Design Technology Productivity in the DSM Era*

Andrew B. Kahng

UC San Diego Departments of CSE and ECE, La Jolla, CA 92093-0114 USA
abk@ucsd.edu

Abstract: Future requirements for design technology are always uncertain due to changes in process technology, system implementation platforms, and applications markets. To correctly identify the design technology need, and to deliver this technology at the right time, the design technology community – commercial vendors, captive CAD organizations, and academic researchers – must focus on improving design technology time-to-market and quality-of-result. Put another way, we must address the well-known Design Productivity Gap by addressing the Design *Technology* Productivity Gap. The future of design technology is most uncertain in the physical implementation space, where many complexities must be managed without sacrificing system value or turnaround time. This talk will begin with examples of changes in methodology and design tool infrastructure needed to support future physical implementation. Then, recent initiatives in the MARCO Gigascale Silicon Research Center (GSRC) are described which support the design technology and designer communities in creating the right design technology when it is needed.

1 Background: DSM Challenges

The litany of back-end DSM challenges is well-known. (1) Challenges of the *small* are induced by sub-50nm scaling, and bring to the forefront issues of current density and power, synchronization, manufacturing variability, high-frequency and coupling noise, and yield (manufacturing cost). (2) Challenges of the *large* are induced by $> 10^9$ transistor integration, and center on system complexity and cost (notably design costs along axes of resource, quality and time-to-market). (3) Challenges of the *heterogeneous* are induced by the encapsulation, reuse and integration of CMOS logic along with mixed-signal and RF, MEMS, and memory technologies. For background, we briefly sketch example challenges arising in *design convergence* and the *design-manufacturing interface*. Our purpose in presenting this small sampling of DSM challenges is to illustrate how future technical solutions will require integration across many different types of tools. Indeed, integration and cross-domain optimization are common themes for future design technologies.

Example 1: Design Convergence. *Design convergence* is the convergence of logic, timing and spatial embedding; from a functional viewpoint, it represents the support of front-end signoff by a predictable back-end. Future technology regimes will entail global wires $>$ 100x faster than local wires, single via stacks with $>$ 12 ohms, over 80% of wire capacitance arising from coupling, and worst-case "Miller coupling factors" increasing from 2 to more than 9 as gates switch faster and wire pitch decreases. In such regimes, (static) signal integrity and timing signoff methodologies become problematic due to increased guardbanding. Moreover, front-end signoff of parasitic and performance estimates require *predictability* of distributed RLC interconnect topologies.

Previous statistical prediction methods to decouple logic and physical syntheses (e.g., block-specific wireload models) suffer from increased iterations and non-convergence. Newer "RTL-to-GDSII" methodology variants achieve predictability in several ways. (1) *Correct by construction* methods assume a result, then enforce it. Examples include constant-delay (gain-based) logic synthesis, and the use of hard chip-level global routes to define global timing and pin assignments before block synthesis. Iterations are reduced at the cost of guardbanding. (2) *Construct by correction* methods acknowledge the need to iterate, but architect tools to support this. Examples include unified logic and layout synthesis, or, unified analysis and synthesis (e.g., tying incremental extraction and static timing to incremental place and route). (3) *Ignore by prevention* methods apply "methodology" to make certain issues ignorable. Examples include (i) rules for repeater insertion, slew time control, and layer assignment to solve sizing and signal integrity issues [19, 9], (ii) rules for grounded shielding to improve inductance estimation [9], or (iii) "noise-free fabrics" [21].

These avenues to design convergence, while reasonable and viable, share several technical challenges. (1) RTL partitioning must create blocks that can be placed and budgeted well, and also must recognize and diagnose a "physically challenged" RTL. (2) Block "packing" mindsets must give way to more relevant "shaping" that integrates with wire planning and timing optimization to yield zero-whitespace, zero-overlap results [17]. (3) When block areas and timings are not yet fixed, global interconnect plans must support design changes smoothly. More generally, optimization under design changes or under incomplete design information becomes more critical [12, 18]. (4) The *synthesis, place and route* (SP&R) back end must be highly controllable and handle constraints well. For example, if a local routing resource has been used by a noise-sensitive global bus, the back end must work around this. (5) Since there is always a chicken-egg problem for top-down planning, estimators that drive initial synthesis and budgeting must always be improved (e.g., [6, 11]).

The high-level observation is that future methodology variants require push-button *back-end SP&R. Thus, the most interesting implication of "back-end design convergence" is that synthesis, place and route – along with such supporting technology as extraction and static timing analysis – become* commoditized *(just like delay calculation or Verilog simulation today). Moreover, individual point tools become less valuable as integration becomes the driving consideration.*

Example 2: Design-Manufacturing Interface. Future design technology must address both random and systematic variabilities, with the latter encompassing intrinsic layout-pattern dependent and exposure-system dependent effects (iso-dense or lens aberration impacts on device and interconnect CDs) as well as dynamic effects (temperature, V_{dd}, etc.). Two examples come from use of *aperture* and *phase* in optical lithography to make sub-wavelength feature sizes. (1) Changing *aperture* to correct for line-end shortening, corner rounding, etc. implies more complicated mask shapes in the form of *optical proximity correction* (OPC). OPC affects data volumes and DRC, but more critically impacts mask writing and inspection costs. For example, future design technology must allow "function-aware" OPC that is applied only as needed (e.g., to features in critical paths which require better CD control). This entails passing performance analysis and functional intents from logic-layout synthesis to physical verification. Required flow integrations must span library creation, detailed routing and physical verification (e.g., so that an optical correction is not made independently by several tools, leading to an incorrect result). (2) Changing *phase* to improve contrast and resolution is achieved by *phase-shifting masks* (PSM) [22], which exploit destructive cancellation of diffraction to create more perfect contrasts between light and dark regions on the wafer. Since 1999, the ITRS has specified PSM as a required technology solution that enables Roadmap acceleration (cf. physical bottom gate length in logic, and transistor densities, in the 2000 ITRS ORTCs, Figure 1). With PSM, certain DRC-correct layouts cannot be manufactured, because legal assignment of (e.g., 0 and 180) phases in the mask is impossible [20]. Such *phase conflicts* must be addressed

*Partially supported by the MARCO Gigascale Silicon Research Center.

0-7803-6633-6/01/$10.00 ©2001 IEEE.

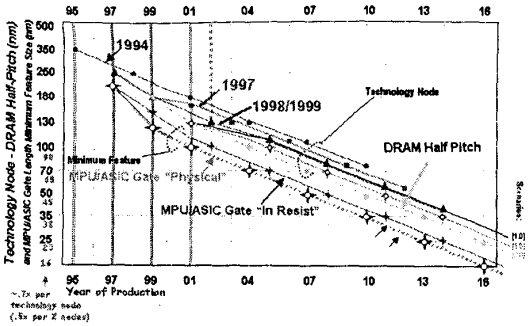

Figure 1: Roadmap acceleration, showing anticipated DRAM half-pitch and logic bottom gate length in the 2000 ITRS Update.

by moving and/or widening some subset of layout features. To maintain back-end layout productivity as well as the layout reuse inherent in, e.g., standard-cell methodology, responsibility for PSM must be distributed across a highly-integrated toolset that spans library creation, full-custom layout support, technology migration, and standard-cell place and route.

Other facets of the design-manufacturing interface also present challenges to future design technology. For example, (3) *pattern density variation* causes dishing of inter-layer dielectrics in chemical-mechanical planarization. With copper interconnect, this is particularly troubling since copper is so soft; the 2000 ITRS therefore has copper dishing and thinning as a fundamental process parameter. At the device level, uneven layout density can result in contact overetch and uneven vapor deposition. The solution is to introduce extra *dummy fill* to make the process result more uniform. This must not only be driven by the best possible models of the manufacturing process, but flow issues abound: (i) dummy metal fill will change RC extraction results, and must be accounted for in the upstream timing verification before the layout goes to physical verification; and (ii) data volumes and design hierarchies must be maintained. Just as with OPC and PSM, dummy fill requires a complete, integrated, front-to-back solution.

The design-manufacturing interface brings design technology much closer to the foundry and capital investment. Tight links with process development and the mask industry must be made, e.g., to transmit functional information to mask writing, inspection and verification, and cost models back up to the design tools. Traditional "physical verification" will become embedded all the way up to library creation, logic synthesis, and place-and-route. Unified, front-to-back solutions will win.

The Challenge: Too Many Challenges. The above are are only a small sampling of the technical, integration, and cross-domain challenges that face future design technology. The key conclusion is that there are too many critical challenges for the design technology community to solve with its available resources.

2 The Design *Technology* Productivity Gap

This section develops the concept and implications of the Design *Technology* Productivity Gap. We start with five observations.

- First, the famous Design Productivity Gap (Figure 2) states that the number of available raw transistors increases by 58%/year while the designer's capability to design them grows by only 21%/year. **Observation 1:** *The cost of designing a transistor increases exponentially relative to the cost of the raw transistor.*
- Second, while the ITRS documents Silicon cost (at volume production, for particular classes of designs), it does not document Product cost (= Non-Recurring (NRE) cost + Silicon cost). We must ask: If design technology is not improving fast enough, then how costly will designs be in the future? Put another way, how many gates can one obtain for $1 or $10? Certainly, OPC and PSM increase the NRE component of system cost. A 25-level mask set will cost around $1M in the 130nm process generation, which arrives this year. At the same time (according to SEMATECH figures), an average of only 500 wafers are processed with a given mask set. Changing reticle reduction factors (e.g., 5X or 6X on 6-inch or 7-inch glass), placing multiple designs on a single mask set, and improved yield learning will mitigate these costs. However, if we acknowledge the exponentially increasing cost of designed transistors, and the need to amortize $5B capital cost per foundry, there is only one possible conclusion. **Observation 2:** *Only high-value, high-volume designs will be affordable in the future.*[1]

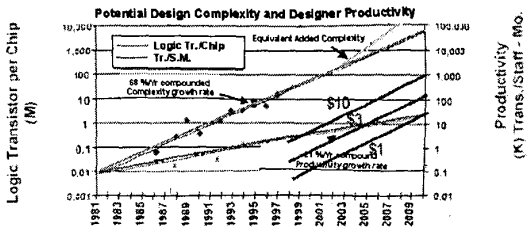

Figure 2: Design Productivity Gap.

- Third, we may consider what is "easy" versus "hard" for design tools. Today, less effort is required to achieve a given return if we design at a higher level. For example, solving power problems is difficult with placement and wiresizing, easier with clock gating, and even easier by adding sleep mode to the system. This difference may be magnified in the future, as DSM physics become very complicated. Design productivity requires designers to design and hand off to implementation at higher and higher levels, but DSM physics forces the implementation platform to remain at a low level. **Observation 3:** *There is a widening* implementation gap, *i.e., separation between where designers need to design, and the reliable (RTL-down) back end.*[2]
- Fourth, while long-term research – such as that within the MARCO GSRC – may eventually address the implementation gap, there is no currently available solution. (1) Making simpler designs (e.g., filling up transistors with memory and reused logic) is one obvious implication of any "productivity gap" between new- or reused-logic productivity and available transistors: if designer resources are not expanded to fill the gap, then large chips *must* contain more "free" (memory) transistors.[3] However, this does not create high-value designs. (2) Making the back-end implementation easier (e.g., with noise-free circuit fabrics [21]) or enabling easier block reuse in SOC (e.g., via communication-based design ("TCP/IP on chip") [25]) improves productivity but harms density and performance. Again, high-value designs are not possible. **Observation 4:** *There is no known solution to the implementation gap that does not compromise design quality and value.*

[1] Works of W. Maly have given detailed analyses of cost contradictions inherent in the NTRS/ITRS, and of the need to temper Moore's Law with cost realities. An example work is [23].

[2] This implementation gap, noted by K. Keutzer and colleagues in the MARCO GSRC, has motivated several core research themes within the GSRC.

[3] STRJ (Japan Semiconductor Technology Roadmap) colleagues have performed one such analysis. Under current productivity growth rate assumptions, 94% of the die will be populated with memory in 2014 if design effort is to remain at current levels.

- Fifth, we recall the economic need for high-volume, high-value designs ("keeping the fabs full"). If high-value designs cannot be created quickly enough, then the semiconductor industry must attempt workarounds. For example, *platform-based* SOC design (cf. [10] and work of Keutzer, Malik, Sangiovanni-Vincentelli et al. in the MARCO GSRC) attempts to realize a handoff level between architecture and microarchitecture. In the platform-based approach, "front end" compilation of application down to architecture goes hand in hand with microarchitecture serving as the compiled abstraction of the silicon process. Where traditional structured-custom design creates a unique mapping of each microarchitecture to silicon (i.e., a unique chip with unique CAD), platform-based design allows compilation of entire application families (e.g., 3G wireless) down to a single platform. Such approaches, if successful, would be high value for the system house, but incompatible with traditional design technology and ASIC business models. On the other hand, platform-based approaches may be unsuccessful because they again leave quality and value on the table.[4] **Observation 5:** *If the "platform-based" design productivity workaround succeeds, it will be will at the cost of design technology's inherent value. On the other hand, platform-based design can fail if it unacceptably compromises design value.*

Design productivity cannot be separated from design quality. Hence, there is no shortcut or workaround: we truly require design technology to deliver high-value silicon with low design cost. This brings us back to the Challenge above: Too many challenges, too few resources.

Initiatives within the MARCO GSRC

The Calibrating Achievable Design (C.A.D.) theme in the MARCO GSRC [4] identifies the Design Productivity Gap with the Design *Technology* Productivity Gap. Aspects of this gap include:

- Design technology lacks a clear industry-wide R&D agenda (e.g., a "roadmap").
- There is a long time-to-market (up to 5-7 years to transfer leading-edge publications to production flows), which forces designers to battle today's design problems using yesterday's design technology.
- There are few means of assessing quality-of-result, e.g., it is difficult to evaluate the impact of new tools on the overall design process, and published descriptions are insufficient for replication or even comparison of algorithms. When R&D cannot identify, evaluate or reuse the design technology leading edge, research and innovation become stalled.

Corresponding causes of the design technology gap include:

- Lack of roadmapping with respect to the ITRS and applications markets.
- Lack of "Foundation CAD-IP": interoperable, reusable, commodity infrastructure upon which new design technology can be built.
- Relative over-resourcing of non-strategic, de facto commodity technology (e.g., GDSII or SPICE format readers/writers, delay calculators, netlist partitioners, etc.) – and a concomitant lack of maturity with respect to control and strategic-vs.-commodity distinctions.
- Lack of standard metrics and benchmarks for design technology.

[4] Notably, the platform-based approach faces challenges for very low-power, very high-throughput applications. For example, J. Rabaey and colleagues have documented well over 1000x quality differences (MOps/mW) between dedicated hardware and, e.g., embedded low-power StrongARM core.

We believe that mature, *coopetive* (= competitive + cooperative) cultures and shared, open infrastructures that lead to improved specification, creation and delivery of design technology. To make this more concrete, we focus on (1) criteria that apply to *any* technology delivery, namely, *time-to-market* and *quality-of-result* (QOR), and on (2) three basic questions that pertain to the design technology life cycle.

- *What will the design problem look like?* What problems do we need to solve, by when? In other words, we must *specify* the required technology.[5]
- *How can we quickly develop the right design technology so that it is available at the right time (i.e., time-to-market)?* In other words, once we know the right design problem, we must be able to quickly develop and deploy a solution.
- *Did we really solve the problem (i.e., QOR)?* Did the design process improve? In other words, we must be able to measure our progress.

The remainder of this paper describes three initiatives whose goal is to enable the design technology community to answer these questions.

3 Technology Extrapolation

What will the design problem look like? Leading-edge VLSI system design aggressively exploits new process technologies, circuit techniques, design methodologies and design tools. It is thus difficult to predict the envelope of *achievable design* – e.g., with respect to power, speed, area, manufacturing cost, etc. – for a given behavior or function, in a given (future) process technology. On the other hand, such *technology extrapolation* activity directly influences the evolution of future VLSI system architectures, design methodologies, and design tools. Via roadmapping efforts such as the International Technology Roadmap for Semiconductors (ITRS) [16], technology extrapolation also influences levels of investment in academic research, career choices for faculty and graduate students, as well as private-sector entrepreneurial activity.

Highly influential technology extrapolation systems are due to Bakoglu and Meindl (SUSPENS) [29] and Sai-Halasz [28]. More recent efforts include GENESYS [27], RIPE [26] and BACPAC [30], along with Roadmap-related efforts [16] and innumerable internal projects throughout industry and academia. Typically, each system provides a plausible "cycle-time model" and estimates of die size and power dissipation, based on a small set of descriptors spanning device/interconnect technology through system architecture. We observe that (i) these systems are often incomparable, (ii) they are "hard-coded" (hence it is difficult to assess their quality and to explore changes through modeling choices), and (iii) their development has entailed a near-total duplication of effort. These observations motivate efforts toward an entirely new level of technology extrapolation capability. The GSRC Technology Extrapolation (GTX) system [5] has been developed with the goals of *flexibility, quality* and *prevention of redundant effort* in mind. GTX addresses these goals by providing an open, portable *framework* for specification and comparison of alternative modeling choices. A fundamental design decision in GTX is to separate model specifications from the derivation engine. This separation is achieved by a human-readable ASCII grammar. As domain-specific knowledge is represented independently of the derivation engine, it can be created and shared by multiple users. Additional extension mechanisms allow specialized prediction methods, technology data sets, and even optimization engines to be encapsulated and shared within GTX; this further reduces the amount of effort that is diverted from actual creation of best-possible prediction models.

[5] For example: Should EDA tools focus on support of dynamic logic synthesis at sub-1 volt supplies Should detailed routers be made PSM-aware? Should circuit design techniques address single-event upsets? Our claim is that in avoiding misguided development efforts, better focus and productivity of the design technology will be achieved.

The GTX framework has prompted development of new SOI and bulk Si device models; models for global interconnect optimizations (incorporating repeater staggering and bus swizzling, shield insertion, etc.), global RLC interconnect delay, and coupling noise; yield and cost models for logic and DRAM; manufacturing variability models; etc. These models, along with studies of power dissipation, delay uncertainty, clock skew, etc. have been released in open-source form along with the GTX system itself. Other efforts have developed design-specific interconnect process optimizations that are based on new models of routability and layer assignment calibrated to industry place-and-route results. A long-term goal is for the (open source, multi-platform) GTX release to literally provide a "living roadmap" (including core portions of the ITRS, as feasible) that extrapolates manufacturing and design technologies and their implications in a transparent, self-consistent way.

4 CAD-IP Reuse

How can we quickly develop the right design technology so that it is available at the right time? As noted above, the "time-to-market" for new design technology – from understanding the design problem to integration of a new solution in mainstream flows – can span several process generations and the birth and death of entire electronics markets. Our goal is to facilitate the development and deployment of new design technology via (i) new IT infrastructure to enable "intellectual property reuse" in the algorithms area, (ii) "social engineering" within the R&D community, and (iii) fundamental technical advances in multiple application domains. An underlying tenet is that uniform representations, appropriate levels of generalizability and reusability, and careful substantiation of effectiveness can facilitate exponential improvements in both the quality and the development time of complex algorithm implementations that are built today.

The need to improve effectiveness of CAD algorithms research is seen not only within phrases such as "design productivity gap", but in more concrete ways as well. (1) At a more technical level, fragmentation of research across too many subcommunities creates a need to regain a well-defined "leading edge": indeed, the 1999 Design Automation Conference instituted a new topic area, **T0.1: Fundamental CAD Algorithms**. (2) At the level of individual problem formulations (specifically, hypergraph bipartitioning), [7] notes cases of undocumented implementation decisions that can change quality of result by 400+%, and the existence of recent works that report 1000+% differences in solution costs returned by implementations of the same well-known algorithm.

Why Reuse? The standard criteria for technology delivery are *time-to-market* and *quality-of-result* achieved within given *resource constraints*. We observe that all of these criteria strongly motivate a culture of reuse throughout the design technology community.

- With respect to time-to-market, design technology productivity improves with better software development processes: "expected" time-to-market grows with project risk, which in turn typically grows with the amount of code that must be written from scratch. Hence, if it is feasible to cheaply reuse existing codes that have documented performance and a history of successful reuse, risk can be lowered, reducing time-to-market even if some new features or interfaces need to be added.

- With respect to quality-of-result (QOR), even the measurement of QOR requires deep knowledge about algorithms and software tools, including common benchmarks, evaluation methodologies, known-good performance results, etc. This leads us to consider a generalized form of reuse, beyond mere code reuse: i.e., reuse of *intellectual property* in CAD (*"CAD-IP reuse"*). The above-cited analysis of hypergraph partitioning [7] shows what can happen when *any* aspect of the "leading edge" is unclear. Such risks increase with maturity of the research domain and size of the literature – in other words, with the importance and attention devoted to a given problem. When individual researchers can no longer keep track of all relevant research in the field, there is a risk of poorly reinventing the wheel. Infrastructure is needed that can clearly identify the best results in the field at any given time; such an infrastructure would implicitly reuse accumulated knowledge about standard benchmarks, evaluation methodologies and performance comparisons. Design technology research also tends to focus on narrow optimizations, e.g., efficient manipulation of binary decision diagrams in logic synthesis. Without the ability to evaluate new results in the context of full design flows ("vertical benchmarking"), the utility of research suffers. A shared public infrastructure for evaluation of specialized algorithms could bring great benefits with small amortized cost to the community.

- Finally, with respect to the short supply of design technology R&D engineers, commonly proposed solutions focus on increasing the supply of engineers via, e.g., outreach to graduate-level educational programs. But at the same time, many companies and research groups reimplement such basic software components as format readers/writers, delay calculators, circuit partitioners, technology mappers, etc. For mature and non-differentiating technology, there is tremendous waste inherent in reimplementation. Reinforcing the trends noted in Section 1, we strongly believe that resource imperatives alone will force data models, polygon database implementations, placers and routers, etc. to inevitably evolve into non-differentiating, commoditized (and nearly free)"foundation IP" (following the course of operating systems, data structures or GUI components before them). Such an evolution of the design technology community is consistent with both the culture of *coopetition* (cf. the history of semiconductor manufacturing and consortia such as SEMATECH) and the impetus toward *disaggregation* (limited resources are allocated to where they add the greatest value).

In summary, (free) intellectual property (including software) reuse has positive impact on all aspects of design technology productivity: time-to-market, quality-of-result, and effective use of given R&D resources. The design technology community requires an adequate level of software reuse, as well as IP reuse in the form of convenient and consistent evaluation methodologies to identify the best available solutions for important problems. Without such infrastructure and practices, the field is hobbled by unnecessary barriers to research progress, as well as unnecessary barriers to the identification and adoption of appropriate research results for use within production design tools and methodologies.

The IP Reuse Paradigm, and CAD-IP. There are strong links between between our proposed objectives and mechanisms, and the successful paradigm of *intellectual property (IP) reuse*. This can be seen via a simple analogy between design/implementation of hardware and design/implementation of applied algorithms. In the (VLSI) design context, the two key goals are time-to-market and quality-of-result. Challenges to design success therefore arise from the increasingly complex electrical engineering, optimization and constraint satisfaction problems that are inherent in the design process. Mistakes become very costly, yet almost inevitable. To make matters worse, the complexity of verification and test grows relative to that of other design tasks. The classic response to the VLSI design complexity challenge is *design IP reuse*, which includes (i) modular design IP explicitly created for reuse, (ii) formal or informal certification and "socketization" of modular IP, (iii) "matching services" that connect IP providers with prospective users, and (iv) IP protection (for commercial applications).

For VLSI CAD tools providers – i.e., the innovators, packagers and purveyors of applied algorithmics results – the goals and challenges are strikingly similar to those faced by VLSI designers: time-to-market (publication, graduation) and quality-of-result are hampered by the complexities of optimization, constraint satisfaction and software engineering. (Accounts of 7-year development cycles from "research result" to "production tool" are not uncommon.) Mistakes can

be show-stoppers since verification and test of both software systems and optimization heuristics are poorly understood. To successfully respond to these challenges, we propose that the (VLSI CAD) applied algorithmics community adopt *(CAD-)IP reuse.*

Just as a design IP has many dimensions (test harness, simulation vectors, library/technology files, constraints, layout geometries, etc.), so does CAD-IP. In reference [8], we have recently identified the following components of CAD-IP. (1) Data models that provide consistent semantics and structuring of data along different steps of design flows (ideally with an accompanying canonical API). (2) Mathematical problem formulations for optimization and constraint satisfaction that isolate the fundamental difficulties in particular design tasks, and that encourage reuse of solvers. (3) Use models and context descriptions for problem formulations. (4) Testcases with corresponding high-quality solutions – both for individual problem formulations and integrated tool flows. (5) Algorithm descriptions and theoretical analyses. (6) Executable implementations, not only for solvers (algorithms) but also for parsers and converters of interchange formats, legality checkers and cost function evaluators of solutions, etc. (7) Leading-edge performance results, as well as standard comparison and evaluation methodologies for algorithms. These are needed to ensure that newly proposed methods are indeed improvements over previous methods. (8) Software design and implementation methodologies. (9) Source code of implementations.

It is important to realize that neither source codes nor executables, nor the traditional medium of written algorithm descriptions (e.g., 6 pages of 2-column 9-point format in a proceedings) dominates CAD-IP. Driving testcases, a range of executables that goes beyond only solvers, appropriate use models and evaluation methodologies are all critical enablers to the IP reuse that can restore efficient progress of the field.

The GSRC Bookshelf. We have developed over the past 12 months a prototype instantiation of a "new publication medium" in the VLSI CAD domain, under the auspices of the MARCO GSRC. The resulting GSRC Bookshelf of Fundamental CAD Algorithms has been a valuable testbed for evaluating the strength of various social and cultural barriers, as well as our assessments of infrastructure priorities. The Bookshelf project maintains an active presence on the Web (see `http://gigascale.org/bookshelf`), which is currently focused on algorithm implementations, evaluation and related information. Slots in the MARCO GSRC Bookshelf are listed at `http://gigascale.org/bookshelf/Slots/`.[6] A number of leading-edge implementations are freely available at the Bookshelf site, accompanied with performance results on standard benchmarks.[7]

5 Design Process Instrumentation and Optimization

Did we really solve the problem? Did the design process improve? In automobile, steel, and even semiconductor manufacturing industries, process optimization is a well-established precept. However, before a process can be optimized on a continuous basis, it must first be *measured.* We observe that **in contrast to other industries, today there are no standards or infrastructure for measuring and recording the semiconductor design process.** Hence, today's design processes tend to be temporary solutions, unique to individual projects and created based on the intuition of senior engineers. Such solutions typically last for one project only, while the basic problem of unpredictable design success remains unaddressed. In this regime, a product team cannot quantify inefficiencies in its design process, and subjective opinions are formulated as to why a given project failed or succeeded (e.g., failure may be generically blamed on "lousy CAD tools" or "inexperienced design team"). Two fundamental gaps prevent measurement of the design process.

- **Data to be measured is not available.** Most visibility into EDA tools is via log files that are typically created by R&D developers for their own use; these vary wildly across different vendors and offer little insight into "what the tool is thinking" or what aspects of the input instance were critical to success/failure.
- **We do not know all the data that should be measured.** Some metrics of tool performance or design artifacts are "obvious", e.g., number of placeable objects, number of unroutes after detailed routing, maximum negative slack over all timing paths, etc. Other metrics are less obviously useful, e.g., it is not clear whether the number of literals after logic optimization has any relationship to the quality of the resulting netlist from a physical implementation perspective. Finally, still other metrics are impossible to discern *a priori*, e.g., perhaps it is the number of spec changes that is the best indicator of project outcomes.

We see that designers cannot obtain necessary design process data because EDA vendor tools do not report the data. On the other hand, EDA vendors do not necessarily know which data is useful to report. These gaps prevent designers and project managers from finding and correcting inefficiencies in their design processes. *Measurement infrastructure is a necessary condition for measuring, then improving.* The GSRC METRICS initiative [4, 13] provides the following open standards and infrastructure (see Figure 3).

- Standard *generic tool metrics*, as well as standard *domain-specific tool metrics* (e.g., for front-end simulation, timing optimization or P&R), that the design technology community can standardize on. Unified naming and semantics of common metrics allow multiple tool developers to report metrics according to the same meanings and conventions.
- Standard system components such as XML (eXtended Markup Language) based metrics transmitters, an Oracle8i-based data warehouse with a standard metrics schema, and Java implementation of a metrics server. This infrastructure enables design process data collection in a "no more log files" manner: design tools and flow scripts transparently write into the design process data warehouse over an inter/intranet, via METRICS-specific standard XML.
- Examples of useful datamining or statistical analyses and reports that have been developed on top of an existing METRICS system implementation.

From a *project management* perspective, the METRICS system offers the potential for (i) accurate resource prediction at any point in the design cycle, (ii) correct go / no-go decisions for projects at earliest possible junctures, (iii) accurate project post-mortems, and (iv) optimization of future projects based on past results. From a *tool development* perspective, benefits include: (i) methodology for continuous tracking data over the entire lifecycle of instrumented tools, (ii) more efficient analysis of realistic data, and real benchmarking via standardized metrics, (iii) easier identification of key design metrics and their effects on tools, as well as design-process relevant optimization objectives, and (iv) identification of tool "sweet spots" (i.e., appropriate field of use) and critical instance attributes. METRICS is not limited to recording of tools-specific or design instance-specific attributes: it includes other design process-related metrics, such as communication

[6] As of November 2000, content is available in the following slots: *Verilog Tools and Resources, Circuit and Microprocessor Design Examples, Fundamental, Generic Optimization, Decision Diagrams, Boolean Satisfiability, Graph Coloring, Network Flow, Computational Geometry, Hypergraph Partitioning, Block Packing, Wirelength-Driven Standard-Cell Placement, Steiner Trees, Bounded-Skew Clock Routing, Grid-Based Global and Detailed Routing, and Single Interconnect Tree Synthesis.* The contents have been created by researchers from UCLA, UC Santa Cruz, University of Illinois at Chicago, University of Michigan, SUNY Binghamton, Trier University, Arizona State University, University of Colorado at Boulder, Technical University of Lisbon, University of Minnesota, Northwestern University and other researchers active in these domains.

[7] One particular open-source release hosted in the Bookshelf — *"UCLA Physical Design tools"* [24] — has been downloaded over 350 times, including to sites at most major electronic design automation tools (EDA) companies worldwide, and to a variety of academic and industrial sites in ten countries. The codes contained in the release are also being used in production code by several EDA companies.

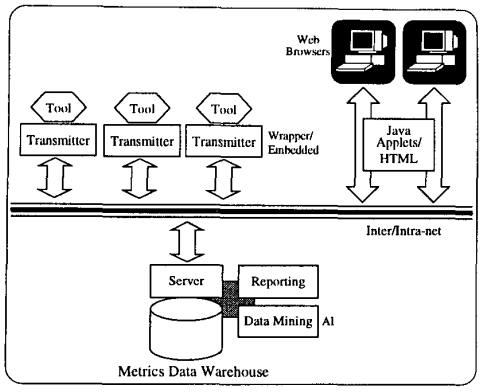

Figure 3: METRICS system architecture.

metrics, tool metrics, design artifact metrics, and design flow metrics. For example, basic instrumentation of the design process would record such information as which version of what tool was called on what revision of what block, by which design engineer at what time on what machine, etc. Given this scope of data collection, the possibilities for ensuing design process optimizations are literally unbounded

6 Summary

The well-lamented Design Productivity Gap is actually identifiable with the Design Technology Productivity Gap. To solve this, the design community must be able to correctly identify the design technology need, and to deliver this technology at the right time with quantified impact. Three recent initiatives in the MARCO Gigascale Silicon Research Center (GSRC) support the design technology and designer communities in reaching these goals. We believe that the following impacts are possible. (1) Accurate roadmapping (technology extrapolation) can help focus already-strained R&D resources onto the most important problems facing the design technology community. (2) Reusable, commodity, foundation CAD-IP (including data models and access methods, intermediate format readers/writers, vertical benchmarks etc.) and accompanying academic publication standards can reduce time-to-market of new design technology, better leverage available R&D and academic resources, and increase the "searched solution space" of design processes through more mix-and-match flow optimizations. Infrastructure such as the Bookshelf can provide new conduits for industry to induce more relevant academic research. (3) Design tool and process metrics, design process instrumentation, and continuous process improvement can additionally improve the "searched solution space" of flow and process optimizations.

7 Acknowledgments

The viewpoint presented in this paper has been influenced by many sources. Many inputs have come from colleagues in the MARCO GSRC, including Richard Newton, Kurt Keutzer, Gary Baldwin, Larry Pileggi, Alberto Sangiovanni-Vincentelli, Wojciech Maly, and Wayne Dai. Key individuals behind the three initiatives (Technology Extrapolation, CAD-IP Reuse, and METRICS) include Igor Markov, Stefanus Mantik, Mike Oliver, Andrew Caldwell and Dirk Stroobandt (for these initiatives, descriptions used in the cited references have been adopted wholesale here). Ted Vucurevich and Jim Hogan at Cadence Design Systems have always provided feedback on these and other ideas. Last, an earlier version of this viewpoint was given in a keynote address at the 12th Japan DA Show in Tokyo, July 2000.

References

[1] C. J. Alpert, "The ISPD-98 Circuit Benchmark Suite", in *Proc. ACM/IEEE International Symposium on Physical Design*, pp. 80-85, 1998. See errata at `http://vlsicad.cs.ucla.edu/~cheese/errata.html`.

[2] C. J. Alpert, "Partitioning Benchmarks for the VLSI CAD Community", `http://vlsicad.cs.ucla.edu/~cheese/benchmarks.html`.

[3] R. S. Barr, B. L. Golden, J. P. Kelly, M. G. C. Resende and W. R. Stewart, "Designing and Reporting on Computational Experiments with Heuristic Methods", *technical report* (extended version of *J. Heuristics* paper), 1995.

[4] `http://vlsicad.cs.ucla.edu/GSRC/`.

[5] A. E. Caldwell, Y. Cao, A. B. Kahng, F. Koushanfar, H. Lu, I. L. Markov, M. R. Oliver, D. Stroobandt and D. Sylvester, "GTX: The MARCO GSRC Technology Extrapolation System", in *Proc. ACM/IEEE Design Automation Conf.*, 2000, pp. 693-698.

[6] A. E. Caldwell, A. B. Kahng, S. Mantik, I. L. Markov and A. Zelikovsky, "On Wirelength Estimations for Row-Based Placement", *IEEE Trans. on Computer-Aided Design*, Vol. 18(9), 1999, pp. 1265-1278.

[7] A. E. Caldwell, A. B. Kahng and I. L. Markov, "Hypergraph Partitioning for VLSI CAD: Methodology for Heuristic Development, Experimentation and Reporting", in *Proc. ACM/IEEE Design Automation Conf.*, 1999, pp. 349-354.

[8] A. E. Caldwell, A. B. Kahng and I. L. Markov, "VLSI CAD Bookshelf", 1999, `http://vlsicad.cs.ucla.edu/GSRC/bookshelf/`. See also "Toward CAD-IP Reuse: The MARCO GSRC Bookshelf of Fundamental CAD Algorithms", to appear in *IEEE Design and Test*, 2001.

[9] Y. Cao, C. Hu, X. J. Huang, A. B. Kahng, S. Muddu, D. Stroobandt and D. Sylvester, "Effects of Global Interconnect Optimizations on Performance Estimation of Deep Submicron Design", in *Proc. IEEE/ACM Intl. Conference on Computer-Aided Design*, 2000, pp. 56-61.

[10] H. Chang et al., *Surviving the SOC Revolution : A Guide to Platform-Based Design*, Boston, Kluwer Academic, 1999.

[11] C. -K. Cheng, A. B. Kahng, B. Liu and D. Stroobandt, "Toward Better Wireload Models in the Presence of Obstacles", in *IEEE/ACM Proc. Asia South-Pacific Design Automation Conf.*, 2001.

[12] J. Cong and M. Sarrafzadeh, "Incremental Physical Design", in *Proc. ACM/IEEE International Symposium on Physical Design*, 2000, pp. 84-92.

[13] S. Fenstermaker, D. George, A. B. Kahng, S. Mantik and B. Thielges, "METRICS: A System Architecture for Design Process Optimization", in *Proc. ACM/IEEE Design Automation Conf.*, 2000, pp. 705-710.

[14] Richard Goering, "Giga-center redefines chip design for new millennium", *EE Times*, 2000, `http://www.eet.com/story/OEG20000103S0011`.

[15] The Gigascale Silicon Research Center, `http://gigascale.org`.

[16] International Technology Roadmap for Semiconductors, 1999, `http://public.itrs.net/`.

[17] A. B. Kahng, "Classical Floorplanning Harmful?", in *Proc. ACM/IEEE International Symposium on Physical Design*, 2000, pp. 207-213.

[18] A. B. Kahng and S. Mantik, "On Mismatches Between Incremental Optimizers and Instance Perturbations in Physical Design Tools", in *Proc. IEEE/ACM Intl. Conference on Computer-Aided Design*, 2000, pp. 17-21.

[19] A. B. Kahng, S. Muddu and E. Sarto, "Tuning Strategies for Global Interconnects in High-Performance Deep-Submicron ICs", *VLSI Design*, Vol. 10(1), 1999, pp. 21-34.

[20] A. B. Kahng and Y. C. Pati, "Subwavelength Lithography and its Potential Impact on Design and EDA", in *Proc. ACM/IEEE Design Automation Conf.*, 1999, pp. 799-804.

[21] S. P. Khatri, A. Mehrotra, R. K. Brayton, A. Sangiovanni-Vincentelli and R. H. J. M. Otten, "A Novel VLSI Layout Fabric for Deep Sub-Micron Applications", in *Proc. ACM/IEEE Design Automation Conf.*, pp. 491-496, 1999.

[22] M. D. Levenson, N. S. Viswanathan and R. A. Simpson, "Improving Resolution in Photolithography with a Phase-Shifting Mask", *IEEE Trans. on Electron Devices* ED-29 (1982), pp. 1828-1836.

[23] W. Maly, "High Levels of IC Manufacturability: One of the Necessary Prerequisites of the 1997 SIA Roadmap Vision", in *Proc. IEDM*, 1998, pp. 759-762.

[24] UCLA Physical Design tools, `http://vlsicad.cs.ucla.edu/software/PDtools`

[25] J. A. Rowson and A. Sangiovanni-Vincentelli, "Interface-based design", in *Proc. ACM/IEEE Design Automation Conf.*, 1997, pp. 178-183.

[26] Rensselaer Interconnect Performance Estimator (RIPE), `http://latte.cie.rpi.edu/ripe.html`

[27] J. C. Eble, V. K. De, D. S. Wills and J. D. Meindl, "A Generic System Simulator (GENESYS) for ASIC Technology and Architecture Beyond 2001," *Proc. ASIC Conf.*, 1996, pp. 193-196.

[28] G.A. Sai-Halasz, "Performance Trends in High-Performance Processors," *Proc. IEEE*, Jan. 1995, pp. 20-36.

[29] H.B. Bakoglu, *Circuits, Interconnections, and Packaging for VLSI*, Chapter 9, Addison-Wesley, 1990.

[30] D. Sylvester and K. Keutzer, "Getting to the Bottom of Deep Submicron," *Proc. ICCAD*, 1998, pp. 203-211.