Seventh Framework Programme — Theme 3.6: Computing Systems

Public Summary

Grant Agreement Number: 248776
Project Acronym: PRO3D
Project Title: Programming for Future 3D Architectures with Manycore
Funding Scheme: STREP
Project Web Site: `http://pro3d.eu`
Perdiod Covered: From January 1^{st} 2010 to December 31^{st}, 2012.

Project Coordinator: Commissariat à l'énergie atomique et aux énergies alternatives
Scientific Representative: Christian FABRE, Senior Research Enginneer,
CEA LETI/DACLE/LIALP
Tel.: +33 4 56.52.03.42
Fax: +33 4 56.52.03.66
Mail: `christian.fabre1@cea.fr`

Report Version: v1.0
Report Status: Delivered
Report Date: May 31, 2013

CEA Reference of Report: LETI/DACLE/13-0364

Versions of the Document

Version	Date	Author	Comment
1.0	May 31^{st} 2013	Christian FABRE	Final version derived straight from Project Final Report. Delivered.

Contributors

David ATIENZA (EPFL), Iuliana BACIVAROV (ETHZ),
Saddek BENSALEM (UJF-VERIMAG), Marius BOZGA (UJF-VERIMAG),
Christian FABRE (CEA), Éric FLAMAND (STM),
Jean-Pierre KRIMM (CEA), Martino RUGGIERO (UNIBO).

PRO3D **Partners**

1	CEA (Coord.)	Commissariat à l'énergie atomique et aux énergies alternatives, Laboratoire d'électronique et des technologies de l'information
2	UJF	Université Joseph Fourier Grenoble 1 – VERIMAG
3	ETHZ	Eidgenössische Technische Hochschule Zürich
4	UNIBO	Università di Bologna
5	STM	STMicroelectronics
6	EPFL	École polytechnique fédérale de Lausanne

Contents

List of Figures

1 Executive Summary

During the last three decades, the performance of microprocessors & microcontrollers has steadily increased at the impressive rate of 100 times per decade. This was fuelled by: (1) The exponential growth in clock speeds; (2) The exponential growth in the number of transistors per die; and (3) The acceleration of instruction flows obtained by various techniques for reducing latency and maximising the amount of computation per clock cycle.

However, this picture was rapidly and radically changing. The shift to parallel architectures was not at all the consequence of a scientific breakthrough. It was primarily a consequence of hitting technology walls that prevented from pushing forward the efficient implementation of traditional uniprocessor designs in silicon. These technologies hitting walls are: (a) Voltage scaling and power reduction techniques, or *Power Wall*; (b) Instruction-level parallelism, or *Complexity Wall*; (c) Memory latency hiding techniques, or *Memory Wall*; (d) Reliable and low-variability silicon technology, or *Yield Wall*.

As an illustration of the rapidly evolving context of PRO3D, we can mention that the expected industrial solutions to the *Memory Wall* changed dramatically during the course of the project. At the start of PRO3D the most promising solution was Wide-IO, where a dedicated memory layer is connected to the computing fabric by vertical interconnects. This solution was expected to outperform the flat, 2D, LPDDR solutions both in speed and energy efficiency. By the time the project completed, the LPDDR roadmap had accelerated to the point of overlapping significantly Wide-IO whith an incremental evolution of the standard, 2D-based, solution: LPDDR3 & LPDDR4.

The PRO3D project proposed a holistic approach for the development activities ranged from programming to architecture exploration and fabrication technologies, and yield the following outcome: (1) **Thermal Modelling & Simulation**. (2) Programming, compilation, verification & deployment for 3D manycore architectures, including **Statistical Model Checking (SMC) of system models**. (3) Exploiting 3D opportunities into multicore architectures. (4) System-level thermal-aware exploration & analysis of 3D designs. (5) Virtual Prototyping.

– Thermal Modelling & Simulation

3D technology allow to achieve smaller footprint in each layer and shorter vertical wires but thermal problems caused by higher power densities and greater thermal resistances contributes to impede the establishment of this technology. Thus, thermal modeling has become increasingly important in 3D chip design.

3D-ICE stands for **3D Interlayer Cooling Emulator**. It is a Linux based Thermal Emulator Library written in C, which can perform transient thermal analyses of vertically stacked 3D integrated circuits with inter-tier Microchannel Liquid Cooling. It is based on the conventional compact modeling of heat transfer by conduction in solids, and advances a novel compact modeling methodology, called the Compact Transient Thermal Modeling (CTTM), for heat transfer by convection in microchannels.

This simulator developped in PRO3D is ideal for situations where a quick estimate of chip temperatures is required. 3D-ICE can be used to assist early stages of architecture design as for floorplanning of 3D multicore architectures or for the enhancement of liquid cooling efficiency through channel modulation. Simulations run with 3D-ICE contributed as well to the development of new strategies for optimal placement of thermal sensors or for the validation of run time thermal-aware management policies.

– Rigorous Design Flow & Statistical Model Checking.

PRO3D developped a rigorous design flow based on the BIP and DOL component framework: The flow is (*i*) model-based, that is, both application software and mixed hardware/software system descriptions are modeled by using a single, semantic framework; (*ii*) it is component-based, that is, it provides primitives for building composite components as the composition of simpler components; (*iii*) it is tool-supported, that is, all steps in the design flow are realized automatically by tools; (*iv*) It supports **Satistical Model Checking of system models (SMC)**.

The core idea of SMC is to **conduct simulations of the system and then use statistical results in order to decide whether the system satisfies the property or not**. For instance, SMC can be used

to **estimate the probability that a system satisfies a given property**. In contrast with exhaustive approaches, a simulation-based solution does not guarantee a correct result. However, it is possible to **bound the probability of making an error**. In PRO3D, SMC has been successfully applied through the Behaviour Interaction Priority (BIP) framework for evaluation of performance properties of mixed hardware/software system models.

– Exploiting 3D Opportunities into Multicore Architectures

The architecture developed in PRO3D project include several computation tiles connected via a mesh Network-on-Chip (NoC). The computation tiles are homogeneous and consist of a network interface for accessing the external NoC, a fast crossbar for quick access to local memory, a set of devices for effcient intra-tile synchronization and a variable number of Processing Elements (PEs). The memory hierarchy comprises several levels and memory types: L1 is the closest to the PE and can also be accessed by non local processing elements with an increased access time. L2 level is Tightly Coupled Data Memory (TCDM) directly connected via a logarithmic interconnect inside each cluster. L3 is a large memory that can be accessed by all the clusters of the fabric implementation as a central 3D stacked memory.

– Performing thermal aware system-level exploration & analysis of 3D designs.

The move towards architectures with tens of cores complicates the process of designing applications. The source of the complexity is two-fold: (1) the specification of the application must expose concurrency, while ensuring the lack of any of the common errors such as race conditions; (2) a highly concurrent application must be mapped on to the architecture with performance and/or temperature considerations.

The DOL formalism was selected as the programming model for the project and has been improved along several axes: (1) Specification and verification, as we specify data flow applications as a set of processes communicating via first-in-first-out (FIFO) buffers; (2) Code generation and calibration of the application for design space exploration of mappings; (3) An analytic approach to compute the worst-case die temperature; (4) Another approach to directly calibrate the thermal performance of an application without need for intermediate parameters. And finaly, (5) Real-time performance with thermal-aware run-time adaptation has been studied. The main options analyzed were: throttling by shaping the input of tasks, and DVS control of a processor.

– Virtual Prototyping

The lack of Electronic Design Automation (EDA) tools that can provide IC designers with efficient simulation of functional and thermal behavior of ICs limits the development of thermal-aware design and run-time approaches for 3D ICs. In PRO3D, we have successfully developed a new virtual platform prototyping framework targeting the full-system simulation of massively parallel heterogeneous 3D system-on-chip, composed by a general purpose processor and a many-core hardware accelerator.

PRO3D started on January 1st, 2010, and lasted 3 years until December 31st, 2012. Its results, including the 119 publications in various conferences, revues and workshops, are online at `http://pro3d.eu`.

2 Context & Objectives

PRO3D has started on January 1st, 2010, and lasted 36 months until December 31st, 2012.

This section details the PRO3D challenged followed by the early context of the project. Then we descibe how the context evolved during the project for memory technologies. Lastly we present the major PRO3D breakthrough on thermal issues and statistical model checking.

The PRO3D Challenges

According to study from Gartner,[1] the specific computing devices for the whole semi-conductor market amount to 100 B$. This was, and still is, one of the key differentiator for semiconductor industry in key markets such as Consumer, Wireless and Automotive. However the basic computing component started moving from simple microcontroller to manycore high performances embedded computing systems. The future of European electronic industry will heavily depend on mastering embedded computing systems.

During the last three decades, the performance of microprocessors & microcontrollers has steadily increased at the impressive rate of 100 times per decade. This was fuelled by:

- The exponential growth in clock speeds;
- The exponential growth in the number of transistors per die; and
- The acceleration of instruction flows obtained by various techniques for reducing latency and maximising the amount of computation per clock cycle.

This picture started to rapidly and radically changing. The shift to parallel architectures was not at all the consequence of a scientific breakthrough. It was primarily a consequence of hitting the technology walls that prevented from pushing forward the efficient implementation of traditional uniprocessor designs in silicon.[2,3]

These technology walls were responsible of the asymptotic diminishing returns of some of the growth curves that drove hardware design in the past, namely:

- Voltage scaling and power reduction techniques, or *Power Wall*;
- Instruction-level parallelism, or *Complexity Wall*;
- Memory latency hiding techniques, or *Memory Wall*;
- Reliable and low-variability silicon technology, or *Yield Wall*.

All these above-mentioned *walls* had to be *broken into* to maintain the growth rate of embedded computing in terms of computational efficiency.

Early Project Context

Provided the high computing density they offer, two key issues of embedded manycore are memory bandwidth and low power consumtpion. In order to cope with those requirements, one

[1] "Forecast Electronic Equipment, 2009", Gartner Analyst & Consultants.

[2] "The Landscape of Parallel Computing Research: A View from Berkeley," Krste Asanovic & al., EECS Department, University of California at Berkeley, Tech. Rep. UCB/EECS-2006-183, Dec. 18th, 2006.

[3] "Computer Architecture: A Quantitative Approach, 4th Edition", John L. Hennessy & David A. Patterson, Morgan Kaufmann, 4th edition.

industry proposition was to move from 2D, flat, memory subsystems like LPDDR to 3D stacking with dedicated memory layers using Wide-IO vertical interconnect. This solutions provided two cumulative advantages: the increase of data width in the vertical interconnect and shorteneing of the wires length. At the system level, the expected benefits of Wide-IO as seen by JEDEC around early 2011 are summarized on Fig. 1.

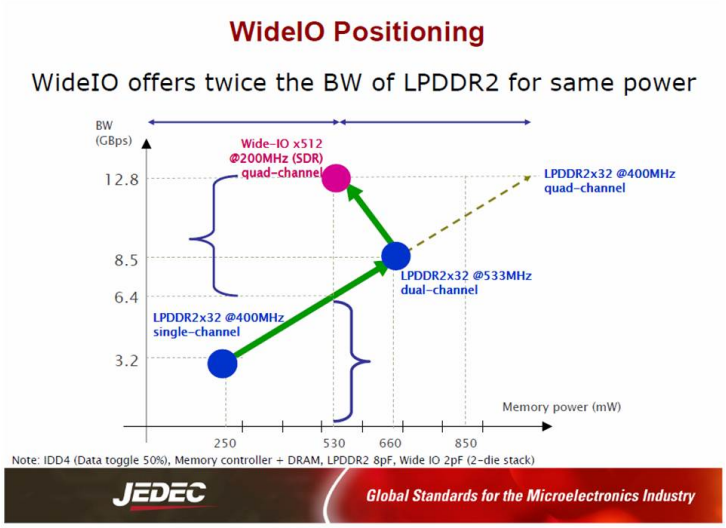

Figure 1: WideIO Positioning – Early 2011

Acceleration of the Low Power Memory Roadmap

Although Wide-IO has its own challenges (TSV limiting memory density, complex business model, power and thermal management, etc.) the progress of its developments was a wake up call for the LPDDR community. To the extend that by 2013, the LPDDR roadmap had been shortened enough to completely overlap the Wide-IO roadmap – See Fig. 2 [11].

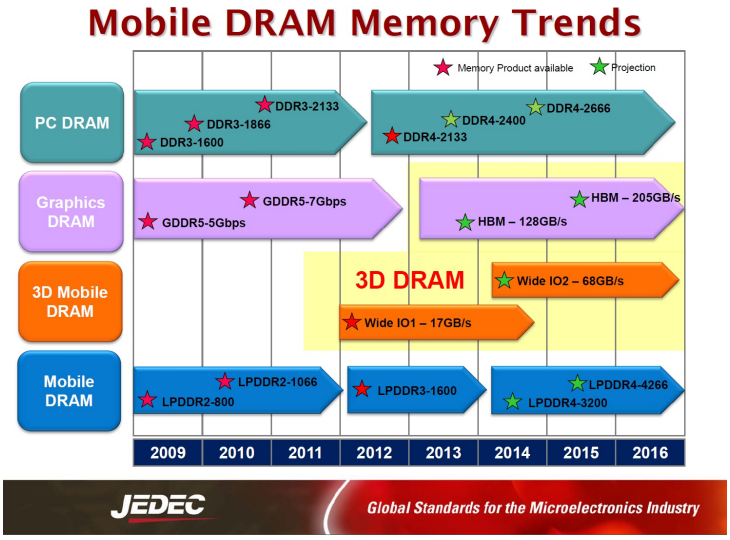

Figure 2: LPDDR vs. Wide-IO, from [11] – Early 2013

3 Main Scientific & Technical Results

Within this rapidely evoling context, PRO3D contributed to the following scientific areas:

1. **Thermal Modelling & Simulation**.
2. High-level programming, compilation, verification & deployment for 3D manycore architectures, including **Statistical Model Checking (SMC) of system models**.
3. Exploiting 3D opportunities into multicore architectures.
4. System-level exploration & analysis of 3D designs.
5. 3D manycore virtual prototyping.

The next two paragraphs give an overview of the two main achievements of PRO3D: thermal modelling and SMC. They are followed by sub-sections that further details the scientific results and advances in each of the four areas mentionned above.

Thermal Modelling and Simulation of 3D Stacks with 3D-ICE

Vertical stacking of multiple silicon layers, referred to as 3D stacking, is emerging as an attractive solution to continue the pace of growth of System on Chips (SoCs). The 3D technology allow to achieve smaller footprint in each layer and shorter vertical wires but thermal problems caused by higher power densities and greater thermal resistances contributes to impede the establishment of this technology. Thus, thermal modeling has become increasingly important in 3D chip design.

3D-ICE stands for **3D Interlayer Cooling Emulator**. It is a Linux based Thermal Emulator Library written in C, which can perform transient thermal analyses of vertically stacked 3D integrated circuits with inter-tier Microchannel Liquid Cooling. It is based on the conventional compact modeling of heat transfer by conduction in solids, and advances a novel compact modeling methodology, called the Compact Transient Thermal Modeling (CTTM), for heat transfer by convection in microchannels [17, 18].

This simulator is ideal for situations where a quick estimate of chip temperatures is required. 3D-ICE can be used to assist early stages of architecture design as for floorplanning of 3D multicore architectures or for the enhancement of liquid cooling efficiency through channel modulation [5, 32]. Simulations run with 3D-ICE contributed as well to the development of new strategies for optimal placement of thermal sensors or for the validation of run time thermal-aware management policies [4, 19].

Statistical Model Checking of System Models with BIP

SMC has recently emerged as an alternative to exhaustive exploration of the state-space for system validation [12, 13, 14]. The core idea is to **conduct simulations of the system and then use statistical results in order to decide whether the system satisfies the property or not**.

Additionally, SMC can also be used to **estimate the probability that a system satisfies a given property**. Of course, in contrast with exhaustive approaches, a simulation-based solution does not guarantee a correct result. However, it is possible to **bound**

the probability of making an error. Simulation-based methods are known to be far less memory and time intensive than exhaustive ones, and are sometimes the only option. SMC is getting widely accepted in various research areas such as systems biology, software engineering.

In PRO3D, it has been successfully applied through the BIP framework for evaluation of performance properties of complex, mixed hardware/system models [25, 29]. There are several reasons for this success. First, **SMC is very simple to implement, understand and use**. Second, **it does not require extra modeling or specification effort, but simply an operational model of the system**, that can be simulated and checked against state-based properties. Third, **it allows validation of properties that cannot be expressed in classical temporal logics**.

3.1 Thermal Modelling & Simulation

Focusing on the thermal modelling of modern 3D ICs, in the last few years, several authors have attempted to address the 3D-IC thermal modeling by using various modeling methodologies. Unfortunately, most of these modeling and simulation solutions are based on either fine-grained finite-element/finite-difference methods. Both methods have a superlinear computational complexity with respect to size of the problem. As more and more number of IC-dies are stacked, the computational times of these solutions would become too long to be practical for effective design-space explorations. In PRO3D, we addressed the problem of computational complexity of existing thermal simulation methodologies by proposing 3D-ICE thermal modeling tool.

3D-ICE is a compact transient thermal model (CTTM) for fast thermal simulation of 3D ICs, targeting also inter-tier microchannel cooling. The major achievements of the proposed model are:

1. The 3D-ICE model is transient and can accurately predict the temporal evolution of chip temperatures when operational system parameters (heat dissipation, flow rate of coolant etc.) change during dynamic thermal management. We have validated the accuracy of the model with a commercial computational fluid dynamics (CFD) simulation tool as well as measurement results from a 3D test IC with microchannel heat transfer geometry (a maximum error of 3.4% in temperature).

2. The 3D-ICE model is compatible with the conventional transient compact thermal models of IC modeling tools. Thus, no major computational overhead is introduced due to the introduction of microchannels in the proposed model. This is critical for speeding up the performance-thermal optimization cycles during the design of 3D ICs (the proposed model shows up to 975x speed-up with respect to commercial CFD tools).

3. The proposed 3D-ICE model offers the designer the freedom to incorporate any suitable heat transfer geometry, such as microchannels or pin fin arrays. Only the empirical correlation set, representing the convective heat transfer has to be adjusted accordingly. These values can be derived from literature or can specifically be computed by conjugate heat and mass transfer CFD modeling of a unit-cell of the heat transfer structure.

4. A thermal simulator based on the proposed 3D-ICE model was implemented for multi-threaded CPU and for massively parallel GPUs. It was found that the parallelization of the simulation resulted in considerable time-savings, especially for large problem sizes (i.e., detailed thermal models for large 3D stacks with liquid cooling).

3.2 High Level Programming Model, Compilation & Statistical Model Checking

Concurrency is paramount for boosting software performance on manycore platforms. Nonetheless, correct and fast development of highly parallel, fine-grain concurrent software is known to be notoriously hard even for expert developers. In general, the inherent complexity of concurrent (handwritten) software is hardly manageable by current verification and validation methods and tools. Moreover, software adaptation and deployment to selected manycore platform targets usually require significant manual transformation, with no strong guarantees about their correctness.

Rigorous Design Flow for Development of Parallel Applications on Manycore

The PRO3D project proposed a holistic approach for the development of embedded applications on top of manycore 3D platforms. PRO3D activities range from programming to architecture exploration and fabrication technologies. The major challenges are the thermal management of 3D platforms and the rigorous, tool-supported design flow of parallel application software.

In the context of PRO3D, a rigorous design flow based on the BIP component framework [6] has been developped. The flow sticks to the general principles of *rigorous design* introduced in [21], namely:

- it is *model-based*, that is, both application software and mixed hardware/ software system descriptions are modeled by using a single, semantic framework. As stated in [21], this allows maintaining the coherency along with the flow by proving that various transformations used to move from one description to another preserve essential properties.
- it is *component-based*, that is, it provides primitives for building composite components as the composition of simpler components. Using components reduces development time by favoring component reuse and provides support for incremental analysis and design, as introduced in [7, 8, 9].
- it is *tool-supported*, that is, all steps in the design flow are realized automatically by tools. This ensures significant productivity gains, in particular due to elimination of potential errors that can occur in manual transformations.

The BIP design flow uses a single language to ensure consistency between the different design steps. This is mainly achieved by applying source-to-source transformations between refined system models. These transformations are proven correct-by-construction, that means, they preserve observational equivalence and consequently essential safety properties. The design flow involves several distinct steps, as illustrated in Fig. 3 page 12 and explained below:

1. The *translation* of the application software into a BIP model. This allows its representation in a rigorous semantic framework. Translations for DOL3D and other programming models (including synchronous, data-flow and event-driven) and domain specific languages into BIP are defined and implemented.
2. *Correctness checking of functional properties* of the application software. Functional verification needs to be done only on high-level models since safety properties and deadlock-freedom are preserved by different transformations applied along the design flow. To avoid inherent complexity limitations, the verification method relies on compositionality and incremental techniques.

3. The *construction of an abstract system model*. This model is automatically generated from 1) the BIP model representing the application software; 2) a BIP model of the target execution platform; 3) a mapping of the atomic components of the application software model into processing elements of the platform. The abstract system model takes into account hardware constraints such as various latencies, mutual exclusion induced from sharing physical resources (like buses, memories and processors) as well as scheduling policies seeking optimal use of these resources.

4. The *construction of a distributed system model*. This model is automatically generated from the abstract system model by expressing high-level coordination mechanisms e.g., interactions and priorities, in terms of primitives of the execution platform. This transformation involves the replacement of atomic multiparty interactions and/or dynamic priorities by protocols using asynchronous message passing (send/receive primitives) and arbiters ensuring semantics preservation. These transformations are proved correct-by-construction [15, 16, 27].

5. The *generation of platform dependent code*, including both functional and glue code for deploying and running the application on the target manycore. In particular, components mapped on the same core can be statically composed thus avoiding extra overhead for (local) coordination at runtime.

6. The *calibration* step, which consists in estimating execution times of actions of the distributed system model. These are obtained through execution and profiling of code fragments compiled on the target platform. They are used to obtain an instrumented system model which takes into account dynamic behavior of the execution platform.

7. The *performance analysis* step involving simulation-based methods combined with statistical model checking on the instrumented system model.

Statistical Model Checking of System Models

The system model captures, besides the pure functionality of the application software, all the non-functional constraints induced on it by the target platform. The system model can therefore be used to analyze non-functional properties such as contention for buses and memory accesses, transfer latencies, contention for processors, etc. In the proposed design flow, these properties are evaluated by simulation of the system model extended with observers. Observers are regular BIP components that sense the state of the system model and collect pertinent information with respect to the properties of interest i.e., the delay for particular data transfers, the blocking time on buses, etc. Actually, we provide a collection of predefined observers allowing to monitor and record specific information for most common non-functional properties.

Simulation is performed by using the native BIP simulation tool. The BIP system model extended with observers is used to produce simulation code that runs on top of the BIP engine, that is, the middleware for execution/simulation of BIP models. The outcome of the simulation with the BIP engine is twofold. First, the information recorded by observers can be used as such to gain insight about the properties of interest. Second, with some caution, the same information can be used to build much simpler, abstract stochastic models. These models can be further used to compute probabilistic guarantees on properties by using statistical-model checking. This two-phase approach combining simulation and statistical model-checking has been already experimented in different contexts, such as validation of system-level Quality of Service (QoS) properties on complex heterogeneous systems. It is fully scalable and allows (at

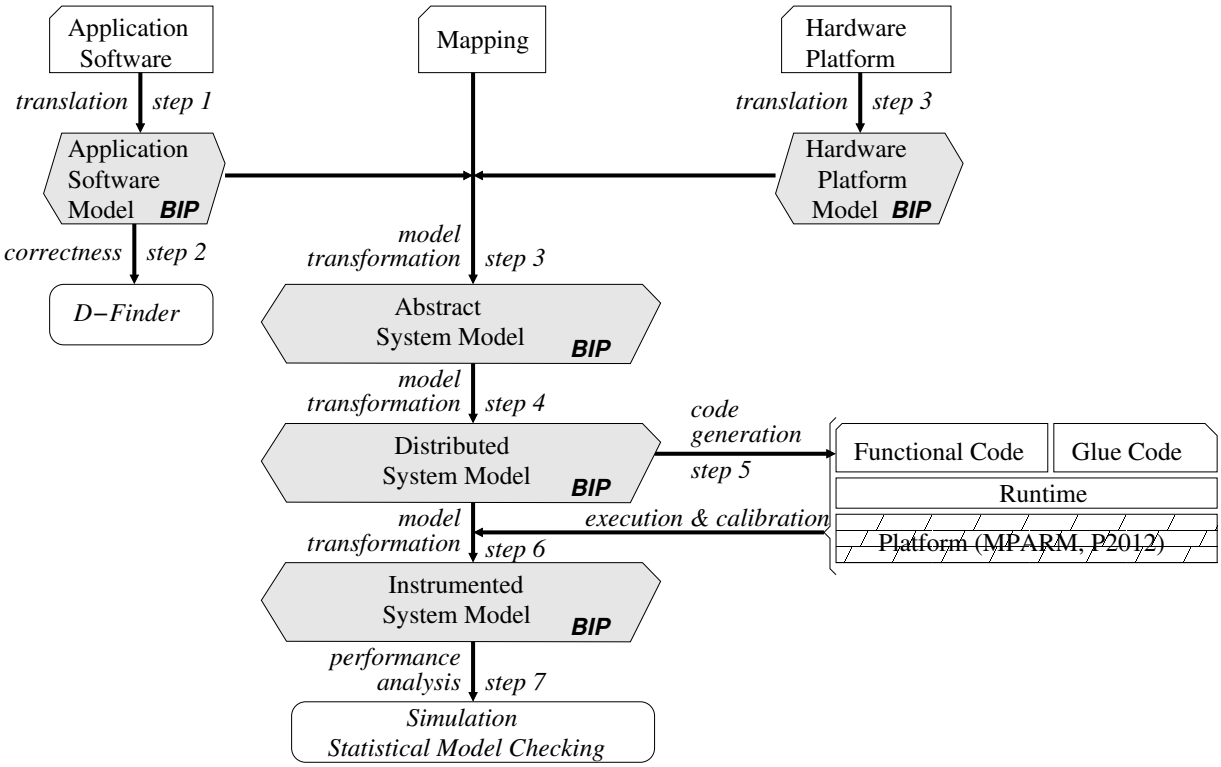

Figure 3: BIP Design Flow for Manycore developped in PRO3D

least partially) overcoming the drawbacks related to simulation-based approaches, that is, the long simulation times and the lack of confidence in the results obtained.

Statistical model checking [12, 13, 14] extends runtime verification capabilities by exploiting statistical algorithms in order to get some evidence that a given system satisfies some property. Given a stochastic system B and a property ϕ, statistical model checking refers to a series of simulation-based techniques that can be used to answer two questions: (1) qualitative: is the probability for B to satisfy Φ greater or equal to a certain threshold θ ? and (2) quantitative: what is the probability for B to satisfy Φ ? The main approaches proposed to answer these questions are based on hypothesis testing and/or estimation using Monte-Carlo methods. The efficiency of the various algorithms is characterized by the number of simulations needed to obtain an answer. This number may change from system to system and can only be estimated. However, heuristics have been developed for combining estimation and testing algorithms, to exploit potential parallelism for simulation or specific characteristics of the system. Their combination make statistical model checking as a very interesting, general and scalable, alternative for verification, which moreover benefit from a lot of interest from the research community.

The BIP design flow allows generation of a correct-by-construction system model for manycore platforms from an application software and a mapping. The flow clearly separates software and hardware design issues. It is also parameterized by design choices related to resource management such as scheduling policies, memory size and execution times. This allows estimation of the impact of each parameter on system behavior. Using BIP as a unifying modeling formalism for both hardware and software confers multiple advantages, in particular rigorousness. The obtained system models are correct-by-construction. This is a main difference from other ad hoc model construction techniques.

Methods for Distributed Implementation of Multiparty Interactions and Priorities

Distributed decentralized implementation of systems of communicating processes raises non-trivial problems. In general, the correct execution of multiparty interactions, subject to priority rules, requires sophisticated mechanisms for runtime conflict detection and resolution.

The construction of optimally distributed and decentralized implementations for BIP models has been thoroughly investigated within the PRO3D project. The BIP design flow includes a distribution/decentralization step (step 4 in Fig. 3), which is meant to transform arbitrary, high-level application models (in BIP) into simpler, low-level models (in BIP as well) that can be effectively deployed on distributed execution platforms. More precisely, this transformation replaces high-level multiparty interactions and/or dynamic priorities that might exist in high-level models by protocols using only asynchronous point-to-point communication primitives. This transformation is implemented as model transformations on BIP and proven correct-by-construction [27, 28].

In particular, the work on the decentralized implementation of dynamic priorities lead to the development of *knowledge-based methods* [26, 30, 31]. These methods proven to significantly reduce the coordination overhead in decentralized implementations. In general, knowledge denotes the information about the current global state of execution a process may know/infer at any time, by combining its local view with some pre-computed information about all the reachable states of the system. The knowledge can be used to get rid of useless arbitration required for false conflict resolution e.g., to restrict communication whenever irrelevant for the distributed protocol.

For example, a method for detection of false conflicts which combines partial observation of the system's state and apriori knowledge extracted from invariants has been presented in [26]. This method relies on heuristics to determinine optimal sets of observations leading to implementations with particular guarantees. It also provides preliminary experimental results on an implementation of the method in the BIP framework. Another way to benefit from knowledge has been investigated in [31]. In this work, information about non-conflicting interactions is obtained by combining the history (the sequence) of past interactions with an apriori known global model of the system behavior. This knowledge can be exploited locally, for every participant in a distributed implementation (e.g., both to simple participant or to coordinator components). Finally, the work in [30] makes one step more towards efficient implementation. It has been shown that priorities can be safely replaced using an additional observation primitive which, moreover, can be used to effectively reduce coordination overhead in combination with standard protocols for distributed execution of multiparty interactions.

The distributed code is generated against the MCAPI runtime.

MCAPI Implementation for STHORM

The Multicore Association is an industry association that includes leading companies implementing products that embrace multicore technology (vendors of processors, operating systems, compilers, development tools, debuggers, ESL and EDA vendors, simulators, application and system developers, and universities. Their primary objective is to define and promote open specifications to enable multicore product development.

The roadmap of the Multicore Association consists of an extensive set of Application Programming Interface (API) that support multicore communications, resource management, task management, and debug facilities. These API provide a foundation for a multitude of services and functions including load balancing, power management, reliability, and quality of service,

elements that are also on the consortium's roadmap.

Currently Multicore Association foundation APIs are divided into three projects:

1. *Communication*: defines an API and a semantic for communication and synchronization between processing cores in embedded systems.
2. *Resource Management*: specifies essential application-level resource management capabilities.
3. *Task Management*: will provide a standard API for leveraging task parallelism on embedded devices containing symmetric or asymmetric multicore processors, covering a wide range of target architectures.

The two first projects were published first and their main results were the delivery of two APIs, the MCAPI and the MRAPI [1, 2]. The later gave birth to MTAPI [3]. For PRO3D, we focused on the MCAPI specification.

The MCAPI specification defines an API and a semantic for communication and synchronization between processing cores in many-core embedded systems. It is based on few basic concepts presented Fig. 4 page 14.

- *Node*. A node is an independent thread of control that can communicate with other nodes. The exact nature of a node is defined by the implementation of the MCAPI, but it can basically be a process, a core, a thread or a HW IP.
- *Endpoint*. An endpoint is a communication termination point and is therefore connected to a Node. One node can have multiple endpoints, but one endpoint belongs to a single node.
- *Domain*. A domain is a set of MCAPI nodes that are grouped together for identification or routing purpose. The semantics attached to a domain is given by the implementation, but it can basically be a cluster, a chip, a terminal, a mobile... Each node can only belong to a single domain.

Compared to MPI, MCAPI offers inter-core communication with low-latency with minimal footprint. MCAPI communication nodes are all statically predefined by the implementation.

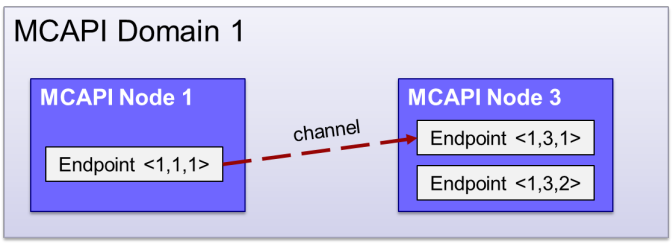

Figure 4: MCAPI Main Concepts

MCAPI offers three fundamental communication mechanisms:

- *Messages*. Messages are streams sent from one endpoint to another. No connection is established between the two endpoints to send a message. This is the easiest way of communicating between two nodes.

- *Packet Channel.* A packet channel is a FIFO unidirectional stream of data packets of variable size, sent from one endpoint to another.
- *Scalar Channel.* A scalar channel is similar to a packet channel, except that only fixed-length word of data can be sent through the channel. A word may be 8, 16, 32 or 64 bit of data.

The communication channels can then be set up between two different *Endpoints*.

The main objective of our implementatuion on P2012/STHORM was to offer a uniform level of abstraction and a homogeneous programming interface for the whole platform, both fabric side and host side. This uniform intermediate representation covering the whole platform can the perfectly integrate in a full application design flow targeting STHORM, contributing to ease the code generation.

The PRO3D implementation of MCAPI for STHORM features five domains: four for the clusters on the computing fabric side, and one for the host (see figure 5).

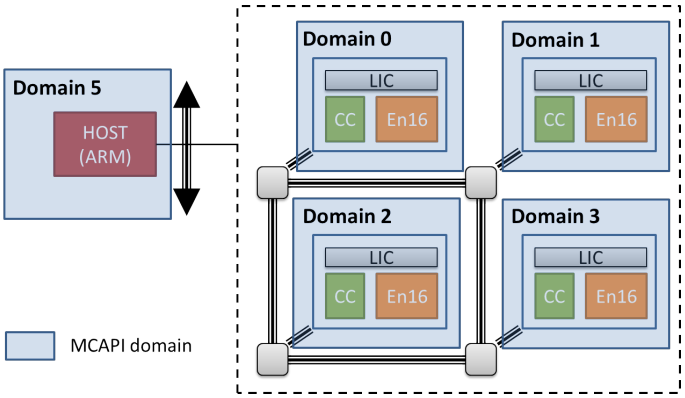

Figure 5: MCAPI Domains on STHORM

The four *cluster* domains are naturally reflecting the STHORM hardware organization. The 16 Processing Elements (PEs) of the ENCore blocks are mapped to 16 available MCAPI nodes in each MCAPI domain. As the PEs are generic processors and as all components in STHORM are memory mapped and can be accessed by any PE, the implementation can decide how many MCAPI endpoints per node it can offer, since no real hardware constraint exists for the number of link between the processors. Currently we set the maximum number of endpoints per node to be 16 as it seems to be a code trade-off between code size and complex application support.

The *host domain* has only a single node which represents the ARM dual-core host processor. The MCAPI host node is responsible for the whole execution, as it is the main entry point of application. The MCAPI initialization of the host node automatically loads the fabric binary code into the L2 memory, and wake up the fabric MCAPI nodes.

This implementation totally hides the complexity of binary and symbols dynamic load and dependencies solving to the application provided. Instead a single semantic for the whole STHORM platform is exposed for homogeneous programming.

The connected channels are implemented using FIFO buffers allocated in any of the available memory region of STHORM. The size of the allocated buffer and its memory mapping can be specified using endpoint attributes, as specified in the MCAPI standard. The sending endpoint and the receiving endpoint must have consistent attributes definition for the creation of a

channel. The creation of the channel is also dependent on the remaining available memory in the platform.

The Cluster Controler (CC) and the Fabric Controler (FC) are not directly visible from the MCAPI mapping on STHORM, but they are used by the implementation to support the various communication mechanisms and synchronizations. DMA engines are also hidden by the implementation but used when transferring data from one region to another.

The STHORM MCAPI implementation is curently used as target for the PRO3D flow described above. The code produced works properly under STHORM simulators from the SDKs 2012.2, 2013.1 and 2013.2. At the time of this report, the implementation passes unitary tests on real STHORM hardware, where it supports breakpoints on the host and traces provided by the hardware platform on the PE.

3.3 System-Level Exploration & Analysis of 3D Designs

Distributed Operation Layer for 3D Integrated Circuits (DOL3D) tool-chain

The move towards highly parallel architectures with tens of cores, such as P2012, complicates the process of designing applications. The source of the complexity is two-fold. On the one hand, the specification of the application must expose concurrency, while ensuring the lack of any of the common errors such as race conditions. On the other hand, a highly concurrent application must be mapped on to the architecture with performance and/or temperature considerations. Such a mapping step is difficult as there are typically a very large number of candidate mappings to choose from. In the project, we have taken a systematic approach to alleviate these complexities with the DOL3D tool-chain.

- *Specification and verification.* We specify the dataflow applications as a set of processes communicating via first-in-first-out (FIFO) buffers. The semantics of the buffers is to have a blocking read and a non-blocking write. The functionality of the communicating processes are authored in a straight forward C interface, with clearly separated communication primitives. The interconnection amongst the processes is specified by a standard xml format. This clear separation between the functionality of the processes and the interconnection amongst processes allows us to formally verify that there are no deadlocks in the applications. This is done with a functional simulator that automatically translates the process network specification to a SystemC specification which is then simulated. This step also allows the application designer to check the data rate amongst different processes. For instance, we specifically verify if the given application satisfies a Synchronous Dataflow (SDF) specification. Thus, this first step of the tool-chain simplifies the effort of the application designer. We demonstrated this tool-chain at the first project review meeting.
- *Mapping optimisation.* Once the application has been specified and verified, we need to find a mapping of the processes on to the cores such that the performance can be optimised. To this end, we calibrate the performance of the application for different candidate mappings and derive a performance model. This step requires automatic code generation with instrumentation for calibrating timing information. Such calibration is done at a fine granularity. Specifically, we calibrate the time required by each continuous code block in the application, and the time required by each communication step. As the final step, the calibrated performance model is used to explore the design space of the very large number of mappings. We use genetic algorithms to explore Pareto-optimal mapping

solutions. To this end, we use an exploration tool developed earlier by our group, called EXPO. We demonstrated this tool-chain at the first project review meeting.

- *Compatibility with partners.* The specification of the application and architecture, as proposed in DOL3D, was adopted in the project. The applications are written and verified for this specification. The BIP tool-chain developed by VERIMAG also supports this specification. We demonstrated the tool-chain first for the MPARM architecture proposed by UNIBO. This architecture and the programming layer it supports, were specifically designed to be similar to the P2012 platform. We also ported the tool-chain to support the MCAPI programming layer developed by CEA to execute on the P2012 platform. Thus, the DOL3D tool-chain was particularly designed to be compatible with the various partners of the project.

Formal analysis of worst-case peak temperature

As has been discussed throughout this project, analysing the on-chip temperatures is necessary when targeting applications for 3D or high performance architectures. Often the thermal requirements only consider the peak or worst-case temperature, i.e., to rate a given architecture and application pair, we only need to verify that the worst-case temperature does not exceed a given system threshold. This concern is similar to the field of hard real-time systems where the main requirement is to ensure that worst-case finish time of a task does not exceed its deadline. With our prior experience in analysis of real-time systems, we aimed to provide a similar formal analysis for computing worst-case temperature.

In this regard, we made the first contribution in [20]. In this paper, we showed that an analytic approach can be used to compute the worst-case temperature, for a given stream of tasks. The tasks can show variability such as jitter in arrival times, or occasional bursts. In all such conditions, we computed the "critical trace" that would lead to the worst-case temperature. A significant part of the result was that the model of the application required for such an analysis is the same as the one used in real-time scheduling analysis. In other words, independent of the thermal model, to compute the worst-case temperature, we need to model the application in the same way as we do to verify real-time properties. Thus, no additional modelling effort on the application-side is required to compute the worst-case peak temperature.

We extended this result in the following directions.

- *Other scheduling algorithms:*
  The original result was presented for a first-come-first-serve (FCFS) scheduling algorithm. Thus, only a single stream of homegenous jobs were analysed. We extended this to several scheduling algorithms in [24]. Here we combined the results of the analysis with our toolbox for Modular Performance Analysis (MPA) available as a MATLAB toolbox at `http://mpa.ethz.ch`. We showed that several different scheduling algorithms such as fixed priority, round robin or time division multiplexing (TDM) can be used. In all cases, we showed how the trace that leads to the worst-case temperature can be analytically computed.

- *Multi-core systems:*
  The original result was presented for a single core system. This result was extended to multi-core systems in [38] and [39]. The main contribution here was to show that an exact computation for the worst-case temperature for multi-core systems is difficult, as the execution of tasks on one core affect the temperature of other cores. We thus provided

good approximation solutions which were then analysed for their accuracy. This result also gave insights into how the thermal analysis of multi-core systems is more involved due to the thermal coupling of the cores.

- *Distributed systems:*
  Finally, we extended the results to consider applications with several tasks connected by dataflow edges executing on multi-core [23] systems. The main contribution was to show that it is not sufficient to have estimates of only the worst-case timing behaviour in order to compute the peak temperature. A shorter execution of a job on a certain processor can lead to a higher temperature on a different processor. This counter-intuitive result must be adequately considered when calibrating distributed applications for peak temperature computations.

This stream of research merged the two branches of formal analysis and thermal-aware design, with results presented in both "design" conferences and "real-time" conferences.

Effective thermal calibration of an application

A crucial component of a design optimisation tool is the calibration step. For instance, we need to know the thermal effects of running different applications. We started the project by showing that different mapping decisions can indeed affect the temperature of the chip [41]. However, this method required that we have detailed information of the power consumption of the different architectural units, the thermal parameters of the chip and so on. The wide scope of these parameters implies that they are not easily available, especially early in the design cycle.

As a response to this, we presented an approach to "directly" calibrate the thermal performance of an application without need for intermediate parameters. The method relies in learning the characteristics of the system ("calibration step") to determine a thermal impulse response of the given system. The crucial assumption is that each application exercises the micro-architectural units of a core in a unique fashion, and the usage scales uniformly with the load, for the same application. The overall results show a high accuracy in the estimated temperatures (at most 5 deg C), as compared to Hotspot, while the computational complexity is about 1000 times smaller. This work was recognised with the Best Paper award at CASES [40] which is part of the Embedded Systems Week.

Real-time performance with thermal-aware run-time adaptation

As has been discussed in other work packages, run-time adaption in response to on-chip temperatures seems to be an essential part of supporting 3d or high performance architecture. Several methods have been proposed earlier in the literature to enable such adaption. Examples include dynamic voltage scaling (DVS), run-time migration and system throttling. Clearly, any such intervention can interfere with the performance offered by the architecture to a supported application. Our aim was then to come up with techniques to analyse and control such interference.

We first studied the problem of throttling. The system can be occasionally throttled to reduce the temperature. This can be implemented at a software level by delaying the incoming tasks, a process known as shaping of the input stream. We studied the problem of identify the shaping parameters which ensure that the deadline of a system is met while optimally minimising the peak temperature. We showed that it is possible to analytically compute an approximation of such a shaper. The approximation was made such that the resultant shaper can be easily implemented. This result was presented at DAC [22], and was nominated for a best paper

award.

As the second problem we studied the DVS control of a processor. We considered an intuitive framework, wherein when the temperature of the processor rises above a threshold the frequency of the processor is reduced. Though the control mechanism is intuitive, it can be difficult to analyse the performance guaranteed by such a processor. We showed that by simulating a single worst-case stream of jobs we can compute guarantees on performance provided by such a processor. The advantage of the result was the non-dependence on the control law. This was presented in RTAS [37] as part of the CPSWeek, and was also nominated for the best paper award. We have subsequently extended the result to consider a given starting initial temperature and sensor errors.

3.4 From 3D Opportunities to 3D Manycore Architectures

In order to increase the performance of electronic devices, Three-Dimensional Integrated circuits (3D-IC) have emerged. 3D-ICs are made of the stack of several functional layers that are electrically connected thanks to Through-Silicon-Vias (TSV). The interconnection architecture and memory hierarchy play a key role in these 3D systems, being responsible not only for communication between the various elements but also for the management of fundamental mechanisms in a multiprocessor environment, such as shared resources utilization, data sharing, synchronization and performance scalability. One of the most important objectives of PRO3D has been focused on exploring and analysing the system-level 3D design space, with the aim of optimizing the overall system performance.

From the interconnection fabric design perspective, there are several approaches already presented in literature: the classical shared bus structures are flanked by a crossbar, ring, and the emergent paradigm of Network-on-Chip. Each design approach has advantages and drawbacks in terms of complexity and performance. For example, the shared bus is the easiest to design, but quickly becomes limiting in terms of bandwidth and latency, when scaling to a high number of bus masters or cores. The crossbar allows communication between several IP modules, but increasing of area makes this approach often not feasible. The Network-on-Chip paradigm instead provides good scalability with regards to the number of processors, providing at the same time high bandwidth, modularity and a high degree of customization; all at the expense of design complexity.

The main platform architecture developed in the PRO3D project can be considered a derivation of the P2012 platform template, where several computation tiles are connected via a mesh NoC. The computation tiles are supposed to be homogeneous and consist of a network interface for accessing the external NoC, a fast crossbar for fast access to local memory, a set of devices for efficient intra-tile synchronization and a variable number of core tiles.

Focusing on the performance of the overall communication architecture, the local crossbar plays a fundamental role. The modelled crossbar is a logarithmic interconnect module. From a behavioural standpoint, it is a Mesh-of-Trees interconnection network to support high-performance communication between processors and memories in L1-coupled processor clusters. The module is intended to connect processing elements to a multi-banked memory on both data and instruction side. Data routing is based on address decoding: a first-stage checks if the requested address falls within the intra-cluster memory address range or has to be directed off-cluster. To increase module flexibility this stage is optional, enabling explicit L3 data access on the data side while, on the instruction side, can be bypassed letting the cache controller take care of L3 memory accesses for lines refill. The interconnect provides finegrained address interleaving on the mem-

ory banks to reduce banking conflicts in case of multiple accesses to logically contiguous data structures. The last log2 (interleaving size) bits of the address determine the destination. The crossing latency consists of one clock cycle. In case of multiple conflicting requests, for fair access to memory banks, a round-robin scheduler arbitrates access and a higher number of cycles is needed depending on the number of conflicting requests, with no latency in between. In case of no banking conflicts data routing is done in parallel for each core, thus enabling a sustainable full bandwidth for processors-memories communication. To reduce memory access time and increase shared memory throughput, read broadcast has been implemented and no extra cycles are needed when broadcast occurs.

From the memory hierarchy perspective, the optimal configuration features a complex organization which comprises several levels and memory types. L1 level is the closest to the processing element and can be accessed locally by its embedding processing element but may also be accessed by non local processing elements with an increased access time. In order to provide high bandwidth and to avoid bottlenecks on memory interface, L1 is a multiported, multibanked memory, exploiting address interleaving among ports. L2 level is an intermediate level typically used for storing data which are shared by several processing elements of the same cluster. L3 is a large memory that can be accessed by all the clusters of the fabric. The implementation of level L3 is a central 3D stacked memory.

Clearly, the organization of memory hierarchy is one of the most important and critical phases in manycore architecture design. Dealing with massively parallel systems such as P2012, instruction caching, data memory and DRAM interfacing play a fundamental role since they must provide the required bandwidth to all cores and software tasks, complying with tight constraints in terms of size and complexity.

Key to providing I-fetch bandwidth for cluster-based architecture is an effective instruction cache architecture design. We analysed and compared the two most promising architectures for instruction caching, namely private instruction caches per core and shared instruction cache per cluster. Experimental results showed that private cache performance can be significantly affected by the higher miss cost, on the other hand the shared cache has better performance. However, it is very sensitive to execution misalignment, which can lead to cache access conflicts and high hit cost.

On the data side, a multi-ported, multi-banked, Tightly Coupled Data Memory (TCDM) has been directly connected to the logarithmic interconnect inside each cluster. The number of memory ports is equal to the number of banks to have concurrent access to different memory locations. Once a read or write requests is brought to the memory interface, the data is available on the negative edge of the same clock cycle, leading to two clock cycle latency for conflict-free TCDM access. As already mentioned above, if conflicts occur there is no extra latency between pending requests, once a given bank is active, it responds with no wait cycles. Banking factor (i.e. ratio between number of banks and cores) can be configured to explore how this affects banking conflicts.

From the DRAM interfacing design perspective, to overcome the pin-limited performance growth, the power vs. bandwidth dilemma and the memory wall 3D integration and 3D-stacked DRAM have been proposed as a very promising solution. 3D-stacked DRAMs reduce the distance between CPU and external DRAM from centimeters to micrometers and improve the bandwidth and access latencies - but more importantly, they provide a major boost in energy efficiency in comparison to standard DRAM devices, such as DDR2, LPDDR2 or DDR3. During PRO3D, we focused on the energy optimization of the 3D-DRAM subsystem for future terminals. We proposed a flexible bandwidth and burst length adaption for the 3D-DRAM and the controller.

With this we are able to handle a large range of access sizes from fine-grained (32b) to coarse-grained (1Kb).

We have investigated different DRAM families. 3D-DRAM densities from 256Mb to 4096Mb and also 2Gb LPDDR/LPDDR2 devices were characterized by several application-relevant workloads. We have then designed a 3D-DRAM channel controller which fits perfectly to the flexible bandwidth and burst length adaption interface of the 3D-DRAMs. By using this flexible interface the total 3D-DRAM subsystem consumes in the fastest configuration for 3D-DRAM (300MHz, 2Gb DDR) and controller (500MHz) less than 240mW, for a workload with PHR = 50%. With very low power settings of 3D-DRAM (167MHz, 2G SDR) and controller the subsystem power is decreased to less than 140mW. The experimental results showed an overall average of 37% power savings by enabling the bandwidth and burst length flexibility for 3D-DRAMs.

3.5 3D Manycore Virtual Prototype

With conventional scaling of CMOS devices fast running into theoretical walls, vertical integration of IC dies seems to be the most viable solution to meet the ever rising demands for more compact and faster electronic products in the short- and medium-term. Even if vertical integration is a promising solution to further increase the performance of future ICs, such 3D ICs present complex thermal issues that cannot be solved only by conventional cooling techniques. Optimal exploitation of such complex computing architectures can be only achieved via efficient programming and management of the entire 3D stack. However, one of the main reasons which still limits 3D technology is the lack of Electronic Design Automation (EDA) tools that can provide IC designers with efficient simulation of functional and thermal behavior of ICs. In particular, this situation becomes a major bottleneck for the development of thermal-aware design and run-time approaches for 3D ICs. One of the main objectives of the PRO3D project has been the development of such innovative tools and their interfacing enabling the efficient design and programming of such 3D manycore architectures, namely the functional and thermal simulation tools.

Performance modelling plays indeed a critical role in the design, evaluation, and development of computing architecture of any segment, ranging from embedded to high performance 3D manycore-based processors. Simulation has historically been the primary vehicle to carry out performance modelling, since it allows for easily creating and testing new designs several months before a physical prototype exists. Performance modelling and analysis are now integral to the design flow of modern computing systems, as it provides many significant advantages: i) accelerates time-to-market, by allowing the development of software before the actual hardware exists; ii) reduces development costs and risks, by allowing for testing new technology earlier in the design process; iii) allows for exhaustive design space exploration, by evaluating hundreds of simultaneous simulations in parallel. Besides the complex hardware, generally modern platforms host also an advanced software eco-system, composed by an operating system, several communication protocol stacks, and various computational demanding user applications.

Unfortunately, as processor architectures get more heterogeneous, complex and the third dimension is added to the picture, it becomes more and more difficult to develop simulators that are both fast and accurate. Cycle-accurate simulation tools can reach an accuracy error below 1-2%, but they typically run at a few millions of instructions per hour. The necessity to efficiently cope with the huge HW/SW design space provided by the 3D target architecture makes clearly full-system simulator one of the most important design tools. Clearly, the use of slow simulation techniques is challenging especially in the context of full-system simulation. In order to perform

an affordable processor design space exploration or software development for the target platform, trade-off accuracy for speed is thus necessary by implementing new virtual platforms that allow for faster simulation speed at the expense of modelling fewer microarchitecture details of not-critical hardware components (like the host processor domain), while keeping high-level of accuracy for the most critical hardware components (like the manycore accelerator domain like P2012).

During PRO3D we have successfully developed a new virtual platform prototyping framework targeting the full-system simulation of massively parallel heterogeneous 3D system-on-chip, composed by a general purpose processor (i.e. intended as platform coordinator and in charge of running an operating system) and a many-core hardware accelerator (like P2012, i.e. used to speed-up the execution of computing intensive applications or parts of them). The developed virtual platform exploits the speed and flexibility of QEMU, allowing the execution of a full-fledged Linux operating system, and the accuracy of a SystemC model for the many-core-based P2012 accelerator.

The specific features of the simulation environment are:

- Since it exploits QEMU for the host processor emulation, unmodified operating systems can be booted on it and the execution of unmodified ARM binaries of applications and existing libraries can be simulated.
- It enables accurate manycore-based accelerator simulation. We designed a full software stack allowing the programmer to exploit the hardware accelerator model implemented in SystemC, from within a user-space application running on top of QEMU. This software stack comprise a Linux device driver and a user-level programming API.
- The host processor (emulated by QEMU) and the SystemC accelerator model can run in an asynchronous way, where a non-blocking communication interface has been implemented enabling parallel execution between QEMU and SystemC environments.
- Beside the interface between QEMU and the SystemC model, we also implemented a synchronization protocol able to provide a good approximation of the global system time.
- The simulator can be also used in stand-alone mode, where only the hardware accelerator is simulated, thus enabling accurate design space explorations.
- The 3D memory hierarchy of the hardware accelerator is fully modelled, as well as its 3D communication architecture.

4 Dissemination

Over the course of its 36 months, PRO3D issued 119 publications in various conferences, revues and workshops. The full list is available on our public web site at `http://pro3d.eu`.

Several public deliverable are also available from the same site in the section *Deliverables*, like: *"D3.1 – Report on 3D Aware Specification of Parallel Applications"*, v1.2, Saddek Bensalem & Marius Bozga (Eds.), 2011-07-01.

An open source software thermal estimator for the low level support of PRO3D, "3D Interlayer Cooling Emulator (3D-ICE)", has been delivered in 2010. It is available at `http://esl.epfl.ch/3d-ice.html`.

5 Contact Points

The PRO3D consortium brought together partners leaders with competencies in software, architecture and 3D integration:

- The *Département d'architecture, conception et logiciel embarqué* from LETI of the *Commissariat à l'énergie atomique et aux énergies alternatives* (CEA), which was the Project's Coordinator (`http://www.leti.fr`);
- The *Distributed & Complex System* (DCS) team at VERIMAG (`http://www-verimag.imag.fr/DCS,31.html`). VERIMAG being a Joint Lab was represented by Université Joseph Fourier Grenoble 1 (UJF);
- The *Computer Engineering Group* (`http://www.tec.ethz.ch`) at *Eidgenössische Technische Hochschule Zürich* (ETHZ);
- The *Microelectronics Research Group* (`http://www-micrel.deis.unibo.it`) from *Università di Bologna* (UNIBO);
- The *Computing Division* (`http://www.st.com`) from *STMicroelectronics* (STM);
- The *Microelectronic Systems Laboratory* (`http://lsm.epfl.ch`), *Embedded Systems Laboratory* (`http://esl.epfl.ch`) & *Integrated Systems Laboratory* (`http://lsi.epfl.ch`) from École polytechnique fédérale de Lausanne (EPFL);

Web Site

The PRO3D consortium has a web site available at `http://pro3d.eu`, and was coordinated by Christian FABRE, from CEA LETI. He can be reached at `christian.fabre1@cea.fr`.

Logo

We have defined and used a PRO3D logo in two shapes: square and rectangular. They are presented Fig. 6 below. Each shape is available in various sizes:

Figure 6: Logos of PRO3D

References

[1] *"The Multicore Communications API (MCAPI™) v2.015"* Available at: `http://www.multicore-association.org`.

[2] *"The Multicore Resource Management API (MRAPI™)"* Available at: `http://www.multicore-association.org`.

[3] *"The Multicore Task Management API (MTAPI™)"* Available at: `http://www.multicore-association.org`.

[4] *"EigenMaps: Algorithms for Optimal Thermal Maps Extraction and Sensor Placement on Multicore Processors"*, J. Ranieri, A. Vincenzi, A. Chebira, D. Atienza, M. Vetterli. Design Automation Conference (DAC 2012), San Francisco.

[5] *"Fast and Scalable Temperature-driven Floorplan Design in 3D MPSoCs"*. Arnaldo, Ignacio; Vincenzi, Alessandro; Rodrigo, Ayala; Luis, José; Risco, José L.; Hidalgo, J. Ignacio; Ruggiero, Martino; Atienza Alonso. Proceedings of the 13th IEEE Latin American Test Workshop (LATW2012). Quito, Ecuador.

[6] *"Modeling Heterogeneous Real-time Systems in BIP"*. A. Basu, M. Bozga, and J. Sifakis. In *SEFM'06 Proceedings*, pages 3–12. IEEE Computer Society Press, 2006.

[7] *"Compositional Verification for Component-based Systems and Application"*. S. Bensalem, M. Bozga, T-H. Nguyen., and J. Sifakis. In *ATVA'08 Proceedings*, volume 5311 of *LNCS*, pages 64–79. Springer, 2008.

[8] *"D-Finder: A Tool for Compositional Deadlock Detection and Verification"*. S. Bensalem, M. Bozga, T-H. Nguyen, and J. Sifakis. In *CAV'09 Proceedings*, volume 5643 of *LNCS*. Springer, 2009.

[9] *"Incremental Component-based Construction and Verification using Invariants"*. S. Bensalem, M. Bozga, A. Legay, T-H. Nguyen, J. Sifakis, and R. Yan. In *FMCAD'10 Proceedings*, pages 257–266. IEEE, 2010.

[10] *"Platform 2012: A Manycore Programmable Accelerator for Ultra-Efficient Embedded Computing in Nanometer Technology"*. Whitepaper. STMicroelectronics and CEA. November 2010.

[11] *"Mobile Memory Technology Roadmap"*. Hung Vuong, JEDEC & Qualcomm Technologies Inc. May 2013, *JEDEC's Mobile Forum 2013*, Santa Clara, CA, USA. `http://www.jedec.org/sites/default/files/files/H_Vuong_Mobile_Forum_May_2013.pdf`.

[12] *"Approximate Probabilistic Model Checking"*. T. Hérault, R. Lassaigne, F. Magniette & S. Peyronnet. VMCAI, LNCS vol. 2937, pp. 73–84, Springer, 2004.

[13] *"Verification and Planning for Stochastic Processes with Asynchronous Events"*. Håkan L. S. Younes. Carnegie Mellon, 2005.

[14] *"Statistical Model Checking: Present and Future"*. S. Bensalem, B. Delahaye & A. Legay. Proc. of *1st Conference on Runtime Verification (RV), Malta, 2010*. Springer-Verlag.

[15] *"Automated Conflict-free Distributed Implementation of Component-Based Models"*. B. Bonakdarpour, M. Bozga, M. Jaber, J. Quilbeuf and J. Sifakis In Proceedings of IEEE Fifth International Symposium on Industrial Embedded Systems - SIES 2010, University of Trento, Italy, July 7-9, 2010, pp. 108–117.

[16] *"From High-Level Component-Based Models to Distributed Implementations"*. B. Bonakdarpour, M. Bozga, M. Jaber, J.Quilbeuf & J. Sifakis. In Proceedings of the 10th International conference on Embedded software, EMSOFT 2010, Scottsdale, Arizona, USA, October 24-29, 2010, pp. 209–218.

[17] *"3D-ICE: Fast Compact Transient Thermal Modeling for 3D-ICs with Inter-Tier Liquid Cooling"*. Sridhar, Mahankali ; Raj, Arvind ; Vincenzi, Alessandro ; Ruggiero, Martino ; Brunschwiler, Thomas ; Atienza Alonso, David. Proceedings of the 2010 International Conference on Computer-Aided Design (ICCAD 2010), vol. 1, num. 1, 2010, p. 1-8. New York: ACM and IEEE Press, 2010. San Jose, California, USA. November 7th-11th, 2010.

[18] *"Compact Transient Thermal Model for 3D ICs with Liquid Cooling via Enhanced Heat Transfer Cavity Geometries"*. Sridhar, Mahankali ; Raj, Arvind ; Vincenzi, Alessandro ; Ruggiero, Martino ; Brunschwiler, Thomas ; Atienza Alonso, David. Proceedings of the 16th International Workshop on Thermal Investigations of ICs and Systems (THERMINIC'10), vol. 1, num. 1, 2010, p. pp. 5 – 11. New York: IEEE Press, 2010. Barcelona, Spain. October 6th-8th, 2010.

[19] *"Fuzzy Control for Enforcing Energy Efficiency in High-Performance 3D Systems"*. Sabry Aly, Mohamed Mostafa, Ayse Kivilcim Coskun, David Atienza Alonso. Proceedings of the 2010 International Conference on Computer-Aided Design (ICCAD 2010), vol. 1, num. 1, 2010, p. 10-16. New York: IEEE Press, 2010. San Jose, California, USA. November 7th-11th, 2010.

[20] *"Worst-Case Temperature Analysis for Real-Time Systems"*. Devendra Rai, Hoeseok Yang, Iuliana Bacivarov, Jian-Jia Chen and Lothar Thiele. Proc. of Design, Automation and Test in Europe (DATE '11), ACM and IEEE Press. Grenoble, France. March 14th-18th, 2011.

[21] *"Rigorous Component-Based System Design Using the BIP Framework"*. Ananda Basu, Saddek Bensalem, Marius Bozga, Jacques Combaz, Mohamad Jaber, Thanh-Hung Nguyen, Joseph Sifakis. IEEE Software, vol. 28, no. 3, May/June 2011. DOI: `http://dx.doi.org/10.1109/MS.2011.27`.

[22] *"Cool shapers: Shaping Real-Time Tasks for Improved Thermal Guarantees"*, Pratyush Kumar and Lothar Thiele, Design Automation Conference 2011, June 5th-10th, 2011. San Diego, California, USA. `ftp://blackbox.ethz.ch/pub/people/kumarpr/DAC11.pdf`

[23] *"End-to-End Delay Minimization in Thermally Constrained Distributed Systems"*, Pratyush Kumar and Lothar Thiele, Euromicro Conference on Real-Time Systems (ECRTS 2011). July 6th–8th, 2011. Porto, Portugal.

[24] *"System-Level Power and Timing Variability Characterization To Compute Thermal Guarantees"*. Pratyush Kumar and Lothar Thiele. Hardware/Software Codesign and System Synthesis, (CODES/ISSS 2011), pp. 179–188. ACM. Embedded Systems Week, 9th-14th October, 2011. Taipei, Taiwan.

[25] *"Component Assemblies in the context of Manycore"*. Ananda Basu, Saddek Bensalem, Marius Bozga, Paraskevas Burgos, and Joseph Sifakis. Presentation at Software Technologies Concertation on Formal Methods for Components and Objects, FMCO 2011, Torino, Italy. October 2011.

[26] *"Knowledge-Based Distributed Conflict Resolution for Multiparty Interactions and Priorities"*. Saddek Bensalem, Marius Bozga, Jean Quilbeuf, and Joseph Sifakis. In Holger Giese and Grigore Rosu, editors, *FMOODS/FORTE - Formal Techniques for Distributed Systems - Joint 14th IFIP WG 6.1 International Conference, FMOODS 2012 and 32nd IFIP WG 6.1 International Conference, FORTE 2012, Stockholm, Sweden, June 13-16, 2012. Proceedings*, volume 7273 of *Lecture Notes in Computer Science*, pages 118–134. Springer, June 2012.

[27] *"A framework for automated distributed implementation of component-based models"*. Borzoo Bonakdarpour, Marius Bozga, Mohamad Jaber, Jean Quilbeuf, and Joseph Sifakis. *Distributed Computing*, 25(5):383–409, October 2012.

[28] *"Model-based implementation of distributed systems with priorities"*. Borzoo Bonakdarpour, Marius Bozga, and Jean Quilbeuf. *Design Automation for Embedded Systems*, pages 1–26, July 2012.

[29] *"Statistical Model Checking QoS Properties of Systems with SBIP"*. Saddek Bensalem, Marius Bozga, Benoît Delahaye, Cyrille Jégourel, Axel Legay, and Ayoub Nouri. In Tiziana Margaria and Bernhard Steffen, editors, *Leveraging Applications of Formal Methods, Verification and Validation. Technologies for Mastering Change - 5th International Symposium, ISoLA 2012, Proceedings, Part I*, volume 7609 of *Lecture Notes in Computer Science*, pages 327–341. Springer, 2012.

[30] *"Optimized Distributed Implementation of Multiparty Interactions with Observation"*. Saddek Bensalem, Marius Bozga, Jean Quibeuf, and Joseph Sifakis. In *2nd International Workshop on Programming based on Actors, Agents, and Decentralized Control Workshop held at ACM SPLASH 2012, Pre-Proceedings*, pages 90–99, October 2012.

[31] *"Knowledge Based Transactional Behavior"*. Saddek Bensalem, Marius Bozga, Jean Quilbeuf, and Doron Peled. In *Eight Haifa Verification Conference, HVC2012*, November 2012. Post proceedings to appear as LNCS volume,

[32] *"Thermal Balancing of Liquid-Cooled 3D-MPSoCs Using Channel Modulation"*. Mohamed M. Sabry, Arvind Sridhar, and David Atienza Alonso. In *EDAA 2012*, 2012.

[33] *"P2012: Building an ecosystem for a scalable, modular and high-efficiency embedded computing accelerator"*. Luca Benini, Eric Flamand, Didier Fuin, and Diego Melpignano. In Rosenstiel and Thiele [34], pages 983–987.

[34] *2012 Design, Automation & Test in Europe Conference & Exhibition, DATE 2012, Dresden, Germany, March 12-16, 2012*. Wolfgang Rosenstiel and Lothar Thiele, editors. IEEE, March 2012.

[35] *The 49th Annual Design Automation Conference 2012, DAC '12, San Francisco, CA, USA, June 3-7, 2012*. Patrick Groeneveld, Donatella Sciuto, and Soha Hassoun, editors. ACM, June 2012.

[36] *"Platform 2012, a many-core computing accelerator for embedded SoCs: performance evaluation of visual analytics applications"*. Diego Melpignano, Luca Benini, Eric Flamand, Bruno Jego, Thierry Lepley, Germain Haugou, Fabien Clermidy, and Denis Dutoit. In Groeneveld et al. [35], pages 1137–1142.

[37] *"Timing Analysis on a Processor with Temperature-Controlled Speed Scaling"*. Pratyush Kumar and Lothar Thiele. In *In Proc. of the 18th IEEE Real-Time and Embedded Technology and Applications Symposium, RTAS 2012*, Beijing, China, April 2012.

[38] *"Worst-Case Temperature Guarantees for Real-Time Applications on Multi-Core Systems"*. Lars Schor, Iuliana Bacivarov, Hoeseok Yang, and Lothar Thiele. In *In Proc. of the 18th IEEE Real-Time and Embedded Technology and Applications Symposium, RTAS 2012*, Beijing, China, April 2012. IEEE Computer.

[39] *"Fast Worst-Case Peak Temperature Evaluation for Real-Time Applications on Multi-Core Systems"*. Lars Schor, Iuliana Bacivarov, Hoeseok Yang, and Lothar Thiele. In *Proc. IEEE Latin American Test Workshop (LATW)*, Quito, Ecuador, April 2012. IEEE.

[40] *"Power Agnostic Technique for Efficient Temperature Estimation of Multicore Embedded Systems"*. Devendra Rai, Hoeseok Yang, Iuliana Bacivarov, and Lothar Thiele. In *CASES*, pages 61–70, 2012.

[41] *"Predictability for Timing and Temperature in Multiprocessor System-on-Chip Platforms"*. Lothar Thiele, Lars Schor, Iuliana Bacivarov, and Hoeseok Yang. *ACM Transactions in Embedded Computing Systems (TECS)*, March 2013.

Glossary

3D-ICE 3D Interlayer Cooling Emulator. 23

API Application Programming Interface. 13, 14, 29

BIP Behaviour Interaction Priority. 5, 9–13, 17

DOL3D Distributed Operation Layer for 3D Integrated Circuits. 16, 17

JEDEC The *Joint Electron Devices Engineering Council* is an industry association that develops open standards for the microelectronics industry – See `http://www.jedec.org` for details. 7, 29

LPDDR Low Power DDR. 3, 7

Multicore Association The *Multicore Association®* is an industry association that includes leading companies implementing products that embrace multicore technology. The Multicore Association's roadmap consists of an extensive set of API, like MCAPI that support multicore communications; MRAPI for resource management and MTAPI for task management. These API will provide a foundation for a multitude of services. See `http://www.multicore-association.org` for more details. 13, 14, 29

MCAPI stands for *Multicore Communications API* – See [1] for details. 13–17, 29

MPARM is a multi-processor platform developed by Microelectronics Research Group from Università di Bologna. 17

MPI stands for *Message Passing Interface* – See `http://www.mcs.anl.gov/research/projects/mpi` for details. 14

MRAPI stands for *Multicore Resource Management API* – See [2] for details. 14, 29

MTAPI stands for *Multicore Task Management API* – See [3] for details. 14, 29

NoC Network-on-Chip. 5

P2012 stands for *Platform 2012*. A manycore platform architecture template developed by STmicroelectronics [10, 33, 36]. 15–17, 29

PE Processing Element. 15, 16

QoS Quality of Service. 11

SMC Statistical Model Checking. 4, 5, 8, 9

SoC System on Chip. 8

STHORM stands for *ST Heterogeneous lOw poweR Manycore*. A manycore platform specialised for video analytics developed by STmicroelectronics, based on P2012. 13, 15, 16

Wide-IO is an inter-die vertical interconnect technology specified by JEDEC. 3, 4, 7