Shared Resource Contention Aware Schedulability Analysis for Multiprocessor Real-Time Systems

Jatin Arora
Eduardo Tovar
Cláudio Maia

CISTER-TR-221202

2023/04/17

Shared Resource Contention Aware Schedulability Analysis for Multiprocessor Real-Time Systems

Jatin Arora, Eduardo Tovar, Cláudio Maia

CISTER Research Centre

Polytechnic Institute of Porto (ISEP P.Porto)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8321159

E-mail: jatin@isep.ipp.pt, emt@isep.ipp.pt, clrrm@isep.ipp.pt

https://www.cister-labs.pt

Abstract

Multicore platforms share the hardware resources such as caches, interconnects, and main memory among all the cores. Due to such sharing, tasks running on different cores compete to access these shared resources which can potentially result in shared resource contention. This shared resource contention can increase the execution times of tasks in a non-deterministic manner. Consequently, the shared resource contention is problematic for hard real-time systems, i.e., systems that run tasks with stringent timing requirements. To address this issue, this PhD dissertation builds novel solutions to model and analyze the shared resource contention that can be suffered by tasks executing on a multicore system. The shared resource contention aware schedulability analysis is then derived by integrating the maximum shared resource contention that can be suffered by the tasks.

Shared Resource Contention Aware Schedulability Analysis for Multiprocessor Real-Time Systems

DATE PhD Forum 2023

Jatin Arora
CISTER Research Centre, ISEP, IPP
Porto, Portugal
jatin@isep.ipp.pt

Advisor: Eduardo Tovar
CISTER Research Centre, ISEP, IPP
Porto, Portugal
emt@isep.ipp.pt

Co-advisor: Cláudio Maia
CISTER Research Centre, ISEP, IPP
Porto, Portugal
crrm@isep.ipp.pt

Hard real-time systems are special types of embedded systems in which the correctness of the system depends not only on the logical results, i.e., functional behavior, but also at the time on which the results are being produced, i.e., *temporal behavior*. These systems typically run tasks with *stringent timing requirements*, and the consequences of not meeting the given time requirements can be catastrophic. A common example is the car airbag system in which activating the airbag within a given time is important; otherwise, it may have serious negative consequences. Therefore, it is necessary to perform the *schedulability analysis* of such tasks at design time to determine whether tasks will meet their timing constraints or not at run time. The schedulability analysis can be performed using various methods but the *Worst-Case Response Time* (WCRT) based schedulability analysis is the most common approach. Specifically, the WCRT of a task determines the maximum time taken by that task from its release to completion in the worst-case scenario.

Traditionally, most of the systems (including hard real-time systems) were deployed on top of single-core processors, and several single-core processors were deployed to meet the computing power requirements. *Multicore processors* were then introduced that integrate several cores on the same die in order to meet the growing demand for computational power and energy efficiency. This led to a shift towards commercial-off-the-shelf (COTS) multicore platforms as they became a preferable choice for modern systems. However, the adoption of multicore platforms in hard real-time systems is still under scrutiny. The main challenge that hinders the use of COTS multicore platforms in hard real-time systems is their unpredictability, which originates from the sharing of different hardware resources. A task executing on one core of a multicore platform has to compete with other co-running tasks (tasks running on other cores) to access *shared hardware resources* such as the last-level cache (LLC), the interconnect (e.g., memory bus), and the main memory [14]. This competition is problematic as it can negatively influence the temporal behavior of tasks in a non-deterministic manner. This phenomenon is known as the *shared resource contention*. Analyzing the shared resource contention can be extremely complex as it depends on the specific properties of tasks, i.e., the number of shared resource requests, the maximum service time of each request, type of request (read/write) etc., and run-time state of the system and shared resources. This shared resource contention is problematic as it brings temporal unpredictability in the system, which makes the WCET and WCRT analysis very challenging when tasks are running on the COTS multicore platform.

To circumvent this problem, the concept of the *3-phase task execution model* [16, 9, 13] was proposed in the literature. In the 3-phase task execution model, the execution of each task is divided into three phases, namely *Acquisition* (A), *Execution* (E), and *Restitution* (R). The A- and R-phases are considered *memory phases*, i.e., the time intervals in which the task can fetch and write-back data/code from/to the main memory via the memory bus, and the E-phase is the *computation phase*, i.e., the time interval in which the task only performs computations using the preloaded data and does not issue any main memory request. When a task is released, it executes its A-phase to fetch the required data/code from the main memory and store it in cache memory. It then executes its E-phase by accessing the data/code that is already available in the cache, without the need to access bus/memory. Finally, the task writes the modified data back to the main memory during the R-phase. Leveraging such a model, tasks can be scheduled in a manner such that while a task is executing its memory phase, another task on a different core can execute its computation phase concurrently without suffering shared resource contention. Some works [15, 7] proposed frameworks to generate a system-level offline schedule such that no two tasks can execute their memory phases at the same time to avoid the shared resource contention. However, such approaches may not be applicable in some scenarios, e.g., due to the event-triggered/sporadic nature of tasks. Furthermore, scalability is an issue for such approaches, as the system-level offline schedule may not be valid and needs to be reconstructed if some changes take place in the system, e.g., variation in the length of the memory phases, addition of a new task, etc. In the absence of such an offline schedule, tasks running on multiple cores can execute their memory phase at the same time, potentially resulting in shared resource contention. Consequently, a few works [12, 8, 19] have focused on analyzing the shared resource contention for the 3-phase task model. However, these approaches have some limitations. For example, the existing works are either limited to the global scheduling [12, 19] or consider an architecture that provides a point-to-point connection between each core and the main memory [8]. Furthermore, these works [12, 8] analyze each shared resource independently. For example, the work in [12, 19] analyzes *bus contention*, i.e., shared resource contention due to sharing of memory bus, and the work in [8] analyzes *memory contention*, i.e., shared resource contention due to sharing of main memory. Analyzing each shared resource independently can be pessimistic because contention at a given shared resource is interdependent on the behavior of other shared resources. For example, the bus contention (resp. memory contention) also depends on the number of bus requests (resp. memory requests) which in turn also depends on the number of LLC misses. This implies that considering that each memory phase of every task will access all memory blocks from the main memory without analyzing the maximum number of LLC misses can be pessimistic. Therefore, the existing works [12, 8, 19] can overestimate shared resource contention.

To address these limitations of the state-of-the-art in the computation of shared resource contention for the 3-phase task model, this PhD dissertation has the following contributions.

Contribution 1: Bus Contention-Aware WCRT Analysis for the 3-Phase Task Model

In this contribution, the bus contention analysis is proposed for the 3-phase task model considering *partitioned scheduling*. Specifically, we analyze the bus contention analysis for 3-phase tasks considering two different memory access models, i.e., *dedicated* and *fair memory access models*, built on top of the *first-come-first-served (FCFS)* bus arbitration policy. Furthermore, we show that how the bounds on the bus contention can be improved by considering Round-Robin (RR) bus arbitration scheme. The bus contention analysis is then derived for the 3-phase task model considering the RR bus arbitration policy. We also show that if the blocking caused by a lower priority task in the multicore platform is computed in a manner similar to that of the uniprocessors, it can yield unsafe bounds. We then show how to correctly quantify the maximum blocking that can be caused by a lower priority task under the 3-phase task model that execute on a multicore platform. Finally, the *bus contention-aware WCRT analysis* is formulated by integrating the maximum bus contention that can be suffered by tasks. The detailed analysis and results of this contribution are published in several reputed conferences and journals, i.e., [1, 2, 3, 4].

Contribution 2: Cache-Aware Bus Contention Analysis Framework for the 3-phase Task Model

In this contribution, we present a *holistic* overview of the relationship between the memory bus and LLC. We show that the bus contention strongly depends on the LLC misses and considering the worst-case LLC misses without analyzing the LLC may lead to pessimistic bounds on the bus contention. In particular, we use the notion of *cache persistence* [17], i.e., memory blocks that once loaded into the cache by the task can be reused by its subsequent jobs without the need to access the main memory. We then compute the maximum number of LLC misses and integrate it in the bus contention analyses considering various bus arbitration policies. Finally, the maximum bus contention suffered by 3-phase tasks is integrated into their WCRT. Evaluations show that the cache-aware bus contention analyses can provide significantly tighter bounds in comparison to their respective cache-oblivious counterparts. The preliminary results of this contribution is published in [5] and the manuscript containing detailed results is under review.

Contribution 3: Fixed Task-Priority based Memory-Centric Scheduling

A memory-centric scheduler ensures that at most one task access the shared bus/memory at a time to minimize/eliminate the shared resource contention suffered by tasks. The existing memory-centric scheduler considers (i) a TDMA-based memory scheduler [20], i.e., tasks' memory requests are served under a static TDMA schedule, and (ii) Processor-Priority (PP) based memory scheduler [18], i.e., tasks' memory requests are served depending on the priority of the processor/core on which the task is executing. Although these works provide important solutions, they can potentially overestimate the memory interference, e.g., 1) TDMA-based MCS is built on top of TDMA which is a *non-work conserving* policy; 2) PP-based MCS schedule memory accesses on the basis of *priority of cores* and does not take into account task priorities. To address these issues, we propose and analyze a fixed Task Priority (TP) based memory-centric scheduler, i.e., tasks' memory requests are served depending on the priority of the task, which can reduce the shared resource contention in comparison to existing memory schedulers. The detailed analysis and results of this contribution are published in [6].

Contribution 4 (in progress): Memory Contention Aware WCRT analysis for the 3-Phase Task Model

The existing work [8] that analyzes the main memory contention for the 3-phase task model is limited to *in-order-pipeline*, i.e., there can be at most memory request pending at a given core at a time. It has been shown in [21, 11, 10] that there can be architectures with the *out-of-order pipeline*, i.e., multiple memory requests can be pending at a given core at a time. Consequently, there is a need to derive the main memory contention that can be suffered by 3-phase tasks running on such architectures. Furthermore, we realize that in architecture with the out-of-order pipeline, memory address mapping of tasks can have an impact on the memory contention suffered by tasks. Consequently, we analyze the main memory contention for the 3-phase task model considering the out-of-order pipeline and leverage memory address mapping of tasks. Finally, a new WCRT analysis is presented by integrating the maximum memory contention that can be suffered by tasks. The manuscript containing detailed analysis/results will be submitted to a conference/journal in the near future.

Publications: This PhD dissertation has resulted in six publications in which one of the papers received **Best Paper Award** at ICESS 2021. All the papers were published in reputed venues for real-time systems research, i.e., RTSS 2020, RTNS 2021, ICESS 2021, RTCSA 2022, RTSS 2022, Elsevier's Journal of System Architecture. Two more papers are expected to be published soon.

Acknowledgments

This work was supported by the CISTER Research Unit (UIDP/UIDB/04234/2020), financed by National Funds through FCT/MCTES (Portuguese Foundation for Science and Technology); by project ADACORSA (ECSEL/0010/2019 - JU grant nr. 876019) financed through National Funds from FCT and European funds through the EU ECSEL JU. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Austria, Sweden, Spain, Italy, France, Portugal, Ireland, Finland, Slovenia, Poland, Netherlands, Turkey - Disclaimer: This document reflects only the author's view and the Commission is not responsible for any use that may be made of the information it contains. This work is also a result of the work developed under project Aero.Next Portugal (nº C645727867-00000066) and FLY-PT (grant nº 46079, POCI-01-0247-FEDER-046079), also funded by FCT under PhD grant 2020.09532.BD.

References

[1] J. Arora, C. Maia, S. Aftab Rashid, G. Nelissen, and E. Tovar. "Bus-Contention Aware Schedulability Analysis for the 3-Phase Task Model with Partitioned Scheduling". In: *29th International Conference on Real-Time Networks and Systems*. RTNS'2021. NANTES, France: Association for Computing Machinery, 2021, pp. 123–133.

[2] J. Arora, C. Maia, S. A. Rashid, G. Nelissen, and E. Tovar. "Bus-contention aware WCRT analysis for the 3-phase task model considering a work-conserving bus arbitration scheme". In: *Journal of Systems Architecture* 122 (2022), p. 102345.

[3] J. Arora, C. Maia, S. A. Rashid, G. Nelissen, and E. Tovar. "Schedulability analysis for 3-phase tasks with partitioned fixed-priority scheduling". In: *Journal of Systems Architecture* 131 (2022), p. 102706.

[4] J. Arora, C. Maia, S. A. Rashid, G. Nelissen, and E. Tovar. "Work-In-Progress: WCRT Analysis for the 3-Phase Task Model in Partitioned Scheduling". In: *2020 IEEE Real-Time Systems Symposium (RTSS)*. 2020, pp. 407–410.

[5] J. Arora, S. A. Rashid, C. Maia, G. Nelissen, and E. Tovar. "Work-in-Progress: A Holistic Approach to WCRT Analysis for Multicore Systems". In: *2022 IEEE Real-Time Systems Symposium (RTSS)*. 2022, pp. 511–514.

[6] J. Arora, S. A. Rashid, C. Maia, and E. Tovar. "Analyzing Fixed Task Priority Based Memory Centric Scheduler for the 3-Phase Task Model". In: *2022 IEEE 28th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*. 2022, pp. 51–60.

[7] M. Becker, D. Dasari, B. Nicolic, B. Akesson, V. Nélis, and T. Nolte. "Contention-Free Execution of Automotive Applications on a Clustered Many-Core Platform". In: *2016 28th Euromicro Conference on Real-Time Systems (ECRTS)*. 2016, pp. 14–24.

[8] D. Casini, A. Biondi, G. Nelissen, and G. Buttazzo. "A Holistic Memory Contention Analysis for Parallel Real-Time Tasks under Partitioned Scheduling". In: *2020 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*. 2020, pp. 239–252.

[9] G. Durrieu, M. Faugère, S. Girbal, D. Gracia Pérez, C. Pagetti, and W. Puffitsch. "Predictable Flight Management System Implementation on a Multicore Processor". In: *Embedded Real Time Software (ERTS'14)*. TOULOUSE, France, Feb. 2014.

[10] M. Hassan and R. Pellizzoni. "Analysis of Memory-Contention in Heterogeneous COTS MPSoCs". In: *32nd Euromicro Conference on Real-Time Systems (ECRTS 2020)*. Ed. by M. Völp. Vol. 165. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020, 23:1–23:24.

[11] M. Hassan and R. Pellizzoni. "Bounding DRAM Interference in COTS Heterogeneous MPSoCs for Mixed Criticality Systems". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 37.11 (2018), pp. 2323–2336.

[12] C. Maia, G. Nelissen, L. Nogueira, L. M. Pinho, and D. G. Pérez. "Schedulability analysis for global fixed-priority scheduling of the 3-phase task model". In: *2017 IEEE 23rd International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*. 2017, pp. 1–10.

[13] C. Maia, L. Nogueira, L. M. Pinho, and D. G. Pérez. "A closer look into the AER Model". In: *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*. 2016, pp. 1–8.

[14] C. Maiza, H. Rihani, J. M. Rivas, J. Goossens, S. Altmeyer, and R. I. Davis. "A Survey of Timing Verification Techniques for Multi-Core Real-Time Systems". In: *ACM Comput. Surv.* 52.3 (June 2019).

[15] C. Pagetti, J. Forget, H. Falk, D. Oehlert, and A. Luppold. "Automated Generation of Time-Predictable Executables on Multicore". In: *Proceedings of the 26th International Conference on Real-Time Networks and Systems*. RTNS '18. Chasseneuil-du-Poitou, France: Association for Computing Machinery, 2018, pp. 104–113.

[16] R. Pellizzoni, E. Betti, S. Bak, G. Yao, J. Criswell, M. Caccamo, et al. "A Predictable Execution Model for COTS-Based Embedded Systems". In: *2011 17th IEEE Real-Time and Embedded Technology and Applications Symposium*. 2011, pp. 269–279.

[17] S. A. Rashid, G. Nelissen, D. Hardy, B. Akesson, I. Puaut, and E. Tovar. "Cache-Persistence-Aware Response-Time Analysis for Fixed-Priority Preemptive Systems". In: *2016 28th Euromicro Conference on Real-Time Systems (ECRTS)*. 2016, pp. 262–272.

[18] G. Schwäricke, T. Kloda, G. Gracioli, M. Bertogna, and M. Caccamo. "Fixed-Priority Memory-Centric Scheduler for COTS-Based Multiprocessors". In: *32nd Euromicro Conference on Real-Time Systems (ECRTS 2020)*. Ed. by M. Völp. Vol. 165. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020, 1:1–1:24.

[19] T. Thilakasiri and M. Becker. "An Exact Schedulability Analysis for Global Fixed-Priority Scheduling of the AER Task Model". In: *Proceedings of the 28th Asia and South Pacific Design Automation Conference*. ASPDAC '23. Tokyo, Japan: Association for Computing Machinery, 2023, pp. 326–332.

[20] G. Yao, R. Pellizzoni, S. Bak, E. Betti, and M. Caccamo. "Memory-centric scheduling for multicore hard real-time systems". In: *Real-Time Systems* 48 (Nov. 2012).

[21] H. Yun, R. Pellizzon, and P. K. Valsan. "Parallelism-Aware Memory Interference Delay Analysis for COTS Multicore Systems". In: *2015 27th Euromicro Conference on Real-Time Systems*. 2015, pp. 184–195.