Hierarchical Thermal Management Policy for High-Performance 3D Systems With Liquid Cooling

Francesco Zanini, Mohamed M. Sabry, David Atienza, *Member, IEEE*, and Giovanni De Micheli, *Fellow, IEEE*

(Invited Paper)

***Abstract*—Three-dimensional (3D) integrated circuits and systems are expected to be present in electronic products in the short term. We consider specifically 3D multi-processor systems-on-chips (MPSoCs), realized by stacking silicon CMOS chips and interconnecting them by means of through-silicon vias (TSVs). Because of the high power density of devices and interconnect in the 3D stack, thermal issues pose critical challenges, such as hot-spot avoidance and thermal gradient reduction. Thermal management is achieved by a combination of active control of on-chip switching rates as well as active interlayer cooling with pressurized fluids.**

In this paper, we propose a novel online thermal management policy for high-performance 3D systems with liquid cooling. Our proposed controller uses a hierarchical approach with a global controller regulating the active cooling and local controllers (on each layer) performing dynamic voltage and frequency scaling (DVFS) and interacting with the global controller. Then, the on-line control is achieved by policies that are computed off-line by solving an optimization problem that considers the thermal profile of 3D-MPSoCs, its evolution over time and current time-varying workload requirements. The proposed hierarchical scheme is scalable to complex (and heterogeneous) 3D chip stacks.

We perform experiments on a 3D-MPSoC case study with different interlayer cooling structures, using benchmarks ranging from web-accessing to playing multimedia. Results show significant advantages in terms of energy savings that reaches values up to 50% versus state-of-the-art thermal control techniques for liquid cooling, and thermal balance with differences of less than 10 °C per layer.

***Index Terms*—Hardware/software co-design, multilayer, multiprocessor system-on-chip (SoC), power modeling and estimation, thermal.**

I. Introduction

THREE-DIMENSIONAL (3D) integrated circuits and systems are becoming mainstream for a variety of reasons. In the high-performance processor market, chip stacking enables short connections and high bandwidth between processors and memories [5]. In the mobile market, 3D integration enables designers to package together chips that come from different processes and/or have different operational parameters, such as analog and low-voltage digital components [14], [20].

We consider in this work 3D *multi-processor systems-on-chips* (MPSoCs), realized by stacking silicon CMOS chips and interconnecting them by means of *through-silicon vias* (TSVs) [31]. Because of the high volumetric density of devices and interconnect, thermal issues are a critical challenge. Indeed heat generation grows with the number of stacked layers and heat extraction is harder because of the three-dimensional nature of the system. Thus, challenges in 3D design include mitigating temperatures, reducing hot-spots as well as thermal gradients, which would otherwise reduce the *mean time to failure* (MTTF) of the 3D stack [5], [46] and, as a limiting case, burn it.

In 3D stacks, cooling cannot be handled and managed by conventional air cooling methods [5], [29] over the stack surface. Interlayer liquid cooling is a potential solution to address thermal problems, due to the higher heat removal capability of liquids in comparison to air [8] and to the possibility to extract heat at various layers of the stack. There are several ways to support liquid cooling, e.g., by adding/inserting to the stack a plate with built-in microchannels and/or by etching a porous-media structure between the tiers of the 3D stack [8], [9]. Experiments have shown that when a coolant fluid is pumped through the microchannels, up to 3.9 kW/cm^3 [9] of heat can be extracted. Fig. 1 shows the cross section and prototype of a manufactured 3D test-chip with thermal emulators and microchannel-based interlayer cooling structure [9].

Porous-media structures can be designed with different forms according to the TSVs spacing requirements and the desired fluidic path [39], [40]. Fig. 2 shows a planar view of two dif-

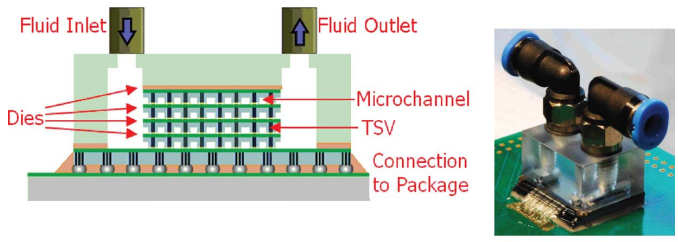

Fig. 1. Manufactured prototype and cross section of a test stack with interlayer liquid cooling [9]. (a) Cross section. (b) Prototype.

Manuscript received December 31, 2010; accepted March 21, 2011. Date of publication June 30, 2011; date of current version August 19, 2011. This work was supported in part by ERC Senior Grant 246810, by the PRO3D EU FP7-ICT-248776 project, and by the Nano-Tera.ch RTD Project CMOSAIC (ref. 123618), which is financed by the Swiss Confederation and scientifically evaluated by SNSF. This paper was recommended by Guest Editor V. Narayanan.

F. Zanini and G. De Micheli are with the Laboratory of Integrated Systems (LSI), Ecole Polytechnique Fédérale de Lausanne (EPFL), 1015-Lausanne, Switzerland (e-mail: Francesco.Zanini@epfl.ch; Giovanni.DeMicheli@epfl.ch).

M. M. Sabry and D. Atienza are with the Embedded Systems Laboratory (ESL), Ecole Polytechnique Fédérale de Lausanne (EPFL), 1015-Lausanne, Switzerland (e-mail: Mohamed.Sabry@epfl.ch; David.Atienza@epfl.ch).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JETCAS.2011.2158272

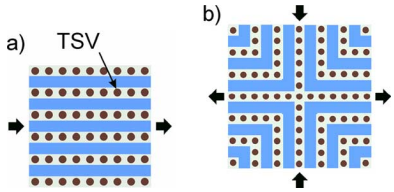

Fig. 2. Top view of: (a) 2-port and (b) 4-port microchannel fluid delivery architecture compatible with area-array interconnects.

ferent structures. Although these structures use microchannels to guide the fluid, one of them uses straight channels with two ports [Fig. 2(a)], while the other exploits bent channels and four ports [Fig. 2(b)]. In the following we will refer to these structures as "straight" and "bent" channels. To the best of our knowledge, this is the first time that bent channels are considered in 3D-MPSoC designs, and this paper reports on results with both straight (traditional) and bent microchannels. The main advantage of the bent structure is having channels with different flow rates due to their different lengths. However because of its manufacturing complexity straight channels are preferred in small footprint dies.

Overall, thermal management of a 3D stack is achieved by a combination of active control of on-chip switching rates (the heat source) as well as active interlayer cooling with pressurized fluids (the heat sink). It is important to remember that the cooling system requires one (or more) pumps to circulate the fluid, as well as a heat exchanger to cool the fluid. The latter may be passive (e.g., fin structure) or active (e.g., fan). At any rate, a relevant part of the system energy spent for cooling is due to the pump [32] and a minor part by the exchanger. Therefore, we consider this energy in the overall energy balance of the system. The possibility to adjust the flow rate dynamically, thus the cooling power, adds an important dimension and novelty in addressing thermal issues.

The other important knob in controlling the 3D system is the active monitoring and control of switching and voltage swings, as effected by *dynamic voltage and frequency scaling* (DVFS) [28]. Keeping core frequencies and voltages of each core to a minimum to satisfy workload requirements is crucial for minimizing the heat generated by computing, storing and transferring information. Combining DVFS with interlayer variable-flow cooling is a complex problem, especially because of the 3D distribution of the cores on various layers of the 3D stack [5], [16], [32].

For the aforementioned reasons, we introduce here a new hierarchical approach to thermal management for 3D stacks, using both DVFS and variable-flow liquid cooling. Our approach uses: 1) a global controller (for the 3D stack) that regulates the active cooling and 2) local controllers, one for each layer, performing DVFS and interacting with the global controller. On-line control is achieved by policies that are computed off-line by solving an optimization problem that considers the thermal profile of the system, its evolution over time and current time-varying workload requirements. The proposed hierarchical scheme is scalable to complex (and heterogeneous) chip stacks.

The contributions of these paper can be summarized as follows. We introduce a novel hierarchical thermal management system specifically suited for 3D stacks. Thus, we are able to exploit new technologies for cooling, using new structures for microchannels (straight and bent) and we characterize their thermal properties. Moreover, we use a variable-flow liquid cooling scheme, where we control (and trade off) the pump power and related fluid pressure. We regulate frequencies and voltages on each layer through local controllers, that are leaves of the hierarchical thermal management scheme.

We perform extensive experiments on a 3D-MPSoC case study with different interlayer cooling structures using benchmarks ranging from web-accessing to playing multimedia. Results show that using bent channels reduces thermal gradients by up to 58% with respect to using static worst-case liquid flow rates. Also, these results show that our policy achieves better thermal balance by reducing the thermal gradients below 10 °C per layer. Moreover, our policy results in energy savings up to 50% with respect to state-of-the-art thermal management techniques for 3D stacks with liquid cooling [15], [16], [32]. Our proposed policy reaches up to 38% additional pump energy savings when using bent channels, which have so far not been proposed for 3D-MPSoCs.

The remainder of this paper is organized as follows. We revise the related work on thermal management techniques in Section II. Section III elaborates on the mathematical formulation used to model 3D-MPSoCs. In Section IV, the architecture and problem formulation of the proposed hierarchical thermal control policy is shown. The experimental setup is described in Section V. Next, we show the simulation results of our proposed policy, as well as state-of-the-art thermal management techniques in Section VI and, finally, Section VII summarizes the main conclusions of this work.

II. Related Work

A. Power and Thermal Management for Chips

Early methods for power management were based on monitoring the idle time of processors [19], and control frequency and voltage of processors. Power management system are modeled as stochastic optimum control, and policies were determined as solutions to these problems, using discrete-time [2] and continuous-time [30] Markov decision processes respectively. Simunic *et al.* [33] show a methodology for managing power consumption in networks-on-chips (NoCs). More recently researchers focus on combined power and thermal management by presenting a set of scheduling mechanisms for MPSoCs that perform temperature management at the system-level [21], using thread migration techniques to achieve reduction in localized hot-spots [17], or using a temperature-aware dynamic scheduling algorithm with negligible performance overhead [11]. Lu *et al.* [26] present a software architecture that allows system designers to investigate power management algorithms in a systematic fashion. The aforementioned methods do not exploit history information and take reactive control actions based on the current thermal profile and frequency setting of the MPSoC. However, recent works exploit history information to improve thermal management policies. Coskun *et al.* [12] exploit a temperature-forecast technique based on an autoregressive moving average model. Another

work proposes a novel technique that adapts the thermal management policy to the current workload characteristics [13], where the adaptation is done online exploiting information related to the workload history. Two recent approaches [10], [45] describe two methodologies to achieve thermal prediction by combining the information of thermal model, thermal sensors and power consumption statistical properties.

All these approaches rely on open-loop search or optimization where it is assumed that power can be estimated accurately. More advanced solutions apply the concepts of *model-predictive control* (MPC) to turn the control from open loop to closed loop [1]. A chip-level power control algorithm based on optimal control theory is proposed [42], where the power consumption of the MPSoC is controlled to maintain the temperature of each core below a specified threshold. A similar concept is tailored for multi-modal video sensor nodes [27]. A recent work [43] proposes the idea of using MPC to solve the frequency assignment problem of a planar MPSoC.

However, most previous policies do not completely avoid hotspots, but they simply reduce their frequency, because the interaction among the prediction method, the thermal behavior of the MPSoC and the frequency assignment of the MPSoC have not been addressed as a joint optimization problem.

B. 3D MPSoCs Thermal Management

In 3D MPSoCs, prior work on thermal management mainly addresses design stage optimization, such as thermally-aware floorplanning [20] and integrating thermal via planning in the 3D floorplanning process [25]. Recent work considers dynamic thermal management for 3D MPSoCs. Zhu *et al.* evaluate several policies for task migration and DVFS [47]. They explore thermal profiles of adjacent processing elements being on the same vertical column (interlayer adjacent) or within the same layer (intralayer). Based on their analysis, they implement a combined DVFS and a task migration policy, named *THERMOS*, but they have not considered interlayer liquid cooling effects. Zhou *et al.* [46] integrate a thermally aware task scheduler with DVFS on a 2-tier system with 8 cores. A recent paper proposes a temperature-aware scheduling method specifically designed for air-cooled 3D systems [14]. This method takes into account the thermal heterogeneity among the different layers of the system, but there is no study on the effect of interlayer cooling as an active thermal management parameter. The resulting temperatures obtained in these papers are significantly high (85 °C–110°C). These results imply that 3D MPSoCs are prone to high temperatures, and with increasing power densities conventional thermal management techniques and air-based cooling are incapable of controlling the temperature while preserving system performance.

The use of convection in microchannels to cool down high power density chips has been an active area of research since the initial work by Tuckerman and Pease [41]. The heat removal capability of interlayer heat-transfer with pin-fin in-line structures for 3D chips is investigated in [8]. Also, several works [4], [24] have explored the feasibility of having liquid cooling as cooling method for 3D MPSoCs. Then, prior liquid cooling work [15] evaluates existing thermal management policies on a 3D system with a fixed-flow rate setting, and also investigates the benefits of variable flow using a policy to increment or decrement the flow rate based on temperature measurements, but without considering pump energy consumption as we do in this work.

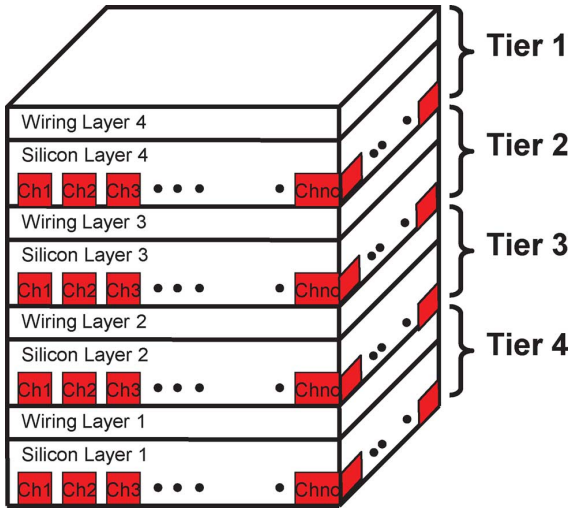

Fig. 3. Structure of the 4-tier 3D-MPSoC with interlayer liquid cooling we target in this paper.

Accurate thermal modeling of liquid cooling is critical in the design and evaluation of systems and policies. HotSpot [36] is a thermal model tool that calculates transient temperature response given the physical and power consumption characteristics of the chip. Its latest releases include 3D modeling capabilities and basic liquid-cooled systems as well [15]. 3D-ICE [39] is a new thermal modeling tool specifically designed for 3D stacks, and includes detailed inter-layer liquid cooling modeling capabilities. However, the tool is limited to modeling single porous-media structures. Finally, the same thermal modeling concept used in 3D-ICE is extended to model more complex structures, such as pin fin-based porous media [40], but there is no exploration of bent channels for MPSoC design in the literature, as we propose in this work.

Thermal management methods for 3D MPSoCs using a variable-flow liquid cooling have been recently proposed [16], [32]. These policies use experimentally sets of rules to control the temperature profile of the 3D MPSoC while ensuring performance requirements to be satisfied. These approaches use a centralized control concept, which is inappropriate if the controlled parameters increase [18], as in the 3D MPSoC designs we target in this work with bent channels.

III. 3D-MPSoC Model

In this paper, we focus on modeling and energy-efficient thermal management of 3D-MPSoCs with interlayer liquid cooling. A typical structure of 3D-MPSoCs consists of two or more more silicon tiers, with the processing and storage elements of the system. Interlayer liquid cooling is realized by etching *microchannels* in silicon and creating porous structures of different form and shapes. Etching must take into account the TSVs allocation and spacing requirements. Fig. 3 shows an example of a 4-tier 3D-MPSoC with multiple inlets and outlets in different parts of the tiers, as we target in this paper. In this figure, the wiring layer is explicitly shown, as thermal

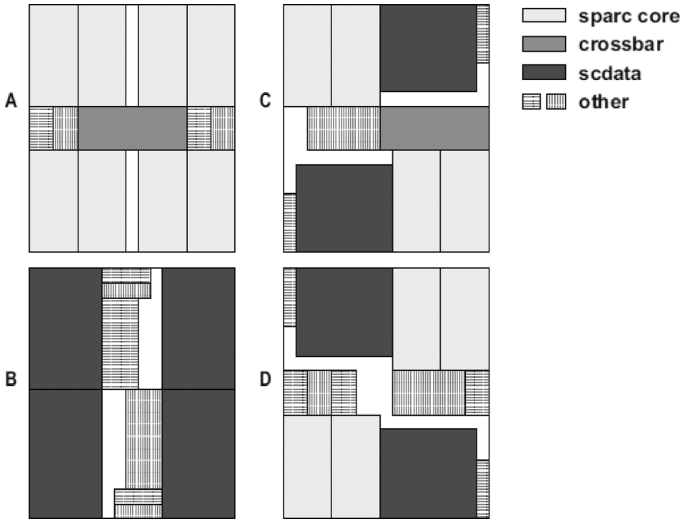

Fig. 4. Floorplan of the used silicon tiers in our target 3D-MPSoC.

properties of interconnect (usually copper) are different from silicon. In this example, we show four different floorplans (A, B, C, D) (see Fig. 4) in the silicon tiers with various processing cores (i.e., UltraSPARC Niagara T1 [22]), with independent clock frequency and voltage supplies, interconnects (crossbar) and memories (scdata).

In our approach, we experiment with both straight and bent microchannels, having 2 and 4 ports, respectively (cf. Fig. 2). In both cases the microchannel cross section is constant. The straight microchannels have all equal length (i.e., they go from side to side of the chip). The length varies in bent channel structures. All microchannels (in all layers) are connected to a pumping network that injects the fluid with the same input pressure as well as pressure difference between the inlet and the outlet. Nevertheless more complex microfluidic circuitry can be used. For example, the assumption of a single pump and equal input pressure can be removed for the sake of generality. However, a more complex microfluidic circuitry complicates the manufacturing process and, as we shown in this work, bent channels already enable energy-efficient hierarchical thermal balancing approaches for 3D-MPSoC designs. Therefore, to perform a system-level modeling of 3D-MPSoCs with interlayer liquid cooling, the following stages are involved:

1) modeling the interlayer structure that includes the interconnecting TSVs and the microchannels;
2) modeling the heat propagation of the building blocks of 3D-MPSoC;
3) power and frequency modeling of the processing elements in 3D-MPSoCs;
4) modeling the workload assigned to processing elements.

In the following subsections we elaborate on each of these stages.

A. Interlayer Cooling Layer Modeling

Previous works on thermal modeling and management of 3D-MPSoCs with interlayer cooling use straight microchannels as the cooling layer structure [8], [16], [32]. Moreover, the liquid is assumed to be injected from a single port and flows through the microchannels to a single outlet port. In this work, however, we extend this structure to bent channels. Thus, we extend the previous compact thermal modeling concept [39] to account for two major factors. First, the fluid flow is no longer constant among the channels of the same layer, but it is related to the channel length [8]. Thus, different lengths of the channels lead to different fluid velocities [9]. The channels with the shortest length have the highest fluid velocity, while the longest channels have the lowest velocity. Second, we assume that the fluid flow is not a single dimensional flow. Thus, we apply the fluid flow representation concept of 3D-ICE [39] to model the new multi-directional fluid flows. In this case the fluid enters from a direction that lies in one Cartesian axis (e.g., south) and can leave from another direction that lies on another axis (e.g., east). Indeed, our results show that using multi-port bent channels is more beneficial than using straight channels, if the straight channel length is longer than the thermal developing length of the fluid [9].

TABLE I
PARAMETERS DEFINITION USED TO RELATE THE FLOW RATE TO THE CHANNEL LENGTH

Parameter	Definition
w_{ch}	Channel width ($50\mu m$)
h_{ch}	Channel height ($100\mu m$)
ε	Cavity porosity (0.5)
κ	Cavity permeability ($7.17E-11m^2$)
μ	Dynamic viscosity ($1E-3Pascal \cdot sec$)
ΔP	Pressure difference between the inlet and outlet ports (1 bar)

In addition, in the target 3D-MPSoC stacks the microchannels have different lengths, which implies that the pumped flow rate is not distributed homogeneously between the microchannels. The relation between the flow rate Fl and the channel length L is as follows:

$$Fl = w_{\text{ch}} \cdot h_{\text{ch}} \cdot \nu_{\text{bulk}} \tag{1}$$

$$\nu_{\text{bulk}} = \frac{\nu_{\text{darcy}}}{\varepsilon} \tag{2}$$

$$\nu_{\text{darcy}} = \frac{\kappa}{\mu} \cdot \nabla P \tag{3}$$

$$\nabla P = \frac{\Delta P}{L}. \tag{4}$$

where the parameters in these equations are shown in Table I. Hence, the flow rate and channel length are inversely proportional, i.e, the shorter the channel length is, the higher the flow rate. We validate the flow velocity obtained for each channel by comparing the analytical model in (1)–(4) with the experimental values shown in [9]. As shown in Fig. 5, the proposed analytical model provides us with an acceptable method to calculate the flow rate for different channel lengths.

Since we use varying flow rate as a control variable for energy-efficient thermal management, it is crucial to study the thermal capability of interlayer liquid cooling with respect to different pumping power values. Thus, each pumping power value is translated to a specific flow rate in our system. First, we use Bernoulli's equation to describe the pump power P_{pump} as follows:

$$P_{\text{pump}} = \frac{\Delta P \cdot Fl}{\zeta} \tag{5}$$

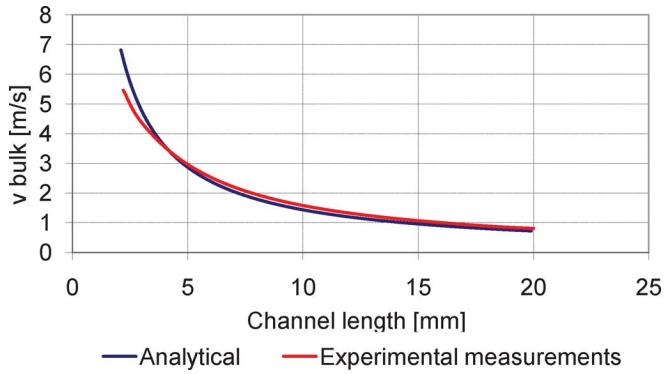

Fig. 5. Comparison of the fluid flow velocity in different channel lengths between the analytical method [see (1)–(4)] and the experimental results shown in [9].

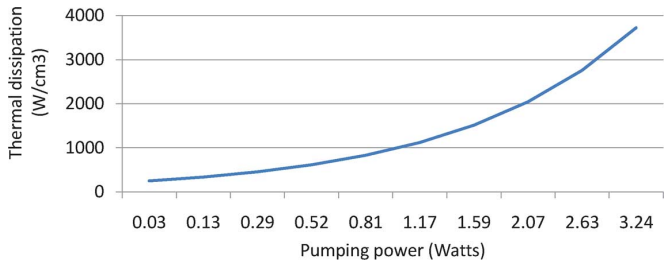

Fig. 6. Rate of change of thermal capability of interlayer liquid cooling TC with respect to pumping power P_{pump}.

where ΔP is the pressure difference required, Fl is the fluid flow rate, and ζ is the pumping power efficiency. We use $\zeta = 0.7$, as it is a normal pump efficiency value [23], [34]. Since there is a linear relation between the pressure difference and the flow rate injected in the stack ((1)–(4)), we can say that $P_{\text{pump}} \propto Fl^2$.

Next, we define the thermal capability of interlayer liquid cooling as the maximum heat flux absorbed by the fluid to keep the maximum temperature within the stack below 85 °C. To estimate this thermal capability, we use 3D-ICE [39] to record the maximum temperature of the stack at different thermal dissipation values, and with different flow rates. We limit the maximum flow rate injected to be the one at $\Delta P = 1$ bar, since it is the maximum safe pressure requirement within the stack [8].

Therefore, Fig. 6 shows the amount of minimum pumping power applied to keep the maximum temperature of the stack below 85 °C, at different thermal dissipation rates.

B. 3D Heat Propagation Model

Our 3D thermal model is based on finite-element analysis, as used by typical system-level thermal analysis tools [39]. Heat propagation is modeled by thermal resistances and capacitances. By discretizing the differential equations, we can model the heat propagation process as follows:

$$\mathbf{t}_{\tau+1} = \mathbf{A}\mathbf{t}_\tau + \mathbf{B}\mathbf{p}_\tau \tag{6}$$

$$\tilde{\mathbf{t}}_\tau = \mathbf{C}\mathbf{t}_\tau. \tag{7}$$

We assume to have p layers (also called tiers) and that they are divided into n cells in total. Matrices $\mathbf{A} \in \Re^{n\times n}$ and $\mathbf{B} \in \Re^{n\times(p+1)}$ describe the heat propagation properties of the 3D structure and they depend on the integration method used. At time τ, the temperature of the next time step of cell i, i.e., $(\mathbf{t}_{\tau+1})_i$ can be computed by (6). The vector $\mathbf{p} \in \Re^{p+1}$ is the input vector. The first p entries are the normalized power consumption for each of the p layers. The last entry is the normalized power consumed by the liquid cooling system, i.e., by the pump and any other cooling active structure.

Matrix $\mathbf{C} \in \Re^{s\times n}$ relates the temperature value of each cell to the temperature measurement of a particular sensor. In this model we assume that the temperature can be measured only in a limited number of locations. We assume that s is the total number of thermal cells in the model where the temperature can be measured. Equation (7) describes the choice of temperature sensors inside the 3D-MPSoC. It is important to mention that we have validated this model with the thermal analysis tool, 3D-ICE [39], and we have found that the maximum offset error between our model and 3D-ICE is below 5%.

On each layer, clock frequencies (for the cores) can take only specific discrete values in the range from a minimum to a maximum frequency values ($f_{\min}$, $f_{\max}$). The relation between the frequency and the power consumption is assumed to be quadratic [36].

C. Workload Model

The workload is generated from higher level software layers (e.g., the operating system). For our purposes, it is defined as the minimum value of the clock frequency that the functional unit should have to execute the required tasks within the specified system constraints.

The workload requirement at time τ is defined as a vector $\mathbf{w}_\tau \in \Re^p$, where $(\mathbf{w}_\tau)_i$ is the workload requirement value for input i at time τ. In other words, it is the frequency $(\mathbf{w}_\tau)_i$ that processing units associated with input i from time τ to time $\tau + 1$ should have in order to satisfy the desired performance requirement coming from the scheduler.

Our model is assumed to be continuous and ranging from a minimum to a maximum frequency values ($f_{\min}$, $f_{\max}$) at which the cores can process data, namely

$$f_{\min} \preceq \mathbf{f}_\tau \preceq \mathbf{f}_{\max} \ \forall \tau. \tag{8}$$

When $(\mathbf{w}_\tau)_i > (\mathbf{f}_\tau)_i$, the workload cannot be processed and so it needs to be stored and rescheduled in the following clock cycles. This leads to an increase in both the task delay before execution and the undone workload $\mathbf{u}$. At time τ, this second performance parameter is given by the vector $\mathbf{u}_\tau \in \Re^p$.

$$\mathbf{u}_\tau = \mathbf{w}_\tau - \mathbf{f}_\tau. \tag{9}$$

It expresses the difference at time τ between the requested and the executed workload by the MPSoC.

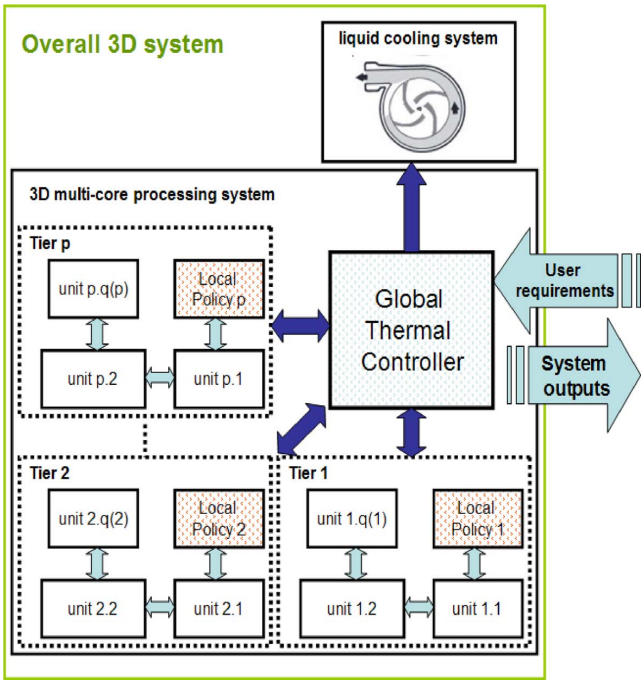

Fig. 7. Structure of the proposed hierarchical thermal management system.

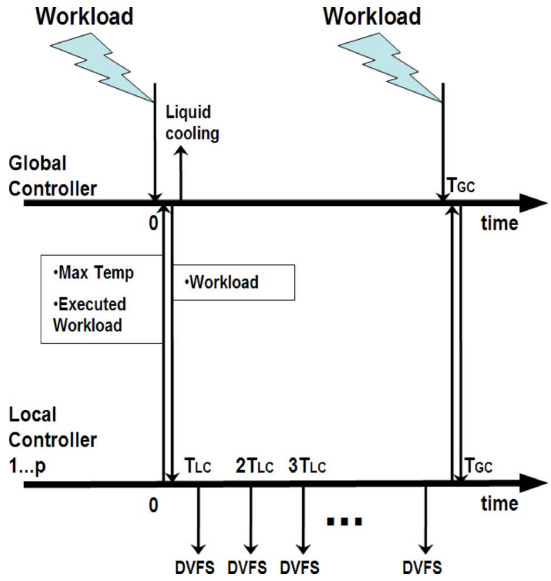

Fig. 8. Communication protocol between the global and the local controllers of the proposed method.

IV. Hierarchical Thermal Control

A. Hierarchical Structure

The structure of the proposed hierarchical thermal management system is shown in Fig. 7: the 3D-MPSoC architecture is partitioned into p tiers (or layers) where, without loss of generality, each tier is a subsystem of the 3D-MPSoC. In our exploration, we define a tier as a complete layer. Moreover, any tier consists of several units. These units could be cores, memory storage units, or other computational units (e.g., ASIC or custom hardware blocks). Then, the units inside each tier, say tier i, are partitioned into $q(i)$ frequency islands, and a local thermal controller manages the $q(i)$ islands, i.e., sets the frequencies and voltages to all (controllable) components inside the tier. Objectives of local controllers include preventing hot-spots and minimizing undone workload. Specific requirements (e.g., workload) come from a centralized unit (i.e., the *global thermal controller* in Fig. 7), which is responsible for the holistic coordination of the p local thermal controllers, and which regulates the heat extraction of the cooling system by setting the pressure of the coolant liquid (by controlling the cooling pump and/or the controlling valve).

This hierarchical structure is crucial for scalability and feasibility of large MPSoCs [18]. Indeed by using this hierarchical approach, we can significantly simplify the function and overhead of the global controller by using local thermal controllers. Moreover, this structure allows the global and local controllers to be executed with different rates, e.g., the optimization of the global controller can be executed at least one order of magnitude less frequently as compared to the local regulators. The global controller manages the pumping flow rate, which is much slower process than DVFS.

B. Run-Time Interaction Between Global and Local Controllers

The communication protocol between the local controllers and the global one is shown in Fig. 8. Initially, the global controller receives a workload requirement from the scheduler as well as a data vector containing their workload fulfillment status in each specific tier from all the p local controllers. This data vector contains two pieces of information: 1) the maximum temperature measured on line in the corresponding tier and 2) the already executed workload. Indeed this last information provides the global controller with an overview about how well the local controllers are performing in trying to fulfill overall requirements.

Moreover, as the workload fulfillment data from all the local controllers are collected and processed, the global unit splits the overall workload into p components. Hence, for each local controller, the global unit sets the amount of workload it has to execute. It is important to notice that the controller does not perform detailed task assignment, but just sets individual targets for each tier to satisfy the overall workload. The pressure of the coolant liquid is set during this process by the global controller, which performs this operation periodically, with a period of T_{GC}. Once these tasks are performed, the global controller stays still for the rest of the period T_{GC}. Concurrently each local controller sets periodically the DVFS value of all related islands, but with another period T_{LC}, such that $T_{\mathrm{GC}} = n \cdot T_{\mathrm{LC}}$, $n \in Z^{+}$. The local controllers manage independently the corresponding subsystems and they can communicate with the global thermal management unit only once in the period T_{GC}.

C. Design and Implementation

The design and implementation of the proposed management scheme consists of two phases, i.e., *design phase* and *run-time phase* , which are shown in Fig. 9. The *design phase* is performed off-line to compute and generate the optimized control

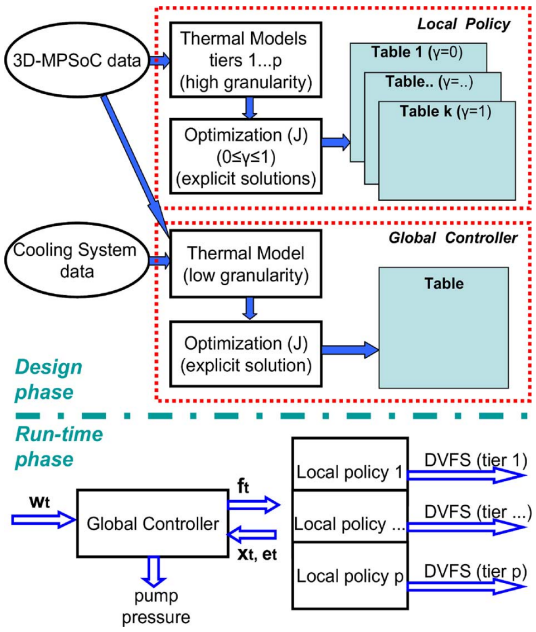

Fig. 9. Design phase and run-time phase of the proposed hierarchical thermal management.

decisions of both *local* and *global* controllers. Afterwards, these decisions are allocated in a look-up table-based implementation, at *design phase* , to be used by the global and local controllers at the *run-time phase* .

To compute the tables needed for the implementation of the local controllers, all design data related to the structure of the 3D-MPSoC (e.g., elements layout, thermal conductivity of materials,\ldots) are used to create an accurate thermal model for each one of the p tiers composing the 3D-MPSoC (upper part of Fig. 9). This model has a fine granularity and can be formulated as an optimization problem. Different explicit solutions (for various values of the input parameters and optimization goals) are then stored into tables (cf. Section IV-F) to be used at run-time. To compute the table needed for the implementation of the global controller, we build a coarse-grained thermal model of the 3D-MPSoC and of the cooling system (e.g., the available pumping power values, microchannels layout,. . .).

During *run-time*, both the global and the local controllers apply the rules stored in the aforementioned look-up tables. Each local controller generates the frequency setting for its tier elements at the processing element-level granularity, while the global controller sets the pressure for the cooling system pump.

The overall system uses software-driven thermal management, namely, the control action is done by software routines (for both the local and global controller) that access the pre-computed data in the tables. These tables represent the control *policies*. Their computation is described in the following subsections.

The global controller communicates with all local controllers, sets the liquid flow rates and assigns the workload to each tier. This routine is always active and is performed by a dedicated task. In order to guarantee its permanent and reliable operation, we assign this task to a dedicated processing unit.

D. Policy Computation: Global Thermal Controller

The *global thermal controller* is the unit responsible for the global joint operation of all local controllers and for the pump control which sets the coolant pressure.

As described in Section III-C, the workload to be dispatched to each local controller is stored in vector $\mathbf{f}_\tau$. The p entries of this vector contain the average frequency of operation at which each local controller has to work in order to execute the workload assigned to its controlled tier by the global unit.

The global controller policy minimizes power and undone workload [see (9)]. Furthermore, the performance requirements coming from the scheduler have to be fulfilled and the maximum temperature constraint satisfied. The problem can be defined as follows:

$$J = \sum_{\tau=1}^{h} \left(||\mathbf{R}\mathbf{p}_{\boldsymbol{\tau}}|| + ||\mathbf{T}\mathbf{u}_{\boldsymbol{\tau}}|| \right) \tag{10}$$

$$\min \ J \tag{11}$$

$$\text{subject to}: f_{\min} \preceq \mathbf{f}_\tau \preceq \mathbf{f}_{\max} \ \forall \tau \tag{12}$$

$$\mathbf{x}_{\tau+1} = \mathbf{A}\mathbf{x}_\tau + \mathbf{B}\mathbf{p}_\tau \ \forall \tau \tag{13}$$

$$\tilde{\mathbf{C}}\mathbf{x}_{\tau+1} \preceq \mathbf{t}_{\max} \ \forall \tau \tag{14}$$

$$\mathbf{u}_\tau \succeq 0 \ \forall \tau \tag{15}$$

$$\mathbf{u}_\tau = \mathbf{w}_\tau - \mathbf{f}_\tau \ \forall \tau \tag{16}$$

$$\mathbf{l}_\tau \succeq \mu \mathbf{f}_\tau^2 \ \forall \tau \tag{17}$$

$$-\mathbf{w} \preceq \mathbf{m}_{\tau+1} - \mathbf{m}_\tau \preceq \mathbf{w} \ \forall \tau \tag{18}$$

$$0 \preceq \mathbf{m}_\tau \preceq \mathbf{1} \ \forall \tau \tag{19}$$

$$\mathbf{p}_\tau = [\mathbf{l}_\tau; \mathbf{m}_\tau] \ \forall \tau \tag{20}$$

where matrices $\mathbf{A}$, $\mathbf{B}$ are related to the overall 3D-MPSoC system description (cf. Section III). These matrices represent the 3D-MPSoC system using a coarse granularity of the thermal cells and where the sampling time of the resulting discrete-time system is T_{GC}. We define the horizon of this predictive policy as h [1]. Then, the objective function J is expressed by a sum over the horizon.

In this equation, the first term $||\mathbf{R}\mathbf{p}_{\boldsymbol{\tau}}||$ is the norm of the power input vector p weighted by matrix $\mathbf{R}$. Power consumption is generated here by two main sources: i) the workload setting and ii) the liquid cooling pumping power. Vector p is a vector containing normalized power consumption data the p tiers and the pumping power. Matrix $\mathbf{R}$ contains the maximum value of the power consumption of the tiers (first p diagonal entries) and the cooling system (last entry). The second term $||\mathbf{T}\mathbf{u}_{\boldsymbol{\tau}}||$ is the norm of the required workload, but not yet executed. To this end, the weight matrix $\mathbf{T}$ quantifies the importance that executing the required workload from the scheduler has in the optimization process. Then, Inequality (12) defines a range of working frequencies to be used, but this does not prevent from adding in the optimization problem a limitation on the number of allowed frequency values.

Equation (13) defines the evolution of the 3D-MPSoC according to the present state and inputs. Equation (14) states that temperature constraints should be respected at all times and in all specified locations. Since the system cannot execute jobs that have not arrived, every entry of $\mathbf{u}_\tau$ has to be greater than or equal

to 0 as stated by (15). The undone work at time τ, u_τ is defined by (16). Equation (17) defines the relation between the power vector $\mathbf{l}$ and the working frequencies. μ is a technology-dependent constant.

Then, (18)-(19) define constraints on the liquid cooling management. The normalized pumping power value ($\mathbf{m}$) scales, and any time instance τ, from 0 (no liquid injection) to 1 (power at the maximum pressure difference allowable), as shown in (19). Moreover, we limit the maximum increment/decrement change in the pumping power value from time (τ) to $(\tau+1)$ by a another normalized value $\mathbf{w}$, as shown in (18), which models the mechanical dynamics of the pump. Although we assume one pump in the target 3D-MPSoCs, since we use a vector notation for the pumping power and its constraints, our formulation is valid for multiple pumps as well.

Equation (20) defines formally the structure of vector $\mathbf{p}$, as proposed in Section III-B. Vector $\mathbf{l} \in \Re^p$ is the power input vector, where p is the number of tiers of 3D-MPSoC.

Finally, we formulate the control problem over an interval of h time steps, which starts at current time τ. Therefore, our approach is predictive. Indeed the result of the optimization is an optimal sequence of future control moves (i.e., amount of workload to be executed in average for each tier of the 3D-MPSoC which is stored in vector $\mathbf{f}$). Then, we only apply to the target 3D-MPSoC the first samples of such a sequence; the remaining moves are discarded. Thus, at each next time step, a new optimal control problem based on new temperature measurements and required frequencies is solved over a shifted prediction horizon (e.g., the "receding-horizon" [1] mechanism), which represents a way of transforming an open-loop design methodology into a feedback one,as at every time step the input applied to the process depends on the most recent measurements.

It has been shown by [1] and [43] that this problem can be transformed so that the solution is given by the linear system

$$\mathbf{y}_{\tau+1} = \mathbf{F}_j \begin{bmatrix} \mathbf{x}_{\tau+1} \\ \mathbf{f}^2\tau \\ \mathbf{w}_\tau^2 \end{bmatrix} + \mathbf{g}_j \tag{21}$$

where $\mathbf{y}$ is the desired solution as a vector containing the workloads and the pump power, matrix $\mathbf{F_j}$ is a suitable matrix, and $\mathbf{g_j}$ a suitable vector defined over subregions of the solution space indexed by j. We refer the reader to [1] and [43] for details. In [43] an approximate computation method of the regions shows a consistent reduction in the number of storage space with a negligible performance loss.

E. Policy Computation: Local Controllers

The p local controllers are responsible for the thermal management (e.g., DVFS) of the p tiers of the target 3D-MPSoC. Then, for each tier i the local controller sets frequency and voltage for the $q(i)$ frequency islands (cf. Fig. 10).

In our hierarchical design, the local controller i receives as input the vector $\mathbf{f}_{t+1}$, which is the average frequency at which island i has to run to execute all the workload assigned to it by the global unit. As a second input data, we use information from a minimum set of specifically located thermal sensors, which provide the minimal feedback run-time input needed to estimate the global thermal profile of the 3D-MPSoC island. In our work, we assume that the thermal sensors are optimally allocated on the die as shown in previous work [44]. Thus, the impact of thermal sensor quality and allocation on the management policy is beyond our scope. Then, the local policy computes the frequencies and voltages for all the $q(i)$ units inside island i, as sketched in the dotted box of Fig. 10. Input data are used as both computing and selection parameters to choose one of the k functions stored in pre-computed look-up tables (cf. Section IV-C).

The local controller decides on the type of optimization to perform: either performance or power-oriented optimization (cf. Section IV-F) and the related policies are stored in the corresponding look-up tables. Specifically, the control policies optimize power and undone workload [see (9)]. We use an optimization parameter γ that weights these two objectives. At the same time, performance requirement coming from the global controller has to be fulfilled and the maximum temperature constraint satisfied.

The control function is expressed by a policy that is the solution of the following optimization problem:

$$J = \sum_{\tau=1}^{h} \left(\|\mathbf{R}\mathbf{p}_{\boldsymbol{\tau}}\| + \gamma \|\mathbf{T}\mathbf{u}_{\boldsymbol{\tau}}\| \right) \tag{22}$$

$$\min \ J \tag{23}$$

$$\text{subject to} : \mathbf{v}_{\min} \preceq \mathbf{v}_\tau \preceq \mathbf{v}_{\max} \ \forall \tau \tag{24}$$

$$\mathbf{x}_{\tau+1} = \mathbf{A}\mathbf{x}_\tau + \mathbf{B}\mathbf{p}_\tau \ \forall \tau \tag{25}$$

$$\tilde{\mathbf{C}}\mathbf{x}_{\tau+1} \preceq \mathbf{t}_{\max} \ \forall \tau \tag{26}$$

$$\mathbf{u}_\tau \succeq 0 \ \forall \tau \tag{27}$$

$$\mathbf{u}_\tau = (\mathbf{f}_\tau)_i - \sum \mathbf{v}_\tau \ \forall \tau \tag{28}$$

$$\mathbf{p}_\tau \succeq \mu \mathbf{v}_\tau^2 \ \forall \tau \tag{29}$$

where matrices $\mathbf{A}, \mathbf{B}$ are related to the thermal modeling of the specific tier that the local controller is supervising. The objective function J expresses the minimization problem by a weighted sum of two terms, in a similar vein as the global policy is computed, except for the tuning parameter γ. Parameter γ changes according to the specific type of optimization criteria for each tier. It ranges from 0 to 1 in steps of 0.1. We set this parameter at run-time based on the maximum temperature recorded according to a heuristic rule: the hotter the thermal profile is , the lower γ, and vice versa. Thus, the controller performs performance-oriented optimization in the case of cold thermal profile, but power saving oriented optimization in case of a hot thermal profile. γ is used at run-time to choose from a set of tables, as shown in Fig. 10. In the next subsection, we present the generated design space by the parameter γ, and quantify how it significantly affects the power and performance trade-offs of the local policy design.

As in the previous case, this problem can be transformed such that the solution is given by the following linear system:

$$\mathbf{y}_{\tau+1} = \mathbf{F}_j \begin{bmatrix} \mathbf{x}_{\tau+1} \\ \mathbf{v}^2\tau \\ \mathbf{f}_\tau^2 \end{bmatrix} + \mathbf{g}_j \tag{30}$$

where $\mathbf{y}$ is the desired solution as a vector containing the frequencies of the various islands for the tier under consideration.

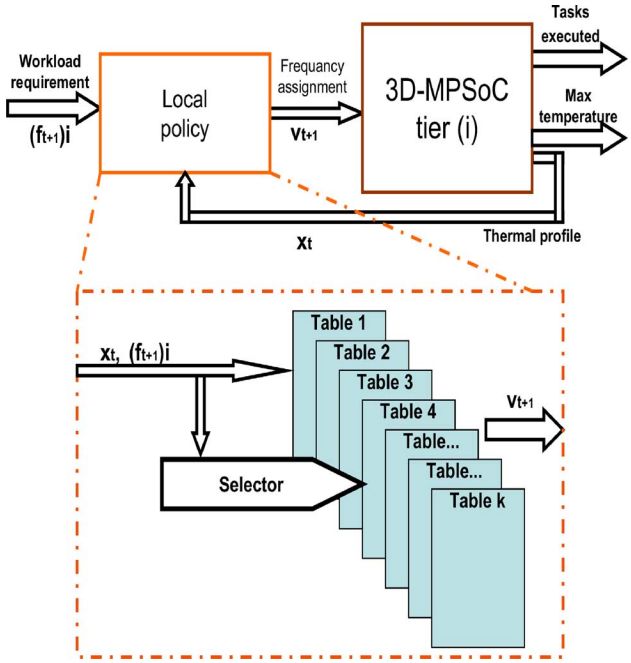

Fig. 10. Local policy controller block diagram.

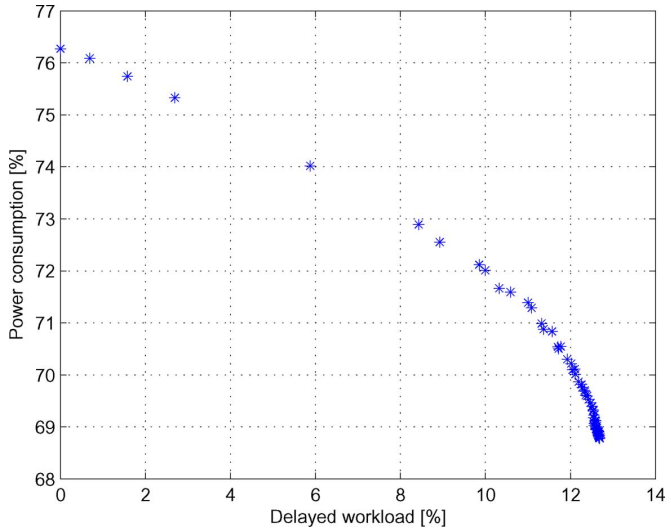

Fig. 11. Example of normalized power consumption versus delayed workload for different optimization criteria ranging from power- to performance-oriented optimization.

F. Run-Time Local Policy Optimization Criteria

The empirical law that expresses power consumption as a function of the frequency setting (see [36], [38], [43]) can be approximated by a quadratic function. Furthermore, this equation is convex[7]. Thus, by applying basic properties of convex functions, we obtain the following:

$$\mathbf{p}_\tau + \mathbf{p}_{\tau+\epsilon} \geq 2 \cdot \mathbf{p}_{(\tau+\epsilon)/2} \; \forall \epsilon \in [0, 1) \tag{31}$$

where $\mathbf{p}_\tau$ is the power consumption at time τ. Since frequency setting and executed workload are positively correlated, then potential energy savings demand a uniformly distributed workload. Unfortunately workloads are usually not uniformly distributed during the run-time execution of the policy and scheduling task uniformly would increase latency.

Inequality (31) expresses this issue as follows:

$$\mathbf{p}_{\text{pow}} \leq \mathbf{p} \leq \mathbf{p}_{\text{perf}} \tag{32}$$

by indicating that power consumption $\mathbf{p}$ is bounded between two values. On the one hand, the lower bound ($\mathbf{p}_{\text{pow}}$) is the power value consumed when the workload is uniformly distributed. In this case, we optimize the execution for power minimization by allowing a nonzero task execution delay, but at the same time we require that the complete workload has to be executed. On the other hand, the upper bound ($\mathbf{p}_{\text{perf}}$) is the power consumed when all tasks are executed at the same time they arrive. In this case, we optimize the execution for performance and the resulting task execution delay is zero. Clearly, the gap between these two numbers is highly dependent on the workload properties. Fig. 11 shows an example of the resulting power consumption versus delayed workload for different optimization criteria [γ in (22)], ranging from power- to performance-oriented optimization.

TABLE II
THERMAL AND FLOORPLAN PARAMETERS DEPLOYED IN THE MODEL.

Parameter	Value
Silicon conductivity	$130 W/(m \cdot K)$
Silicon capacitance	$1635660 J/(m^3 \cdot K)$
Wiring layer conductivity	$2.25 W/(m \cdot K)$
Wiring layer capacitance	$2174502 J/(m^3 \cdot K)$
Water conductivity	$0.6 W/(m \cdot K)$
Water capacitance	$4183 J/(kg \cdot K)$
Die thickness (one stack)	$0.15 mm$
Area per core	$10 mm^2$
Area per L2 cache	$19 mm^2$
Total area of each layer	$115 mm^2$

V. SIMULATION SETUP

A. 3D-MPSoC Specifications

The 3D-MPSoC architecture we are considering is presented in Figs. 3 and 4. This architecture is based on the 90-nm UltraSPARC T1 (i.e., Niagara-1) processor [22]. The power consumption, area, and the floorplan of UltraSPARC T1 are available in [22]. UltraSPARC T1 has 8 multi-threaded cores, and a shared L2-cache for every two cores. In our architecture, we use twice the existing elements in UltraSPARC T1 (e.g., 16 multi-threaded cores) since we use four silicon tiers in our targeted 3D-MPSoC. In our thermal model of this 3D-MPSoC, the used parameters are provided in Table II. This table contains the thermal conductance and capacitance values of different materials used in modeling the stack.

To implement the voltage and frequency scaling techniques, we use frequencies ranging from a minimum (166 MHz) to a maximum value (1.2 GHz), as specified by [22]. In this range, only specific values of frequencies are allowed. These values are generated from the integer division of the maximum clock frequency on a linear scale, as presented in [3].

We dynamically calculate the leakage power of processing cores as a function of their area and actual run-time temperature. We use a base leakage power density of 0.25 W/mm^2 at 383 K for 90-nm technology [6]. Thus, the leakage power at a

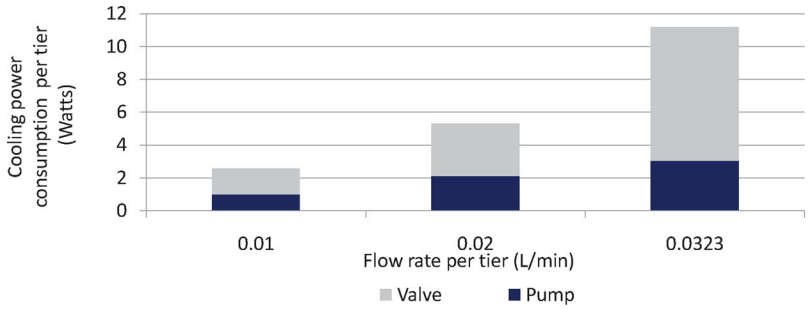

Fig. 12. Power consumption and flow rates of the cooling infrastructure per one tier.

TABLE III
MICROCHANNEL-BASED PARAMETERS USED IN DIFFERENT TOPOLOGIES

Parameter	straight channel	Bent channel
Channel width	$50\mu m$	$50\mu m$
Channel height	$100\mu m$	$100\mu m$
Channel pitch	$150\mu m$	$150\mu m$
Channel length	$11mm$	$2.5 - 11mm$
Number of channels per tier	67	90
Max flow rate per tier	$0.0323l/min$	$0.0686l/min$

temperature T °K is given by: $P(T) = P_o \cdot e^{\beta(T-383)}$, where P_o is the leakage power at 383 K, and β is a technology dependent coefficient. We set $\beta = 0.017$ [32].

B. Cooling Model

The geometry of the cooling layer is related to the following factors: the microchannel topology and dimensions, and the TSVs' sizes and spacing requirements. In our model, we use 50 μm diameter TSVs with 100 μm spacing requirements. The microchannel-related parameters is shown in Table III for both straight and bent channels. This table shows that the amount of injected fluid in the case of bent channels is more than that of straight channel. This increase implies better heat removal capabilities, but the increase in flow rate comes with an increase of the pumping power.

We assume that there is only one pump connected to all microchannels of all the layers, such as a centrifugal pump EMB MHIE [34], is responsible for the fluid injection to the whole system. This pump has the capability of producing large discharge rates at small pressure heads. Liquid is injected to the stack from this pump via a pumping network. To enable using different flow rates for each stack, we control the fluid via control valves we include in the network. We assume *normally closed valves* (NCV) provided by Festo group [35]. NCVs use external power to reduce the pressure drop and to increase the flow rate. Without loss of generality, this configuration is scalable into different pumping networks, where different valves are used to control the fluid in every tier. Fig. 12 shows the power consumed by the pump and valve per tier to inject the fluid from a single at a certain flow rate and pressure difference. In the case of straight channels, we use the same plotted values. However, in the case of bent channels, we increase the energy consumed by the pump only to account for the increased amount of injected fluid at the same pressure difference. Unlike the valve energy which is a function of the pressure difference, not the flow rate. Thus, the valve energy remains the same for both straight and bent channels.

C. Virtual Platform Environment

The 3D-MPSoC simulation framework is a SystemC-based simulation platform. The main device consists of 16 (8 per tier) 32-bit cores, 16 private memories and 16 shared memories distributed in the 4 tiers of our target 3D-MPSoC (cf. Fig. 4). All these units communicate among each other by a crossbar interconnect. A floating point unit is also connected to it. The virtual platform environment provides also power statistics for the several hardware modules in the simulated platform. The simulation is based on applications generating functional data traffic on the target architecture. Dynamic power consumption data are coming from the 3D simulation platform while temperature data are extracted using the publicly available 3D-ICE thermal modeling tool [39], as described in the previous sections. Afterwards, the leakage power is computed as stated before and added to the dynamical power to estimate the total power consumption. Modern OSes have a multi-queue structure, where each CPU core is associated with a dispatch queue, and the job scheduler allocates the jobs to the cores according to the current policy. In our simulator, we implement a similar infrastructure, where the queues maintain the threads allocated to cores and execute them.

We use workload traces collected from real applications running on an UltraSPARC T1. We record the utilization percentage for each hardware thread at every second using *mpstat* for several minutes for each benchmark. We use various real-life benchmarks including web server, database management, and multimedia processing. The web server workload is generated by SLAMD [37] with 20 and 40 threads per client to achieve medium and high utilization, respectively. For database applications, we experiment with MySQL using sysbench for a table with 1 million rows and 100 threads. Finally, we run several instances of the mplayer (integer) benchmark as typical examples of multimedia processing. The utilization ratios are averaged over all cores throughout the execution.

D. Policy Setup

The *global thermal controller* activation period is $T_{\mathrm{GC}} = 1$ s, while the local policies are applied every $T_{\mathrm{LC}} = 10$ ms. The simulation step for the discrete time integration of the thermal model has been set to 200 μs. The maximum temperature limit is set to 370 K. The room temperature and fluid temperature (T_{fluid}) are set to 300 K. In the problem formulation, to establish the relation between the frequency setting and the power consumption, we use a quadratic relation as in [38]. The time

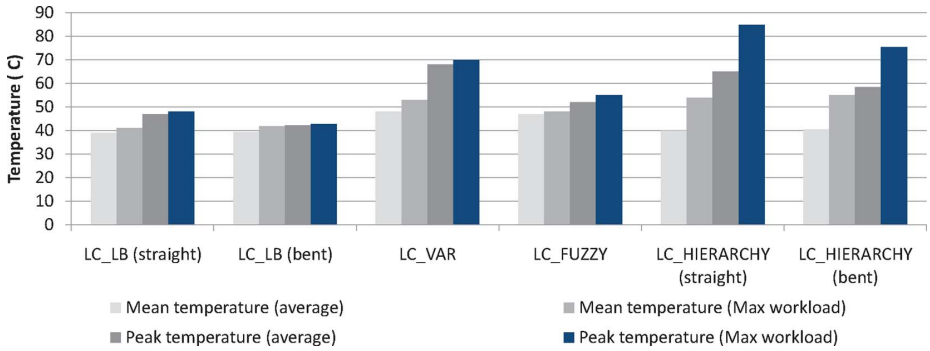

Fig. 13. Peak and average temperatures observed using all the policies, both for the average case across all workloads and maximum workload on 4-tier 3D-MPSoC.

constants needed by the mechanical dynamics of the cooling pumps to go from 0 to maximum power is set to 400 ms.

We vary the parameter γ in the local policies from 0 to 1 with steps of 0.1. Since every value of this parameter is associated with a different look-up table, there are 11 tables in the local controller.

E. Compared 3D-MPSoC Thermal Management Policies

In our evaluation of the proposed hierarchical management policy thermal and energy efficiency, we implement different state-of-the-art thermal management techniques that we elaborate on them as follows:

- *Liquid cooling with load balancing (LC_LB)* [15], [17] (the default implementation in most operating systems): Applies the maximum flow rate (0.0323 l/min per tier), while balancing the workload by moving threads from a core's queue to another if the difference in queue lengths is over a threshold.
- *LUT-based varying flow rate with TALB (LC_VAR)* [16]: Changes the flow rate (between 0.01–0.0323 l/min per tier) based on the predicted maximum temperature, but the jobs are scheduled with *temperature-aware 3D load balancing* [16].
- *Fuzzy control-based thermal management (LC_FUZZY)* [32]: Uses a run-time fuzzy control to alter the flow rate (between 0.01–0.0323 l/min per tier) and tunes the voltage and frequency values of the processing elements.

In our evaluation, we use the straight microchannel-based cooling layer with all policies, while we use the bent microchannel-based cooling layer with LC_LB and our proposed hierarchical policy.

VI. Results

In our evaluation of different thermal management policies, we compare our proposed policy with respect to the other management techniques mentioned above based on the following:

- maximum and average temperatures;
- thermal gradients;
- power consumption and performance degradation.

In the following subsections, we elaborate on each of the aforementioned metrics.

A. Maximum and Average Temperatures

Thermal impact of all the policies the 4-tier 3D-MPSoC is shown in Fig. 13. In this figure, we show in the peak and average temperature recordings of the same workloads mentioned before (cf. Section V-C). Interlayer liquid cooling has the ability to absorb the heat flux between different tiers surrounding the cooling layer, regardless the used structure. LC_LB reduces the peak temperature to 47 °C, whereas LC_FUZZY and LC_VAR push the system into a higher peak of 52 °C and 67 °C, respectively, but still avoids any hot-spots. This is the similar case in our proposed hierarchical policy, where the peak temperature reaches 84 °C. The alteration between the peak temperature comes from the fact that main target is to reduce the peak temperature to any value below 85 °C. However, since each technique has a different management policy, with different control elements, the peak and average temperatures are affected.

B. Thermal Gradients

We compute thermal gradients in the stack in addition to computing the peak/average temperatures. We calculate the maximum thermal gradient in the whole stack as well as the average intralayer thermal gradient of the different source layers in the stack. We define the thermal gradient threshold by 15 °C, hence the policy objective is to minimize the maximum thermal gradient to any value below 15 °C.

Figs. 14 and 15 show the maximum thermal gradient and the intralayer thermal gradient of the 3D-MPSoC with different management policies. Although interlayer liquid cooling diminishes the thermal hot-spots, it increases both the intralayer and the maximum thermal gradient of the stack. This is based on the fact that the fluid grows thermally from the inlet to the outlet such that the elements near the inlet have more heat removed than the ones at the outlet. Moreover, varying the flow rate, as in LC_FUZZY and LC_VAR, increases the thermal gradient, since reducing the flow rate increases the thermal gradient of the system.

Our hierarchical policy manages to reduce intralayer thermal gradients below 10 °C per layer. This is due to that fact that the local controllers distributed the assigned workload among the controlled elements, taking into consideration their thermal state. Thus, elements with lower temperature gets more load, while high temperature elements are assigned lower workload.

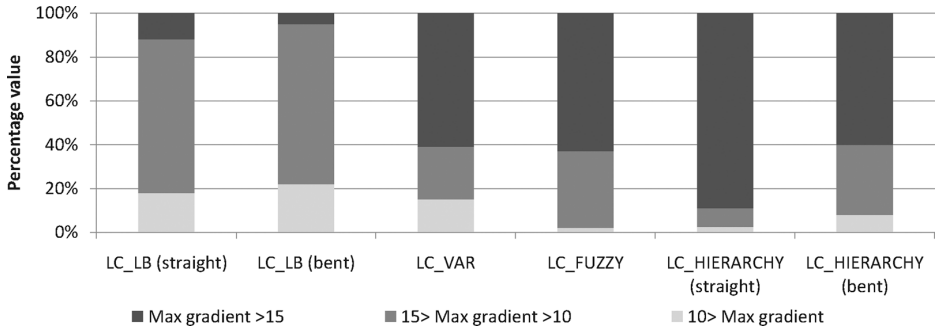

Fig. 14. Maximum thermal gradient of the whole 3D-MPSoC stack, using the average case of all workloads.

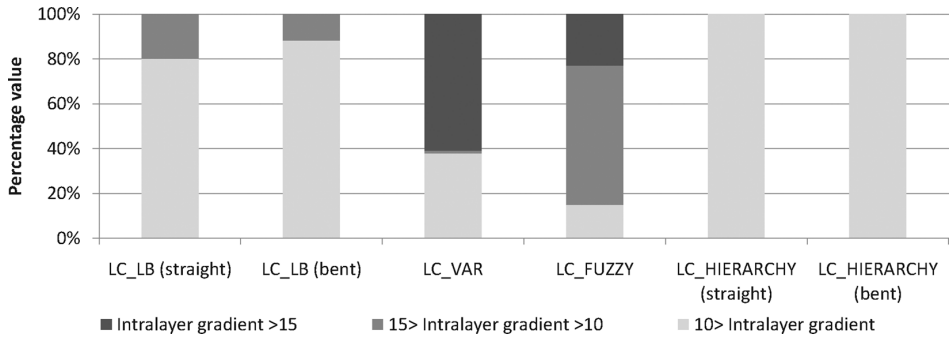

Fig. 15. Average intralayer thermal gradient of the whole 3D-MPSoC stack, using the average case of all workloads.

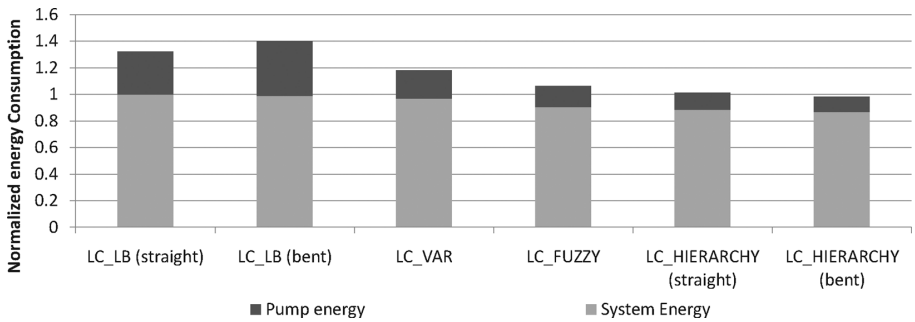

Fig. 16. Normalized energy consumption in the whole system (chip and cooling network) averaged per stack.

While using straight microchannels has an impact on the thermal gradient, the usage of bent microchannels aids in diminishing the maximum thermal gradient. The peak thermal gradient in LC_LB with bent channels, is reduced, by 58% with respect to using the same policy with straight channels. This enhancement is based on the fact that there is more fluid pumped to the bent structure than the straight structure. Moreover, the fluid path is in the bent channels is relatively shorter than that of straight channels. Thus, the fluid thermal growth is lower in the bent channel case. Furthermore, the use of bent channels with our policy aids in reducing the maximum gradient by an additional 32% with respect to using straight channels.

C. System and Cooling Power Consumption

Fig. 16 shows the total consumed power when running the various policies on the 4-tier MPSoC with the average workload. Energy consumption values are normalized with respect to the load balancing policy on the 3D-MPSoC with LC_LB. In this figure, we show that our proposed policy manages to reduce the cooling power and the overall system power by 60% and 23%, respectively, with respect to LC_LB. Moreover, our policy even reduced the cooling energy more than LC_VAR and LC_FUZZY by 40% and 22%, respectively.

When the bent channels are used with LC_LB, the pumping power consumed is higher than the case with straight channels. This is based on the same fact that more fluid is pumped in this case, hence more power is needed. However, when the bent channels are used, our policy does not apply the maximum flow rate since the objective goal is achievable with lower flow rates. Thus, the consumed pumping power in the bent channel case is of the same order as the case with straight channels.

VII. Conclusion

The contribution of this work is a novel online thermal management policy for high-performance 3D systems with liquid cooling. The proposed controller uses DVFS and adjusts the liquid flow rate to meet the desired performance requirements and to minimize the overall MPSoC energy consumption.

The proposed controller has an innovative hierarchical structure that allows the policy to be both effective in terms of

achieving its performance requirements and simple in terms of hardware implementation. Moreover, the hierarchical structure of the policy allows the thermal management system to be easily scalable to any 3D systems. The optimization problem is executed and it considers the thermal profile of the system, its evolution over time and current time-varying workload requirements. The implementation is done using look-up tables.

We implemented the policy on a hardware simulation platform and performed experiments on a 3D-MPSoC case study using benchmarks ranging from web-accessing to playing multimedia. Results show significant advantages in terms of energy savings that reach values up to 50% with respect to state-of-the-art thermal control techniques for 3D stacks with liquid cooling, and a thermal balance with differences of less than 10 °C per layer.

ACKNOWLEDGMENT

The authors would like to thank L. Thiele and T. Brunschwiler for their suggestions.

REFERENCES

[1] A. Bemporad *et al.*, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.
[2] L. Benini, A. Bogliolo, G. A. Paleologo, and G. De Micheli, "Policy optimization for dynamic power management," *IEEE Trans. Comput.-Aided Des. Circuits Syst.*, vol. 18, no. 6, pp. 813–833, Jun. 1999.
[3] L. Benini *et al.*, "MPARM: Exploring the multi-processor SOC design space with SystemC," *J. VLSI Signal Process.*, vol. 41, no. 2, pp. 169–182, 2005.
[4] A. Bhunia *et al.*, "High heat flux cooling solutions for thermal management of high power density gallium nitride HEMT," in *JTHERM*, 2004, pp. 75–81.
[5] B. Black *et al.*, "Die stacking (3D) microarchitecture," *IEEE MICRO*, pp. 469–479, 2006.
[6] P. Bose *et al.*, "Power-efficient microarchitectural choices at the early design stage," presented at the PACS, 2003.
[7] S. Boyd *et al., Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[8] T. Brunschwiler *et al.*, "Interlayer cooling potential in vertically integrated packages," *Microsyst. Technol.*, vol. 15, no. 1, pp. 57–74, 2008.
[9] T. Brunschwiler *et al.*, "Validation of the porous-medium approach to model interlayer-cooled 3D chip stacks," in *3DIC*, 2009, pp. 1–10.
[10] R. J. Cochran *et al.*, "Consistent runtime thermal prediction and control through workload phase detection," in *DAC*, 2010, pp. 62–67.
[11] A. K. Coskun *et al.*, "Temperature-aware task scheduling," in *DATE*, 2007, pp. 1–6.
[12] A. K. Coskun *et al.*, "Proactive temperature balancing for low cost thermal management in MPSOCs," in *ICCAD*, 2008, pp. 250–257.
[13] A. K. Coskun *et al.*, "Temperature management in multiprocessor SoCs using online learning," in *DAC*, 2008, pp. 890–893.
[14] A. K. Coskun *et al.*, "Dynamic thermal management in 3D multicore architectures," in *DATE*, 2009, pp. 1410–1415.
[15] A. K. Coskun *et al.*, "Modeling and dynamic management of 3D multicore systems with liquid cooling," in *VLSI-SoC*, 2009, pp. 60–65.
[16] A. K. Coskun *et al.*, "Energy-efficient variable-flow liquid cooling in 3D stacked architectures," in *DATE*, 2010, pp. 111–116.
[17] J. Donald *et al.*, "Techniques for multi-core thermal management: Classification and new exploration," in *ISCA*, 2006, pp. 78–88.
[18] T. Emi *et al.*, "Tape: Thermal-aware agent-based power economy for multi/many-core architectures," in *ICCAD*, 2009, pp. 302–309.
[19] T. R. Halfhill *et al.*, "Transmeta breaks X86 low power barrier," *Microprocessor Rep.*, 2000.
[20] M. Healy, M. Vittes, M. Ekpanyapong, C. S. Ballapuram, S. K. Lim, H.-H. S. Lee, and G. H. Loh, "Multiobjective microarchitectural floorplanning for 2-D and 3-D ICs," *IEEE Trans. Comput.-Aided Des. Circuits Syst.*, vol. 26, no. 1, pp. 38–52, Jan. 2007.
[21] W. Hung *et al.*, "Thermal-aware allocation and scheduling for systems-on-chip," in *DATE*, 2005, pp. 898–899.
[22] P. Kongetira *et al.*, "Niagara: A 32-way multithreaded SPARC processor," *IEEE MICRO*, vol. 25, no. 2, pp. 21–29, 2005.
[23] *Laing 12 Volt DC Pumps Datasheets*, [Online]. Available: http://www.lainginc.com/pdf/DDC3_LTI_USletter_BR23.pdf
[24] H. Lee *et al.*, "Package embedded heat exchanger for stacked multichip module," in *Transducers, Solid-State Sensors, Actuators and Microsyst.*, 2003, pp. 1080–1083.
[25] Z. Li *et al.*, "Integrating dynamic thermal via planning with 3D floorplanning algorithm," in *ISPD*, 2006, pp. 178–185.
[26] Y. Lu *et al.*, "Software controlled power management," in *CODES*, 1999, pp. 157–161.
[27] M. Magno *et al.*, "Adaptive power control for solar harvesting multimodal wireless smart camera," in *ICDSC*, 2009, pp. 1–7.
[28] R. Mukherjee *et al.*, "Physical aware frequency selection for dynamic thermal management in multi-core systems," in *ICCAD*, 2006, pp. 547–552.
[29] K. Puttaswamy *et al.*, "Thermal herding: Microarchitecture techniques for controlling hotspots in high-performance 3D-integrated processors," in *HPCA*, 2007, pp. 193–204.
[30] Q. Qiu, Q. Wu, and M. Pedram, "Stochastic modeling of a power-managed system: Construction and optimization," *IEEE Trans. Comput.-Aided Des. Circuits Syst.*, vol. 20, no. 10, pp. 1200–1217, Oct. 2001.
[31] R. Reif *et al.*, "Fabrication technologies for three-dimensional integrated circuits," in *ISQED*, 2002, pp. ??–??.
[32] M. M. Sabry *et al.*, "Fuzzy control for enforcing energy efficiency in high-performance 3D systems," in *ICCAD*, 2010, pp. 642–648.
[33] T. Simunic, S. P. Boyd, and P. Glynn, "Managing power consumption in networks on chips," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 12, no. , pp. 96–107, Jan 2004.
[34] *WILO MHIE Centrifugal Pump*, [Online]. Available: http://www.wilo.com/cps/rde/xchg/en/layout.xsl/3707.htm
[35] *Festo Electric Automation Technology*, [Online]. Available: http://www.festo.com
[36] K. Skadron *et al.*, "Temperature-aware microarchitecture: Modeling and implementation," in *TACO*, 2004, pp. 94–125.
[37] *SLAMD Distributed Load Engine*, [Online]. Available: http://www.slamd.com
[38] S. Murali *et al.*, "Temperature control of high performance multicore platforms using convex optimization," in *DATE*, 2008, pp. 110–115.
[39] A. Sridhar *et al.*, "3D-ICE Fast compact transient thermal modeling for 3D-ICs with inter-tier liquid cooling," in *ICCAD*, 2010, pp. 463–470.
[40] A. Sridhar *et al.*, "Compact transient thermal model for 3d ICS with liquid cooling via enhanced heat transfer cavity geometries," in *THERMINIC*, 2010, pp. 1–6.
[41] D. B. Tuckerman *et al.*, "High-performance heat sinking for VLSI," *IEEE Electron Device Lett.*, vol. EDL-2, no. 5, pp. 126–129, May 1981.
[42] Y. Wang *et al.*, "Temperature-constrained power control for chip multiprocessors with online model estimation," in *ISCA*, 2009, pp. 314–324.
[43] F. Zanini *et al.*, "Online convex optimization-based algorithm for thermal management of MPSOCs," in *GLSVLSI*, 2010, pp. 203–208.
[44] F. Zanini *et al.*, "Temperature sensor placement in thermal management systems for MPSOCs," in *ISCAS*, 2010, pp. 1065–1068.
[45] Y. Zhang *et al.*, "Adaptive and autonomous thermal tracking for high performance computing systems," in *DAC*, 2010, pp. 68–73.
[46] X. Zhou *et al.*, "Thermal management for 3D processors via task scheduling," in *ICPP*, 2008, pp. 115–122.
[47] C. Zhu, Z. Gu, L. Shang, R. P. Dick, and R. Joseph, "Three-dimensional chip-multiprocessor run-time thermal management," *IEEE Trans. Comput.-Aided Des. Circuits Syst.*, vol. 27, no. 8, pp. 1479–1492, Aug. 2008.

Francesco Zanini received three Masters degrees in electronic engineering from the University of Parma, Parma, Italy, the National University of Ireland, Dublin, Ireland, and the Advanced Learning and Research Institute, University of Lagano, Switzerland, in 2005, 2006, and 2007, respectively. He is currently working toward the Ph.D. degree from the Laboratory of Integrated Systems (LSI), Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland.

His research interests include design methodologies for embedded MPSoC with particular emphasis on thermal management policies and algorithms.

Mr. Zanini received the Best Student Award from the faculty of engineering of the University of Parma in 2004. He won the Franchetti Award for his excellent school career.

Mohamed M. Sabry received the B.Sc. degree (with honors) as well as the best student award, the M.Sc. degree from AinShams University, Egypt, in 2005 and 2008, respectively. He is currently working toward the Ph.D. degree from the Embedded Systems Laboratory (ESL), Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland.

His research interests include Multi-Processor Systems-on-Chip (MPSoCs) and embedded systems design and resource management methodologies. His recent work focuses on thermal and reliability management of 2D and 3D-MPSoCs.

Mr. Sabry received the Best Student Award from AinShams University, Egypt, in 2005.

David Atienza (M'05) received the M.Sc. degree from the Complutense University of Madrid (UCM), Madrid, Spain, in 2001, and the Ph.D. degree from the Inter-University Microelectronics Center (IMEC), Belgium, and UCM, in 2005, both in computer science and engineering.

He is currently Professor and Director of the Embedded Systems Laboratory (ESL) at Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, and adjunct professor at the Computer Architecture Department, Complutense University of Madrid (UCM). His research interests focus on design methodologies for high-performance Multi-Processor Systems-on-Chip (MPSoCs) and embedded systems, including new 2D/3D thermal-aware design, wireless sensor networks, dynamic memory optimizations and Network-on-Chip (NoC) design. In these fields, he is co-author of more than 150 publications in prestigious journals and international conferences.

Dr. Atienza has received a Best Paper Award at the IEEE/IFIP VLSI-SoC 2009 Conference and two Best Paper Award Nominations at the ICCAD 2006 and DAC 2004 conferences. He is also Associate Editor of IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, IEEE EMBEDDED SYSTEMS LETTERS, and Elsevier's *Integration*. He is an elected member of the Executive Committee of the IEEE Council of Electronic Design Automation (CEDA) since 2008, and of the Board of Governors of the IEEE Circuits and Systems Society (CASS) since 2010.

Giovanni De Micheli (S'79–M'83–SM'89–F'94) is Professor and Director of the Institute of Electrical Engineering and of the Integrated Systems Centre at EPF Lausanne, Switzerland. He is program leader of the Nano-Tera.ch program. Previously, he was Professor of Electrical Engineering at Stanford University. His research interests include several aspects of design technologies for integrated circuits and systems, such as synthesis for emerging technologies, networks on chips and 3D integration. He is also interested in heterogeneous platform design including electrical components and biosensors, as well as in data processing of biomedical information. He is author of: *Synthesis and Optimization of Digital Circuits* (McGraw-Hill, 1994), and co-author and/or co-editor of 8 other books and of over 450 technical articles.

Prof. De Micheli is a Fellow of ACM, and member of the Academia Europaea. He is the recipient of the 2003 IEEE Emanuel Piore Award for contributions to computer-aided synthesis of digital systems. He received the Golden Jubilee Medal for outstanding contributions to the IEEE CAS Society in 2000. He received the 1987 D. Pederson Award for the best paper on the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, two Best Paper Awards at the Design Automation Conference, in 1983 and in 1993, and a Best Paper Award at the DATE Conference in 2005. He has been serving IEEE in several capacities, namely: Division 1 Director (2008–2009), co-founder and President Elect of the IEEE Council on EDA (2005–2007), President of the IEEE CAS Society (2003), Editor in Chief of the IEEE Transactions on CAD/ICAS (1987–2001). He is and has been Chair of several conferences, including DATE (2010), pHealth (2006), VLSI SOC (2006), DAC (2000), and ICCD (1989).