# System-Level Exploration Tools for MPSoC Designs

Peter Flake
Imperas, Inc.
flake@imperas.com

Simon Davidmann
Imperas, Inc.
simond@imperas.com

Frank Schirrmeister
Imperas, Inc.
franks@imperas.com

## ABSTRACT
In this paper, we describe the challenges users face for software development on Systems on Chip (SoC) involving multiple cores. We will briefly review the trends and challenges for MPSoC design and will derive from them the resulting requirements. We will then discuss briefly the required exploration tools and close with an analysis of which type of providers will be able to provide appropriate solutions to the MPSoC challenge.

## Categories and Subject Descriptors
D.2.10 [**Software Engineering**] Design - *methodologies*
D.3.2 [**Programming Languages**] Language Classifications - *concurrent, distributed, and parallel languages, data-flow languages, design languages, specialized application languages, very high-level languages*
D.3.4 [**Programming Languages**] Processors – *code generation, compilers, debuggers, optimization, retargetable compilers*

## General Terms
Performance, Design, Economics, Languages, Verification

## Keywords
Multicore, Simulation, Debugging, Software Development

## 1. INTRODUCTION
Today users are listening to their favorite information channels as pod casts on devices which used to be their cell phone and now have fallen victim to convergence combining a myriad of consumer functions in a single unit. The next wave of convergence will bring even tougher integration challenges. The electronics industry is at risk not being able to deliver the next generation of convergence. The methodologies to develop cheaper, faster, more power efficient and reliable chips that drive consumer electronics are simply running out of steam. Innovative thinking is required to address design and verification across the multiple specializations hardware, software, architecture and verification.

## 2. TRENDS AND CHALLENGES
Examples of the first generations of platforms targeted to consumer markets are Philips nExperia [1], Texas Instruments OMAP [2] and STMicroelectronics Nomadik [3]. All of these platforms require a fairly complicated ecosystem and exhaustive design chain interaction as described in [4]. On the nExperia silicon users already find 3 processors and about 20 dedicated hardware components. Given the huge investment necessary for a platform easily ranging between $20M and $100M not only Time to Market is important but also Time in Market has to be optimized to allow proper returns on the investment. This in turn has driven the trend to enable more flexibility using software on processors.

However, power consumption has since then become the key limiting factor. While the computing world has stopped at about 4GHz, processors in the embedded consumer world have reached 1 GHz only to run into similar power and clock frequency limitations. Using 10 processors instead, each of them consuming 100 times less power, leads to an overall 10 times reduction in power consumption. In reality however, the industry is lacking the methodologies to efficiently parallelize and distribute the software across those processors in an automatic fashion.

The current situation suggests great discontinuity in the electronics industry short term and in turn offers a tremendous long-term opportunity for those willing to seize it. As the industry moves forward, developing and programming Multi Processor Systems on Chip (MPSoC) will require a unified Systems Design Automation approach in which hardware and software technologies and design processes are combined in a seamless development environment.

This change will also cause a major restructuring of the electronics industry as business units developing software programmable platforms effectively have already become software companies with the number of software developers exceeding the number of hardware designers. And this trend will only become stronger with average number of processors per MPSoC growing and programmers finding that their existing tools do not work for the new class of multi-processor chips.

## 3. REQUIREMENTS FOR MPSoC DESIGN
Assessing who will be able to seize this opportunity requires understanding of the fundamental requirements a software development environment for MPSoCs has to meet. Software developers can no longer wait for the chip to debug the software because the chip effectively is becoming the software. Hardware developers realize that the MPSoC, its structure and partitioning, has become part of the software problem. Not quite meeting the requirements of an application market has tragic consequences for a silicon platform given the development effort. Hence even more flexibility is required leading to more processors. However, the answer to the question which number of processors to use in which configuration is in turn again highly dependent on the software running on it.

First, to assess different options these design teams require very fast simulation, crossing the hardware/ software boundaries and far exceeding today's capabilities. Today software on an

Instruction Set Simulator (ISS) for single processors can be simulated at multiple MIPS. Those capabilities are simply not scaling appropriately when using 10, 20 or even 50 processors and trying to meet the requirement of running a significant portion of the application (think 30 seconds of video and audio in a multimedia application).

Second, today's embedded debug solutions are built with single processors in mind, not taking into account the requirements of debugging parallelism in an application distributed to 10, 20 or 50 processors.

Lastly today's compilers are targeted to single processors only. The problems of expressing parallelism in application programming, then mapping and distributing software on a MPSoC platform remain largely unaddressed.

## 4. EXPLORATION TOOLS

Exploration tools for MPSoC design fall into two categories: for the platform designer and the platform user

The platform user wants to add a new application to an existing platform. The feasibility of meeting performance constraints must be checked, and the power consumption optimized. This requires the exploration of various partitioning options of the software

The platform designer wants to ensure that selected applications can be run at the required performance and efficiency. This involves programming the computation-intensive parts of the applications.

Both users therefore require compilation, simulation, debug, performance analysis and power analysis tools. The degree of integration and the raw speed both determine the productivity achieved by the tool users.

An excellent overview of existing ESL offerings for MPSoC design is provided in [5] and the author concludes that "the design of complex embedded systems with multiple configurable, extensible processors demands new ESL tool capabilities that go well beyond current offerings."

## 5. SOLUTION PROVIDERS

Who is up for the challenge? Will it be driven by the software side or the hardware side? Who can afford the investments? Who is willing to pay the price? While hardware designers are used to paying a price for fast simulation and debug, software designers are used to free or almost free tools in the age of GCC and GDB. This trend is exacerbated by the expectation of software developers to get a software development environment presented in conjunction with the silicon which they are programming. Vertically integrated semiconductor houses like Texas Instruments and Philips have realized that some time ago and hired more software developers recently than hardware developers to support and differentiate their silicon.

The next generation System Design Automation paradigm can only be addressed in close cooperation with hardware and software designers. It will have to be funded by the hardware world in which the users – the software developers - as part of the silicon offering expect a state of the art software development environment supporting the platform. Otherwise they will simply switch platforms or remap the application to a new, highly programmable MPSoC specifically taking into account their application requirements.

Providing this new methodology for System Design Automation is the next great challenge, particularly with the need to cross traditional industry boundaries and attitudes between EDA and embedded software. But when an MPSoC design team can apply functionality across a range of processors, successfully debug and validate software and hardware simultaneously, this new design paradigm will yield significant benefits for all and enable accelerated growth in consumer electronics based on cheaper, faster, more power efficient and reliable chips.

## 6. CONCLUSION

If - and only if - System Design Automation becomes reality then the next generation of convergence in consumer devices can be developed with appropriate time to market and reliability.. Those companies that have the best System Design Automation methodology and tools will be able to beat their competitors both in time to market and in product quality

Users will be able to look back at today's methodologies and smile while double-checking that their PDA has computed directions correctly using the integrated GPS and being on the phone with their children discussing their homework, all using one integrated device.

## 7. REFERENCES

[1] http://www.semiconductors.philips.com/products/nexperia/about/index.html.

[2] http://focus.ti.com/omap/docs/omaphomepage.tsp.

[3] http://www.st.com/stonline/books/pdf/docs/9306.pdf

[4] Martin, G., Schirrmeister, F. *A Design Chain for Embedded Systems.* IEEE Computer 35, 3 (March 2002), 100-103

[5] Martin, G. ESL Requirements for Configurable Processor-based Embedded System Design, Design And Reuse, http://www.us.design-reuse.com/articles/article12444.html.

.