System-on-Chip Beyond the Nanometer Wall

Philippe Magarshack
Central R&D, STMicroelectronics
Ave. Jean Monnet
38031 Crolles cedex, France
+33-4-7692-6498
philippe.magarshack@st.com

Pierre G. Paulin
Central R&D, STMicroelectronics
16 Fitzgerald Road
Ottawa, ON, Canada, K2H 8R6
+1-613-768-9069
pierre.paulin@st.com

ABSTRACT

In this paper, we analyze the emerging trends in the design of complex Systems-on-a-Chip for nanometer-scale semiconductor technologies and their impact on design automation requirements, from the perspective of a broad range SoC supplier.

We present our vision of some of the key changes that will emerge in the next five years. This vision is characterized by two major paradigm changes. The first is that SoC design will become divided into four mostly non-overlapping distinct abstraction levels. Very different competences and design automation tools will be needed at each level.

The second paradigm change is the emergence of domain-specific S/W programmable SoC platforms consisting of large, heterogeneous sets of embedded processors. These will be complemented by embedded reconfigurable hardware and networks-on-chip. A key enabler for the effective us of these flexible SoC platforms, is a high-level parallel programming model supporting automatic specification-to-platform mapping.

Categories and Subject Descriptors

B7.1 [**Integrated Circuits**]: Types and Design Styles – *VLSI, Advanced technologies*, *Microprocessors and microcomputers*

C3 [**Special-Purpose and Application-Based Systems**]: *Real-time and embedded systems*

General Terms

Algorithms, Design, Economics

Keywords

System-on-chip, network-on-chip, reconfigurable systems, multi-processor systems, embedded software technologies, design automation tools.

1. INTRODUCTION

The continued increase in the non-recurring expenses (NRE) for the manufacturing and design of nanoscale systems-on-chip (SoC), in the face of continued time-to-market pressures, is leading to the need for significant changes to their design and manufacturing. The SoC mask set manufacturing NRE cost has been multiplied by a factor of ten in about three process technology generations, exceeding 1M$ for current 90nm process. At this cost, many smaller design houses cannot afford the financial risk of a tape-out. For example, for a chip sold at a price of $5, and a profit margin of 20%, this implies selling over one million chips simply to pay for the mask set NRE. This does not even account for the accompanying increase in design NRE, which ranges from 10M$ to 100M$ for *today's* complex 0.13 micron designs. Using the same assumptions as above, this implies volumes of 10 to 100 million chips to break even.

These figures partially explain the strong growth rate of Field-Programmable Gate Arrays (FPGA) and application-specific standard products (ASSP) in certain markets. This is particularly true in the communications infrastructure space for example, where medium volumes (below 100K chips/year) preclude the development of specialized ASICs.

A radical change is needed to allow small-to-medium entrants in the market, or to support products with volumes well below the multi-million chip threshold needed to make a profit on low-cost IC's. Somehow, the mask-set NRE needs to be reduced or amortized over many more products. FPGA's are one solution to this, but their higher power and cost preclude high-volume and low-power applications. Recent approaches using a gate-array style fabric and top metal-level configuration will also help provide an intermediate point on the NRE-flexibility continuum. Finally, 'systems-in-a-package' (SiP) approaches, which contain multiple dies of various process technologies (e.g. logic and DRAM) will also help address the manufacturing NRE.

However, neither of these solutions address the *design* NRE and time-to-market needs for today's SoC's which can have over 100 million transistors – enough to theoretically place the logic of over one thousand 32 bit RISC processors on a die. Leveraging these capabilities is a major challenge. For this reason, a SoC design platform needs to be amortized over many variants and generations of a product family, to help amortize both the mask *and* the design NRE's. Moreover, platform users need better productivity tools to reduce the end-product design NRE.

2. EVOLUTION OR REVOLUTION?

For many of the traditional CAD companies supporting the semiconductor business, there seems to be an underlying assumption that we will continue doing design in essentially the same way we are doing it now, albeit with (much) more complexity. We would be using essentially the same type of components, namely, a slowly evolving mix of hardware and software: a few general-purpose processors, a few DSP's and still many H/W IP blocks (digital and analog). It also seems assumed that evolving current design and CAD technologies will be able to address the complexity growth.

The reality is that we are not adequately solving the 0.13 micron design problem now, and it is unrealistically optimistic to try to solve the sub-90nm problem using extensions of today's approaches. In fact, it could be argued that for 90nm technologies and beyond, the design productivity (transistors designed per man-year) will actually decline due to the new deep submicron effects discussed later.

It is also assumed that most new SoC products will be a novel assembly of (hopefully reused) IP's. Even if the SoC is entirely made of reused IP's, this does not solve manufacturing cost NRE, the deep submicron physical design issues, and the combined verification and design-for-test issues of the resulting SoC.

Given the repeated message of exponentially rising complexity, we strongly advocate complementing this evolutionary roadmap with an exploration of significantly different design methods for complex SoC's. We believe a major paradigm change will be needed – and will occur – in the next five years or less. This is already happening in some markets. This paradigm change will be driven by three requirements:

1. *Faster time-to-market for SoC platform implementation*. In particular, through the use of higher-level off-the-shelf IP's, connected via a modular, scaleable SoC interconnect topology and standard communication interfaces.
2. *Increased flexibility in SoC platforms* to amortize the mask and design NRE over more products. This can be achieved by a combination of S/W programmability and configurable H/W, leading to more reusable platforms.
3. *Dramatically increased productivity for the platform user*. This will be the key requirement and will have the highest impact on the application S/W structure and the underlying platform architecture. This will drive the development of new parallel programming models to enable automated application-to-platform mapping.

As a result of these requirements, we believe that two major paradigm changes will emerge in the next five years:

- Embedded SoC design will become divided into four mostly non-overlapping distinct abstraction levels.
- The emergence of domain-specific S/W programmable SoC platforms consisting of large, heterogeneous sets of embedded processors, reconfigurable H/W and networks-on-chip.

We examine the first paradigm shift in sections 3 to 5, the second in section 6, and a vision of emerging solutions in section 7 and section 8.

3. MULTI-LEVEL SOC DESIGN

In order to manage the complexity explosion, SoC design will become divided into four mostly non-overlapping distinct abstraction levels:

1. <u>System application design</u>: This level involves application specialists, writing embedded S/W at a high level, using general-purpose and domain-specific embedded S/W productivity tools. This includes the initial algorithm design task. No hardware design is done here. At most, this might involve the specification of configurations of an existing platform.
2. <u>Multi-processor SoC (MP-SoC) platform design</u>: This consists of highly flexible S/W-programmable and reconfigurable platforms for well-defined application areas: wireless, multimedia, networking, automotive. Specialists at this level assist with the (re)configuration of the platform for the system application developer. As a rule, no IP design is done here, but specification, assembly and configuration of existing IP blocks.
3. <u>High-level IP block design</u>: This includes embedded processors (RISC, DSP, MCU, ASIPs), interconnect technologies (with a trend towards networks-on-chip, and away from traditional shared buses), domain-specific standard I/O's (PCI-variants, SPIx variants, HyperTransport, I2C, FireWire, QDR, etc.), and finally, well defined H/W IP for standards (e.g. an MPEG4 video codec).
4. <u>Semiconductor technology & basic IP</u>: Standard cells, I/O, memories and the basic technology processes supporting them. The trend here is for more heterogeneous technologies, combining embedded DRAM, embedded Flash, mixed-signal BiCMOS, RF, analog.

These four abstraction levels will require mostly orthogonal competences. Or put another way, they *must* be orthogonal in order to solve the complexity explosion. The underlying divide-and-conquer approach implies very different needs for designers working at each level.

In order to achieve this, better tools will be needed to feed the power, performance and area figures up to higher abstraction levels to better quantify the effect of the mapping of a system application onto a MP-SoC platform. The two main design issues will be power optimization and embedded memory architecture tradeoffs (embedded SRAM, eDRAM and eFlash, v.s. external memories).

4. EVOLUTIONARY SOLUTIONS

The two lowest levels (high-level and basic IP) will require most of the evolutions underway in the CAD industry today. Here, an evolution of existing design and verification tools is appropriate: e.g. faster simulators, more IP reuse, integrated logic and physical design synthesis, etc. Of particular importance in this space are the following issues:

- Deep sub-micron effects that are becoming predominant in 90nm and below. These include: electro-migration, voltage-drop, and on-chip variations, all of which will lead to statistical design, self-repair and various forms of redundancy.

- The integration of analog and RF IP's, which, when integrated with digital logic on a SoC, can save the cost of an additional die in the bill of materials.
- DFT has to evolve together with SoC complexity. The IEEE 1500 class of on-chip test bus is an example of this trend. In addition, BIST will need to support all sorts of IP's: Not only memories, but also digital logic, analog and RF.
- Increased use of formal proof between abstraction layers, as well as the use of unique verification testbenches and environments across abstraction layers [7].
- Continued improvement of H/W-S/W codesign tools, but extended to include reconfigurable H/W as a design option.
- Transaction-level modeling (TLM) of mixed H/W-S/W systems to anticipate the step when effective HW-SW co-simulation is effective before RTL, reduce the time to develop executable specifications of HW blocks and increase the simulation speed [10]. Standardization of TLM approaches and API's is urgently needed.
- Finally, low-power is a must, not just an added-value feature. This includes techniques such as on-chip voltage control, back-bias to master leakage, and multi-Vt transistors. The objective of low-power will favor the use of hardware over software in many cases, when design time permits. This tradeoff of productivity versus lower power will be a key consideration in the design of next generation SoC platforms.

The list above addresses many of today's problems at the two lower abstraction levels. However, for the system application and multi-processor SoC platform levels, new approaches need to be considered, and very different design automation tool needs will emerge for each of these levels. These are examined below.

5. SYSTEM APPLICATION DESIGN TOOL NEEDS AND SOLUTIONS

The two key requirements in this space are: 1. system application development productivity, and 2. higher independence from the implementation platform.

5.1 Use of Domain-specific Specification and Modeling Tools

A variety of effective domain-specific tools already exist. For example, the Matlab environment is one of the most widely used set of tools, and it effectively covers a wide range of analog, digital and mathematical problems. Other domain specific tools and abstractions include the SDL-based tools from Telelogic, Esterel Studio, and a variety of queuing and dataflow simulators. It is our belief that these tools provide sufficient productivity for high-level application development, in their specific application domain. Better interoperability is needed though. More importantly, there is a need for a more automated refinement to the MP-SoC implementation platform, as discussed below.

5.2 Use of Leading Edge S/W Tools

Many of the leading ideas of the 'traditional' (non-embedded) S/W development approaches are demonstrating promising productivity gains. For example, Java and Microsoft .NET illustrate the potential for higher S/W productivity via encapsulation and reuse. Object-oriented formalisms like CORBA provide many clean abstractions for distributed systems that we believe are adaptable for complex SoC's.

Nevertheless, embedded S/W productivity and reuse remain key challenges. One big issue is the proliferation of S/W specification languages (e.g. UML, SDL), object-oriented distributed system formalisms (CORBA, DCOM, RMI), message passing formalisms (MPI), general-purpose programming languages (C, C++, Java), and embedded operating systems (Embedded Windows, Linux, VxWorks). There is a huge overlap in the concepts and capabilities across all of these.

Some simplification and rationalization is needed for their effective use in SoC's. Hopefully, the experience - and mindset - of the VLSI H/W community in raising abstraction levels and defining standards can be put to benefit here. SystemC approaches this objective from the bottom-up, but more work is still needed.

In the O/S domain, the main additional need is for ultra-lightweight versions of these O/S's, which supply a level of services tuned to the application domain. In some cases, part of the O/S services will need to be performed in hardware.

5.3 The System Application to Platform Gap

The domain-specific and general-purpose tools above will help mostly with high-level specification, modeling and platform-independent S/W development. When used in the context of MP-SoC platforms, a common issue for these tools is the difficulty in refining and mapping the application to the platform.

An optimal system solution will require the "correct" mapping of high-level abstractions on to the lower layers. This mapping process involves optimizations and trade-offs between many complex constraints, including quality of service, real time response, power consumption, area, and other factors impacting device cost. Tools are urgently needed to explore this mapping process, and assist and automate optimization where possible. It is also necessary to establish correctness between the various abstraction levels, ideally using formal proofs where possible, and allow reuse of test bench and verification environments across the layers.

One obstacle to achieving more automation has been the abstraction gap – perhaps more accurately referred to as the abstraction 'grand canyon' - between the system specification and most of the SoC platforms available today.

This is particularly true for today's ad-hoc, heterogeneous, low-level, H/W-S/W platforms. The issue is compounded when there is no defined SoC platform programming model, or not even a set of well-defined API's to interact with the platform. In this case, the platform user is directly exposed to the low-level hardwired, reconfigurable and S/W programmable components. This is time consuming and also makes the application non-portable.

The next sections will discuss the second major paradigm change, namely the emergence of flexible multi-processor SoC platforms. In particular, we will address the need for developing appropriate parallel programming models for these platforms, in order to simplify the automated application-to-platform mapping.

6. DOMAIN-SPECIFIC MULTI-PROCESSOR SOC PLATFORMS

The growth of hardware complexity in SoC's has tracked Moore's law, with a resulting growth of 56% in transistor count per year. However, industry studies show that the complexity of embedded S/W is rising at a staggering 140% per year. In many leading SoC's today, the embedded S/W development effort has surpassed that of the H/W design effort. Moreover, in consumer multimedia SoC products, such as set-top box, DVD, and audio, the actual cost of licenses and royalties for the application S/W (O/S, audio and video licenses) largely exceeds the chip manufacturing cost in many applications.

Based on the requirements for flexibility, rapid platform development and platform end-user productivity, our belief is that, within five years, the large majority of end-user SoC product *functionality* will run on heterogeneous embedded processors. This does not translate to comparable proportions of area or total performance though. Low-power and/or performance requirements will dictate partitions where the majority of performance will come from optimized H/W or FPGA, implementing critical inner loops and parallel operations, but of comparatively lower functional complexity.

MP-SoC platforms will include ten to hundreds of embedded processors. These will come in a wide diversity, from general-purpose RISC to specialized application-specific instruction-set processors (ASIP), with different trade-offs in time-to-market versus product differentiation (power, performance, cost), as depicted in Figure 1.

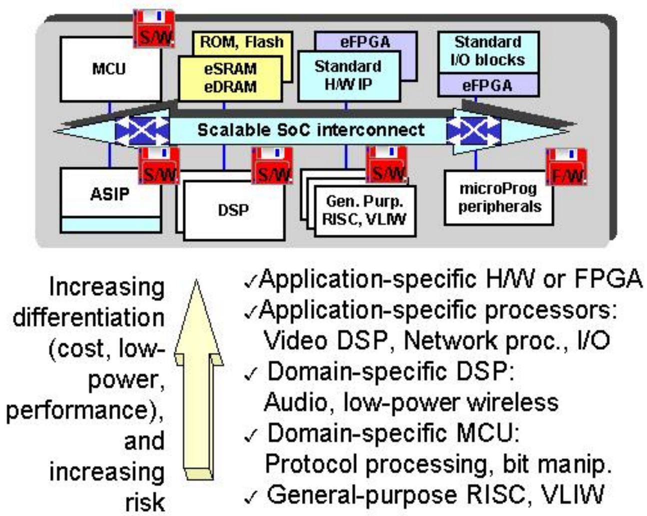

Figure 1. MP-SoC Platform Component Options

Current generation platforms in consumer multimedia (e.g. set-top box, DVD, digital video, camera and imaging), and wireless handsets already include over a half-dozen processors. New designs are appearing with much larger numbers of embedded processors, ranging from 8 to 32 in communications and network processing, security processors, storage array networks, and wireless base stations; to over 100 processors in recent platforms in consumer image processing, and high-end network processors.

All this leads to the increasing importance of effective programming tools for these platforms.

6.1 Network-on-Chip

A key component of the MP-SoC platform is the interconnect technology. An orthogonal, scaleable, interconnect approach with predictable bandwidth and latency is essential for many reasons:

1. It provides a regular, plug-and-play methodology for interconnecting various hardwired, reconfigurable or S/W programmable IP's.
2. It supports the high-level communication between processes on multiple processors, and simplifies the automatic mapping onto the interconnect technology.

We advocate the recent so called 'network-on-chip' (NoC) approaches currently under development [13]. We also strongly support the need for a standard NoC interface definition. ST is evolving its proprietary STBUS configurable interconnect towards NoC. We are currently using the proposed OCP-IP standard [11] in our MP-SoC platform experiments, as discussed in [1], [10].

However, there is still much remaining work to be done to characterize the various topologies – ranging from bus, ring, tree to full-crossbar – and their effectiveness for different application domains.

A common issue with all NoC topologies is communication latency. In 50 nm technologies, it is predicted that the intra-chip propagation delay will be between six and ten clock cycles [12]. A complex NoC could therefore exhibit latencies many times larger. Latency hiding is therefore a key aspect of in achieving efficient parallel processing.

6.2 Heterogeneous Multi-Processors

We believe that the large scale use of software programmable embedded processors will emerge as a key means to improve flexibility and productivity. As depicted in Figure 1, a range of processors will be used, to achieve different tradeoffs in time-to-market versus power, area or speed.

General-purpose processors will continue to play an important role, in particular for the most complex upper layers of the application stack, where real-time constraints are not as tight. Conventional real-time operating systems will run on these processors. Domain- or application-specific processors will also play an important role in bridging the gap between the required ease-of-use and high flexibility of general-purpose processors on one end, and the higher speed and/or lower power of hardware on the other. Configurable processors (like Arc or Tensilica) are one possible means to achieve processor specialization from a RISC-based platform. Reconfigurable processors take this one step further, by allowing run-time changes to the architecture [4].

Independent of the degree of processor instruction-set specialization, a common requirement is the efficient handling of the latencies of the interconnect, memory and co-processors. A variety of approaches can be used, including multi-threading, memory pre-fetching, and split-transaction interconnect. Multithreading lets the processor execute other streams while another thread is blocked on a high latency operation. A *hardware multithreaded* processor has separate register banks for different threads, with hardware units that schedule threads and swap them in one cycle.

6.3 Embedded FPGA's

Embedded FPGA's (eFPGA) will complement the processors, but only with limited scope (less than 5% of the IC functionality). The 10X cost and power penalty of eFPGA's will restrict their further use. Also, eFPGA's are like hardware in that they are really suited to a well-defined, repeatable function. They are not well-suited to small scale time division multiplexing of different tasks. Embedded processors can execute a much wider variety of tasks than an eFPGA. They are also more amenable to large-scale changes in product specs or user requirements. Nevertheless, for high-speed and simple functions, or highly parallel and regular computations, eFPGA's can play an important role. An important question here is what is the best level of granularity of the basic reconfigurable component. The evolution of current stand-alone FPGA platforms suggest that a heterogeneous mix of datapath and fine-grain fabrics will emerge.

6.4 Hardware IP

Of course, hardware will not disappear! But increasingly, it will exist in the form of highly standardized functions, which communicate via a standard protocol. Examples include high-performance video processing, e.g an MPEG2 video codec. Another main category of standard H/W is the I/O component. Increasing standardization of I/O's for different market spaces will leave a dozen main I/O families: e.g. PCI evolutions, RapidIO, HyperTransport, SPI-x, USB, FireWire, QDR, etc. Their integration into the SoC will be facilitated by the network-on-chip's standardized protocol and scalability.

7. MULTI-PROCESSOR SOC PLATFORM: EMERGING SOLUTIONS AT ST

7.1 FPPA Architecture Platform

In order to address the real objective, namely the productivity of the platform end-user, we believe that the system application, the platform architecture *and* the programming tools must be considered as an interdependent whole. For this reason, we have developed an environment to enable exploration of the interactions between these three domains.

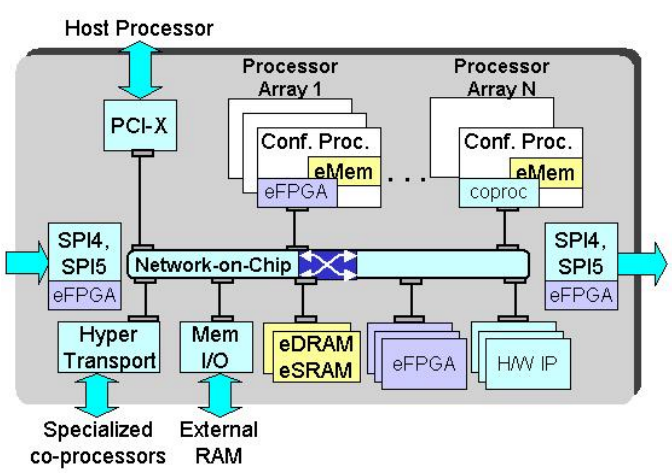

Figure 2. Flexible Communications Platform Example

Figure 2 depicts an example of a domain-specific flexible architecture platform, oriented towards networking applications. It is derived from ST's *StepNP*^{TM} exploratory NPU platform [1]. This is not a product, but an experimentation vehicle used in ST's Central R&D organization as a means to explore MP-SoC automation tool requirements. This platform includes models of configurable processors, a network-on-chip, reconfigurable H/W and standard H/W, as well as communication-oriented I/O's. We refer to this platform as a 'Field-Programmable Processor Array', or *FPPA*. We believe that FPPA's embody many of the characteristics of the emerging high-productivity platforms we will need in nanoscale technologies.

7.2 MultiFlex MP-SoC Tools

It is our conviction that the success of an FPPA platform will depend mostly on its ability to support a high-level programming model, therefore enabling higher productivity tools. This is the key means to bridge the gap between system specifications and the platform capabilities, as discussed in Section 5.3.

Within ST's Central R&D organization, we have been working on the 'MultiFlex' toolset for multi-processor SoC systems, with networking and communications applications as the key drivers. Over the past three years, our previous work was concerned with the development of multi-processor modeling, debug and analysis tools [1], [14], using StepNP as the reference platform. This environment leverages our existing system-level design tools [7], [10], and embedded software technologies [6], but also adds several MP-oriented capabilities.

Our recent MP-SoC automation research work has focused on parallel programming models and automatic mapping to MP-SoC platforms. The programming model should be platform independent in order to ease the porting of an application to different platform (re)configurations. It should also express parallelism in a natural and intuitive manner for the application domain.

We have developed a lightweight Distributed System Object Component (DSOC) programming model inspired by CORBA-like concepts. DSOC objects can be executed on a variety of processors supported in the StepNP environment, as well as on hardware or on the eFPGA. Using the DSOC methodology, the application design is largely decoupled from the details of a particular FPPA target mapping.

To demonstrate the DSOC concepts, we have successfully mapped a DSOC model of a complete IPv4 fast-path application onto a large-scale multi-processor and H/W multi-threaded instance of the StepNP platform. We achieved near 100% utilization of the embedded processors and threads, even in presence of NoC interconnect latencies of over 100 cycles, while processing worst-case traffic at a 10 Gbit line rate. The first results are presented in [2]. This is an early demonstration of the feasibility of the application-to-platform mapping we are advocating, at least for the networking application domain.

We believe the DSOC framework provides a very natural programming model, immediately familiar and intuitive to software developers exposed to mainstream distributed software techniques such as Java RMI or CORBA. In addition, the framework allows capture of characteristics of objects in a way that can be exploited by tools. Given base properties of the architecture, such as predictable NoC latency and throughput, the tools can vastly simplify the mapping of the DSOC objects on to the architecture, enabling rapid exploration and optimization.

8. CURRENT ACTIVITIES & OUTLOOK

Beyond the MultiFlex MP-SoC automation tools referred to above, the several ST R&D organizations have been active on a number of other fronts, which also address the emerging flexible MP-SoC platform needs. This includes component development, architecture platforms and system design and embedded systems technologies:

- The development and manufacturing of a 1 GOPS reconfigurable signal processing IC [4]. This combines a commercial configurable RISC core with an embedded FPGA fabric which implements the application-specific instruction extensions. This IC also includes an embedded Flash memory component [5].
- We are also exploring tradeoffs in configurable fabrics which allow us to optimize the balance of processing done in dedicated blocks versus software processors. The use of coarse and fine grain configurable fabrics allows the system designer to optimize performance versus power consumption. We are exploring these issues in the application of low-power wireless LAN's.
- The development of a 6.4 Gbps/channel on-chip communication network using Flash-EEPROM switches and elastic interconnects. This approach implements a configurable crossbar, using non-volatile memory [3].
- The design of a high-performance network packet search engine optimized for IPv4/IPv6 forwarding. In comparison with CAM-based look-up methods, it relies on an SRAM-based approach that is more memory and power-efficient [9].
- In cooperation with the UPMC/LIP6 laboratory in Paris, we have developed a 32 port version of the SPIN network-on-chip [8], implemented using ST's 0.13 micron process.
- The development of system-level design methods to support mixed H/W-S/W systems, from TLM to RTL [7], [10].
- The development of the 'FlexWare' high-performance embedded software development tools, which is quickly retargetable to a range of domain-specific processors [6].

Future activities in ST include the evaluation and manufacturing of a range of network-on-chip topologies, further exploration of reconfigurable H/W fabrics, and the extension of the MP-SoC programming models and compilers for consumer multimedia applications like image processing and digital video.

9. CONCLUSION

The continued increase in the non-recurring expenses for the manufacturing and design of nanoscale systems-on-chip (SoC) is leading to the need for significant changes to their design and manufacturing. As a result, SoC design will become divided into four, mostly non-overlapping, distinct abstraction levels. Different competences and tools will be needed at each level.

In order to address flexibility and time-to-market needs, we will see the emergence of domain-specific flexible platforms consisting of a large, heterogeneous set of embedded processors, reconfigurable H/W and standardized H/W IP's, all connected via a scalable network-on-chip.

10. ACKNOWLEDGMENTS

We thank our ST colleagues Laurent Bergher, Marco Cornero, Faraydon Karim, Chuck Pilkington and Roger Shepherd for their contributions to the themes discussed in this paper.

11. REFERENCES

[1] P. G. Paulin, C. Pilkington, E. Bensoudane, "StepNP: A System-Level Exploration Platform for Network Processors", *IEEE Design & Test of Computers*, vol. 19, no.6, Nov. 2002.

[2] P. G. Paulin, "StepNP: A Driver for Multi-processor SoC tools", *Presentation at the Multi-Processor SoC Seminar*, Chamonix, July 2003. See http://tima.imag.fr/mpsoc.

[3] M. Borgatti et al, "A Multi-Context 6.4Gbps/Channel On-Chip Communication Network using 0.18um Flash-EEPROM Switches and Elastic Interconnects", *Proc. of ISSC*, San Francisco, Feb. 2003.

[4] M. Borgatti et al, "A 0.18um, 1GOPS Reconfigurable Signal Processing IC with embedded FPGA and 1.2GB/s, 3-Port Flash Memory Subsystem", *Proc. of Intl. Solid-State Circuits Conference (ISSC)*, San Francisco, Feb. 2003.

[5] M. Pasotti et al, "An Application Specific Embeddable Flash Memory System for Non-Volatile Storage of Code, Data and Bit-Streams for Embedded FPGA Configurations", *Proc. of Symposium on VLSI Circuits*, Kyoto, June 2003.

[6] P. G. Paulin and M. Santana, "FlexWare: A Retargetable Embedded-Software Development Environment," *IEEE Design & Test of Computers*, vol. 19, no. 4, July 2002.

[7] A. Clouard et al., "Towards Bridging the Gap between SoC Transactional and Cycle-Accurate Levels," *Proc. Design, Automation, and Test in Europe—Designer Forum*, 2002, pp. 22-29.

[8] A. Greiner et al, "SPIN: a Scalable, Packet-switched, On-chip Micro-network, *Proc. of Design Automation and Test in Europe (Designer Forum)*, Munich, March 2003.

[9] N. Soni et al, "NPSE: A High Performance Network Packet Search Engine", *Proc. of Design Automation and Test in Europe (Designer Forum)*, Munich, March 2003.

[10] A. Clouard, K. Jain, F. Ghenassia, L. Maillet-Contoz, J.-P. Strassen, "Using Transactional Level Models in a SoC Design Flow", in "SystemC Methodologies and Applications", eds. W. Muller, W. Rosentiel, J. Ruf, *Kluwer Academic Publishers,* 2003.

[11] See OCP-IP web site: http://www.ocpip.org.

[12] L. Benini and G. De Micheli, "Networks on Chip: A New SoC Paradigm," *Computer*, vol. 35, no. 1, Jan. 2002.

[13] A. Jantsch, H. Tenhunen (Eds.), "Networks on Chip", *Kluwer Academic Publishers*, 2003.

[14] P. G. Paulin, "Trends and Requirements for Network Processor SoC Tools", *Presentation at Multi-Processor SoC Seminar*, Pizay, June 2002. See http://tima.imag.fr/mpsoc/2002/slides/paulin02.pdf