* Single-chip integration of subsystems implemented in multiple design technologies must be made simple, reliable and predictable.
- *Embedded SW.* As HW platforms emerge and become increasingly fixed, embedded SW will become the main vehicle for product differentiation. Embedded SW will then become more than a first-class citizen in system design: it becomes the main focus.

Table 14 lists system-level design challenges and issues. The remainder of this section develops these in more detail.

[8] *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.*
[9] *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.*
[10] *'Hardware' corresponds to implemented circuit elements, and 'software' corresponds to logical abstractions (instructions) of functions performed by 'hardware'.*

Table 14 System-Level Design Difficult Challenges

Difficult Challenges ≥ 65 nm / Through 2007	*Summary of Issues*
System complexity	Higher-level abstraction and specification (SOC) Dynamism and softness (SOC) System-level reuse (SOC, AMS) Design space exploration and system-level estimation (SOC, AMS MPU) Efficiency of behavioral synthesis and software compilation (SOC) Automatic interface synthesis (SOC)
System power consumption	Energy-performance-flexibility tradeoffs (MPU, SOC) Novel data transfer and storage techniques (MPU,SOC)
Integration of heterogeneous technologies	Codesign (HW-SW, chip-package-board, fixed-reprogrammable) (SOC, MPU, AMS) Non-scalability of analog circuits; analog behavioral modeling and synthesis (AMS, SOC) Top-down implementation planning with diverse fabrics (digital, AMS, RF, MEMS, EO, SW) (MPU, SOC, AMS)
Embedded software	SW-SW codesign onto highly programmable platforms (SOC) System capture and abstraction (SOC) New automation from high level description to HW-SW implementations, including SW synthesis (SOC) Formal verification for SW (SOC) HW-SW coverification (SOC)
Links to verification, test and culture	Integration-oriented verification and test architectures (SOC) Divergent design practices and cultures (SOC, MPU, AMS)
Additional Difficult Challenges < 65 nm / Beyond 2007	
System complexity	Communication-centric design and network-based communications on chip (SOC, MPU) Design robustness (SOC)
System power consumption	Non-scaling of centrally organized architectures (MPU) Building large systems from heterogeneous SOCs (SOC)
Integration of heterogeneous technologies	Total system integration including new integrated technologies (e.g., MEMS, electro-optical, electro-chemical, electro-biological or organic) (SOC)

System-Level Design Challenges—Near Term (> 65 nm)

(1) System complexity. The challenges of system complexity and design productivity create 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 and others) 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 (e.g., 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 – 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 (e.g., 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, e.g., design methodologies for processors that run "standard" software but are specialized to different cost-performance tradeoffs 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 (e.g., 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 (e.g., 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 tradeoffs 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. Tradeoffs 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 environment. More generally, design optimization must increasingly consider a device in its system context rather than as a standalone entity.
- *Non-scalability of analog circuits; analog behavioral modeling and synthesis.* Analog continues to be the most non-scalable aspect of circuit design, and accounts for an increasing portion of design cost. 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, vs. 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.
- *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 (a) the creation of programmable platforms by a team of hardware-centric and hardware-dependent software-centric designers, and (b) use of the platforms in derivative products by a team of applications and software-centric designers. The system platform (one

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

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:

- *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, bitstreams, 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 which, 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. (a) 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] (b) 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 (e.g., 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 (< 65 nm)

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

- *Communications-centric design.* At 65nm 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

[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. This culture gap might be bridged by a "tall, fat" designer who spans all levels of abstraction with wide and deep knowledge of all design disciplines. However, even if such a person were to exist, this is not a scalable solution.

via network-oriented protocols. This matches requirements of system-level power optimization schemes (e.g., avoidance of switching large clock nets at the maximum chip frequency). The communication-centric perspective implies that (a) algorithms must be redesigned as concurrent collaborating data transformations controlled by high-level occasional control events, and (b) 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 65nm, 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.

Difficult challenges for logical, circuit and physical design are summarized in Table 15. 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, single-event upset, 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 15. Note that some analog/mixed-signal discussions have been consolidated within the System Drivers Chapter.

Table 15 Logical, Circuit and Physical Design Difficult Challenges

Difficult Challenges *≥ 65 nm / Through 2007*	*Summary of Issues*
Efficient and predictable implementation	Scalable, incremental analyses and optimizations (MPU) Unified implementation/interconnect planning and estimation/prediction (MPU, SOC) Synchronization and global signaling (MPU, SOC) Heterogeneous system composition (SOC, AMS, MPU) Links to verification and test (MPU, SOC)
Variability and design-manufacturing interface	Uncertainty of fundamental chip parameters (timing, skew, matching) due to manufacturing and dynamic variability sources (MPU, SOC, AMS) Process modeling and characterization Cost-effective circuit, layout and reticle enhancement to manage manufacturing variability (MPU, SOC)
Silicon complexity, non-ideal device scaling and power management	Leakage and power management (MPU, SOC) Reliability and fault tolerance (MPU, SOC, AMS) Analysis complexity and consistent analyses / synthesis objectives (MPU, SOC)
Circuit design to fully exploit device technology innovation	Support for new circuit families that address power and performance challenges (MPU) Implementation tools for SOI (MPU, SOC) Analog synthesis (AMS, SOC)
Additional Difficult Challenges *< 65 nm / Beyond 2007*	
Efficient and predictable implementation	Reliable, predictable fabric- and application-specific silicon implementation platforms (SOC) Cost-driven implementation flows
Variability and design-manufacturing interface	Increasing atomic-scale variability effects (MPU, SOC, AMS)
Silicon complexity, non-ideal device scaling and power management	Recapture of reliability lost in manufacturing test (MPU)
Circuit design to fully exploit device technology innovation	Increasing atomic-scale effects (MPU, AMS) Adaptive and self-repairing circuits (MPU, SOC, AMS) Low-power sensing and sensor interface circuits; micro-optical devices (AMS, SOC)

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 2X 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 (cf. Figure 18) 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 (a) topology-sensitive global interconnect delays that do not scale with device switching times, (b) capacitive and inductive coupling interactions among devices and interconnects that dominate timing/noise validations, and (c) "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 16 FO4 inverter delays (approximately 225X 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 (e.g., 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 trend to more robust and power-efficient hybridizations of synchronous and asynchronous design. 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, e.g., 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 crcuits with respect to noise and interference, along with constraint-dominated formulations (e.g., matching, symmetry, and electrical rules in layout). 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, e.g., optimization of system-level interconnect from die through package and board. The

[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 tradeoffs among various metrics such as design TAT, area, timing, power, etc.*

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, e.g., "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 leverage of available on-chip processors (e.g., 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: e.g., 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 (cf. 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 and performance verification tools* must comprehend parasitics, delays and geometrics 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 the amount of overdesign.

- *Reticle enhancement technology* (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. (a) 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, e.g., the RLC extraction impact of downstream dummy metal insertion in the post-tapeout layout database. (b) Handoff to 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

[14] *But not quadrillions of cycles, as would be the case with reliability-related forms of dynamic variability (threshold shift due to hot carrier-induced oxide breakdown, resistivity change due to electromigration, etc.).*
[15] *Up to 200+ GB uncompressed MEBES file sizes are anticipated for a single mask layer at the 90nm node.*

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

- *Leakage and power management.* Lower supply voltages, along with larger currents stemming from increased power densities, lead to larger relative supply rail inductive noise. 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 100nm, 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 ~100ms. 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, total and standby power management "gaps" in MPU and SOC-LP reach 25X and 800X 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. Other challenges are as follows. (1) A near-term discontinuity is with respect to *single-event upsets* (SEU, or soft errors) caused by ionizing radiation, which will increasingly impact not only bistables, but random logic as well. With decreasing feature sizes, Q_{crit} values decrease to levels such that even the noise pulse from an alpha particle can be trapped as a logic fault. Protection from SEU will likely require dedicated analyses (performed as a separate characterization step) of statistical likelihood by cell, device type, and placement. Automated methods are then needed to modify logical, circuit and physical design (e.g., by automatic introduction of error correction, sizing, etc.) to prevent or manage SEU without violating design constraints. (2) 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 (a) decreasing supply voltages require exponentially increasing burn-in times, and (b) 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 (cf. the discussion of self-checking cores in the Design Verification discussion). (3) 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.
- *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-100nm nodes – reflections, EMI and substrate coupling. In this context, today's timing/noise library characterizations have big holes (e.g., 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 tradeoffs 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 (e.g., 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, e.g., (a) mitigation of gate insulator tunneling and breakdown integrity (which affects floating body devices, dynamic circuits, etc.) requires a high-permittivity gate insulator, and (b) 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. (a) 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). (b) Operation at extremely low voltage – possibly entirely subthreshold – will become more common as device performance requirements are alleviated by use of parallelism. (c) Increased compiler- and OS-based control of such parameters as body bias, clock, 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 and is particularly attractive for mixed-signal SOC. However, many DT issues must be solved before designers can switch transparently from bulk MOS devices. The floating body effect 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 become 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 Vdd-GND decoupling capacitance; hence, maintaining power supply integrity (on-chip decoupling roughly 10X 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.

[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.*
[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.*

- *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 as well as automatic *layout* syntheses. Analog syntheses must handle increased opportunities for distortion and nonlinearity due to impact ionization, thermal nonlinearity, body contacts acting as bandpass filters, etc. Syntheses must also handle future regimes of increased manufacturing variability, e.g., by hybrid analog-digital compensation for device mismatch. In the near term, new synthesis tools for optical interface circuits and high-Q tunable CMOS on-chip inductors/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 (e.g., beam-steering) devices. Verification and test, being "to specification" rather than structural in nature, are problematic for analog circuits. While ever-faster simulation has been the historical solution, new verification solutions must include statistical techniques, better compact models that speed up simulation without sacrificing accuracy, and acceptance criteria. BIST/BOST techniques will be increasingly synthesized around, and laid out with, analog circuitry (e.g., high-speed optical networking interfaces). Such techniques must be nonintrusive, i.e., they must not degrade performance of embedded analog blocks.

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 (e.g., automated tools for logic synthesis, place-and-route, and test pattern generation), 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 16 summarizes the main challenges in design verification. The remainder of this section provides more details. For expository reasons, the discussion is structured with respect to the System Drivers.

[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 16 Design Verification Difficult Challenges

Difficult Challenges ≥ 65 nm / Through 2007	*Summary of Issues*
Increase verification capacity (SOC, MPU, AMS)	Verification exponential in design size Need high coverage Need to handle large designs Semi-formal techniques
Robust verification tools (SOC, MPU, AMS)	Verification algorithms highly unpredictable Improved heuristics Characterization of problem difficulty
Verification metrics (SOC, MPU, AMS)	Behavior coverage Realistic bug model Algorithms to determine bug coverage
Software verification (SOC)	Software intrinsically more difficult to verify Traditional software verification techniques inapplicable Integrated hardware/software systems Design for verifiability (long-term)
Verification reuse (SOC)	Must allow reuse of verification of IP blocks Specify abstract behavior of IP blocks Specify environmental constraints of IP blocks Hierarchical verification algorithms
MPU verification methodology (MPU, +long-term)	Different cost-benefit trade-off (higher cost acceptable) Need exceptionally high capacity Must be very predictable due to long design cycle and pipelined development teams
MPU design-for-verifiability (MPU, +long-term)	Will be necessary sooner than for other system drivers Specialized techniques likely possible
Greater concurrency (MPU, + long-term)	New processors far more concurrent Greatly increased verification complexity
Anything is progress (AMS)	Extremely primitive state-of-the-art Forces difficult hybrid-systems issues into near term
Additional Difficult Challenges < 65 nm / Beyond 2007	
Design for verifiability (SOC, MPU, AMS)	New methodology needed Characterize and minimize performance and area impact
Higher levels of abstraction (SOC, MPU, AMS)	New algorithms needed Complexity of designs enabled by higher-level design Equivalence checking vs. RTL
Human factors in specification (SOC, MPU, AMS)	Specifications of correctness will become unmanageable Need to understand what kinds of specifications are most understandable Need to consider how to make specifications modular and modifiable
Verification of non-digital systems (SOC, MPU, AMS)	Hybrid systems verification for analog effects Hybrid systems verification for analog properties Verification of probabilistic systems
Heterogeneous systems (AMS, SOC)	How to model, analyze, and verify MEMS, EO devices, and electro-biological devices

DESIGN VERIFICATION CHALLENGES—GENERAL

Many of the most important challenges for verification are relevant to most or all system drivers. In the near-term, three primary issues (1-3 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 (4-7 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.

(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 18 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.

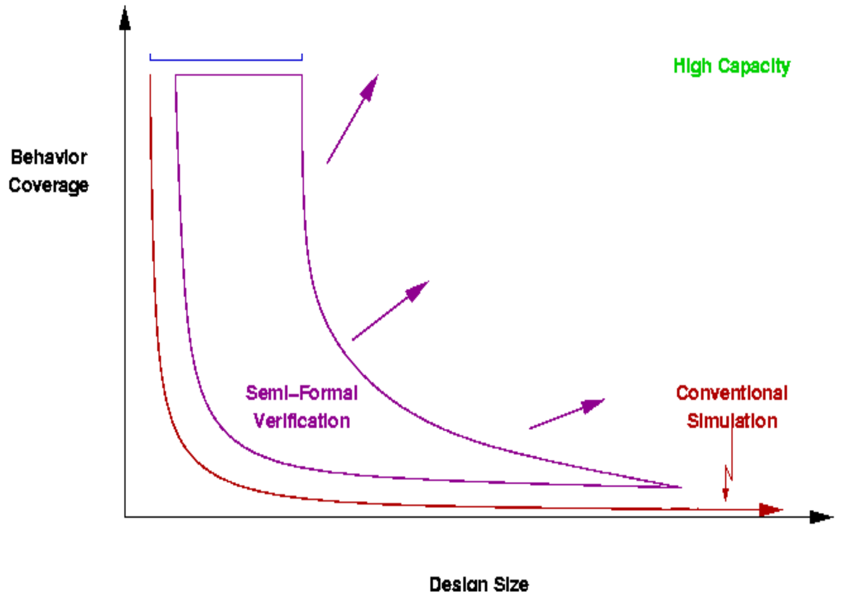

Figure 18 Current Verification Tool Landscape.

(2) Robustness. Figure 18 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.

[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.*

- 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) 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.

(5) 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).

(6) 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 the development of the languages needed 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.

(7) 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 (e.g., 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 (e.g., quality of service guarantees in a network processor). Thus, there will be the challenges of hybrid systems and probabilistic verification.

DESIGN VERIFICATION CHALLENGES—SOC-SPECIFIC

All of the broader verification challenges apply to SOC. In addition, the following verification challenges are especially important for SOC.

(1) Software. Much of the functionality of SOCs will be defined by software, so a major SOC verification challenge is how to verify software and hardware/software systems. Traditionally, software development is not as rigorous as hardware development in terms of design reviews, analysis tools, and testing. Furthermore, software is intrinsically harder

[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.*

to verify, due to more complex, dynamic data and the 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; new verification methods for low-level software will have to be developed. The near-term challenge will be to develop techniques that allow verification of even elementary 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.

(2) 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 is most likely for standardized IP interconnects, such as on-chip buses, but the general problem for arbitrary IP block interfaces must be solved eventually.

DESIGN VERIFICATION CHALLENGES—MPU-SPECIFIC

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 used in incomparably high volumes. As a result, different, domain-specific verification challenges and opportunities exist, both near- and long-term.

(1) 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, but are already 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 the 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?

(2) Specialized design-for-verifiability. Similarly, the complexity of MPU designs will likely require design-for-verifiability sooner than other system drivers will, 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 watch-dog 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.

(3) New kinds of concurrency. As MPU designs become more complex, new kinds of concurrency are becoming important. Already, many of the bugs that elude verification relate to cache coherence and other concurrency issues. New designs increase the level of concurrency via techniques such as chip-level multiprocessing (and on-chip cache coherence protocols) and simultaneous multithreading, which greatly complicates the verification process. The challenge is to develop techniques to handle these new forms of concurrency.

DESIGN VERIFICATION CHALLENGES—AMS-SPECIFIC

AMS designs share the broader verification challenges with the other system drivers. However, the main difference is that they force the long-term issue of hybrid-systems verification into the near term. Unfortunately, research on hybrid-systems verification is still in its infancy. The near-term challenges are, therefore, to provide any support possible to improve the current ad hoc approaches and to develop the basic research needed to understand how to move beyond a grab-bag of standalone simulations to more powerful techniques. In the long term, analog might not mean only electronics. If MEMS, electro-optic, and electro-biological devices become more than simple transducers, a major verification challenge will be how to model, analyze, and verify a system with such heterogeneous parts.

DESIGN TEST

Nanometer process technology, increasing clock rate and SOC integration present severe challenges to design for test (DFT). The test industry must cope with an enormous spectrum of problems ranging from high-level test synthesis for component-based design to noise/interference and power dissipation in extremely high performance (in reality, 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 (e.g., 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.

(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) designs. Integration of pre-existing design blocks into larger integrated devices produces non-linear complexity growth for design tools, DFT, and manufacturing test, even when the blocks are homogeneous (e.g., all logic). Increasingly, a wider variety of circuits are being integrated. Logic and 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, e.g., manufacturing test costs 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 may also 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. Thus, testing embedded flash, RAM, or AMS on an integrated logic device typically breaks these optimizations. 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 not available 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 (e.g., 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.

(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 180nm and 130nm, thermal runaway limits the use of, and drives non-linear cost increases for, burn-in for 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 also 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—e.g., to the extent that foundries can inform and support integrated test strategies, they will become preferred vendors. The resulting Design Test Difficult Challenges are summarized in Table 17.

Table 17 Design Test Difficult Challenges

Difficult Challenges ≥ *65 nm / Through 2007*	*Summary of Issues*
At-speed test with increasing frequencies	Continuation (avoidance) of at-speed functional test with increased clock frequencies (MPU, (ASIC/SOC)) At-speed structure test with increased clock frequencies (MPU, ASIC/SOC) Test and on-chip measurement techniques for multi-gigahertz serial ports (AMS)
Capacity gap between DFT/Test generation/fault grading tools and design complexity	Better EDA tools for advanced (open, delay, etc.) fault models (MPU, ASIC/SOC) DFT to enable low-cost ATE (MPU, ASIC/SOC) Non-intrusive logic BIST (including advanced fault models) (MPU, ASIC/SOC) AMS DFT/BIST, especially at beyond-baseband frequencies (AMS)
Quality and yield impact due to test equipment limits	Power and thermal management during test (MPU, ASIC/SOC) Fault diagnosis and design for diagnosability (MPU, ASIC/SOC) Yield improvement and failure analysis tools and methods (ASIC/SOC)
Signal integrity testability and new fault models	Signal integrity (noise, interference, capacitive/inductive coupling, etc.) testability (MPU, ASIC/SOC) Fault models for analog (parametric) failures (AMS)
SOC test	Integration of SOC test methods onto test equipment platform Integration of multiple fabric-specific test methodologies DFT, BIST and test methods compatible with core-based SOC environment and constraints (AMS) Embedded memory built-in self-diagnosis and self-repair Test reuse
Additional Difficult Challenges < 65 nm / Beyond 2007	
Integrated self-testing for heterogeneous SOCs	Test of multi-gigahertz RF front ends on chip (AMS) Use of on-chip programmable resources for SOC self-test (SOC) *Dependence on self-test solutions for SOC (including RF and analog)* (Analog) signal integrity test issues caused by interference from digital to analog circuitry (AMS) Test methods for heterogeneous SOC including MEMS and EO components (SOC)
Diagnosis and reliability screens	Diagnosis and failure analysis for AMS parts (AMS) Design for efficient and effective burn-in to screen out latent defects (MPU, SOC) Quality and yield impact due to test equipment limits (MPU, ASIC/SOC) New timing-related fault models for defects/noise in nanometer technologies (MPU, ASIC/SOC)
Fault tolerance and on-line testing	DFT and fault tolerant design for logic soft errors (MPU, ASIC/SOC) Logic self-repair using on-chip reconfigurability (SOC) System-level on-line testing (SOC)

DESIGN TEST CHALLENGES—NEAR TERM (> 65 nm)

(1) At-speed test with increasing frequencies.

- *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 test 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 ports*. Protocols such as Gigabit Ethernet and Firewire are adopted with, e.g., embedded processors with multi-GHz peripheral buses using unconventional signaling techniques. 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. 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. New I/O test methodologies, DFT and on-chip measurement techniques for such serial communications buses and components are needed. 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.

(2) Capacity gap between DFT/Test generation/fault grading tools and design complexity. Design databases of current microprocessors and the largest ASICs exceed data size limits of industry EDA tools, forcing complex patch schemes and partitioning and making 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 (moreover, 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-100nm 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 (e.g. 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 extend to cover beyond stuck at and transition faults (e.g. to directly cover modeled opens and bridges and other fault types).

In most mixed-signal ICs, analog circuitry accounts for only a small fraction of the total silicon. However, the total production cost is dominated by the analog costs. Several sources predict that the ratio of analog testing cost to total mixed-signal product cost will continue to increase unless changes to analog testing are made. Developing cost-effective test methodologies and techniques for analog and mixed-signal components is therefore an essential part of devising a complete solution for testing of heterogeneous SOCs.

- *Analog/mixed-signal DFT/BIST, especially at higher resolution and/or higher frequencies (beyond baseband).* Cost-effective test techniques for on-chip, high-resolution (e.g., >16 bits) ADC and RF components are needed. 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. DSP-based DFT and BIST solutions for AMS components must be extended to frequencies significantly beyond baseband.

(3) Quality and yield impact due to test equipment limits.

- *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 will require 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 life-time 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, to provide corrective actions and to improve the yield.

(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 for evaluating a given analog test program. It is crucial to develop meaningful figures of merit for evaluating the 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 which can minimize the computation complexity of fault simulation while maintaining high simulation accuracy.

(5) SOC test.

- *Integration of SOC test methods onto test equipment platform.* Multiple insertions using a cheaper, dedicated tester per integrated block type (analog, RAM, flash, logic, etc.) can be traded off against integration of all measurement hardware and software onto a single platform and using a single insertion. 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 taken into account. Historically, integration of additional hardware complexities 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 testing of different blocks is not executed 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 (e.g., IEEE P1500 or equivalent) is required. In the long run, non-linear manufacturing test costs will drive as many blocks as possible to full BIST solutions, else the economies of SOI and silicon integration will be lost. A design engineering focus on solutions in this space will provide very significant ROI in terms of SOC product cost reduction, but must be addressed in the earliest phases of product planning.
- *Integration of multiple fabric-specific test methodologies.* SOC 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 testing embedded blocks should provide equivalent or faster test times than those for standalone versions of such blocks.

DESIGN TEST CHALLENGES—LONG TERM (< 65 nm)

(1) Integrated self-testing for heterogeneous SOCs.

- *Test of multi-gigahertz RF front ends on chip.* The manufacturing test process of RF components requires significant instrumentation and human efforts. 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, e.g., 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 support crosstalk testing during operation at 2 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 and reliability screens.

- *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.* To control latent defects at acceptable levels, burn-in thermal runaway should be mitigated via design-for-burn-in or new alternate schemes (e.g., 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 increasingly 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 18 Additional Design Technology Requirements

YEAR OF PRODUCTION	*2001*	*2002*	*2003*	*2004*	*2005*	*2006*	*2007*	*2010*	*2013*	*2016*	*DRIVER*
DRAM ½ PITCH (nm)	*130*	*115*	*100*	*90*	*80*	*70*	*65*	*45*	*32*	*22*	
MPU / ASIC ½ PITCH (nm)	*150*	*130*	*107*	*90*	*80*	*70*	*65*	*50*	*35*	*25*	
MPU PRINTED GATE LENGTH (nm)	*90*	*75*	*65*	*53*	*45*	*40*	*35*	*25*	*18*	*13*	
MPU PHYSICAL GATE LENGTH (nm)	*65*	*53*	*45*	*37*	*32*	*28*	*25*	*18*	*13*	*9*	
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)	**1.2**			**2.6**			**5.9**	**13.5**	**37.4**	**117.3**	SOC
SOC dynamic power reduction beyond scaling (X)	**0**			**1.5**			**2.5**	**4**	**7**	**20**	SOC
SOC standby power reduction beyond scaling (X)	**2**			**6**			**15**	**30**	**150**	**800**	SOC
%Test covered by BIST	**20**			**30**			**45**	**60**	**75**	**90**	MPU, SOC

Mtx—Mtransistors

White—Manufacturable Solutions Exist, and Are Being Optimized	⬜
Yellow—Manufacturable Solutions are Known	🟨
Red—Manufacturable Solutions are NOT Known	🟥

APPENDIX: DT COST AND VALUE

The vital contribution of DT to semiconductor product realization is shown by the following analysis. At the request of the 2001 ITRS Design ITWG, Gartner/Dataquest measured designer productivity at 4K gates (= 16K transistors) per year in 1990—the year in which the so-called "RTL methodology" originated – and calibrated design productivity improvements for seven major DT innovations that have occurred or are anticipated since then. These improvements are: in-house place-and-route (1993; +38.9% productivity improvement to 5.55K gates per designer-year); "tall-thin engineer" (1995, +63.6% improvement to 9.09K gates), small-block (2,500 – 74,999 gates) reuse (1997, +340% improvement to 40K gates), large-block reuse (75,000 – 1M gates) (1999, +38.9% improvement to 56K gates), IC implementation suite (2001, +63.6% improvement to 91K gates); "intelligent testbench" (2003, +37.5% improvement to 125K gates); and "electronic system-level (ES-level) methodology" (2005, +60% improvement to 200K gates). Definitions of these DT capabilities are as follows. (1) The IC Implementation Suite is characterized as a set of tightly integrated tools that takes an IC design from RTL synthesis through IC place-and-route and GDSII output. (2) The Intelligent Test Bench is an RTL verification automation tool ("cockpit") which takes a design description from the ES-level and partitions it into verifiable blocks, then executes various verification tools, on the appropriate blocks, while tracking and reporting code coverage. (3) ES-level is immediately above RT-level and includes both hardware and software design. It consists of two sub-levels, behavioral (where the system function has not been partitioned) and architectural (where hardware and software are identified and handed off to the appropriate design teams).

Figure 15 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 *GTX* model sets the historical rate of increase in engineer cost at 5% per year (salary and overheads starting at $181,568 in 1990), and the rate of increase in EDA tool cost at 3.9% per year (starting at $99,301 per engineer in 1990). Refer to the supplemental material for the *study of productivity*. The number of designers per million logic gates is 250 in the year 1990 and 11 in 2001 (implying an average of 32.8% per year improvement in number of logic gates per designer-year). The low-power SOC PDA model (System Drivers chapter) has 3M logic gates in 2001, implying an SOC PDA design cost (designers + tools) of $15.1M. Without the five major DT innovations that occurred between 1993 and 2001, the design cost for the same SOC in 2001 would be approximately $342.4M. The gap is 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 System Drivers chapter; that estimate sets new (reuse) logic productivity to be 360K (720K) gates/designer-year in 1999, a 6X (12X) factor higher than the Dataquest estimate for the same year.

We conclude that without a continued DT innovation pipeline, design cost would quickly become prohibitive or designs will be forced to have less valuable content.