Enabling low-cost and lightweight zero-copy offloading on embedded heterogeneous many-core accelerators: the PULP experience

Alessandro Capotondi, University of Bologna

Andrea Marongiu, University of Bologna, ETH Zurich

Luca Benini, University of Bologna, ETH Zurich

Architectural heterogeneity has proven an effective design paradigm to successfully tackle many technology walls in the past decade. Multi- and many-core heterogeneous designs, initially proposed in the high-performance computing domain, are nowadays widely available among commercial and academic embedded systems. One of the most common heterogeneous system templates envisions single-chip coupling of a powerful, general-purpose *host* processor to one (or more) *programmable many-core accelerator(s)* (PMCA).

Historically, the complex memory systems adopted by heterogeneous embedded systems-on-chip has brought the biggest difficulties in application development. On the host side, coherent caches and *Memory Management Units* (MMUs) make the memory hierarchy completely transparent. On the PMCA side, however, *scratchpad* memories are physically addressed and need to be explicitly managed via DMA transfers. Nowadays, virtually all major embedded GPU vendors are equipping their SoCs with full-fledged HW support for Unified Virtual Memory (UMV), consisting of I/O MMUs and coherent interconnections. On the other hand, in the low-end embedded SoC domain UVM support is missing or only partially supported.

The *Parallel, Ultra-Low-Power Platform* (PULP) has been recently used as a research platform to demonstrate lightweight support for UVM in the context of resource-constrained, low-end heterogeneous many-cores. Here, a simple *Input/Output Translation Lookaside Buffer* (IOTLB) is used as a replacement for sophisticated system MMUs to enable virtual-to-physical address translations for the PMCA. To retrieve such translations, the PMCA program explicitly instruments every *Load/Store* into the main memory via software functions that query the IOTLBs. In case of a miss, these functions stall the requesting PMCA core while a request for the page miss handling is propagated to the host. While similar solutions have demonstrated the importance of UVM for constrained embedded many-cores, their use still requires a lot of programmer effort to manually orchestrate offloading sequences and PMCA code instrumentation. *Directive-based* programming models have shown their effectiveness in simplifying programmability while enabling high performance targets on heterogeneous many-core architectures. Since the release v4.0, the OpenMP Specification offers constructs to deploy and to distribute work from the host to the PMCA using directives.

The presentation will cover the hardware extensions to the PULP platform for supporting UVM and our experience in porting the OpenMP (specification v4) programming model to a heterogeneous embedded system based on a four-cluster, thirty-two-core instance of the PULP accelerator and featuring lightweight UVM support. Moreover, the presentation will describe the GNU GCC based toolchain extensions that enables: i) the automatic generation of host and accelerator binaries from a single, high-level, OpenMP parallel program; ii) the automatic instrumentation of the accelerator program to transparently manage UVM, enabling up to 4x faster execution compared to traditional copy-based offload mechanisms.