Using All Processor Cores While Being Confident about Timing

Today, almost all computers use multicore processors. Unfortunately, satisfying hard real-time requirements of software executing on such computers is challenging because the timing depends on how resources in the memory system are shared, and this information is typically not publicly available. This project addresses this problem.

Multicore processors

Today, almost all computers use multicore processors. These computers have many processor cores such that one program can execute on one processor core and another program can execute on another processor core simultaneously (true parallelism). Typically, processor cores share memory. In today's memory system, a large number of resources are used to make memory accesses faster in general but, unfortunately, also make execution time more unpredictable and dependent on execution of other programs (because these other programs use shared resources in the memory system). A simplified view of a multicore processor with the memory system is shown in Figure 1.

Embedded real-time cyber-physical systems

These systems are pervasive in society in general, as shown by the fact that 99% of all processors produced are used in embedded systems. In many of these systems, computing the correct result is not enough; it is also necessary to compute the correct result at the right time.

These methods assume that **one knows** the resources in the memory system; unfortunately, most chip vendors do not make this information available.

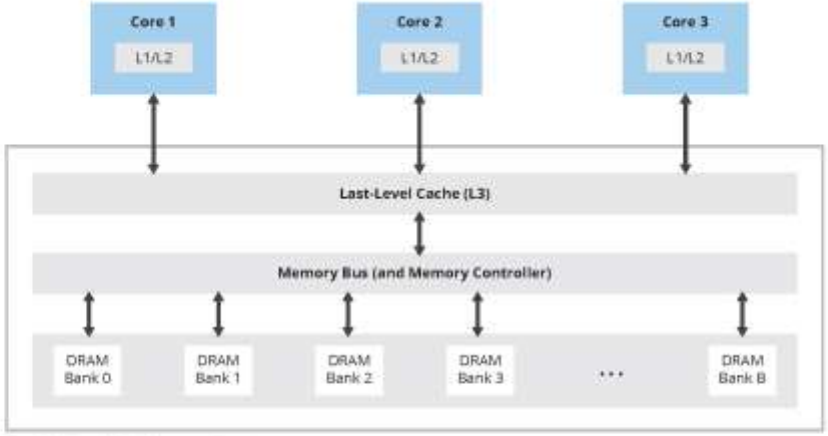

Figure 1: A simplified view of a multicore processor with shared memory

Department of Defense (DoD)

Embedded real-time cyber-physical systems are pervasive in the DoD. Because of the importance of achieving predictable timing, it is common for practitioners to disable all processor cores except one (hence making a multicore processor behave as a single processor system). The importance of timing was recently stressed by AMRDEC's S3I director [1]:

> *"The trick there, when you're processing flight critical information, it has to be a deterministic environment, meaning we know exactly where a piece of data is going to be exactly when we need to—no room for error," [Jeff] Langhout says. "On a multi-core processor there's a lot of sharing going on across the cores, so right now we're not able to do that."*

Current solutions

The current state of the art makes solutions available for managing contention for resources in the memory system and for analyzing the impact of this contention on timing for the case that we know the resources in the memory system.

Problem addressed

In this project, we have addressed the problem of verifying timing of software executing on a multicore processor assuming that we do not know the resources in the memory system.

Results

We have developed a preliminary method—see Andersson, B. et al., "Schedulability Analysis of Tasks with Co-Runner-Dependent Execution Times." ACM Transactions on Embedded Computing Systems. 2018.

[1] "Army still working on multi-core processor for UH-60s" May 2017. Available at https://www.flightglobal.com/news/articles/army-still-working-on-multi-core-processor-for-uh-6-428855/

Dr. Björn Andersson | bandersson@sei.cmu.edu