Modeling and Dynamic Management of 3D Multicore Systems with Liquid Cooling

Ayse K. Coskun[†], José L. Ayala[⋆], David Atienza[‡], Tajana Simunic Rosing[†]

[†]Computer Science and Engineering Dept. (CSE), University of California, San Diego, USA.

[⋆]Computer Architecture and Automation Dept. (DACYA), Complutense University of Madrid, Spain.

[‡]Embedded Systems Laboratory (ESL), Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland.

Abstract— **Three-dimensional (3D) circuits reduce communication delay in multicore SoCs, and enable efficient integration of cores, memories, sensors, and RF devices. However, vertical integration of layers exacerbates the reliability and thermal problems, and cooling efficiency becomes a limiting factor. Liquid cooling is a solution to overcome the accelerated thermal problems imposed by multi-layer architectures. In this paper, we first provide a 3D thermal simulation model including liquid cooling, supporting both fixed and variable fluid injection rates. Our model has been integrated in HotSpot to study the impact on multicore SoCs. We design and evaluate several dynamic management policies that complement liquid cooling. Our results for 3D multicore SoCs, which are based on 3D versions of UltraSPARC T1, show that thermal management approaches that combine liquid cooling with proactive task allocation are extremely effective in preventing temperature problems. Our proactive management technique provides an additional 75% average reduction in hot spots in comparison to applying only liquid cooling. Furthermore, for systems capable of varying the coolant flow rate at runtime, our feedback controller increases the improvement to 95% on average.**

I. Introduction

Continuous technical advances in chip design increase functionality and clock rates while shrinking the feature sizes. Interconnects have not followed the same scaling curve as transistors, and as a result they have become a limiting factor in performance and power consumption [16]. One solution to the rising interconnect power consumption is 3D stacking [3], which reduces wirelength through vertical integration of circuit blocks. However, 3D stacking substantially increases thermal resistances due to the placement of computational units on top of each other. High power densities are already a major concern in 2D circuits [1], and in 3D systems the problem is even more severe [3], [22]. The 3D stacked systems exacerbate temperature-induced problems, leading to degraded performance and reliability if not handled properly.

Conventionally, cooling of microprocessors is performed by attaching a heat sink on the package and removing the heat from the sink via fans. To assess the effectiveness of the cooling solution, the thermal resistance from the junction to ambient has to be reduced dramatically. As predicted by the ITRS road map, this thermal resistance should reach the 0.18 $^oC/W$ for the year 2010 and should continue decreasing in the future to address the increasing power density. Considering the high power densities in 3D systems and the shrinking space for thermal management devices, a shift from air cooling technology to liquid cooling will provide significant benefits. Liquid cooling is performed by attaching a heat sink with built-in microchannels, and also by fabricating microchannels within the interface material between the layers of the 3D architecture. Then, a coolant fluid (i.e., water or other fluids) is pumped through the microchannels to remove the heat.

* This work was partly supported by the Swiss Confederation through the Nano-Tera.ch NTF Project nr. 123618 - CMOSAIC, the Spanish Government Research Grants TIN2008-00508 and CSD00C-07-20811, Sun Microsystems, UC MICRO, Center for Networked Systems (CNS) at UCSD, MARCO/DARPA GSRC and NSF Greenlight.

Dynamic thermal management techniques developed for 2D chips keep the temperature under a given threshold or reduce the operating temperature as much as possible to avoid the appearance of local hot spots. Dynamic voltage and frequency scaling (DVFS), temperature-aware job allocation, and thread migration are examples of such techniques [10], [9]. For thermal management of 3D circuits, most of the prior work has only addressed design stage optimization, such as thermal-aware floorplanning [12]. For dynamic thermal management in 3D systems, Zhu et al. [30] evaluate several task migration and DVFS policies that respond to the feedback provided by the thermal sensors and performance counters.

The new and advanced heat removal capabilities of microchannel cooling for a full-scale multiprocessor SoC (MPSoC) and the dynamic thermal management opportunities for such systems have not been studied before. In this work, we first provide a compact thermal modeling approach for 3D stacked architectures with liquid cooling. Our approach can be integrated into widely-used automated thermal simulators such as HotSpot [24]. We then utilize the proposed model for the thermal evaluation of MPSoCs.

In our experimental results, we demonstrate that liquid cooling provides much more efficient cooling in terms of reducing and balancing the temperature in comparison to conventional heat-sink based cooling solutions. We couple the liquid cooling control with several dynamic thermal management policies to further increase the efficiency of cooling. We experiment with both fixed and variable coolant flow rates to investigate the benefits of cooling systems capable of dynamically adjusting the flow rate. Our results confirm that integration of liquid cooling and a dynamic management policy with feedback control achieves further reduction and balancing of temperature, increasing the system lifetime and performance.

The rest of the paper starts with a discussion of prior art in thermal modeling and management for 2D and 3D systems. In Section III, we provide the details of the 3D liquid cooling model. Section IV explains the dynamic management policies we implement and the closed-loop control mechanism. Experimental methodology and results are demonstrated in Sections V and VI, respectively, and we conclude in Section VII.

II. Related Work

Optimizing multicore scheduling with energy and performance (or timing) constraints has been studied quite extensively in the literature (e.g. [23], [29]). However, as power-aware policies are not always sufficient to prevent temperature-induced problems, thermal modeling and management methods have been developed. One of the first automated thermal models is HotSpot [24], which calculates transient temperature response given the physical characteristics and power consumption of the units in the die. To reduce simulation time even for large multicore systems while maintaining accuracy, a thermal emulation framework for FPGAs was proposed in [1].

Dynamic thermal management in microprocessors has been introduced in [5], where the authors explore performance trade-offs between different dynamic management mechanisms to tune the thermal profile at runtime. Computation migration and fetch toggling are other examples of dynamic management techniques [24]. Heo et al. reduce peak junction temperature by activity migration between multiple replicated units [13]. Heat-and-Run performs temperature-aware thread assignment and migration for multicore multithreaded systems [11]. Kumar et al. propose a hybrid method that coordinates clock gating and software thermal management techniques, such as temperature-aware priority management [18]. The multicore thermal management method introduced in [10] combines distributed DVS with process migration. For multicore systems, the temperature-aware task scheduling method proposed in [9] achieves better thermal profiles than conventional thermal management techniques without introducing a noticeable impact on performance.

For the thermal management of 3D circuits, most of the prior work has addressed design stage optimization, such as thermal-aware floorplanning (e.g. [12]). In [30], the authors evaluate several policies for task migration and DVS, which respond to the feedback provided by thermal sensors and integrated performance counters. Their approach also includes an offline workload profiling phase.

The work presented in this paper is the first to address dynamic thermal management for 3D multicore SoCs with liquid cooling. We first describe our modeling approach for microchannel cooling, and then propose policies that adjust workload allocation or coolant flow rates based on the feedback collected from the system.

III. Liquid Cooling Model

Modeling of the 3D stacked architecture with liquid cooling is accomplished in the following steps: 1) Forming the grid-level thermal R-C network, 2) Detailed modeling of the interlayer material, including the through-silicon-vias (TSVs) and the microchannels. In this section, these steps are described in detail.

A. System Overview

The standard air-cooling heat removal methods are inadequate for high performance 3D ICs, especially for systems with more than two stacked layers. The main challenge for 3D integration is to remove the high concentration of heat produced by the stacked microprocessor chips while also minimizing the thermal stresses imposed on the architecture, but still achieving a microprocessor design with high performance computing characteristics. For example, in a high performance 3D system, each chip on its own can produce heat at the rate of $100-150W/cm^2$, so for a stack of ten $2.25cm^2$ chip layers, 2.2-3.3 kW is dissipated. Microchannel cooling channels, integrated into the chip, offer significant advantages in addressing such heat removal challenges. In addition to water, which we assume in this paper, other coolants can be utilized for the microchannel cooling.

The use of liquid microchannels to cool down high power density electronics has been an active area of research since the initial work by Tuckerman and Pease [28]. Their liquid cooling system could remove 1000 W/cm^2; however, the volumetric flow rate and the pressure drop are both very large. More recent works show how back-side liquid cold plates, such as staggered microchannel and distributed return jet plates, can handle up to 400 W/cm^2 in single-chip applications with more realistic implementation details [6].

The heat removal capability of interlayer heat-transfer pin-line structures for 3D chips is investigated in [7]. At a chip size of 1 cm^2 and a $\Delta T_{jmax-in}$ of 60 K, the heat-removal performance is shown to be more than 200 W/cm^2 at pitches bigger than 50 μm. Finally, [2], [19] describe how to achieve variable flow rate for the

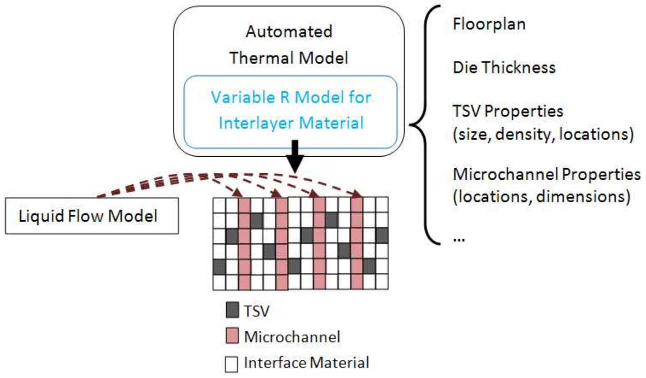

Fig. 1. Simulating 3D System with Microchannel Cooling

coolant. Having variable coolant flow rate allows the development of policies that use this mechanism to improve the cooling.

Figure 5 shows the 3D system targeted in this paper. The microchannels are distributed uniformly and fluid flows through each channel with the same flow rate.

B. Thermal Modeling for 3D

3D thermal modeling can be accomplished using an automated model that forms the R-C circuit for given grid dimensions. In this work, we utilize HotSpot v.4.1. [24], which is extended to include 3D modeling capabilities as discussed in [14]. The 3D version of HotSpot has been validated through comparisons with a commercial tool, Flotherm, which showed an average temperature estimation error of 3^oC, and a maximum deviation of 5^oC [15].

The extension we have developed for the existing multi-layered thermal modeling provides a new interlayer material model to include the TSVs and the microchannels. The conceptual flow of our extended model is shown in Figure 1. The TSV modeling is embedded in the interlayer material model by modifying the thermal resistivity of the TSV locations accordingly (see next sub-section).

In a typical automated thermal model, the thermal resistance and capacitance values of the blocks or grid cells are computed at the beginning of the simulation. To model the heterogeneous characteristics of the interlayer material including the TSVs and microchannels, we introduce two novelties: (1) As opposed to having a uniform thermal resistivity value of the layer, our infrastructure enables having various resistivity values for each grid cell, (2) The resistivity value of the cell can vary at runtime. Each grid cell except for the cells of the microchannels has a fixed thermal resistance value depending on the characteristics of the interface material and TSVs. The thermal resistivity of the microchannel cells is computed based on the liquid flow rate at runtime.

C. Modeling Through-Silicon-Vias (TSVs)

To model the effect of through-silicon-vias (TSV) on the thermal behavior, first, we perform a study to determine what granularity of modeling is required. The three investigated approaches in our work are: (1) Assuming a homogeneous TSV density all across the chip surface, (2) Providing a TSV density per each unit (i.e., core, cache, crossbar, etc), (3) Providing the exact TSV locations in the interlayer material. We assume that the effect of the TSV insertion to the heat capacity of the interface material is negligible, which is a reasonable assumption considering the total area of TSVs constitutes a very small percentage of the total area of the material.

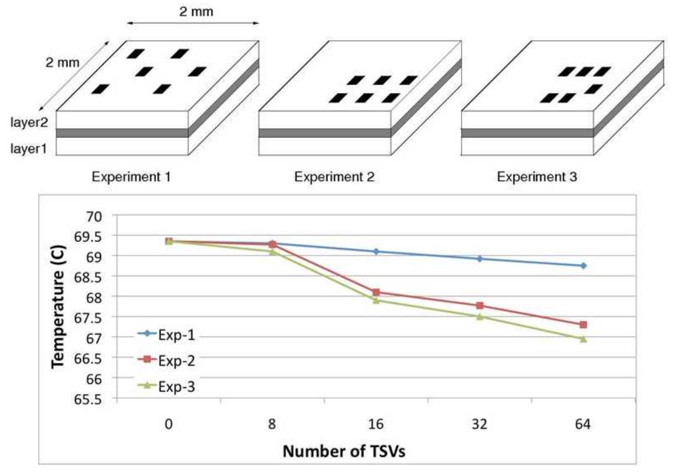

Fig. 2. TSV Modeling Granularity

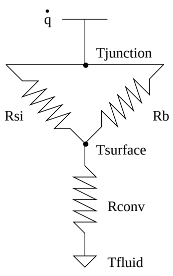

Fig. 3. Equivalent 3D Resistive Network Including Liquid Cooling

In (1), we assume a homogeneous via density and homogeneous distribution of TSVs across the die. The insertion of TSVs is expected to change the thermal characteristics of the interface material, thus, we compute the "combined" resistivity of the interface material based on the TSV density. In experiment (2), we differentiate among the different block functionalities to adjust the TSV density. For example, a crossbar structure will require a high TSV density to connect different layers in the 3D system, whereas a cache buffer communicating only with its associated cache may not require any vertical connections. Therefore, we assign a TSV density to each unit based on its functionality and system design choices. Finally, in experiment (3), we determine the exact location of the TSVs, and modify the thermal resistivity of each grid cell in the interlayer material if there is a TSV located at that position.

Since experiment (3) imposes a high computation overhead to build the thermal R-C circuit, we use a smaller die size in comparison to a full-scale MPSoC. The maximum steady state temperature results for each of the experiments are shown in Figure 2, for a chip with 2mm x 2mm footprint and two vertical layers. The TSVs are assumed to be located in the $1mm^2$ square shown in the floorplan of the original design. We simulate a total TSV count of $n = \{8, 16, 32, 64\}$ for all the three experiments. The TSV dimensions are set to $10\mu m \times 10\mu m$, with a spacing requirement of $10\mu m$ from each side of the TSV.

We observe that using block-level granularity provides very similar results to providing the exact locations of TSVs. Considering the similarity of accuracy between experiments (2) and (3), and that the overhead of locating each TSV in a large MPSoC in the simulator is prohibitively time consuming, we use approach (2).

D. Liquid Cooling Model

In a 3D system with liquid cooling, the local junction temperature can be computed using a resistive network shown in Figure 3. In this figure, the thermal resistance of the wiring layers (R_b), the thermal resistance of the silicon (R_{Si}) and the convective thermal resistance are combined to model the 3D stack. Considering the heat flux ($\dot{q}$) as the source and the chip back-side temperature (T_{fluid}) as the ground, the electrical circuit is solved to get the junction temperature ($T_{junction}$). Thus, the total thermal resistance (R_{tot}) of the junction is computed as in Equation 2 [7]. The parameters of the equation are provided in Table I, and the constant parameter values are taken from [7]. We assume a base flow rate of $15ml/min$ as in [17]. For the variable flow experiments, we use four flow rate settings of 10, 15, 20 and 25 ml/min. Each channel has a width of $700\mu m$ and a depth of $300\mu m$.

$$R_{tot} = R_{cond} + R_{conv} + R_{heat} \tag{1}$$

$$R_{tot} = \frac{1}{k_{si}/t + 1/R_b} + \frac{A}{\bar{h} \cdot A_t} + \frac{A}{\dot{V} \cdot \rho \cdot c_p} \tag{2}$$

TABLE I. PARAMETERS IN EQUATION 2

Parameter	Definition
R_{th}	Thermal resistance
R_{cond}	Conductive R_{th}
R_{conv}	Convective R_{th}
R_{heat}	Effective R_{th} representing $T_{in} - T_{out}$
k_{si}	Thermal conductivity of Si
t	Si base thickness
R_b	R_{th} of wiring levels
A	Heater area (i.e., total area consuming power)
h	Heat transfer coefficient
A_t	Total surface area
$\dot{V}$	Volumetric flow rate
ρ	Density
c_p	Heat capacity

IV. Dynamic Management of 3D Multicore Systems with Liquid Cooling

We complement the liquid cooling enabled 3D system with dynamic management policies. Even though liquid cooling provides significantly better cooling in comparison to traditional packages, integrating the liquid cooling with a dynamic monitoring and management technique further increases the efficiency. In this section, we provide the details of the proposed policies.

We propose two sets of policies. The first set of techniques assumes a fixed flow rate, and uses dynamic management techniques (such as thermally-aware job scheduling) to reduce and balance the temperature. The second set adjusts the flow rate of the liquid in the microchannels to maintain temperature at a safe level and also to smooth out the temperature variations on the chip.

A. Fixed Flow Rate

Using a fixed flow rate is the commonly proposed approach for 3D liquid cooling. This way, the liquid is distributed via a pump to all the microchannels at the same rate simultaneously.

We investigate both reactive and proactive methods. In **Fixed-Reactive**, the MPSoC workload scheduler directs the next arriving job to the coolest core on the die. This policy is similar to the policy of sending the workload to the coolest core in 2D chips [9].

Fixed-Proactive forecasts future temperature, and based on the predictions it then moves jobs from *projected* hotter cores to cores that are expected to be cooler. Due to thermal time constants, after we tune any system parameter such as workload scheduling or power consumption via dynamic management, there is a delay before we

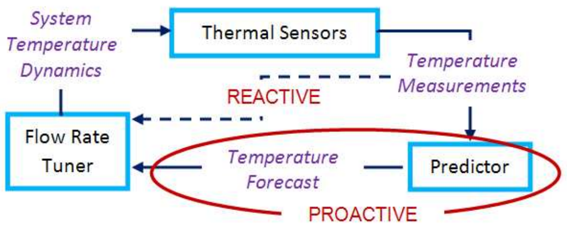

Fig. 4. Closed-Loop Control

see the effect on temperature. Therefore, proactive management can reduce and balance the temperature on the die more effectively.

Proactive workload allocation has previously been proposed in [8]. On each core, we use an ARMA (Autoregressive Moving Average) predictor to forecast temperature. As the workload characteristics are correlated during short time windows and temperature changes slowly due to thermal time constants, we assume the underlying data for the ARMA model is stationary, and use a recent history window of temperature values to form the ARMA predictor model. An ARMA model is described by Equation 3. In the equation, y_t is the value of the series at time t (i.e., predicted temperature value), a_i is the lag-i auto-regressive coefficient, c_i is the moving average coefficient and e_t is called the noise, error or the residual. p and q represent the orders of the auto-regressive (AR) and the moving average (MA) parts of the model, respectively. ARMA prediction is highly accurate for temperature forecasting, and runtime adaptation methods can also be integrated with ARMA as discussed in [8].

$$y_t + \sum_{i=1}^{p}(a_i\ y_{t-i}) = e_t + \sum_{i=1}^{q}(c_i\ e_{t-i}) \tag{3}$$

B. Varying Flow Rate

As discussed in Section III, it is possible to adjust the flow rate at runtime. We assume that a single pump distributes the liquid to all the channels. Thus, the flow rates in all the channels are the same, however, controlling the flow rate at runtime is possible.

The set of policies discussed in this section targets achieving uniform temperature distribution across the chip at a safe operating temperature (e.g. at 80^oC) by varying the flow rate in the microchannels. As the volume of liquid flow increases, the thermal resistivity decreases, easing cooling at that particular location. We discuss several policies which adjusts the flow rate based on the temperature measurements of the die.

We utilize closed-loop control to adjust the coolant flow rate to meet the temperature characteristics of the system. The flow of the closed-loop control is provided in Figure 4. We assume that a discrete set of flow rates are available, as noted in Section III.

Variable-Reactive reactively increases / decreases the flow rate in channels, based on the maximum temperature observed during the last measurement interval.

Variable-Proactive utilizes the ARMA predictor for temperature forecasting, and increases the flow rate proactively based on the forecast. Hence, if temperature is projected to increase in the next interval, we increase the flow rate by one step further, and decrease the flow rate similarly if temperature is decreasing.

V. Methodology

The 3D multicore systems we use in our experiments are based on the UltraSPARC T1 (i.e., Niagara-1) processor [20], which is manufactured using 90nm technology. The average power consumption, area distribution of the units on the chip and the floorplan of UltraSPARC T1 are available in [20]. The architecture is composed of 8 cores with multithreading capability, and a shared L2-cache for every two cores. Inter-core communication is accomplished by a shared memory infrastructure.

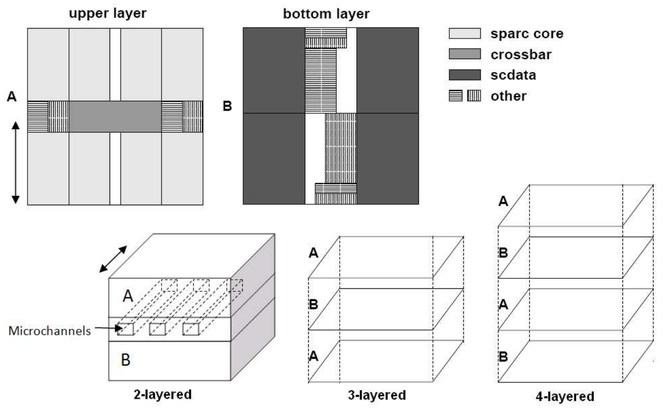

Fig. 5. Floorplans of the 3D Systems.

The simulations are carried out with 2-, 3- and 4-layered stack architectures. A typical approach to design the 3D system is to place the logic units (i.e., the processing cores) and the memory blocks (i.e., caches, etc.) on separate layers. Placing cores and their associated memories on separate layers is a preferred scenario for systems with a large number of memory accesses, such as systems targeting multimedia applications. In this way, the length of interconnections between the cores and their caches can be reduced, achieving higher performance. Such an architecture also allows the use of different technologies for manufacturing the cores and memories, and can result in more optimized designs. Thus, we place cores and L2 caches (i.e., scdata) of the UltraSPARC T1 on separate layers (see Figure 5).

The first step to construct our experimental framework is gathering detailed workload characteristics of real applications on an UltraSPARC T1. We sampled the utilization percentage for each hardware thread at every second using `mpstat`. During this profiling, we recorded half an hour long traces for each benchmark. Also, the length of user and kernel threads were recorded using `DTrace` [21] to determine the active/idle time slots of cores more accurately.

We have used various real-life benchmarks including web servers, database management, and multimedia processing. The web server workload was generated by SLAMD [25] with 20 and 40 threads per client to achieve medium and high utilization, respectively. For database applications, we experimented with MySQL using `sysbench` for a table with 1 million rows and 100 threads. We also ran the `gcc` compiler and the `gzip` compression/decompression benchmarks as samples of SPEC-like benchmarks. Finally, we ran several instances of the `mplayer` (integer) benchmark with 640x272 video files as typical examples of multimedia processing. A detailed summary of the benchmarks workloads is shown in Table II. The utilization ratios are averaged over all cores throughout the execution. We also recorded the cache misses and floating point (FP) instructions per 100K instructions using `cpustat`. The workload statistics collected on the UltraSPARC T1 was replicated for the 16-core systems with 3 and 4 stacked layers.

The peak power consumption of SPARC is close to its average power [20]. Thus, we assumed that the instantaneous power consumption is equal to the average power at each state (active, idle, sleep). The `active` state power is taken as 3 Watts, based on [20]. The cache power consumption is 1.28W per each L2, which is computed with CACTI [27], and verified by the values in [20]. We modeled

TABLE II. WORKLOAD CHARACTERISTICS

	Benchmark	Avg Util (%)	L2 I-Miss	L2 D-Miss	FP instr
1	Web-med	53.12	12.9	167.7	31.2
2	Web-high	92.87	67.6	288.7	31.2
3	Database	17.75	6.5	102.3	5.9
4	Web & DB	75.12	21.5	115.3	24.1
5	gcc	15.25	31.7	96.2	18.1
6	gzip	9	2	57	0.2
7	MPlayer	6.5	9.6	136	1
8	MPlayer&Web	26.62	9.1	66.8	29.9

TABLE III. THERMAL MODEL AND FLOORPLAN PARAMETERS

Parameter	Value
Die Thickness (one stack)	$0.15mm$
Area per Core	$10mm^2$
Area per L2 Cache	$19mm^2$
Total Area of Each Layer	$115mm^2$
Convection Capacitance	140 J/K
Convection Resistance	0.1 K/W
Interlayer Material Thickness	0.02 mm
Interlayer Material Thickness (with channels)	0.4 mm
Interlayer Material Resistivity (without TSVs)	0.25 mK/W

the crossbar power consumption by scaling the average power value according to the number of active cores and the memory accesses. Except for the crossbar, we did not explicitly model the on-chip interconnects in this work.

The leakage power of the processing cores is calculated according to different structural areas of the system and their temperature. We assume a base leakage power density of $0.5W/mm^2$ at 383K as in [4]. To account for the temperature and voltage effects on leakage power, we used the second-order polynomial model proposed in [26]. We empirically determined the coefficients in the model to match the normalized leakage values shown in [26].

Many current systems have power management capabilities to reduce the energy consumption. Even though the power management techniques do not directly address temperature, they affect the thermal behavior considerably. We implement `Dynamic Power Management (DPM)`, especially to investigate the effect on thermal variations. We utilize a fixed timeout policy, which puts a core to sleep state if it has been idle longer than the timeout period. We set a `sleep` state power of 0.02 Watts, which is estimated based on `sleep` power of similar cores.

As noted earlier, HotSpot Version 4.1 [24] has been used as the thermal modeling tool. We used the 3D capability available in the grid model of the tool, utilizing the chip layouts discussed earlier. The thermal simulations were run with a sampling interval of 100 ms, which provided sufficient precision. HotSpot was initialized with steady state temperature values. The model parameters are provided in Table III. Modeling methodology for the interlayer material to include TSVs and the microchannels has been described in Section III.

We perform a comparison between a 3D system with conventional cooling and a system with microchannel cooling. For the conventional system, the default package characteristics in HotSpot V.4.1. were used, as these represent a modern CPU package.

The management policies we propose utilize temperature measurements from the die. We assume that each core has a temperature sensor, which is able to provide temperature readings at regular intervals of 100ms. Modern OSes have a multi-queue structure, where each CPU core is associated with a dispatching queue, and the job scheduler allocates the jobs to the cores according to the current policy. We implement a similar infrastructure, where the queues maintain the threads allocated to cores and execute them.

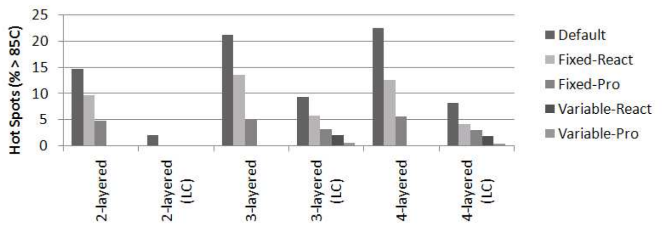

Fig. 6. Thermal Hot Spots for Conventional and Liquid Cooling Methods.

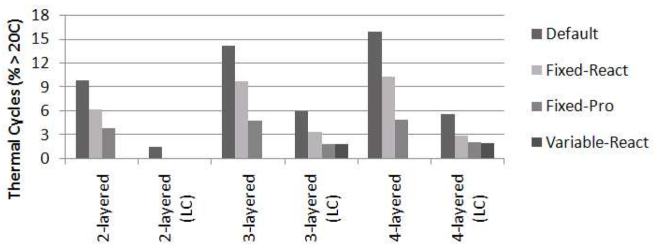

Fig. 7. Temporal Thermal Variations.

VI. EXPERIMENTAL RESULTS

In this section, we evaluate the benefits of liquid cooling in comparison to conventional heat-sink based cooling solutions. We also show how the active temperature management policies integrated with the cooling mechanisms in 3D systems impact the thermal behavior. The experiments including liquid cooling are labeled as *LC* in all the plots, and the rest of the results assume a conventional heat-sink based cooling. The *Default* policy refers to the temperature results obtained by directly using the workload traces collected on UltraSPARC T1, without applying any thermal management policy. Note that the system we experimented with runs a Solaris scheduler which applies a performance-aware load balancing policy.

The first set of results shows the percentage of time for which thermal hot spots (i.e., above 85^oC) are observed. Figure 6 presents the hot spot profiles for all policies and for various number of layers in the 3D stack. For the experiments with conventional cooling, the Fixed-Reactive and Fixed-Proactive policies refer to applying the management policies, but there is no liquid cooling available. Also, for these systems, the variable flow rate policies do not exist.

We observe that, for the 2-layered MPSoC, liquid cooling eliminates all the hot spots. Therefore, it is not necessary to implement the infrastructure for variable flow rate on the chip. However, as the number of layers increases, there is a substantial increase in temperature, and the control-loop including variable flow rate provides additional reduction in thermal hot spots.

Both the 3- and 4-layer configurations have the same number of cores integrated in the 3D stack and similar thermal profiles. However, in the 4-layer configuration there is an extra layer of coolant, which explains the slightly better results due to the heat removal occurred in these microchannels.

We also investigate the effect of policies on temporal and spatial thermal variations, which also have adverse affects on system reliability. All the experiments shown in Figures 7 and 8 utilize DPM, which increases the thermal variations on the chip due to the difference in temperature between the sleep and normal power consumption modes. For temporal variations, we demonstrate the frequency of thermal cycles with magnitude above 20^oC, and we report spatial gradients over 15^oC.

Similar to thermal hot spots, thermal cycles becomes more visible

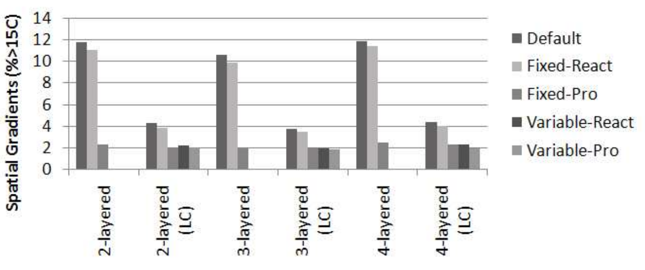

Fig. 8. Intralayer Spatial Gradients.

for higher number of layers. This increase in the magnitudes of cycles is due to the increase in temperature (see Figure 7). The liquid cooling mechanism is able to reduce this impact. *Variable-Proactive* eliminates all large thermal cycles and is not shown in the figure.

Figure 8 provides the frequency of intralayer spatial gradients observed on the chip. Even with *Variable-Proactive*, it is not possible to fully prevent large gradients. This suggests that the granularity of the channels is not enough for perfectly balancing the temperature across the die. The vertical gradients are not considered in this experiment, as we have observed that the vertical gradients between two adjacent layers is limited to a few degrees at most. Still, liquid cooling further reduces the inter-layer differences due to the low thermal resistivity in the channels.

Note that by using a sufficiently high (fixed) flow-rate, such as $25ml/min$ and above in our experiments, potentially it would be possible to further improve the thermal behavior. However, designing a microchannel infrastructure to support high flow rates increases the cost of 3D design. Therefore, developing combined liquid cooling and dynamic management policies is an attractive solution as the dynamic management comes at negligible design cost, and reduces the cooling burden on the microchannels.

Our experimental results have shown that the implementation of microchannels for liquid cooling is an effective mechanism to reduce the hot spots in 3D architectures, and liquid cooling also smooths out temperature temporally and spatially. Our proactive dynamic management approach, which utilizes closed-loop control, can provide an additional 75% average reduction in hot spots in comparison to applying only liquid cooling. For systems capable of applying variable coolant flow rate, the improvement reaches 95% in average. We have observed that such gains become more significant as number of layers increase in the 3D stack.

VII. Conclusion

3D design reduces communication delay between the functional units. However, integrating more than two layers in the 3D chip exacerbates the reliability and thermal problems, and cooling becomes a limiting factor. Liquid cooling is a solution to overcome the accelerated thermal problems imposed by the multi-layer architecture. This paper has presented a novel thermal simulation model including liquid cooling support for both fixed and variable liquid injection rates. We propose two dynamic management policies to complement the liquid cooling, and evaluate them on 3D stacked systems based on UltraSPARC T1. The first policy assumes a fixed flow rate, and uses proactive job scheduling to minimize and balance the temperature on the die, decreasing the frequency of the hot spots by 75% in average, in comparison to applying only liquid cooling. The second one is able to dynamically adjust the liquid flux to achieve a uniform distribution of temperature on the 3D stack of the multicore SoC, decreasing the frequency of the hot spots by 95% on average.

References

[1] D. Atienza, P. D. Valle, G. Paci, F. Poletti, L. Benini, G. D. Micheli, and J. M. Mendias. A fast HW/SW FPGA-based thermal emulation framework for multi-processor system-on-chip. In *DAC*, 2006.
[2] A. Bhunia, K. Boutros, and C.-L. Che. High heat flux cooling solutions for thermal management of high power density gallium nitride HEMT. In *Inter Society Conference on Thermal Phenomena*, 2004.
[3] B. Black, M. Annavaram, N. Brekelbaum, J. DeVale, L. Jiang, G. H. Loh, D. McCaule, P. Morrow, D. W. Nelson, D. Pantuso, P. Reed, J. Rupley, S. Shankar, J. Shen, and C. Webb. Die stacking (3d) microarchitecture. In *MICRO 39: Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 469–479, 2006.
[4] P. Bose. Power-efficient microarchitectural choices at the early design stage. In *Keynote Address on PACS*, 2003.
[5] D. Brooks and M. Martonosi. Dynamic thermal management for high-performance microprocessors. In *HPCA*, pages 171–182, 2001.
[6] T. Brunschwiler and et al. Direct liquid-jet impingement cooling with micron-sized nozzle array and distributed return architecture. In *ITHERM*, 2006.
[7] T. Brunschwiler and et al. Interlayer cooling potential in vertically integrated packages. *Microsyst. Technol.*, 2008.
[8] A. K. Coskun, T. Rosing, and K. Gross. Proactive temperature balancing for low-cost thermal management in MPSoCs. In *ICCAD*, 2008.
[9] A. K. Coskun, T. S. Rosing, K. A. Whisnant, and K. C. Gross. Static and dynamic temperature-aware scheduling for multiprocessor socs. *IEEE Transactions on VLSI*, 16(9):1127–1140, Sept. 2008.
[10] J. Donald and M. Martonosi. Techniques for multicore thermal management: Classification and new exploration. In *ISCA*, 2006.
[11] M. Gomaa, M. D. Powell, and T. N. Vijaykumar. Heat-and-Run: leveraging SMT and CMP to manage power density through the operating system. In *ASPLOS*, 2004.
[12] M. Healy and et al. Multiobjective microarchitectural floorplanning for 2-d and 3-d ICs. *IEEE Transactions on CAD*, 26(1), Jan 2007.
[13] S. Heo, K. Barr, and K. Asanovic. Reducing power density through activity migration. In *ISLPED*, pages 217–222, 2003.
[14] Hs3d thermal modeling tool. http://www.cse.psu.edu/ link/hs3d.html.
[15] W.-L. Hung, G. Link, Y. Xie, N. Vijaykrishnan, and M. Irwin. Interconnect and thermal-aware floorplanning for 3d microprocessors. In *ISQED*, pages 98–104, 2006.
[16] P. Kapur, G. Chandra, and K. Saraswat. Power estimation in global interconnects and its reduction using a novel repeater optimization methodology. In *DAC*, pages 461–466, 2002.
[17] J.-M. Koo, S. Im, L. Jiang, and K. E. Goodson. Integrated microchannel cooling for three-dimensional electronic circuit architectures. *Journal of Heat Transfer*, 2005.
[18] A. Kumar, L. Shang, L.-S. Peh, and N. K. Jha. HybDTM: a coordinated hardware-software approach for dynamic thermal management. In *DAC*, pages 548–553, 2006.
[19] H. Lee and et al. Package embedded heat exchanger for stacked multi-chip module. In *Transducers, Solid-State Sensors, Actuators and Microsystems*, 2003.
[20] A. Leon and et al. A power-efficient high-throughput 32-thread SPARC processor. *ISSCC*, 2006.
[21] R. McDougall, J. Mauro, and B. Gregg. Solaris Performance and Tools. *Sun Microsystems Press*, 2006.
[22] K. Puttaswamy and G. H. Loh. Thermal herding: Microarchitecture techniques for controlling hotspots in high-performance 3d-integrated processors. In *International Symposium on High Performance Computer Architecture (HPCA)*, pages 193–204, 2007.
[23] M. Ruggiero, A. Guerri, D. Bertozzi, F. Poletti, and M. Milano. Communication-aware allocation and scheduling framework for stream-oriented multi-processor system-on-chip. In *DATE*, pages 3–8, 2006.
[24] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. Temperature-aware microarchitecture. In *ISCA*, 2003.
[25] SLAMD Distributed Load Engine. www.slamd.com.
[26] H. Su and et al. Full-chip leakage estimation considering power supply and temperature variations. In *ISLPED*, 2003.
[27] D. Tarjan, S. Thoziyoor, and N. P. Jouppi. CACTI 4.0. Technical Report HPL-2006-86, HP Laboratories Palo Alto, 2006.
[28] D. B. Tuckerman and R. F. W. Pease. High-performance heat sinking for VLSI. *IEEE Electron Device Letters*, 5:126–129, 1981.
[29] Y. Zhang, X. S. Hu, and D. Z. Chen. Task scheduling and voltage selection for energy minimization. In *DAC*, pages 183–188, 2002.
[30] C. Zhu, Z. Gu, L. Shang, R. P. Dick, and R. Joseph. Three-dimensional chip-multiprocessor run-time thermal management. *IEEE Transactions on CAD*, 27(8):1479–1492, August 2008.