Towards a Predictable Execution Model for Heterogeneous Systems-on-a-Chip

The interest in autonomous vehicles is growing constantly, with lots of practical applications appearing on the marketplace and many more being actively studied in academia, industry and the military. Autonomous systems are characterized by a plethora of computation-heavy and time-critical tasks, which are often inherently concurrent.

The deployment of such real-time workloads on commercial off-the-shelf (COTS) hardware is attractive, as it reduces the cost and time-to-market of new products. However, because of non-predictable hardware arbiters designed to maximize average or peak performance, it is very difficult to provide timing guarantees on such systems.

Most modern high-end embedded systems-on-a-chip (SoCs) rely on a heterogeneous design, coupling a general-purpose multi-core CPU to a massively parallel accelerator, typically an integrated GPU (iGPU). In such designs, the coupling of CPU and GPU is very tight, as they physically share the main DRAM memory, as opposed to traditional discrete GPUs. Main memory sharing complicates the deployment of real-time workloads, as contention introduces huge variability in execution time that is difficult to predict without a major loss in performance. This is particularly true in the view of the high bandwidth requirements of GPUs. To harness the advantages of COTS hardware and integrated accelerators in the context of real-time applications, new techniques that arbitrate memory requests are required.

In this presentation, we discuss a software technique that enables predictable arbitration of main memory usage in heterogeneous SoCs. Borrowing from the Predictable Execution Model (PREM), this technique operates by orchestrating CPU and GPU accesses to DRAM based on the division of accelerator workloads into memory and computation phases. Compiler support simplifies the generation of specialized heterogeneous codes, while the underlying runtime system synchronizes CPU and GPU operation.

We present an early evaluation of the technique using a prototype LLVM implementation targeting the NVIDIA Tegra TX1 SoC, wher an ARM bigLITTLE multi-core CPU and an NVIDIA Maxwell GPU share a unified global memory. Our experiments show that the presented scheme is indeed able to reduce the adverse effects of memory sharing, while retaining a high throughput on the accelerator and the benefits of programming with widespread programming models for COTS heterogeneous platforms.