Open Questions for the Bus-Blocking Problem in the 3-Phase Task Model under Partitioned Scheduling

Jatin Arora
Cláudio Maia
Syed Aftab Rashid

CISTER-TR-210503

2021/06/04

Open Questions for the Bus-Blocking Problem in the 3-Phase Task Model under Partitioned Scheduling

Jatin Arora, Cláudio Maia, Syed Aftab Rashid

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, clrrm@isep.ipp.pt, syara@isep.ipp.pt

https://www.cister-labs.pt

Abstract

Modern multicore processors have the potential to provide raw computing power while being energy efficient. However, the use of multicore processors in systems with stringent timing requirements (e.g., avionics, automotive, and railways) still needs to be handled in a cautious manner. The reason for this is that, in multicore platforms, resources such as system bus, last-level cache, and main memory, are shared among all the cores. Let us consider the system bus as an example. The system bus is typically responsible for connecting all the cores to the main memory (thus, it is shared among all cores in the system). When a task wants to perform a read/write operation from/to the main memory it needs to do so by accessing the shared system bus. The direct consequence of using a shared system bus is that a task may suffer bus-blocking if the bus is already busy serving memory requests of co-running tasks (tasks running on the other cores). The 3-phase task model was introduced to reduce the unpredictability caused by shared resources in multicore systems. However, the bus-blocking can still happen when memory phases running on multiple cores wants to access the system bus at the same time instant. To analyze this, a bus blocking analysis for the 3-phase task model was proposed in the state-of-the-art. This paper highlights the open questions and possible solutions related to the bus blocking problem in the 3-phase task model under partitioned scheduling that has not been addressed in the state-of-the-art.

Open Questions for the Bus-Blocking Problem in the 3-Phase Task Model under Partitioned Scheduling

Jatin Arora, Cláudio Maia, Syed Aftab Rashid
CISTER Research Centre, ISEP, IPP, PORTO, Portugal.
{jatin,crrm,syara}@isep.ipp.pt

1. Introduction

Multicore processors became the preferable choice over single-core processors for various applications (e.g. mobile phones, computers, smart TVs, etc.) due to their distinguished features, such as computing power, energy efficiency, etc. However, the use of multicore processors in systems with stringent timing requirements (e.g., avionics, automotive, and railways) still needs to be handled in a cautious manner. The reason for this is that, in multicore platforms, resources such as system bus, last-level cache, and main memory, are shared among all the cores. Let us consider the system bus as an example. The system bus is typically responsible for connecting all the cores to the main memory (thus, it is shared among all cores in the system). When a task wants to perform a read/write operation from/to the main memory it needs to do so by accessing the shared system bus. The direct consequence of using a shared system bus is that a task may suffer *bus-blocking* if the bus is already busy serving memory requests of co-running tasks (tasks running on the other cores). An important aspect to be considered in the analysis of such platforms, in particular the ones that run applications with stringent timing requirements is the impact of bus-blocking. The bus blocking suffered by a task is dependent on the specific properties of tasks that are running on the other cores at the same instant, (e.g., number of memory requests, WCET of each memory request, etc.). This may bring temporal unpredictability to the system under analysis and make the analysis of the worst-case timing behavior of any given task extremely difficult.

The 3-phase task model [1] was introduced to reduce the unpredictability caused by shared resources in multicore systems. In the 3-phase task model, the execution of each task is divided into three phases, i.e., *Acquisition* (A), *Execution* (E), and *Restitution* (R) phases. During the A-phase, the task fetches data from the main memory via the system bus and stores it into the core's local memory. During the E-phase, the core executes the task using the available data (previously fetched) in the core's local memory. Finally, during the R-phase, the task writes back the modified data to the main memory, by using the system bus. Thus, the A- and R-phases are the memory phases in which accesses to the system bus are required, whereas the E-phase is a computation phase in which system bus access is not required. Consequently, a core can execute a memory phase by accessing the system bus without suffering bus-blocking when other cores execute E-phases (which can happen in parallel).

2. Open Questions

Even though the 3-phase task model can reduce the temporal unpredictability due to the sharing of system bus in multicore systems, tasks can still suffer bus-blocking under certain circumstances. For instance, when a task wants to access the system bus to execute a memory phase while the system bus is busy handling the memory phase of a co-running task. There exists work that focuses on the bus-blocking problem for the 3-phase task model using partitioned scheduling (i.e., [2]). In [2], the authors compute the bus-blocking by considering different cases that contemplate the different scenarios that may occur between the memory phases executing on a given core and the memory phases of other cores. This information is then used along with the given bus arbitration policy to derive bounds on the maximum bus blocking tasks can suffer. Even though works like [2] compute sound upper-bound on the bus-blocking for the 3-phase task model using partitioned scheduling, the following questions are still open for discussion:

- **Q1:** The bus-blocking analysis of [2] assumes the bus arbitration policy is First-Come First-Served (FCFS). It is not known whether such an analysis can be directly applied to other bus arbitration policies, such as round-robin, TDMA, processor-priority, fixed-task priority, etc.
- **Q2:** The bus-blocking analysis of [2] assumes that if there are pending A-phases that are to be executed on a given core, they can execute immediately after the completion of a R-phase without releasing the bus (i.e., back-to-back execution of R and A-phases on the same core). Although such an assumption

can be considered realistic [3], it is still not clear how to derive the bus-blocking if the system cannot comply with such an assumption and what is the impact concerning the results.

3. Towards Possible Solutions

Possible Solutions for Q1: The bus-blocking analysis of [2] assumes FCFS bus arbitration in which each bus request can be composed of at most a combination of one R and one A-phase running on a given core. The implication of this bus arbitration policy is that the worst-case is derived by assuming that each bus request of the core under analysis is the last to arrive on an FCFS basis and it can suffer bus-blocking from at most one bus request running on each of the other cores. Unlike FCFS, using a round-robin bus arbitration policy makes use of a bus access slot in which each core can only access the system bus during the assigned slot. The challenge is how to deal with different bus access slot sizes as they have a direct impact on the worst-case timing analysis. In the worst-case, the core under analysis may have to wait for all the other cores to complete their bus access slots. Similarly, the bus-blocking analysis for TDMA can be derived.

The bus-blocking analysis of [2] may not be directly applied to fixed-priority bus arbitration policies such as processor-priority and fixed-task priority. This is because the bus-blocking analysis of [2] assumes that the task under analysis may suffer bus-blocking from all the tasks running on all the other cores. Such an assumption may not be true in the case of processor-priority bus-arbitration policy as the task under analysis can only suffer bus-blocking from tasks running on higher-priority processors. Similarly, under fixed-task priority bus arbitration policy, the task under analysis can only suffer bus-blocking from higher-priority tasks running on all the other cores. To achieve this goal, work is needed to develop a general framework that can compute bus-blocking for 3-phase tasks considering different bus arbitration policies.

Possible Solutions for Q2: The bus-blocking analysis of [2] is not applicable to systems that cannot comply with the assumption of back-to-back execution of R and A-phases. To address this issue, the bus-blocking can be derived by relaxing the assumption of back-to-back execution of R- and A-phases. Similarly to Q1 above, if a round-robin policy is selected, then the bus access slot sizes will have a direct impact on the results obtained. Maybe it is possible to define them in such a way that each core can execute at most one memory phase during its bus access slot. Under FCFS, each bus access request can be composed of at most one memory phase. Consequently, each memory phase running on the core under analysis can suffer the bus-blocking from at most one memory phase running on each of the other cores. This aspect is still not considered in the analysis presented in [2] and by relaxing the assumption of back-to-back execution of R and A-phases, it may be possible to derive even tighter bounds on bus blocking in comparison to [2]. To investigate these properties, we are currently developing a bus-blocking analysis by relaxing the assumption of back-to-back execution of R and A-phases and comparing it to [2], by using a set of examples and experiments.

References

[1] Girbal, Sylvain & Durrieu, Guy & Faugere, Madeleine & Gracia Pérez, Daniel & Pagetti, Claire & Puffitsch, Wolfgang. (2014). Predictable Flight Management System Implementation on a Multicore Processor.

[2] Jatin Arora, Cláudio Maia, Syed Aftab Rashid, Geoffrey Nelissen and Eduardo Tovar, "Bus-Contention Aware Schedulability Analysis for the 3-Phase Task Model with Partitioned Scheduling", in the 29th International Conference on Real-Time Networks and Systems (RTNS'21) (Accessed on 28/04/2021. URL: https://easychair.org/publications/preprint/gdNJ.

[3] A. Alhammad and R. Pellizzoni, "Schedulability analysis of global memory-predictable scheduling," 2014 International Conference on Embedded Software (EMSOFT), 2014, pp. 1-10, doi:10.1145/2656045.2656070.

Acknowledgments

This work was partially supported by National Funds through FCT/MCTES (Portuguese Foundation for Science and Technology), within the CISTER Research Unit (UIDB/04234/2020); also by the Operational Competitiveness Programme and Internationalization (COMPETE 2020) under the PT2020 Partnership Agreement, through the European Regional Development Fund (ERDF), and by national funds through the FCT, within project POCI-01-0145-FEDER-029119 (PREFECT); also by FCT, under PhD grant 2020.09532.BD.