MonTM: Monitoring-Based Thermal Management for Mixed-Criticality Systems

Marcel Mettler
Chair of Electronic Design Automation, Technische Universität München, Germany

Martin Rapp
Chair for Embedded Systems, Karlsruhe Institute of Technology, Germany

Heba Khdr
Chair for Embedded Systems, Karlsruhe Institute of Technology, Germany

Daniel Mueller-Gritschneder
Chair of Electronic Design Automation, Technische Universität München, Germany

Jörg Henkel
Chair for Embedded Systems, Karlsruhe Institute of Technology, Germany

Ulf Schlichtmann
Chair of Electronic Design Automation, Technische Universität München, Germany

Abstract

With a rapidly growing functionality of embedded real-time applications, it becomes inevitable to integrate tasks of different safety integrity levels on one many-core processor leading to a large-scale mixed-criticality system. In this process, it is not sufficient to only isolate shared architectural resources, as different tasks executing on different cores also possibly interfere via the many-core processor's thermal management. This can possibly lead to best-effort tasks causing deadline violations for safety-critical tasks. In order to prevent such a scenario, we propose a monitoring-based hardware extension that communicates imminent thermal violations between cores via a lightweight interconnect. Building on this infrastructure, we propose a thermal strategy such that best-effort tasks can be throttled in favor of safety-critical tasks. Furthermore, assigning static voltage/frequency (V/f) levels to each safety-critical task based on their worst-case execution time may result in unnecessary high V/f levels when the actual execution finishes faster. To free the otherwise wasted thermal resources, our solution monitors the progress of safety-critical tasks to detect slack and safely reduce their V/f levels. This increases the thermal headroom for best-effort tasks, boosting their performance. In our evaluation, we demonstrate our approach on an 80-core processor to show that it satisfies the thermal and deadline requirements, and simultaneously reduces the run-time of best-effort tasks by up to 45% compared to the state of the art.

2012 ACM Subject Classification Hardware → On-chip resource management; Computer systems organization → Embedded and cyber-physical systems

Keywords and phrases Dynamic thermal management, mixed-criticality, monitoring

Digital Object Identifier 10.4230/OASIcs.PARMA-DITAM.2023.5

Funding This work was partly funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Projektnummer 146371743 – TRR 89 "Invasive Computing".

1 Introduction

New applications such as autonomous driving increase the complexity for modern embedded real-time systems. Hence, it becomes increasingly challenging to reconcile functional requirements with non-functional requirements such as cost, weight, power consumption and heat generation. In order to still meet both, functional and non-functional requirements, there is an increasing trend in industry and academia to integrate tasks of different safety integrity

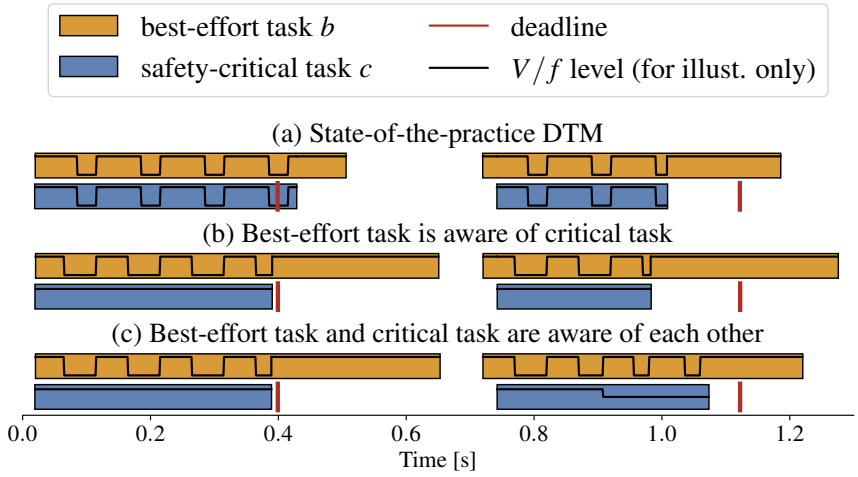


Figure 1 The temporal behavior of two periodic applications that run concurrently on a many-core processor prototype (a) using a state-of-the-practice DTM (b) using a DTM that is aware of neighboring safety-critical tasks, and (c) using a DTM that is aware of neighboring best-effort and safety-critical tasks.

levels (SILs) in one mixed-criticality system (MCS) [3]. As a result, either all tasks must be developed according to the highest SIL or tasks of different SILs must be isolated to meet the requirements of the safety standards [1]. In practice, it is too expensive to develop all tasks according to the highest SIL since large systems comprise only few safety-critical tasks and many best-effort tasks [7].

A prominent solution to provide the required isolation are virtualization technologies [5]. However, on many-core processors, an isolation of architectural resources, such as processing elements (PEs), caches, buses, etc. is not sufficient. Different tasks can not only interfere via shared resources but also via the many-core processor's thermal manager. In the following motivational example, we demonstrate that this interference can lead to deadline violations and propose potential solutions and optimizations.

1.1 Motivational Example

This example analyzes the impact of the thermal interference between a best-effort task b and a safety-critical task c. Both tasks are executed periodically on neighboring cores of a tiled many-core processor, with per-core dynamic voltage frequency scaling (DVFS). In Fig. 1, we evaluate the measured run-times of both tasks for three different scenarios, where c is first executed along a timing-critical path through its control flow graph (CFG) and subsequently along a non-critical path.

In Scenario (a), each tile uses a state-of-the-practice hardware dynamic thermal manager (DTM) [9]. The DTM monitors the temperature of all cores on a tile and reacts on a thermal violation by throttling down the voltage/frequency (V/f) level. Once the tile temperature decreased, the V/f levels are reset. In this example, the tasks running on neighboring cores of different tiles cause thermal violations, which require to throttle both tasks periodically as shown in the figure. In this case, c does not meet its deadline when it operates along a timing-critical path.

In Scenario (b), the DTM is aware of the SIL of c. As a result, it does not only throttle b on a local thermal violation but also on a thermal pre-error, i.e. an imminent thermal violation, of c. Hence, the DTM prevents thermal interference by downscaling the V/f level of

b more frequently so that no thermal violations occur on the tile of c, which guarantees that c does not miss its deadline. However, the price for this guarantee is an increased run-time of b.

As this technique is overly pessimistic when the run-time of c is significantly shorter than the worst-case execution time (WCET), we present a Scenario (c) where the DTMs of c and b mutually interact. In this scenario, the DTM adjusts the V/f level of c based on the progress of the task. As soon as the critical task progresses along a non-critical path through its task graph, the DTM reduces the V/f level accordingly. This decreases the power consumption of c, thereby increasing the thermal budget of b, such that the DTM of b is triggered less frequently. Consequently, this reduces the run-time of b compared to Scenario (b).

1.2 Challenges and Novel Contributions

A major challenge in the design of a thermal manager for a many-core MCS is the thermal coupling between different cores that execute tasks of different criticality. As this coupling decreases with an increase of the distance between the cores, it is crucial to especially throttle those best-effort task that are located in the neighborhood of a safety-critical task with an imminent thermal violation. To solve this challenge an interconnect for the DTMs is required to communicate the monitored thermal status of the safety-critical tasks. A second challenge is that the actions must be applied immediately when the temperature of the cores change to avoid thermal violations. To solve this challenge, a hardware implementation for the DTM is required since it is faster than a software solution. This is also the state of the practice in the industry [9]. Furthermore, the execution time of a safety-critical task is unknown at design-time. Therefore, monitors that evaluate the slack of safety-critical tasks are required to enable a safe reduction of their V/f level. In this work, we present a monitoring-based thermal manager, called MonTM consisting of one DTM per core. MonTM aims to maximize the performance of best-effort tasks in MCSs guaranteeing no thermal violations for safety-critical tasks. It is based on two novel components reflecting the two key contributions of this paper:

- A hardware-based thermal management strategy for MCSs that prevents best-effort tasks from inducing thermal violations into safety-critical tasks. For that purpose, MonTM uses a novel interconnect to communicate the thermal status of safety-critical tasks. Thereby, the DTMs are able to choose a V/f level for best-effort tasks that avoids thermal violations on cores running safety-critical tasks without leading to unnecessary performance losses.
- A hardware-based slack monitor that determines the minimal V/f requirement of safety-critical tasks based on their current progress. This enables the DTMs in scenarios with a high slack to safely reduce the V/f level, which increases the available thermal headroom.

2 Related Work

Resource management strategies mainly apply DVFS to reduce the power consumption to a thermally safe level. They range from reactive to proactive strategies. A reactive technique is Intel's Turbo Boost [2], which upscales the V/f level if the current, the power and the temperature are below a predefined threshold and downscales it otherwise. A drawback of reactive DTMs is that it is not predictable. Hence, it is not possible to give timing guarantees for safety-critical tasks at design time, since the DTM can be triggered at any point at run-time (see Scen. (a) in our motivational example).

For a more predictable behavior, proactive power budgeting can be employed. In the most basic form, the thermal design power (TDP) is used to allocate a static power budget to each core of the chip. As this budget does not consider the activity of cores, thermal safe

power (TSP) [17] has been proposed, which determines the maximal power budget for each task based on the worst-case mapping scenario. To additionally consider the actual mapping of the tasks, power density-aware resource management (PdRM) [10] can be employed. Greedy based dynamic power budgeting (GDP) [21] and T-TSP [14] additionally consider the transient temperature of the cores, enabling an increased power budget for cold cores. Finally, distributed power budgeting (DBP) [22] uses the local temperature model of each core to compute the power budgets in a decentralized fashion. A major drawback of power budgeting is that the core frequency must be determined by the maximal power consumption to prevent thermal violations. As a result, a high variance in their power consumption results in an under-utilization of the power budget. While some works instead control the V/f levels re-actively such that the power consumption of a task stays within the assigned budget, this may cause thermal violations in two ways. First, the controller cannot react infinitely fast on fluctuations of the power consumption such that the task may exceed its power budget. Secondly, thermal violations may occur even if the power budgets have been proven to be thermally safe in the steady-state as shown in [16].

Besides the general literature on thermal management, there also exist scheduling works for MCSs that additionally consider the thermal behavior of the processor. A popular approach is to use DVFS to either reduce the chip temperature by minimizing the average power consumption [20] or to fulfill the TDP requirement by reducing the peak-power consumption [18]. A more recent thermal management method has been employed in [19] where the scheduling strategy uses the concept of TSP to fulfill the thermal requirements. A common shortcoming of these works is that they rely on periodic application models. However in practice, best-effort tasks, such as the infotainment system, need to be executed on demand and not periodically. Furthermore, the literature on general thermal management has shown that the TDP constraint does not guarantee to prevent thermal violations [17] and that TSP is overly pessimistic compared to the state of the art in power budgeting [21, 10].

To the best of our knowledge, this is the first work that combines the predictability advantages of power budgeting with the run-time advantages of reactive thermal management for MCS.

3 MonTM Thermal Management Strategy

3.1 Problem Formulation

Given a many-core MCS, we differentiate between safety-critical tasks and best-effort tasks. Safety-critical tasks must be mapped to an exclusive resource PE_i to guarantee their schedulability at any time. As safety-critical tasks are typically subject to timing requirements, we model their service level agreements (SLAs) by a tuple (C_i, D_i), where C_i corresponds to the WCET using the maximal frequency $f_{PE_i,max}$ of its exclusive resource PE_i and D_i to the deadline. Best-effort tasks, such as the infotainment system, are typically not subject to timing requirements and, therefore, do not require a specific application model. They can be executed on any available PE that is not reserved for a safety-critical task. Furthermore, we allow both safety-critical and best-effort tasks to be scheduled on demand. As the underlying platform, we consider a network on chip (NoC)-based many-core processor with per-PE DVFS on which all tasks are executed. Our objective is to maximize the performance of the best-effort tasks, i.e. to minimize their execution time, under the constraint that all safety-critical tasks meet their deadline. Hence, we need to carefully balance the V/f level of best-effort tasks to ensure that they do not induce thermal violations into safety-critical tasks.

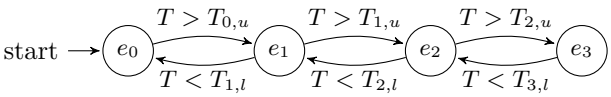


Figure 2 Thermal pre-error final state machine (FSM).

3.2 Dynamic Thermal Manager for Safety-Critical Tasks

For a worst-case scenario, the upper bound of the minimal frequency requirement of a safety critical task can be computed by its WCET C_i and its deadline D_i.

$$ub(f_{i,min}) = \frac{C_i}{D_i} f_{PE_i,max} \tag{1}$$

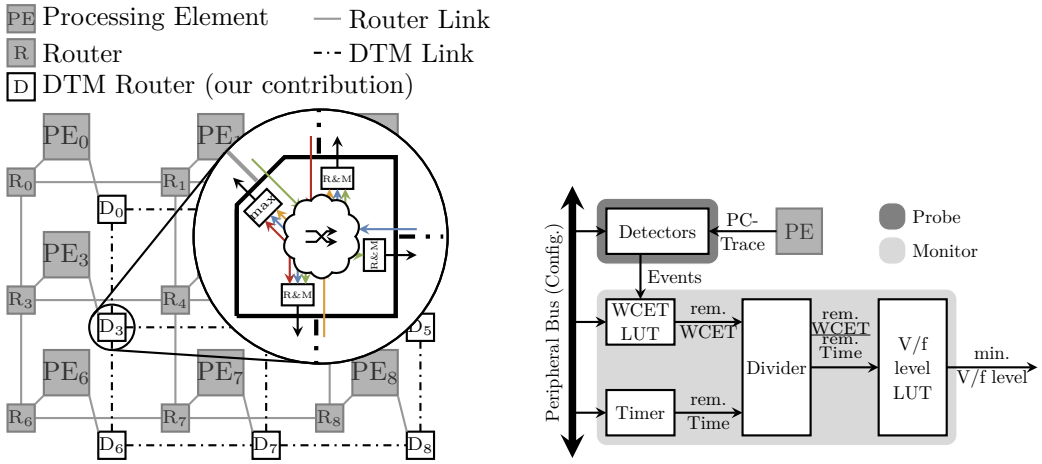
While $ub(f_{i,min})$ is especially accurate for compute-bound tasks, it can also be used for memory-bound tasks, where the minimal frequency is even lower [4]. To be able to guarantee this upper bound, two conditions must be fulfilled: the first condition is a general prerequisite for MCSs. It must be possible to run the combination of all safety-critical tasks (in absence of the best-effort tasks) such that all deadlines can be met. To verify this condition, methods from power budgeting can be used. In this process, we model the power budget of each core running a safety-critical task by its peak-power consumption and the power budget of all others cores by the static power consumption at the lowest V/f level. Using the thermal RC-thermal model, which is commonly applied in state-of-the-art thermal simulators [16], it is then possible to compute the remaining temperature headroom of each core according to Eq. 2, where T_{head}, $T_{chip,max}$ and $T_{amb,max}$ are vectors storing the headroom, the maximal chip and the maximal ambient temperature for each component of the chip, B is a matrix storing the thermal conductances between the components and P is a vector describing the power budget of each component.

$$T_{head} = T_{chip,max} - T_{amb,max} - B^{-1}P \tag{2}$$

If T_{head} is positive for all components on the chip, the condition is fulfilled. Otherwise, the safety-critical load on the system is too high and it cannot be guaranteed that all deadlines are met.

The second condition for this objective is that safety-critical and best-effort tasks maintain a sufficient thermal isolation. In this process, the thermal isolation is sufficient if and only if no best-effort task induces a thermal violation in a safety-critical task. To ensure this condition, we propose a DTM interconnect that communicates the thermal pre-error e of a safety-critical task to neighboring best-effort tasks such that these can be throttled in favor of the safety-critical task. A pre-error indicates that a thermal violation is imminent. We define several levels of urgency, ranging from e_0 to e_3. To determine the urgency, we use a simple final state machine (FSM), as illustrated in Fig. 2, which increases the urgency of the thermal pre-error once the temperature T exceeds the temperature bounds $T_{0,u}$, $T_{1,u}$ and $T_{2,u}$, and decreases the urgency once T falls below $T_{1,l}$, $T_{2,l}$ and $T_{3,l}$. For the communication of thermal pre-errors between the DTMs, we use the DTM interconnect, illustrated in Fig. 3a. Here, we use a reduce and maximize (R&M) intellectual property (IP) block at each output port of the router to first reduce the pre-error level according to Eq. 3 and subsequently to forward the maximal pre-error. Here, e_r corresponds to the reduced thermal pre-error and e_i to the thermal pre-error of the input channel i.

$$e_r = \begin{cases} e_i & \text{if } i = \text{local} \vee e_i \in \{e_0, e_3\} \\ e_i - 1 & \text{otherwise} \end{cases} \tag{3}$$



(a) The NoC-based DTM interconnect, integrated in a many-core processor.

(b) Hardware design of the slack monitor.

Figure 3 Hardware architecture of MonTM.

Hence, the IP only reduces the thermal pre-error of input ports that are non-local and of input ports where the thermal pre-error is below e_3. This procedure determines the smallest possible number of neighboring best-effort tasks that must be throttled in favor of safety-critical tasks. Given a thermal pre-error of e_0, no throttling of neighboring tasks is required. For e_1, the best-effort tasks within a hop distance of one must be throttled. If this countermeasure is not sufficient and e_2 is reached, the throttled region is further increased by one hop. In a worst-case situation with a thermal pre-error of e_3, all best-effort tasks are halted. Please note that this is an emergency measure that moves the system to the case that only the safety-critical tasks are executed, which are known to be run in combination without thermal violations. As usually there is some thermal headroom available for best-effort tasks in MCSs, this mode should rarely be triggered during operation. Furthermore, it should be noted that the pre-errors are broadcasted according to the XY-routing scheme to prevent a deadlock scenario where e_3 continues to be broadcasted in cycles even though the urgency of the thermal pre-error already decreased.

3.3 Dynamic Thermal Manager for Best-effort Tasks

The objective of best-effort tasks is to minimize their latency without inducing thermal violations into critical tasks. In order to satisfy this requirement, the DTM must implement two actions: First, the reduction of the V/f level of the core to a minimum if it gets informed about a thermal pre-error e_i with $i > 0$. Second, the DTM must additionally halt the core if the thermal pre-error reaches its maximal urgency, i.e. $i = 3$. This strategy ensures that the thermal isolation between best-effort and safety-critical tasks is sufficient to guarantee that all deadlines can be met and additionally enables the best-effort tasks to fully utilize the remaining temperature headroom.

3.4 Slack Monitoring of Safety-Critical Tasks

In average and best-case scenarios of safety-critical tasks, the minimal frequency requirement, presented in Eq. 1, could be further reduced to increase the thermal headroom that is available for best-effort tasks. Therefore, we propose to determine the minimal frequency requirement of safety-critical tasks based on their progress.

This can be achieved using techniques from the field of run-time monitoring [11, 12]. Run-time monitoring is a light-weight verification technique that can be used to monitor system requirements. Thereby, the system under verification is instrumented to extract its current status based on events. The trace of events is then analyzed by a run-time monitor, which either derives a verdict about the current status of the requirement or actions to continue to comply with the requirements. Similarly, we instrument safety-critical tasks at specific points of interest and measure the remaining WCET for each of the points. As accurate static WCET analysis of multi- and many-core processors is still an open research problem [15], we employ a measurement-based WCET and add a safety-margin. In practice, it is advisable to instrument the application especially after branching points in the CFG. Thus, it is possible to identify at run-time whether the task chooses the worst-case path through the CFG or not and to optimize the V/f level accordingly.

Fig. 3b illustrates the hardware architecture used for this monitoring approach. The approach consists of a probe to non-intrusively instrument the task at run-time and a monitor to update the frequency requirement. The detectors in the probe are configured using the program counter (PC) addresses of the points of interest and their respective identification (ID). Hence, a detector that matches the PC trace with the PC address of a point of interest sends an event with the respective ID to the run-time monitor. Here, a lookup table (LUT) is used to identify the remaining WCET based on the ID of the detected point of interest. Furthermore, the run-time monitor comprises a countdown timer, which issues the remaining time until the deadline of the task is reached. Together, the remaining WCET and the time remaining until the deadline can be used to compute the frequency requirement using a divider. As the implementation of a divider is expensive in terms of area, it is also possible to share a single divider between multiple monitors. Finally, a V/f level LUT translates the frequency requirement into the minimal V/f level of the task.

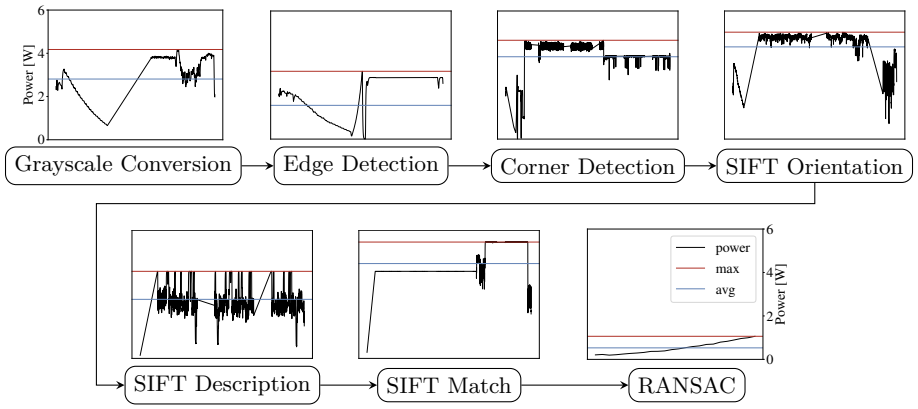
Now, the DTMs of safety-critical tasks have the choice between the original frequency requirement, presented in Eq. 1, and the potentially lower frequency requirement from the proposed monitoring approach. The monitored frequency requirement is only advantageous over the original frequency requirement, if the released thermal budget is actually used by best-effort tasks. Otherwise, it is advantageous to follow a race to idle strategy with the original frequency requirement to minimize future overlaps between best-effort and safety-critical tasks. To account for this trade-off at run-time, we additionally extend the DTM-interconnect by a second physical layer (identical to the layer in Fig. 3a) , which communicates the activity of best-effort tasks to all DTMs within a hop-distance of two. Thus, the DTM of safety-critical tasks can use the original frequency requirement if there is no best-effort task in its direct neighborhood and otherwise use the potentially lower frequency requirement of the monitoring approach.

4 Experimental Results

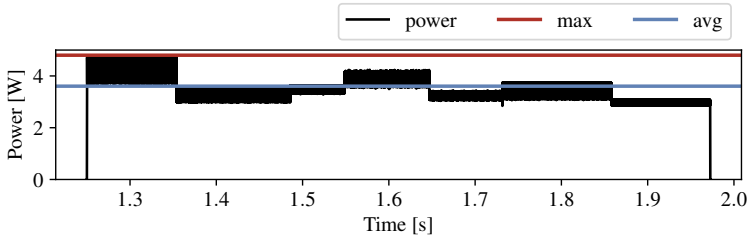
In this section, we present the experimental setup followed by evaluations of the MonTM approach.

4.1 Experimental Setup

For the following evaluations, we implement a field programmable gate array (FPGA) prototype of a tiled many-core processor with an application specific integrated circuit (ASIC) target technology of 14 nm and a target frequency of 4 GHz on a proFPGA platform [6] consisting of four Virtex-7 FPGAs. The processor consists of 16 tiles, which are connected via



(a) Object detection chain containing of multiple actors.



(b) Synthetic application.

Figure 4 Power consumption measured by the emulator on the FPGA prototype.

a NoC. Each tile implements five LEON3 cores from which one is reserved for the operating system (OS), a shared 512 kB L2 cache for remote tile memory accesses and an 8 MB tile-local memory. We emulate the ASIC behavior of the processor, similar as e.g. proposed in [13]. Thereby, the ASIC power monitor of the cores is emulated based on an instruction-level power model, which has been obtained by gate-level simulations using the Synopsys tool PrimePower. For the temperature monitor, we fit an ASIC temperature emulator based on the RC-thermal network, which we obtained from the thermal simulator MaTex [16]. Both, the power and the temperature emulators update their output every 256 cycles. Finally, we also emulate DVFS on a per-core basis, which supports emulated frequencies in multiples of 100 MHz from a minimum of 1.0 GHz up to a maximum of 4.0 GHz. As phase-locked loop (PLL) locktime, we use 2 us based on [8]. The DTM is implemented according to the proposed thermal management techniques, presented in Sec. 3. As a base technique for the best-effort task, we use a DTM, which reacts on a temperature rise to T_{crit} by throttling down the V/f level of the core to a minimum value. Once the core temperature decreases below a lower thermal threshold, the V/f level of the core is reset to its peak value.

4.2 Workload Modelling

In order to model real world workloads and load the 80 cores, we generated a library of code blocks as proposed in [13] with different run-time properties in terms of cache access and floating-point instruction rates. Those code blocks are then embedded into random CFGs, consisting of one to 16 code blocks. In order to demonstrate that the generated workloads mimic the power consumption of real-world workloads sufficiently, we compare their power consumption with the power consumption of an object detection chain, consisting of multiple independent actors, where each actor implements an algorithm from the field of computer vision.

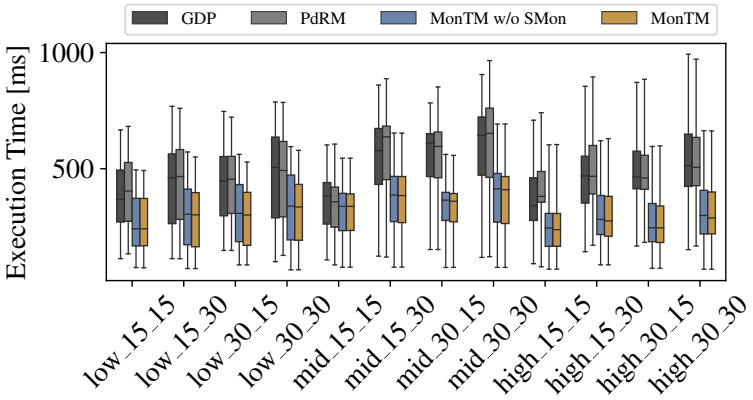
The actor graph of the application and the respective power traces are illustrated in Fig. 4a. The traces show that the power consumption of all algorithms varies significantly. As a result, thermal management techniques that exclusively consider the maximal power consumption of a task, as it is done for power budgeting [10, 21], do not utilize the full thermal headroom of the chip. In comparison to that, Fig. 4b illustrates the power consumption of a generated workload. The trace demonstrates that the individual code blocks show a uniform behavior in their power consumption. However, with the combination of multiple code blocks into one task, we are able to model a variance in the power consumption and generate different load scenarios for the 80-core system. Even though it is not possible to exactly replicate the power consumption of the object detection chain with the synthetic applications, the comparison shows that synthetic benchmarks can be used as a conservative replication since the state of the art profits from a low variance in the power consumption.

4.3 Comparison to the State of the Art

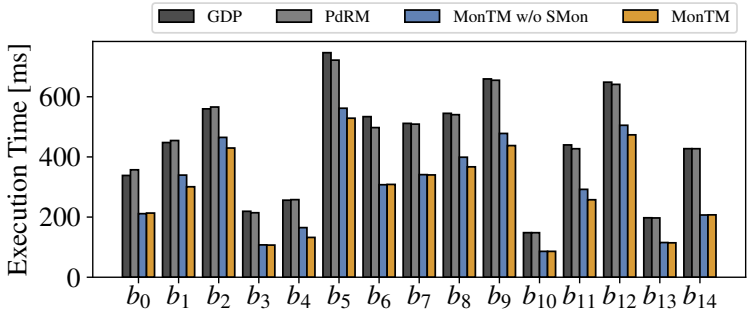
We compare MonTM with and without slack monitoring to the state-of-the-art algorithms GDP [21] and PdRM [10] (described in Sec. 2). Even though these methods have not been designed for MCSs, both are more competitive than the state of the art for MCSs that still relies on TDP, which does not guarantee to prevent thermal violations, or TSP, which has been shown to be less competitive than GDP and PdRM. As MonTM does not rely on a specific mapping of the best-effort tasks, we use the same random mapping for all methods. Furthermore, we set the V/f level of the safety-critical tasks based on Eq. 1 for GDP and PdRM as well. In our implementation of GDP and PdRM, this is reflected in a fixed power budget for the safety-critical tasks.

For our evaluations, we generate various use cases by varying the load on the system and the variance of the power consumption. We influence the load of the system by the number of best-effort tasks N_b and safety-critical tasks N_c that we run in the use case. Here, we use $N_b \in \{15, 30\}$ and $N_c \in \{15, 30\}$. To influence the variance in the power consumption, we construct the CFGs of best-effort and safety-critical tasks out of code blocks with $P_{i,max} \in [5.0\ W, 5.2\ W]$ for a low variance, with $P_{i,max} \in [4.0\ W, 5.2\ W]$ for a medium variance and with $P_{i,max} \in [3.0\ W, 5.2\ W]$ for a high variance. The name of the use case forms from the used configuration according to $< var(P) >\_< N_c >\_< N_b >$.

As all techniques satisfy the thermal requirements of the chip and the deadline requirements of the safety-critical tasks, Fig. 5a presents the execution times of the best-effort tasks as a boxplot. Here, the box marks the upper and lower quantile, the bar in the plot the median and the whiskers the maximal and minimal execution time. It can be seen that the median of the execution time for all techniques increases with the load on the system. Furthermore, both MonTM without slack monitoring and MonTM with slack monitoring outperform the state-of-the-art techniques for all uses cases. This observation is also valid across the best-effort tasks within the benchmarks as MonTM also reduces the extrema and the quantiles of the execution times. While GDP and PdRM rely on the peak-power consumption of the tasks, MonTM can fully exploit the available thermal headroom and thereby reduce the average run-time by 7%-44% without slack monitoring. In addition, the slack monitor further reduces the average run-time of the best-effort tasks by another 1%-6%. Here, the reduction is limited by the fact that not all best-effort tasks overlap with nearby safety-critical tasks and that not all CFGs consist of multiple execution paths. By analyzing the individual execution times of the best-effort tasks in the low_30_15 use case in Fig. 5b, this observation can be confirmed. The execution time of some best-effort tasks can be reduced by an additional 12%.



(a) Average execution time of the best-effort tasks for different use cases.



(b) The execution time of the best-effort tasks for the low_30_15 use case.

Figure 5 Execution times of the best-effort tasks measured on the FPGA prototype.

4.4 Overhead

Finally, we evaluate the run-time and the hardware overhead of MonTM. Tab. 1 presents the average and the maximal run-time overhead of MonTM compared to PdRM and GDP for the use cases with a low variance in the power consumption. It should be noted that the other use cases show similar overheads as the run-time overhead is independent of the variance in the power consumption. It can be seen that the run-time overhead of MonTM, required for the configuration of the hardware, is neglectable compared to the state of the art. The main reason for that is that the technique is purely implemented in hardware. In contrast to that, GDP introduces the largest overhead due to its large computational complexity. As the maximal run-time overhead is even of the same order of magnitude than the execution times of some tasks, GDP is not suited for the studied MCS use cases.

Table 1 The measured run-time overhead of MonTM per load change.

	PdRM [10]		**GDP** [21]		**MonTM**	
	avg.	max.	avg.	max.	avg.	max.
low_15_15	292 µs	316 µs	15 ms	30 ms	3 µs	8 µs
low_15_30	309 µs	354 µs	22 ms	73 ms	3 µs	8 µs
low_30_15	296 µs	325 µs	17 ms	42 ms	4 µs	8 µs
low_30_30	314 µs	364 µs	24 ms	92 ms	4 µs	8 µs

Table 2 The resource consumption of MonTM relative to the implementation of the FPGA prototype.

	Slice LUTs		Slice Registers	
	abs.	rel.	abs.	rel.
Router	101	$< 0.1\%$	208	0.2%
Thermal Manager	70	$< 0.1\%$	60	$< 0.1\%$
Progress Monitor	1465	1.0%	3176	3.3%
Probe	356	0.2%	830	0.9%
Total	1997	1.3%	4274	4.4%

Tab. 2 presents the absolute and relative hardware overhead for each component of MonTM for one tile. With a total overhead of 1.3% in terms of slice LUTs and 4.4% in terms of slice registers, the hardware overhead is well justified considering the performance improvements that MonTM offers.

5 Conclusion

In this paper, we presented MonTM, a monitoring-based thermal management strategy for MCSs. MonTM uses a DTM interconnect to communicate the thermal pre-error of safety-critical tasks. Hence, it is possible to throttle best-effort tasks in favor of safety-critical tasks. Furthermore, MonTM uses a slack monitor to monitor the minimal V/f requirement of safety-critical tasks based on their progress and their deadline. Thereby, the DTMs are able to safely reduce the frequency of critical tasks in order to increase the thermal budget of best-effort tasks. In our evaluation, we show that MonTM reduces the average run-time of best-effort tasks by up to 45% compared to the state of the art without violating thermal and deadline requirements.

References

1 IEC SC 65A. Functional safety of electrical/electronic/programmable electronic safety-related systems. Technical Report IEC 61508, The International Electrotechnical Commission, Geneva, Switzerland, 1998.

2 J. Casazza. Intel turbo boost technology in intel core microarchitecture (nehalem) based processors. Technical report, Intel Corporation, November 2008.

3 Hongxia Chai, Gongxuan Zhang, Jin Sun, Ahmadreza Vajdi, Jing Hua, and Junlong Zhou. A review of recent techniques in mixed-criticality systems. *Journal of Circuits, Systems and Computers*, 28(07):1930007, 2019.

4 Kihwan Choi, R. Soma, and M. Pedram. Dynamic voltage and frequency scaling based on workload decomposition. In *International Symposium on Low Power Electronics and Design (ISLPED)*, 2004.

5 M. Cinque, D. Cotroneo, L. De Simone, and S. Rosiello. Virtualizing mixed-criticality systems: A survey on industrial trends and issues. *Future Gener. Comput. Syst.*, 129(C):315–330, April 2022.

6 PRO DESIGN. `https://www.profpga.com/products/systems-overview/virtex-7-based/profpga-quad-v7`.

7 R. Ernst and M. Di Natale. Mixed criticality systems – A history of misconceptions? *IEEE Design & Test*, 33(5):65–74, 2016.

8 A. Hoban. Designing real-time solutions on embedded intel architecture processors. Technical report, Intel Corporation, May 2010.

9 Intel. Intel xeon phi processor 7250, 2016. URL: `https://ark.intel.com/content/www/us/en/ark/products/94035/intel-xeon-phi-processor-7250-16gb-1-40-ghz-68-core.html`.

10 H. Khdr, S. Pagani, É. Sousa, V. Lari, A. Pathania, F. Hannig, M. Shafique, J. Teich, and J. Henkel. Power density-aware resource management for heterogeneous tiled multicores. *IEEE Transactions on Computers*, 66(3):488–501, 2017.

11 Martin Leucker and Christian Schallhart. A brief account of runtime verification. *The Journal of Logic and Algebraic Programming*, 78(5):293–303, 2009. The 1st Workshop on Formal Languages and Analysis of Contract-Oriented Software (FLACOS'07).

12 M. Mettler, D. Mueller-Gritschneder, and U. Schlichtmann. A distributed hardware monitoring system for runtime verification on multi-tile mpsocs. *ACM Trans. Archit. Code Optim.*, 18(1), December 2021.

13 M. Mettler, M. Rapp, H. Khdr, D. Mueller-Gritschneder, J. Henkel, and U. Schlichtmann. An fpga-based approach to evaluate thermal and resource management strategies of many-core processors. *ACM Trans. Archit. Code Optim.*, 19(3), May 2022.

14 Sobhan Niknam, Anuj Pathania, and Andy D. Pimentel. T-tsp: Transient-temperature based safe power budgeting in multi-/many-core processors. In *International Conference on Computer Design (ICCD)*, 2021.

15 Vincent Nélis, Patrick Meumeu Yomsi, and Luís Miguel Pinho. Methodologies for the wcet analysis of parallel applications on many-core architectures. In *2015 Euromicro Conference on Digital System Design*, pages 748–755, 2015.

16 S. Pagani, J. Chen, M. Shafique, and J. Henkel. Matex: Efficient transient and peak temperature computation for compact thermal models. In *Design, Automation Test in Europe Conference Exhibition (DATE)*, 2015.

17 S. Pagani, H. Khdr, J.-J. Chen, M. Shafique, M. Li, and J. Henkel. Thermal Safe Power (TSP): Efficient Power Budgeting for Heterogeneous Manycore Systems in Dark Silicon. *IEEE Transactions on Computers*, 66(1):147–162, 2017.

18 B. Ranjbar, A. Hosseinghorban, M. Salehi, A. Ejlali, and A. Kumar. Toward the design of fault-tolerance-aware and peak-power-aware multicore mixed-criticality systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(5):1509–1522, 2022.

19 S. Safari, H. Khdr, P. Gohari-Nazari, M. Ansari, S. Hessabi, and J. Henkel. Therma-mics: Thermal-aware scheduling for fault-tolerant mixed-criticality systems. *IEEE Transactions on Parallel and Distributed Systems*, 33(7):1678–1694, 2022.

20 Amir Taherin, Mohammad Salehi, and Alireza Ejlali. Reliability-aware energy management in mixed-criticality systems. *IEEE Transactions on Sustainable Computing*, 3(3):195–208, 2018.

21 H. Wang, D. Tang, M. Zhang, S. X.-D. Tan, C. Zhang, H. Tang, and Y. Yuan. Gdp: A greedy based dynamic power budgeting method for multi/many-core systems in dark silicon. *IEEE Transactions on Computers*, 68(4):526–541, 2019.

22 Hai Wang, Wenjun He, Qinhui Yang, Xizhu Peng, and He Tang. Dbp: Distributed power budgeting for many-core systems in dark silicon. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2022.