From Single Core to Multi-Core: Preparing for a new exponential

Jeff Parkhurst
Intel Corporation
Folsom, CA 95630
1-916-356-2508
Jeff.parkhurst@intel.dot.com

John Darringer
IBM T J Watson Research Center
Yorktown Heights, NY 10598
1-914-945-2742
jad@us.ibm.com

Bill Grundmann
Intel Corporation
Hillsboro, OR 97123
1- 971-214-1769
bill.grundmann@intel.dot.com

ABSTRACT

In the past, processor design trends were dominated by increasingly complex feature sets, higher clock speeds, growing thermal envelopes and increasing power dissipation. Recently, clock speeds have tapered and thermal and power dissipation envelopes have remained flat. However, the demand for increasing performance continues which has fueled the move to integrated multiple processor (multi-core) designs. This paper discusses this trend towards multi-core processor designs, the design challenges that accompany it and a view of the research required to support it.

1. INTRODUCTION

In April 1965, Gordon Moore wrote an article for Electronics magazine titled "Cramming more components onto integrated circuits" [1]. He predicted that the number of transistors on a chip would double every 12 months into the near future. Although this exponential trend has "tapered" to doubling transistors every 18 months, it remains the driving force behind the integrated circuits industry including memory, microprocessors, and graphics processors and has become known as Moore's law. This law over the years has provided a roadmap for product designers as they plan efficient and effective usage of the transistors at their disposal. It has stood the test of time predicting an exponential trend for over 40 years. Process scaling has been the underlying enabler of this trend starting in the early 90's and continuing to today. Starting with 0.8 um process circa 1992, process scaling enabling feature size reduction by a factor of 0.7 has occurred approximately every 24 months. This trend along with Moore's Law has spawned a number of exponentials all in the name of increasing performance. However, not all these residual exponentials can claim this longevity and many are not nearly as benign as Moore's Law.

2. EXPONENTIALS PRIOR TO MULTI-CORE

Consider clock frequency which was on an exponential trend in the mid 90's. From about 1993 with the Intel® Pentium® processor and continuing through mid 2003 with the Intel® Pentium® IV processor, clock frequency doubled every 18 months to 2 years. This was a driving force for increasing performance of microprocessors during this timeframe. However, due to increased dynamic power dissipation and design complexity, this trend tapered with maximum clock frequencies around 4GHz. Along with increased dynamic power, static power continues to increase due to transistor source to drain leakage along with gate leakage. This leakage has been exponentially increasing with scaling but has only recently been a concern as it became a significant portion of the overall power budget. Circuit designers have used stacked gates, body bias, and sleep transistors to mitigate the S-D leakage problem and high K dielectrics to address the gate leakage problem.

Power density is another exponential closely associated with power dissipation and decreasing feature size. Both power dissipation and power density trends have essentially flattened by requiring designers to remain within a particular power budget and relaxing density requirements.

Voltage scaling began in the early 90's when processor supply voltages began to deviate from the 5V standard. This scaling was required to avoid oxide stress as oxide thicknesses scaled with transistor feature size. Gate capacitance scaled with feature size as well resulting in overall lower power dissipation and counteracted the adverse affects on power dissipation by increased clock frequency. Threshold voltage was also required to scale with power supply in order maintain transistor performance. However, decreasing the threshold voltage led to increased sub threshold leakage requiring designers to rethink the tradeoff between increased performance with lower thresholds and increased leakage leading to larger static power dissipation. In the end, voltage scaling was short lived as was threshold voltage scaling to better control leakage and overall static power dissipation.

Design complexity also continued on an exponential trend. Although it is very difficult to define a metric which encapsulates the resultant design effort, the increased design complexity has revealed itself in the past via exponential design team growth. Consider the feature set progression over the last several years including speculative execution followed by speculative execution and branch prediction. Soon after, dynamic execution was introduced which included the aforementioned features plus super scalar and out of order execution. Then came multi-threading followed by hyper-threading. All of these features were added to increase the performance of the microprocessor at the expense of increased design and validation complexity.

ICCAD'06, November 5-9, 2006, San Jose, CA

Low-end and embedded processors have historically trailed in micro-architectural and performance enhancements. Many of these enhancements such as increased pipelining and increased cache size exhibit diminishing returns in the light of increased area and power consumption. This has allowed low-end and embedded processors to close the overall performance gap while providing a reasonable tradeoff of area and power. When considering the limitations associated with voltage supply scaling, threshold scaling, and clock frequency scaling, along with design complexity increases, companies were already looking for an alternative to the single core paradigm. Multi-core was therefore the natural next evolutionary step in staying on the ever increasing performance driven curve. In the next section, the multi-core design will be discussed along with how it addresses the need for increased performance while abiding by strict power dissipation guidelines.

3. THE MIGRATION TO MULTI-CORE

The first microprocessor had only 2200 transistors. Through the 1980s and 1990s and on to today, increasing performance has been the driver. Designers have been limited only by the number of transistors at their disposal. Only recently has power become the over riding limiting factor. One of the byproducts of technology scaling is that the cost per transistor continues to decrease. If we assume that as we continue to scale, transistors become basically "free", we open new avenues to achieve performance without breaching power dissipation requirements. Parallelism is one of the best ways to address the issue of power while maintaining performance where higher data throughput may be achieved with lower voltage and frequency. The result is a larger transistor count, but overall lower power dissipation and power density. This is one of the ideas behind the movement to multi-core.

Other advantages with multi-core are inherent redundancy which lends itself to resilient architectures. Instead of binning based upon speed, one could bin products based upon the number of working cores or overall data throughput. Also, the integration of multiple cores on a chip allows for lower interconnect latency and therefore higher bandwidth between cores than their discrete counterparts. Finally, time to market is a key driver in this industry along with increased performance. Approaching this design problem from an SoC perspective provides for IP re-use of cores and overall reduced design effort.

Multi-core may take on a number of forms. One form would be dedicated heterogeneous cores that could address the variety of applications executed by the computer. For instance, a DSP core could address multimedia applications, a complex core could address computationally intensive applications, and finally a remedial core could address less computationally intensive applications like spreadsheets and word processing. Another possibility would be a large number of remedial homogeneous cores which divide and conquer computationally intensive applications and yet individually address less computationally intensive applications. This would be a natural extension of using the embedded or low-end performance processor families talked about in the previous section. Multi-core could also take the form of few complex homogeneous cores in which a single core could multitask between several remedial applications or individually handle computationally intensive applications. Of course, it could end up being a mix of these options. Consider the CELL microprocessor chip which contains a single "general purpose" microprocessor and eight area and power efficient accelerators targeted for specific applications [2].

The factors determining which form multi-core designs will take is based upon which design (a) best provides maximum throughput (b) meets DFM/DFY requirements, (c) meets thermal and power envelopes, (d) meets a cost target, and (e) allows for validation. On this last point, it cannot be overemphasized that DFT and Design for Validation (DFV) must be foremost in the design methodology due to limited I/O and the inability for complete controllability and observability. In any case, no matter what form multi-core designs take, they all allow for natural partitioning for power down of unused cores and overall power dissipation savings.

This doesn't mean that all questions concerning this move have been answered. For instance, the performance increase in part is directly related to software applications effectively using the multi-core hardware. HW/SW co-design will take on new importance. Also, multi-core allows the opportunity for IP re-use. No longer is increased processor core complexity required for higher performance. Instead, higher performance is obtained by adding cores which could require only minor changes from previous generation designs. This will require discipline on the part of the designer to avoid the tendency to optimize an already design compliant core. Design complexity in the circuitry that controls communication between the cores and polices cache access, could also begin to grow exponentially which will be discussed in the next section. Finally, integration provides higher performance and lower cost advantages driving the integration of graphics, wireless, and other functions into a true SoC design. Achieving this on a high performance CMOS "digital" process requires further research.

4. CHALLENGES FACING MULTI-CORE

Assuming that designers are able to achieve high percentage core re-use in next generation designs, exponentials directly related to core design complexity diminish. However, new challenges arise with respect to the non-core or glue logic of the design. Whether we assume a dedicated cache per core or cache sharing, the need to maintain cache coherency will create added complexity. In the near term, there likely will not be enough available transistors to allow for a distributed cache scenario forcing cache sharing. In this case, complexity will increase as designs try to manage access as well as cache hits and misses.

Glue logic will be required for optimal communication between the cores. Architectural exploration will be required to comprehend tradeoffs of various interconnect strategies and protocols. If this can be made modular, it would lend itself to scalability as well as re-use. Re-use is the key to both handling the exponential growth of RTL, its resultant design complexity, and its subsequent validation. On this last topic, post silicon functional and performance validation complexity increases are unavoidable due to limited I/O which prevents complete controllability & observability. One core could be used to test another at the full chip level, but individual blocks will still need to be tested for fault diagnosis and self repair purposes. With increasing transistors comes increased test vectors and increasing difficulty to reach an acceptable fault grade. All these reasons create a strong need for on chip self test, diagnosis, and repair for cores,

cache, and glue logic. In the past, validation and design have been considered separate entities and yet success in the future is dependent upon tight synergy between the two. The enabling of test, diagnosis, and self repair must be considered part of the overall design. The danger, of course, is that while increasing the number of cores along with glue logic, graphics, wireless, and other functions, and then adding self test circuitry, designs move back on to a new exponential power dissipation and density growth curve obviating the advantages of moving to multi-core designs in the first place.

Software will take a more prominent role in the multi-core era. Reference [3] states that the new exponential will be the increasing number of cores on chip. However, there will be diminishing returns in this paradigm if software applications do not fully utilize the processing power at their disposal. Software's ability to efficiently partition and distribute workload among the cores will be the key enabler. The Research Accelerator for Multiple Processors (RAMP) project is one example of academia's response to this need [4].

Besides component cost, TTM, system performance and power, micro-architects must design products with increasingly unreliable components due to ever increasing process parameter variations and soft-error rates. This will force micro-architecture to think about how to make designs that are resilient to any intermittent or permanently undependable sub-component in addition to achieving normal logic behavior and performance. They will have to utilize different forms of redundancy and logic replay along with a host of continuous self-checking error-detection techniques to make products more resilient to error with increased product reliability.

The availability of different 3D silicon methods (stacked die and layered-transistor die) provide both product advantages and challenges. Both of these promise the possibility of allowing the choice of using the best technology for a particular function without incurring huge interconnect latency and pin utilization constraints. For example, this would allow the use of stacked or layered DRAM technology for implementing local DRAMs instead of compromising digital technology to accommodate on-die DRAMs. Analog on analog technologies or I/O where its voltage operation is significantly different than that of the digital core is another example. The challenge becomes managing the large technology canvas to execute a design and the resultant exploding solution permutation choices. Functions that normally would simply be built in a common die might be better spread between technologies. Architectural planning, analysis, and assessment in 3D across potentially heterogeneous processes will be essential. In addition, controlling the ROI due to the increase in manufacturing cost caused from mixing heterogeneous technologies will have to be assessed.

Another challenge is how to better accommodate the ever increasing role of on-chip communication latency, both in protocol and simple interconnects. Chip architects generally decide very early in a design project's schedule exactly how a design is going to behave. They make these decisions based upon their past design experiences, new research discoveries, and feedback on implementation feasibility from silicon circuit designers. Experienced chip architects combine expertise in silicon physical floor-planning with some amount of initial global placements and routing of major functional logic blocks. From all of these inputs, the chip architect decides how the function is to be spatially spread out physically, how the function is partitioned between pipelined stages, and how much latency is involved in interconnect. For many complex projects in the past, these decisions weren't based upon reality as much as based upon an educated guess due to the lack of perfect data which required the actual building of the part. However, building the part properly required better data than just an educated guess. Herein was the paradox.

Early micro-architectural decisions create constraints for those actually designing the silicon. When silicon design performance and power was dominated by transistor effects this logic-centric approach was fine. Now with interconnects becoming a major factor for performance and power, these premature guesses are less accurate. Consider the following two implementations of the same function as shown in Figure 1. The position of blocks "A" and "B" change the optimal amount of latency between the blocks to maintain the same clock frequency. In one physical partition, this particular latency has to be 3 cycles but in the other it can be 1 cycle.

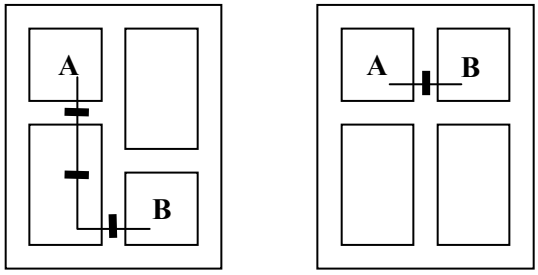

Figure 1. Physical partitioning affect interconnect

Which latency choice is actually better for a product depends upon many other factors for the design. For example, a choice of longer latency for non-critical performance paths may make room for easing the design of a critical performance path. Ultimately, the choice of latency is dependent upon physical partitioning and hence, we have the circular argument of what comes first – the design partitioning and latency or the physical design?

A related challenge for micro-architects is the ease for them to evaluate multiple solutions for a product. It is interesting to note that the number of evaluated permutations of how the design could be built does not track the complexity of a design. In most cases the evaluated cases are about the same from generation to generation of products while complexity trends continue to follow Moore's Law type exponentials. Part of the evaluation problem is the lack of standard methodologies and tools for specifying architectural experiments. This causes the architect to roll their own solutions and painfully create each individual experiment. These experiments have to accommodate rapidly changing functional specification, extensive architectural object libraries, changing functional partitioning, massive re-pipelining of functional stages, iteration of potential physical design, and allow seemingly arbitrary changes to any latency, estimation of power and performance, as well as enable behavioral equivalence to some higher product requirement. These experiments will need standards for design specification, which has to be a higher abstraction than RTL.

5. DESIGN AUTOMATION CHALLENGES

Just as the movement to multi-core systems is essential for system capability to keep pace with demand and the historic growth, further innovation in EDA is required for designers to be able to deliver these multi-core products to the market. Over its lengthy history, design automation has delivered advances in productivity to keep up with silicon with the big steps coming through raising the level of abstraction. Design has progressed from transistor-level to gate-level and on to register-transfer-level, but the industry has been stuck on RTL for decades and it is time to move to a higher level of abstraction. This section first looks at the challenges in the traditional Post-RTL phase of design, then discusses how design automation technology is needed and can be applied to the Pre-RTL phase.

5.1 Post-RTL Implementation

Nearly all of the EDA industry and design automation research is focused on the monumental and ever increasing challenges of converting an RTL specification into a working chip. As discussed above, "timing closure" has evolved into "design closure" that requires the simultaneous optimization of power, performance, cost and reliability. Large high-performance chip design can only be accomplished with a design system that integrates analysis and optimization applications around a common representation of the design stored in memory. File I/O or data translations can not be tolerated in the middle of an optimization loop. Today the EDA vendors are offering integrated systems with their own tools, but the industry needs to integrate the best applications regardless their origin. The Open Access data model and API [12] are aimed at enabling this broader and necessary tight integration.

The second major challenge is the integration of design optimization with manufacturability concerns. Today there are several vendors offering tools to address "design for manufacturability" and eventually the effective techniques will need to also be tightly integrated into the design closure process.

It is not the focus of this paper to inventory the many challenges facing the effective implementation of large complex chips. These issues are well described in the International Technology Roadmap for Semiconductors (ITRS) [5] and leading EDA conferences. Instead the following section discusses some of the ways the structure of multi-core designs can simplify the implementation process by leveraging techniques used in ASICs and SoCs while retaining some of the advantages of the costly custom design methods used today.

5.1.1 Design Reuse

Clearly implementing a chip by assembling reusable components will greatly reduce design time. Of course, this strategy depends on the availability of a library of well architected components that can be configured for a broad range of applications. The library needs to contain not only processors, but accelerators for diverse applications, memory controllers for on- and off-chip memory as well as a set of cores to communicate off-chip. In addition, there needs to be a flexible interconnect method to handle diverse needs from high bandwidth communication to linking pervasive functions such as test and system bring-up. There are challenges for architects and researchers to define the right components, large and small, for composing diverse systems; as well as a need for an improved ability to specify parametric physical structures, such as high bandwidth interconnects.

5.1.2 Validation Reuse:

In most system projects, the largest consumer of resources and time is validating that the system does what was intended via simulation and the careful use of formal methods. Building systems from reusable components that have been subjected to intensive testing can reduce the number of errors in initial designs. Adapting simulation and formal methods to take advantage of reused components is still an open and important area of research. What is needed is the ability to identify the properties of the reused component necessary to formally verify an assembled system.

5.1.3 Layout Reuse

A related benefit to design reuse is the ability to reuse the mask data repeatedly within a design or for derivative chips in the same technology. This is a major productivity gain, especially for high performance chips that contain a great deal of custom layout. Reusing the global interconnect may be more problematic, depending on the style used. Here parametric models or interconnect generators may be the best approach. High speed interconnect together with the on-chip memory hierarchy can be quite application specific and confound reuse. The key is carefully architected parametric components that can support a broad range of applications.

With a significant amount of reuse, large chip design becomes more of an integration task, creating an opportunity for new chip integration tools that operate at higher levels of abstraction and increase productivity.

5.1.4 Test Reuse

With increased leakage and variability, test in the future is a major challenge. Larger multi-core chips with limited I/O further compound the task of test and bring-up after manufacture. Given the reuse of cores, there is an opportunity to reuse tests. In addition, the ability to exploit the multiple processors available on-chip could enable efficient and effective self test to support these large designs. The interconnect fabric presents additional test challenges such as confirming that high speed links which are potentially asynchronous do work correctly at speed. This will require further research to explore the diversity of configurations that may be used.

5.2 Pre-RTL Exploration

Assuming that these large multi-core chips can be implemented, there remains the very important question: "Which design shall we implement"? The fact that processor performance does not simply scale with each new technology generation means that chip and system architects must explore many more novel designs to find the best way to provide the performance demanded in the marketplace. Section 4 described how difficult this task can be for architects working on multi-core designs.

Reuse again can help. When there is a high degree of reuse, the architect is primarily searching for a configuration of existing cores to meet a specific need. As components are reused, models can also be reused with the confidence gained from their earlier application.

But often in the early stages of design, architects are thinking of new ideas for a component that is not in the library. In this case, parametric models or models generated from high-level specifications, or a user friendly language for system modeling are necessary. At times, it is important to incorporate a core designed externally and use the models provided. Fortunately, there appears to be progress on standards in this area: SPIRIT [10] for defining a core's interface and SystemC [11] for defining a core's behavior.

5.2.1 Performance Estimation

Today design tradeoffs are usually facilitated by a modeling team that is expert in writing custom models to predict system performance for traditional software workloads. Often these models are optimized for run-time performance, to handle large traces in manageable times and as a result they do not follow the system organization, sometimes making it difficult to reuse the model or to answer unanticipated questions about specific hardware decisions. To take advantage of design reuse, future performance models should be more modular and aligned with the library of components. Then the task of building a system performance model is a matter of assembling the corresponding performance models. Creating the component performance models is still a daunting task, but SystemC is gaining acceptance in the SoC market because of its modelingl facilities and its wide availability.

5.2.2 Power Modeling

Although performance was mentioned first, it is power that dominates chip trade-off discussions today. Reasonably accurate methods exist for predicting power dissipation when the implementation is complete or near complete. But at the early pre-RTL stages of design, estimates of power are often based on spreadsheet calculations. As a result, there can be surprises later in design that require significant rework. When using an existing component, calibrated power models can be applied. Even in new designs, some smaller pre-characterized components can be used with reasonable accuracy. There is work on predicting power at the "system-level" before an RTL implementation exists, but additional work providing greater accuracy is needed.

5.2.3 Physical Modeling:

Today performance and power models do take into consideration some of the physical constraints, such as the number of cycles along critical paths and the projected leakage for a specific technology. However, the linkage between these models and physical layout is usually manually maintained. Chip size is another important consideration that is usually calculated by a straight forward spreadsheet with constant factors for much of the chip infrastructure. Instead, all of the infrastructure that will be in the final chip must be represented in some form to provide an accurate estimate of area, power, congestion and noise values.

5.2.4 Thermal Hotspots

As was explained earlier, power and power density are the major constraints in modern chip design. With designs approaching the cooling limits of their packages, internal temperatures can approach operational limits or influence system reliability. The multi-core architect will need to consider thermal issues and prevent hotspots over a wide range of workloads and operating conditions. To select the best floorplan and the most effective power management strategy at the early stages of design, the architect needs accurate power estimates discussed above mapped to a floorplan along with thermal analysis that includes an accurate abstraction of the planned package and environment.

5.2.5 Integrated Early Analysis System:

So far, the discussion around the different forms of analysis has been segregated, but designers no longer have the freedom to consider one objective function at a time. Power, performance, cost, yield, reliability, hot spots, etc must be optimized simultaneously in the search for design solutions that meet product requirements. This presents a multi-objective optimization problem that requires an integrated analysis system.

University researches have recognized this need and have responded with interesting work on simultaneously optimizing system performance and layout [6,7], even for 3D packaging [8,9]. Several vendors provide integrated environments for developing SystemC specifications along with strong debugging capabilities. Also, vendors provide some power estimation at this level of abstraction. What is needed, however, is a system that links performance and power with the physical world of implementation[13,14].

Consider the system shown in Figure 2.

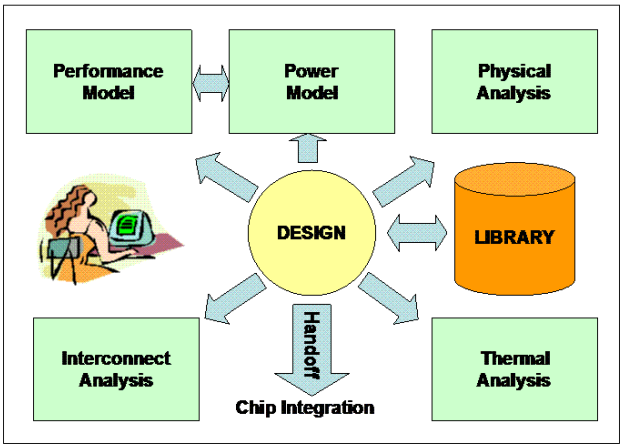

Figure 2. Integrated Early Analysis

The architect enters a simple block diagram that interconnects a set of components from a library. The block diagram drives the generation of a performance model with coupled power modeling. Next the architect sketches an approximate floorplan for the major components and an analysis application determines the latency of the primary interconnect for performance modeling. Other applications can identify congestion or noise concerns. With the component power mapped to the floorplan, another application produces a thermal map indicating potential problems. All information about the design, its environment and analysis results are shared in a common repository based on Open Access [12]. This allows efficient data sharing and provides a path to tools for later implementation. This is the vision of the early analysis system needed for rapid exploration of new multi-core designs.

5.2.6 Optimization

Over the years, there have been many attempts at architectural analysis and even synthesis to RTL implementations. They have typically focused on maximizing performance with little consideration to the many other factors confronting a chip architect.

Once an integrated environment exists to support early analysis there are great opportunities for automated multi-variable optimization. Multi-variable optimization research is not new and in fact can be found in many areas outside of VLSI design including game theory. Algorithms like these need to be characterized to understand their relevance and efficacy to VLSI. In any case, the opportunity for optimization at the important early stage of design is still wide open.

5.2.7 Hand-off to Implementation:

A predictive early analysis system can also speed implementation by directly passing the design decision, constraints and assumption to the RTL implementation phase. This is especially true in the multi-core case, where there is considerable reuse of pre-designed and pre-characterized components. That portion of the design together with its assumed layout can be a first pass implementation. The interconnect can be generated from the configuration and its physical environment. Much if not all of the infrastructure for test, bring-up and power management can be filled in assuming the overall reuse architecture anticipates this particular application. Even when a totally new component is needed, having the environment produced directly will accelerate implementation and enhance the quality of the testing.

5.2.8 Other factors:

This paper has discussed performance, power, area and thermal issues, but there are many other considerations that must be factored into designing a multi-core chip. Reliability is of growing importance and analysis methods to allow system architects to configure a multi-core design that meets requirements would be of high value. Three dimensional packaging is another option that requires all of the analysis tools mentioned so far extended to model the packaging technology being considered.

6. SUMMARY

This paper provides an overview of the reasons for moving to multi-core designs along with the design challenges and a view of the research directions for design automation. We are entering a period of dramatic change. The end of direct performance-scaling has led to an exciting time for system architects, who are inventing new ways to maintain the trend of the historic advances in system performance. Their need to explore more novel architectures provides a great need and opportunity to apply design automation technology to the early, pre-RTL, phase of design. This is the beginning of the long-awaited expansion of design automation to the system-level, but the key enabler is the integration of performance, power and physical analysis. We need to take advantage of the reuse in multi-core designs and the emerging standards to enable this next advance in EDA.

Academia has in the past provided base technologies along with students trained on relevant problems which has fueled the IC industry. Academia must continue to play a unique role in ensuring the continued success of the IC industry by addressing challenges associated with multi-core designs that are both strategic and high risk. Designers also must share the burden of discipline to enable IP re-use in their design as well as the burden of validation to ensure the success of future design and validation efforts both on the test floor and in the field.

7. REFERENCES

[1] Gordon E. Moore, Cramming More Components onto Integrated Circuits. *Electronics*, April 19, 1965.

[2] Pham, D. et al. The Design and Implementation of a First-Generation CELL Processor. In *ISSCC Digest of Technical Papers*. (San Francisco, CA, USA, Feb 6-10, 2005) p. 184-5

[3] Charlie Johnson, Jeff Welser, Future processors: flexible and modular. In *Proceedings of the 3rd IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, CODES+ISSS 2005, Jersey City, NJ, USA, September 19-21, 2005, 4-6

[4] http://ramp.eecs.berkeley.edu/

[5] http://www.itrs.net/

[6] J. Cong, A. Jagannathan, G. Reinman, and M. Romesis, "Microarchitecture Evaluation With Physical Planning," in *Design Automation Conference*, pp. 32–35, June 2003.

[7] J. Cong, A. Jagannathan, K. Konigsfeld, D. Milliron, M. Mohan, G. Reinman, M. Romesis, and H. Yang, "Micro-Architecture Evaluation and Optimization with Interconnect Pipelining," *Proceedings of the Asia South Pacific Design Automation Conference*, pp. 8-15, January 2005.

[8] J. Cong, A. Jagannathan, Y. Ma, G. Reinman and J. Wei, " An Automated Design Flow for 3D Microarchitecture Evaluation, " *Proceedings of the 11th Asia and South Pacific Design Automation Conference (ASP-DAC 2006)*, Yokohama, Japan, pp.384-389, January 2006.

[9] S. S. Sapatnekar , Y. Zhan, and T. Zhang, "Temperature-Aware Routing in 3D ICs," *Proceedings of the Asia-South Pacific Design Automation Conference*, pp. 309 – 314, 2006.

[10] http://www.spiritconsortium.com/

[11] http://www.systemc.org/

[12] http://www.si2.org/?page=69

[13] R. Bergamaschi, S. Bhattacharya, D. Brand, J. Darringer, A. Herkersdorf, J. Morrell, I. Nair, P. Sagmeister, and Y. Shin, "Early Analysis Tools for System-on-a-Chip Design", *IBM Journal of Research and Development*, v.46, no.6, 691-707, November 2001.

[14] Reinaldo A. Bergamaschi, Subhrajit Bhattacharya, John A. Darringer, Nagu R. Dhanwada, , William E. Dougherty, Indira Nair, , Sarala Paliwal, and Youngsoo Shin: SEAS: a system for early analysis of SoCs. *CODES+ISSS,*150-155, 2003.