Quo Vadis, BTSoC (Billion Transistor SoC)?

Nikil Dutt
Center for Embedded Computer Systems
Donald Bren School of Information and Computer Sciences
University of California, Irvine, CA 92697-3435
dutt@uci.edu
http://www.cecs.uci.edu/~dutt

Billion Transistor Systems-on-Chip (BTSoCs) present designers with a classic case of the "embarrassment-of-riches" syndrome: with so many devices at one's disposal, designers may be tempted to integrate functionality willy-nilly, with no strategic rethinking of what this level of integration can both afford, as well as achieve. While many advocate "business-as-usual" – including ad-hoc integration of functionality to achieve application-specific or domain-dependent designs – I believe BTSoCs present us with some opportunities for a paradigm shift in the architectural strategies and design processes for designing such complex chips. I summarize some key principles and ideas below.

Do Less with More

Conventional SoC design has relentlessly pursued a "more-with-less" strategy, cramming ever more functionality on IPs, with the hope of reusing complex, multi-functional IP blocks over a range of applications. I think we should pursue the converse: go with simpler IP blocks that can be easily designed, validated, verified, and composed for scalability. Let's move back to the *KISS* ("Keep It Simple, Stupid!") principle. An obvious by-product is that large numbers of simple IPs can be shut off when inactive, accruing large gains in energy/power, and also allowing for better control of on-chip thermal profiles.

Rethink What MPSoCs Mean

Today, the term MPSoC is commonly interpreted as a Multi-*Processor* SoC, with "multi-core" being the current mantra. I propose that we redefine MPSOCs as **Multi-*Platform* SoCs**, which elevates the design process to the composition of multiple platforms, with each platform tuned for a distinct application or constraint**.** This approach will simplify complex application development and ease the task of mapping system-level functions to System-IPs.

Error-Aware Design

Physical design and manufacturing processes already account for the fact that increasingly, errors cannot be eliminated, but must be tolerated. Similarly, we must develop error-aware system-level strategies for designing reliable BTSoCs using inherently unreliable system-IP components [1].

Instant Recall

Memories are going to dominate the real estate of BTSoCs. We should pursue aggressive customization of memories (tuned to system-IPs). Furthermore, we should deploy these memory resources for aggressive check-pointing and rollback recovery schemes to gain additional fault-tolerance.

Backward Compatibility

Our industry has, by necessity, always followed an evolutionary path towards these types of paradigm shifts, and thus backward-compatibility and reuse of existing flows and IPs is of paramount importance. Thus I believe we will see a gradual migration towards some of the ideas outlined earlier. For instance, consider on-chip communication architectures: clearly traditional bus-based architectures alone will not scale to manage the complex traffic patterns arising from future BTSoCs. However, it is unreasonable to expect that Networks-on-Chip (NoCs) [2] will solve this problem by itself. It may be more realistic to think about an evolutionary/hybrid on-chip communication architecture [3], where long-hauls use NoC links, whereas local transfers employ more traditional bus-based mechanisms [4].

Outlook

BTSoCs will necessitate elevation of the design process to the next level of abstraction, where – using an analogy to logic design – processors are the "gates", customized memories are the "flip-flops" and buses are the "wiring" used to create system-level Platform IPs. BTSoC designers shouldn't think about processors, but instead should compose system-level functions by mapping to system-IPs. A paradigm shift to the "processors-as-gates" model will, in turn, provide a software-centric view of BTSoC design: designers can configure and execute applications using multiple platforms, and conversely, can selectively disable platforms that are not required. This approach also has built-in redundancy, since unused platforms can be dynamically deployed in the face of errors. This modality is in sync with the current trends that show software is the dominant design care-about for complex SoC designs.

References

[1] N. Dutt, F. Kurdahi, A. Eltawil and S. Nassif "Cross-Layer Approaches to Designing Reliable Systems using Unreliable Chips", All-day tutorial, ASPDAC 2008, January 2008.
[2] G. De Micheli and L. Benini, *Networks on Chips*, Morgan Kaufmann, San Francisco, CA, 2006.
[3] S. Pasricha and N. Dutt, "ORB: An On-chip Optical Ring Bus Communication Architecture for Multi-Processor Systems-on-Chip", *Proc. ASPDAC 2008*, Seoul, Korea, January 2008.
[4] S. Pasricha and N. Dutt, *On-Chip Communication Architectures: System on Chip Interconnect*, Morgan Kaufmann, San Francisco, CA, 2008.

978-1-4244-1922-7/08/$25.00 ©2008 IEEE