Dynamic Thermal Management in Many-Core Systems Leveraged by Abstract Modeling

Alzemiro Silva*, Iaçanã Weber*, André Luís del Mestre Martins†, Fernando Gehm Moraes*
*School of Technology - PUCRS – Av. Ipiranga 6681, 90619-900, Porto Alegre, Brazil
†Sul-Rio-Grandense Federal Institute, IFSul, Charqueadas, Brazil
{iacana.weber, alzemiro.silva}@edu.pucrs.br, almmartins@charqueadas.ifsul.edu.br, fernando.moraes@pucrs.br

***Abstract*—For years, transistor size reduction led to a linear decrease in power dissipation. This is the Dennard scaling law, which ended in 2000's technology nodes because the supply voltage no longer scales. The consequence is the increase of the power density in integrated circuits, leading to high temperatures, accelerating aging effects. Dynamic thermal management (DTM) is a technique adopted at runtime to act in the system using the components' current temperature to minimize hotspots and peak temperatures. This paper aims to present a DTM for NoC-based many-cores, using an abstract model, to estimate the temperature at the processing element level and task migration as the main actuation mechanism. Results show up to 12% of temperature reduction and almost 10°C of peak temperature reduction, highlighting the approach's effectiveness, compared to the pattering and spiral mappings.**

***Index Terms*—Many-core, NoC, Dynamic Thermal Management, Mapping.**

I. Introduction

From the beginning of the integrated circuit age, transistors scaled every new generation. Scaling down the supply voltage with the transistor size makes it possible to maintain the power density [1]. However, in the early 2000s, the transistor scaling could not sustain Dennard's law, leading to an increased power density for new transistors' generations. The increase in power density motivated the use of power-aware techniques such as power gating [2], clock gating [3], and dynamic voltage and frequency scaling (DVFS). Even though those techniques brought benefits regarding energy efficiency, it is impossible to keep all hardware blocks powered on simultaneously at maximum speed, an effect named dark silicon [4, 5].

Dark silicon effects defy IC designers to overcome the increasing design cost in new technology nodes in such a way to sustain performance gains for each new transistors' generations [6]. Hence, effective and efficient power and thermal management techniques are gaining momentum, especially for keeping the performance gain with the technology scaling, without incurring high cooling costs and avoiding overheating and hotspots [7]. Dynamic Thermal Management (DTM) is required for dealing with thermal limits. DTM relies on system temperatures monitoring to decide at runtime when to trigger actuation knobs to keep the system below thermal limits while attending a specific goal [8].

The evaluation of DTM heuristics requires accuracy in capturing thermal data to produce confident results and simulation time for periods long enough to observe transient and steady temperature behaviors. Simulations at the RTL level can provide trustful thermal data, but their simulation time is not scalable. In contrast, abstract models allow simulations for long periods. Thus, this work has two main *goals*. The first one is to present the *Chronos* platform, which abstractly models an NoC-based many-core with an instruction-cycle accuracy by using power characterization at the RTL level. The second goal is to apply a DTM heuristic for large-scale systems (64 cores) and perform a comparison of the proposed heuristics with static mapping techniques, such as patterning (dark silicon technique [9, 10]) and spiral (performance-driven heuristic [11]).

The *original contribution* of this paper comprises the proposal and evaluation of a DTM technique for NoC-based systems, using RTL level power characterization. The adoption of a virtual platform enables long simulation periods, in the orders of seconds, compared to milliseconds at the RTL level. Results show that temperature monitoring allied with task migration and DVFS allows keeping the temperature below the designer's predefined limits.

II. Related Works

Singh et al. [12] survey dynamic energy and thermal management for mobile platforms, presenting the challenges for multi-core systems. Works discussed in this Section target NoC-based many-core systems, where the core count increases the DTM complexity.

The first challenge in DTM is to measure or estimate the temperature at runtime. Per-core thermal monitoring is the key for DTM development to manage applications entering and leaving the system dynamically. Although recent proposed DTMs [10, 13] employ analytical thermal models to predict per-core temperature by profiling the applications at design time, these DTMs prevent thermal monitoring due to the high computational costs and the data-dependency algorithms. Another alternative to avoid thermal monitoring is patterning the many-core to map applications [9]. Patterning mapping may lead to either resources underutilization (excessive number of dark cores) or communication performance degradation to meet power and temperature requirements. The DTM herein proposed can deal with the unpredictability inherent in applications' dynamic behavior because the thermal monitoring supports our DTM decisions and does not rely on average data from pre-execution profiling or static mapping pattern.

To enable thermal monitoring in DTMs, temperature sensors are not an option for per-core thermal monitoring or validation purposes due to their size and limited number. Even considering global monitoring, the sensors' noise is a topic of discussion [14]. To face the inherent errors of thermal models and improve DTM benefits, Chen et al. [15] propose an adaptive thermal model to predict the temperature dynamically according to machine-learning algorithms. Another strategy to support thermal monitoring is a neural network-based online thermal estimation based on performance counters [16]. Alternatively, our RC thermal parameters are extracted from a synthesizable VHDL version of the reference platform to overcome the difficulty of validating thermal models and reduce thermal model errors.

Concerning thermal monitoring in DTMs, an adaptation of an RC thermal model [17] aims to mitigate the data-dependency for enabling a distributed thermal management [18]. Each application belongs to a contiguous cluster of cores and has an agent that communicates between other agents to update the cluster borders temperatures and calculate the cluster/application temperature. Per-core thermal monitoring is also a requirement for a thermal-aware management for reliability purposes [19]. Our approach employs a hardware accelerator for fast computation of per-core temperature based on power and performance counters [20].

Khdr et al. [18] propose a distributed thermal management running task-to-core mapping where cores can have their voltage/frequency levels set, and applications can resize at runtime to avoid thermal violations. Haghbayan et al. [19] propose a DTM for reliability purposes by coordinating task mapping, task-to-core scheduling, and per-core DVFS to increase the lifetime of shared-memory many-cores. Our DTM employs a PID-inspired heuristic, considering instantaneous, average, and trend temperature data for taking decisions. Task migration is the coarse-grain actuation mechanism responsible for moving tasks from hot PEs (Processing Elements) to colder PEs. DVFS is the fine-grain actuation policy for keeping PEs under the thermal threshold.

III. Chronos - Virtual Platform Description

The reference of the virtual platform is an RTL model of an heterogeneous many-core [21]. This RTL platform allows collecting performance data, as performance and power, from system characterization. This work adopts the set of tools provided by Imperas (OVP – Open Virtual Platforms [22]) to generate the abstract model of the reference platform.

OVP adopts a quantum-based simulation paradigm, which divides the time into discrete time steps. The number of instructions that each PE execute in a step is called *quantum*. The quantum size is parameterizable, and its size affects the simulation performance and the synchronization between PEs. Each PE simulates sequentially by one quantum. Once every processor executes a quantum, the simulation advances to the next quantum. Summarizing, this paradigm simulates parallelism by alternating each processor's execution. Note that there is a trade-off between the quantum size and the communication synchronism. A large quantum delays the communication between PEs, and a small quantum increases the simulation time. Experimentally, we tuned the quantum to 250 instructions, a value that brought the abstract platform behavior closer to the RTL behavior.

OVP does not provide support for NoCs but provides an API to create peripherals that can be connected to processors. Thus, according to the RTL behavior, we modeled the router and a NI (Network Interface) as a peripheral. As OVP peripherals

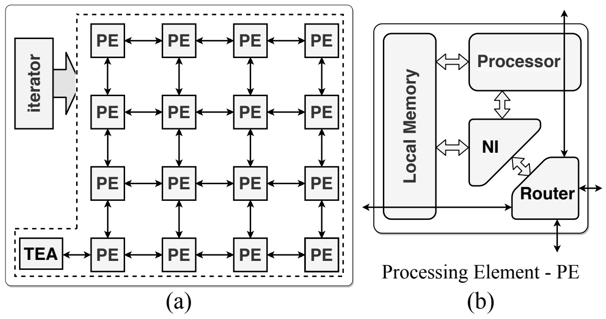

Fig. 1. (a) 4x4 Chronos instance; (b) main PE components.

do not have their execution associated with the quantum size, they are triggered by callbacks executed atomically. The virtual platform received a peripheral called $iterator$ to synchronize the communication between processors at the end of each quantum execution. The $iterator$ evaluates all routers sequentially with flits to transmit, sending one flit to the next router or local port. This process stops when there are no more flits to transmit. After this condition, a new quantum starts.

Figure 1 presents a 4x4 instance of Chronos, which contains PEs connected to a 2D-mesh and peripherals attached in the NoC borders. Each PE has a processor, local scratchpad memory, NI, and router. TEA (Temperature Estimation Accelerator) [23] is a peripheral that periodically receives the power samples from PEs, computes each PE temperature, and sends it to the thermal manager PE, responsible for taking actions to preserve the system TDP (Thermal Design Power). The estimation model presented in the peripheral is a simplified version of the MatEx [24] model.

For generating the power samples, the platform has been instrumented to report which instruction is being executed at each fetch cycle, classifying them into ten classes (branch, arithmetic, jump, move, load, store, shift, nop, logical, and multiplication/division). Each PE periodically generates its power sample by multiplying the amount of executed instructions per class by its pre-characterized power consumption. Besides, memory and router power are also considered, creating the PE power sample sent to TEA to perform the temperature estimation. In the current version, each PE sends its power sample directly to TEA. Larger systems may adopt a hierarchical distribution, where PEs send their power sample to a hub-PE that will concentrate several PEs data in only one packet per hub-PE, reducing the communication with TEA.

After receiving the temperature estimation from TEA, the thermal manager PE executes the DTM. The platform support two actuation mechanisms: DVFS and task migration [25]. The processor characterization for a set of voltage-frequency (VF) pairs enables DVFS support. Chronos supports DVFS by dynamically adjusting the quantum size, simulating in this way changes in the processor frequency. For each VF pair, there is a set of pre-characterized power values. Chronos also supports clock gating, by turning off the instruction counting. Task migration includes suspending the execution in a given PE, restarting it in another PE. Support for task migration is performed through checkpoints, which are points where the task status is stored and transferred to the target processor.

IV. DTM

Temperature-aware mapping aims to balance the temperature distribution in multi-core architectures with available temperature sensors [26, 27]. However, mapping decisions using only the instant temperature can impact the thermal distribution of the system. For example, if a task is mapped in a PE with the lowest temperature, but its temperature is increasing, the resulting peak temperature can be higher than if mapped in other PE with decreasing temperature. For this reason, we proposed a first version of the Proportional, Integral, and Derivative Temperature Management (PIDTM) in [20]. In this work, PIDTM included DVFS, enabling to act on PEs that exceeds a predetermined temperature threshold. The main goal of the PIDTM algorithm is to improve the thermal balance of the system proactively by monitoring the instant temperatures. The DVFS actuation keeps the PE's temperatures below a threshold defined at design time.

The *Proportional* value is related to the instantaneous temperature of the PEs, the *Integral* value is the average temperature value of a predefined number of monitoring windows, and the *Derivative* value is the tendency of the temperature value, and means how fast the temperature is changing. The goal of using the *Derivative* value is to guide mapping and migration decisions, using the temperature tendency of a specific PE. The adoption of the temperature tendency allows the heuristics to avoid PEs that are increasing their temperature or prioritize PEs that are decreasing their temperature. The *Integral* value enables to balance the workload between all PEs since the heuristic avoids PEs that accumulates high average temperatures.

Algorithm 1 presents part of the PIDTM heuristic, responsible for defining a vector with the priorities for task migration, *mapping_prio*. The function is called when there is a new temperature message available, and the input *time_idx* corresponds to the number of times the function is called. There is a circular buffer for each PE, named *integral_buf*, which stores the last INT_WINDOW temperature samples. Line 2 receives the current instantaneous temperature values computed by TEA. The loop between lines 3 and 10 updates the PID score for each PE. Lines 4 and 5 insert the current PE temperature in the circular buffer, while line 6 computes the average PE temperature in the last INT_WINDOW intervals - *integral* value. Line 7 computes the *derivative* value by subtraction the previous temperature from the new temperature value. Line 8 updates the PE PID score, multiplying each value by a constant that defines each component's weight. KP, KI, and KD are proportional, integral, and derivative constants. It is possible to tune those constants according to the control objectives. Using the new PID scores, line 11 sorts the PEs to create the *mapping_prio* vector. Line 12 invokes the task migration, which migrates tasks having their scores above a threshold (TH_{MIG}), respecting an interval between migrations (to avoid a "ping-pong" effect).

The mapping and migration procedures consider virtual regions, named *quadrants*, to minimize the hop-count between tasks, avoiding tasks belonging to the same application spread in the system.

Algorithm 1 PIDTM

```
1: **Inputs**: $time\_idx$
2: $PE\_temps \leftarrow$ **receive_packet**($TEMP\_PERIF$)
3: **for each** $PE_i \in PE\_temps$ **do**
4:     $last \leftarrow time\_idx$ **mod** $INT\_WINDOW$
5:     $integral\_buf_i(last) \leftarrow PE_i.temp$
6:     $integral_i \leftarrow \frac{\sum_{n=1}^{INT\_WINDOW} integral\_buf_i(n)}{INT\_WINDOW}$
7:     $derivative_i \leftarrow PE_i.temp - temperature\_prev_i$
8:     $pid\_score_i \leftarrow KP * PE_i.temp + KI * integral_i + KD * derivative_i$
9:     $temperature\_prev_i \leftarrow PE_i.temp$
10: **end for**
11: $mapping\_prio \leftarrow$ **generateMapPrio**($pid\_score$)
12: **migration**($mapping\_prio$)
```

The migration procedure first verifies if a PE has temperature higher than TH_{MIG} to trigger a task migration. If the number of PEs having temperatures above TH_{MIG} is higher than available PEs to receive tasks, the hottest PEs are selected for migrating. The DVFS is fired when the PE temperature is above TH_{DVFS}, reducing the VF pair. If the temperature is below TH_{DVFS} and the PE is not operating at the maximum VF pair, the VF may be increased. PEs running no tasks compute only the router power, and the processor is considered powered off.

V. DTM Results

We compare our proposed DTM heuristic with a chessboard patterning mapping, which is adopted to improve temperature distribution, and with a spiral mapping [11], that aims to place tasks of the same application next to each other to reduce communication latency and improve performance. Both approaches only adopt DVFS for respecting thermal limits.

Applications use MPI-like communication and are described in C language. We used 7 applications: (*i*) DIJ (7 tasks), Dijkstra algorithm; (*ii*) AV (7 tasks), audio and video decoding; (*iii*) AES (5 tasks), encryption algorithm; (*iv*) Sort (5 tasks), parallel quick sort; (*v*) MPEG (5 tasks), image decoder; (*vi*) DTW (6 tasks), Digital Time Warping algorithm; (*vii*) SYN (6 tasks), a communication intensive synthetic benchmark. The temperature monitoring window is set to 1 ms.

Table I presents the four evaluation scenarios, running in an 8x8 many-core, with four 4x4 quadrants. The AW scenario seeks to use the maximum number of PEs in the patterning mapping. The HW presents a high workload along the simulation time, with more than one application executing simultaneously at each quadrant. Scenarios DW1 and DW2 present dynamic workloads, with a mixed set of long and short execution time applications entering the system at different moments. The number of simultaneously running tasks varies in DW1 and DW2 to present peaks and valleys of system utilization.

Figure 2 compares the peak temperature. The peak temperature is the highest temperature a core achieved during the execution. The Spiral mapping algorithm is the reference heuristics because the temperature is not the primary cost-function. In the patterning approach, the temperature is also not an input of the algorithm, but this approach is usually a reference to produce a better thermal distribution.

TABLE I
THERMAL EVALUATION SCENARIOS.

Scenario		Applications							
		MPEG	DTW	AES	SYN	DIJ	Sort	AV	#Tasks
AW	Average Workload	1	1	0	1	1	1	1	36
HW	High Workload	1	2	0	1	1	2	1	47
DW1	Dynamic workload 1	1	2	10	4	4	2	2	142
DW2	Dynamic workload 2	2	6	10	4	4	6	2	192

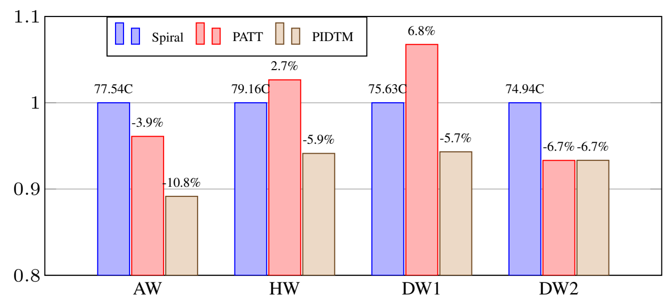

Fig. 2. Spiral, Patterning and PIDTM peak temperature results.

We observed that the patterning mapping, in some scenarios, can produce a peak temperature higher than the spiral mapping but still producing a better thermal distribution with a lower average temperature. The higher peak temperature happens because applications may have a dominant thread, which consumes more power than the other tasks. The spiral mapping usually maps applications far away from each other, while the patterning mapping may map two dominant tasks near to each other, increasing peak temperature. In contrast, the PIDTM heuristic avoids the mapping of dominant tasks near to each other. If this happens, the DTM identifies the temperature increase and migrate one of the tasks.

PIDTM effectively reduced the peak temperature in all scenarios. In AW, we observed a 10.8% peak temperature reduction related to the spiral mapping algorithm, which represents more than 8°C of peak temperature reduction. The HW scenario penalized the patterning mapping, while DTM presented a peak temperature reduction. In DW1, we observed a 12,7% temperature reduction in PIDTM related to the patterning approach, representing almost 10°C of peak temperature reduction. DW2 is the worst-case scenario for PIDTM, with a result equal to the patterning approach. This situation may happen because two dominant tasks have not been placed next to each other in the patterning mapping. Even though PIDTM just matched the patterning algorithm in DW2, its execution time and the average temperature was lower.

This first evaluation shows that the *patterning approach might not be a good option for temperature aware-mapping*. On the other side, an approach using monitoring and actuation, as PIDTM, is a path to follow to reduce peak temperature, improving the system reliability and lifetime.

Figure 3 compares the peak temperature between the PIDTM heuristic and the patterning mapping for HW and DW1 scenarios. We observe that the peak temperature is significantly lower with the proposed heuristic (10°C). The better thermal distribution obtained by using PIDTM causes a lower peak temperature most of the time. This behavior obtained with the PIDTM heuristic can potentially increase

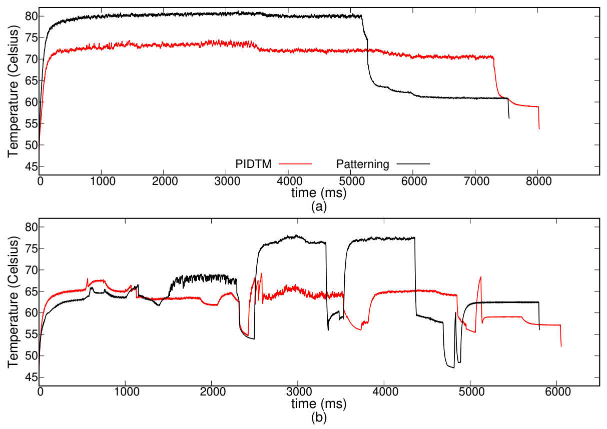

Fig. 3. Comparison between peak temperatures in 2 scenarios: (a) High Workload (HW); (b) Dynamic Workload (DW1).

the chip's lifetime and reliability and open temperature room for more applications to execute.

In addition to temperature monitoring, we also measured the applications' execution time. On average, applications mapped by the spiral algorithm performed 10.2% faster than those mapped by PIDTM. This loss is expected since the spiral algorithm's goal is to approximate communicating tasks, reducing the hop-count and the NoC congestion. On the other hand, compared with the patterning, which has the characteristic of spreading the tasks in a pre-defined pattern to decrease the chip's heat concentration, the PIDTM difference on execution time was minimal (0.01%). These results show that PIDTM has a low impact on application performance while improving thermal distribution.

VI. Conclusion

This paper presented DTM results for the PIDTM algorithm, with results obtained using a virtual platform model. Accuracy was a concern during modeling, with a platform calibration using data from the RTL level. The virtual platform model allowed long simulations, enabling to observe the system behavior during the steady-state temperature operation of the system. Results showed that the proposed heuristic can improve significantly the thermal distribution of the system when compared with a spiral mapping and with a patterning approach. We were able to achieve up to 10.8% in peak temperature reduction without significant performance penalties.

This work showed how important is fine-grain temperature monitoring, i.e., at the PE level, and a constant actuation on the system (migration and DVFS in our work) to keep the temperature at safe levels. Approaches as patterning, using only DVFS, are not thermal efficient with high workloads.

Future works includes: the development and evaluation of new heuristics to provide better DTM approaches, the study of the reliability and lifetime impact using the proposed DTM techniques, as well as the implementation of time sharing multitasking in Chronos platform for further temperature analysis.

Acknowledgment

The authors would like to thank Imperas Software and Open Virtual Platforms for their support and access to their models and simulator. This work was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) – Finance Code 001, and CNPq (grant 309605/2020-2).

References

[1] R. H. Dennard, F. H. Gaensslen, H.-N. Yu, V. L. Rideout, E. Bassous, and A. R. LeBlanc, "Design of ion-implanted MOSFET's with very small physical dimensions," *IEEE Journal of Solid-State Circuits*, vol. 9, no. 5, pp. 256–268, 1974.

[2] Z. Hu, A. Buyuktosunoglu, V. Srinivasan, V. V. Zyuban, H. M. Jacobson, and P. Bose, "Microarchitectural techniques for power gating of execution units," in *ISLPED*, 2004, pp. 32–37.

[3] Q. Wu, M. Pedram, and X. Wu, "Clock-gating and its application to low power design of sequential circuits," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 3, pp. 415–420, 2000.

[4] H. Esmaeilzadeh, E. Blem, R. S. Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in *ISCA*. IEEE, 2011, pp. 365–376.

[5] M. Shafique, S. Garg, J. Henkel, and D. Marculescu, "The EDA challenges in the dark silicon era: Temperature, Reliability, and Variability perspectives," in *DAC*, 2014, pp. 1–6.

[6] J. Henkel, H. Khdr, S. Pagani, and M. Shafique, "New trends in dark silicon," in *DAC*, 2015, pp. 1–6.

[7] S. Pagani, J.-J. Chen, M. Shafique, and J. Henkel, *Advanced Techniques for Power, Energy, and Thermal Management for Clustered Manycores*. Springer, 2018.

[8] M. El Ahmad, M. Najem, P. Benoit, G. Sassatelli, and L. Torres, "PoETE: A Method to Design Temperature-Aware Integrated Systems," *Journal of Low Power Electronics*, vol. 14, no. 1, pp. 1–7, 2018.

[9] X. Wang, A. K. Singh, and S. Wen, "Exploiting dark cores for performance optimization via patterning for many-core chips in the dark silicon era," in *NOCS*, 2018, pp. 1–8.

[10] W. Liu, L. Yang, W. Jiang, L. Feng, N. Guan, W. Zhang, and N. D. Dutt, "Thermal-aware Task Mapping on Dynamically Reconfigurable Network-on-Chip based Multiprocessor System-on-Chip," *IEEE Transactions on Computers*, vol. 67, no. 12, pp. 1818–1834, 2018.

[11] N. Bansal, S. Gupta, N. Dutt, A. Nicolau, and R. Gupta, "Network topology exploration of mesh-based coarse-grain reconfigurable architectures," in *DATE*, 2004, pp. 474–479.

[12] A. K. Singh, S. Dey, K. McDonald-Maier, K. R. Basireddy, G. V. Merrett, and B. M. Al-Hashimi, "Dynamic energy and thermal management of multi-core mobile platforms: A survey," *IEEE Design & Test*, vol. 37, no. 5, pp. 25–33, 2020.

[13] M. Li, W. Liu, L. Yang, P. Chen, and C. Chen, "Chip Temperature Optimization for Dark Silicon Many-core Systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 5, pp. 941–953, 2018.

[14] F. Terraneo, A. Leva, and W. Fornaciari, "An Open-Hardware Platform for MPSoC Thermal Modeling," in *SAMOS*, 2019, pp. 184–196.

[15] K.-C. J. Chen and Y.-H. Liao, "Adaptive Machine Learning-Based Temperature Prediction Scheme for Thermal-Aware NoC System," in *ISCAS*, 2020, pp. 1–4.

[16] M. Rapp, O. Elfatairy, M. Wolf, J. Henkel, and H. Amrouch, "Towards NN-based Online Estimation of the Full-Chip Temperature and the Rate of Temperature Change," in *MLCAD*, 2020, pp. 95–100.

[17] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "HotSpot: A compact thermal modeling methodology for early-stage VLSI design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 5, pp. 501–513, 2006.

[18] H. Khdr, M. Shafique, S. Pagani, A. Herkersdorf, and J. Henkel, "Combinatorial Auctions for Temperature-Constrained Resource Management in Manycores," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 7, pp. 1605–1620, 2020.

[19] M.-H. Haghbayan, A. Miele, Z. Zouv, H. Tenhunen, and J. Plosila, "Thermal-cycling-aware dynamic reliability management in many-core system-on-chip," in *DATE*. IEEE, 2020, pp. 1229–1234.

[20] A. L. da Silva, A. Martins, , and F. G. Moraes, "Mapping and Migration Strategies for Thermal Management in Many-Core Systems," in *SBCCI*, 2020, pp. 1–6.

[21] M. Ruaro, L. Caimi, V. Fochi, and F. G. Moraes, "Memphis: a Framework for Heterogeneous Many-core SoCs Generation and Validation," *Design Automation for Embedded Systems*, vol. 23, no. 3-4, pp. 113–122, 2019.

[22] Imperas, "Open Virtual Platforms - OVP," 2019. [Online]. Available: http://www.ovpworld.org/

[23] A. H. L. da Silva, A. L. D. M. Martins, and F. G. Moraes, "Fine-grain temperature monitoring for many-core systems," in *SBCCI*, 2019, p. 4.

[24] S. Pagani, H. Khdr, W. Munawar, J. Chen, M. Shafique, M. Li, and J. Henkel, "MatEx: Efficient transient and peak temperature computation for compact thermal models," in *DATE*, 2015, pp. 1515–1520.

[25] M. Ruaro and F. G. Moraes, "Demystifying the Cost of Task Migration in Distributed Memory Many-core Systems," in *ISCAS*, 2017, pp. 1–4.

[26] R. Rao and S. Vrudhula, "Efficient Online Computation of Core Speeds to Maximize the Throughput of Thermally Constrained Multi-core Processors," in *ICCAD*, 2008, pp. 537–542.

[27] A. K. Coskun, T. T. Rosing, K. Whisnant, and K. C. Gross, "Static and Dynamic Temperature-aware Scheduling for Multiprocessor SoCs," *IEEE Trans. Very Large Scale Integrated Systems*, vol. 16, no. 9, pp. 1127–1140, 2008.