SAVE: Towards Efficient Resource Management in Heterogeneous System Architectures*

G. Durelli[1], M. Coppola[2], K. Djafarian[3], G. Kornaros[4], A. Miele[1], M. Paolino[5], Oliver Pell[6], Christian Plessl[7], M.D. Santambrogio[1], and C. Bolchini[1]

[1] Politecnico di Milano
{gianlucacarlo.durelli,antonio.miele,marco.santambogio, cristiana.bolchini}@polimi.it
[2] STMicroelectronics
marcello.coppola@st.com
[3] ARM
karim.djafarian@arm.com
[4] Technological Educational Institute of Crete
kornaros@ie.teicrete.gr
[5] Virtual Open Systems
m.paolino@virtualopensystems.com
[6] Maxeler Technologies
oliver@maxeler.com
[7] University of Paderborn
christian.plessl@uni-paderborn.de

Abstract. The increasing availability of different kinds of processing resources in heterogeneous system architectures associated with today's fast-changing, unpredictable workloads has propelled an interest towards systems able to dynamically and autonomously adapt how computing resources are exploited to optimize a given goal. Self-adaptiveness and hardware-assisted virtualization are the two key-enabling technologies for this kind of architectures, to allow the efficient exploitation of the available resources based on the current working context. The SAVE project will develop HW/SW/OS components that allow for deciding at runtime the mapping of the computation kernels on the appropriate type of resource, based on the current system context and requirements.

1 Current Trends and Challenges in System Architecture

Since the 1960s processors have shown an exponential improvement in performance fueled by advances in semiconductor technology, allowing to increase the clock speed, and architectural innovations increasing the amount of work done per cycle, e.g., pipelining, branch prediction and out-of-order execution. Since about 2005 this exponential growth of single-core performance has significantly flattened out. While further increasing the clock frequency would be possible,

* This research is partially supported by the European Commission, EU Seventh Framework Program, Project 610996-SAVE.

D. Goehringer et al. (Eds.): ARC 2014, LNCS 8405, pp. 337–344, 2014.

the energy-efficiency of the computation would drastically decrease. The answer of the semiconductor industry to this challenge was to move away from maximizing single-core performance towards maximizing the aggregated computational power and efficiency of the complete processor. This objective is addressed using two main techniques: *on-chip parallel computing* and *customization*.

On-chip parallel computing exploits the fact that shrinking semiconductor process geometries allow for integrating several CPU cores into a single package while keeping power dissipation constant. We can witness this trend of parallelism by the rapid proliferation of multi- and many-core processors. The former provide *fine-grain* parallelism through SIMD, and *coarse-grain* parallelism by integrating different cores onto the same chip. The concept of on-chip parallel processing is carried further by *many-core processors*, which integrate several dozens of simpler CPU cores in a single processor. This concept is for example used in the Intel Xeon Phi co-processor, Kalray [3], or the STM research platform "Platform 2012" [5]. Massively parallel processor architectures even increase the number of cores to several hundreds, while reducing the complexity of each core and imposing a more restrictive execution model. The prototypical representative of this class of architectures are GPGPUs, which use a SIMD-like execution model that is targeted by dedicated programming and runtime environments, such as, CUDA, STREAM, OpenCL, C++ AMP.

The other direction for improving the performance and computation efficiency is to use programmable hardware devices, in particular field-programmable gate arrays (FPGAs). These devices can be used to implement *customized* application-specific computing engines that are perfectly tailored to the needs of a given application. One instance of this approach that is currently receiving a lot of attention in high-performance computing (HPC) are dataflow engines (DFEs), that implement custom, high-throughput processing pipelines for scientific computing or big data analytics.

In practice, neither massively parallel on-chip processors nor customized DFEs are used as the sole computational resource. Instead, these technologies are used as accelerators in combination with conventional multi-core processors. The resulting heterogeneous computing systems are integrated either as system-on-chip solutions for the embedded and mobile computing segment or by attaching the massively parallel architectures as acceleration co-processors via PCI express/Infiniband for the HPC domain.

While the performance and energy-efficiency potential of heterogeneous computing systems is widely acknowledged and many heterogeneous systems are being deployed, it is still an open research question how to program, operate, and manage these systems to reap the benefits of heterogeneous computing while keeping the complexity in bounds. For example, GPUs' and DFEs' computing resources are currently not managed by the OS but by the applications that use them, preventing a global optimization of resource usage at system-level.

2 The Objectives of the SAVE Project

According to recent technology studies the demand for computing will continue to rise rapidly and the number of computing devices will be more than triple by 2020. Given that each device contributes to our society's energy and carbon footprint, new technologies to address this challenge must be found. Indeed, we are moving towards an on-demand computing scenario, characterized by varying workloads, constituted of diverse applications with different performance requirements, and criticality. With the end of Dennard Scaling [1] in sight and a slowdown in supply voltage scaling, the entire ICT industry is currently going through an inflection point where energy efficiency, which used to be the key design concern of embedded and mobile systems, is becoming the ultimate design constraint for the entire computing spectrum ranging from embedded systems (ES) to HPC systems.

In this scenario, heterogeneous computing is currently the most promising approach to address these challenges. The flip side of heterogeneity is increased complexity. To reach an optimal solution, a system architect needs to take into account the efficiency of the computational units, the workload, the working conditions and so on. As a result, heterogeneous computing is often considered too complex or inefficient, except for very specific application environments, where the working context is delimited and the design space to be explored is suitably contained. Hence, it is key to find answers to the question how to integrate, exploit, and manage heterogeneous resources to reach the desired performance goal at minimal cost while limiting the complexity for development.

The EU FP7 project SAVE (Self-Adaptive Virtualization-aware high-performance/low-Energy heterogeneous system architectures) [2] aims at addressing the challenge of exploiting specialized computing resources of a heterogeneous system architecture (HSA) by pooling them and taking advantage of their individual characteristics to optimize the performance/energy trade-off for the resulting system, without constraining the applications or operation context. To this end, we strive for defining a more general approach for exploiting HSAs, lowering the complexity of managing the available resources, while enabling the overall system to pursue an optimization goal that can depend on the current working conditions (application requirements, workload, ...). More precisely, SAVE will address these limitations by providing *self-adaptivity* and *hardware-assisted virtualization* to allow the system to dynamically and autonomously decide how to optimally allocate the workload generated by applications to the specialized resources for achieving an effective execution of the application while optimizing a user-defined goal (e.g., performance, energy, reliability, resource utilization).

SAVE will define the necessary SW/HW technologies for implementing self-adaptive systems exploiting heterogeneous architectures that include two classes of accelerators: GPUs and DFEs that enhance heterogeneous architectures to cope with the increased variety and dynamics of workloads observed in the HPC and ES domains. Virtualization and self-adaptiveness are jointly exploited to obtain a new self-adaptive virtualization-aware Heterogeneous System Architecture, dubbed *saveHSA*, that exhibits a highly adaptive behavior to

achieve the requested performance while minimizing energy consumption by allocating the tasks to the most appropriate accelerators, based on the current status of the overall system. The effectiveness of SAVE's technologies will be validated in two applications scenarios, financial risk computing and image processing algorithms, to cover the ES and HPC domain. We strive for an energy-efficiency improvement of 20% with respect to today's architecture that use DFEs or GPUs in a traditional fashion. At the same time, system manageability, ease of deployment and resilience will be greatly improved.

3 The SAVE Concept

SAVE will enable the system to decide at run-time, based on the observation of collected information that characterizes the changing scenario, how to use the available resources, to pursue a defined, but changeable, optimization goal. The overall project outcome are technologies suitable for a cross-domain adoption ranging from the data centers and HPC, where virtualization plays a relevant role, to ES, where heterogeneity and customized execution prevail. An important issue is how to move one step forward in the way operating systems view such computational resources, traditionally perceived as secondary components that hide their complexities behind pre-defined device drivers that expose high-level programming API. In this perspective hardware-assisted virtualization offers the opportunity to simplify the sharing of such heterogenous resources, without compromising performance. In SAVE, the various computing HW resources, will execute workloads constituted from multiple applications running in a virtualized environment. Since device drivers determine how accelerators are shared, this restricts scheduling policies and optimization criteria for the computational resources. If resources are shared between several guest OSes running in virtual machines (VMs), issues of contention, fairness and security become further pronounced. In addition the hypervisor has limited control on how the heterogeneous computing resources are used, and whether sharing is possible in time, space or both, because there is no direct control over the scheduler actions beyond the proprietary interfaces.

Therefore, to benefit from the opportunity that heterogeneous platforms offer SAVE exploits virtualization and integrates it in a self-adaptive perspective, to obtain a performance/energy cost-effective solution.

The final goal is to provide a set of technologies implementing a **self-adaptive virtualization-aware Heterogeneous System Architecture**, consisting of:

1. an architecture able to run any kind of applications (through *virtualization*),
2. a *scalable*, *cache coherent heterogeneous* platform that exposes the heterogeneity (different kinds of virtualizable computation islands, such as GPUs and DFEs) reducing power consumption and simplifying management,
3. a *dynamically adaptable* saveHSA management, driven by different environment inputs that may change at run-time, exploiting a runtime and just-in-time code generation infrastructure that allows for dynamically offloading computational hotspots to heterogeneous computing resources, and

4. an integrated hypervisor layer that promotes the specialized islands of computation as schedulable entities, that can be shared by multiple VMs.

This goal will be achieved by developing new technologies deeply integrated to obtain efficiency and optimality: (i) an *advanced run-time self-adaptiveness OS support layer*, (ii) a *hardware-assisted virtualization support for the specialized computing resources*, (iii) new *hypervisor extensions* that expose a virtual computation interface to applications, and (iv) a *just-in-time compilation and offloading infrastructure*.

3.1 Advanced Run-Time Self-adaptiveness OS Support Layer and Adaptive Hardware Layer

The first main target of the SAVE project is the development of a self-adaptive and virtualization-aware HSA, able to modify itself according to the changing scenario, thus exposing *autonomous* **adaptation** capabilities driven by the system itself, as a response to a variation in the workload, in the architecture resources, or in the user (optimization) requirements.

While reconfigurability for some classes of architectures (e.g., homogeneous multi- and many-core platforms, FPGAs) has been tackled to some extent in both the ES and HPC scenarios, each one exploiting the peculiarities of the domain, we aim at tackling the challenge by referring to a more general architectural platform, that is *across the two domains*, to serve as a means for both low-cost HPC or high-end ES. Fig. 1 shows the SAVE self-adaptiveness concept, where a *self-adaptive orchestrator* module integrated in the host operating system dispatches the submitted kernels onto the most appropriate kind of resources available in the saveHSA, where the appropriateness depends on the currently pursued goal (power consumption, performance, overall load) and the kernel nature, in terms of computational requirements. To support the runtime *self-adaptive orchestrator* module, SAVE will develop an enhanced communication infrastructure, monitors combining cache coherent on-chip interconnects with off-chip (PCI-E) with knobs to observe their behavior and configure them to act in a specific mode, along with the design of a common interface towards hardware components, called Dynamically Manageable Component (DMCs), suitable for the realization of adaptive systems. On top of this, a novel parametric versions of existing OS components (e.g., observable and controllable schedulers, device drivers with adaptive and introspective data structures) will be developed, able to orchestrate the execution of the kernels on the heterogeneous architecture, by adjusting at run-time the adopted policies, and set the ground for a novel OS with new self-aware components to support self-awareness and run-time adaptability. Furthermore, the development of a library of optimization policies allowing the system to decide, on the basis of the observed behavior (achieved performance, monitored temperature, ...), the actions (on what resource to map kernels) to be carried out to fulfill the optimization goal, has to be implemented.

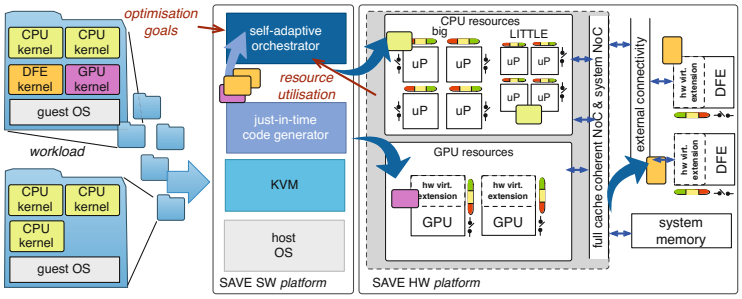

Fig. 1. SAVE self-adaptiveness concept

3.2 Hardware-Assisted Virtualization Support for the Specialized Computing Resources

The second main target of the project is the definition of an innovative hardware support for virtualization for the components constituting the heterogeneous platform, such that their *exploitation and sharing* can be facilitated. More precisely, within the perspective of a self-adaptive system that may migrate VMs running kernels from one resource (e.g., CPU) to a different one (e.g., GPU), as shown in Fig. 2, the foreseen proposed solutions will need to be flexible and sophisticated to allow virtualization not to introduce overheads and prevent the exploitation of the peculiarities of the platform.

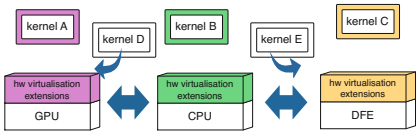

Fig. 2. SAVE virtualization concept

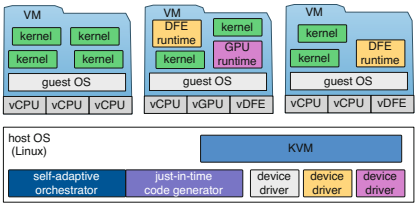

Fig. 3. SAVE hypervisor extensions

This goal will be achieved through the development of APIs to expose to the *self-adaptive orchestrator* the performance, power information of kernels running in VMs on the specific accelerator (GPU, DFE). The APIs shall provide the access to the dynamic characteristics of the kernels being executed to support the decision of the orchestrator on where to map the VM kernels to pursue the optimization goal. Moreover hardware cache coherency between the host processor and the different on-chip islands of computation resource will be introduced. This mechanism will simplify the communication cost related to data and code of executing kernel in specific accelerators. In this way, kernels can be efficiently transferred to accelerators reducing the number of copies between accelerators and host processor. Another advantage is the usage of a common virtual memory system for host processors and accelerators, to allow accelerators to access virtual memory addresses that are not yet available in the physical memory.

3.3 Hypervisor Extensions

The third target of the project is a set of elements (Fig. 3) needed to integrate the two previously discussed objectives in the realization of a self-adaptive virtualization-aware HSA. This integration will be accomplished by developing extensions to enable multiple VMs to directly access virtualized GPU/DFE, by means of the hardware-assisted virtualization. This will add the concepts of vGPU and vDFE to the already existing, in the virtualization scenario, vCPU. The hypervisor will manage these resources, enabling multiple VMs to safely interact with them concurrently. Furthermore, it will be necessary to provide extensions for allowing inner and outer migration. The former enables the "smart" orchestrator to migrate and schedule directly the tasks of different VMs among vCPUs/vGPUs/vDFEs. The latter will provide scalability in the HPC scenario (such as that of Cloud computing data centers), giving to the VMs the possibility to migrate among different saveHSA platforms. Finally, the development of an efficient communication mechanism that permits the interaction between the host and the VMs will be done. In this way, the just-in-time code generator and the orchestrator will be able to dispatch tasks to the appropriate resources and achieve the self-adaptiveness of the system.

3.4 Just-in-Time Compilation and Offloading Infrastructure

Finally, SAVE aims at exploring a novel approach of offloading computational hotspots to heterogeneous computing resources. To this end, a runtime system based on the LLVM compiler and virtual machine infrastructure [4] will be developed to autonomously analyze applications for computational hotspots that could be executed more efficiently with DFEs or GPUs. If such hotspots are identified, the smart orchestrator can initiate a just-in-time compilation for the optimal resource. Once the compilation has completed, the runtime system transparently migrates the computation to the targeted computing resource.

4 Conclusions

The heterogeneity, generated by the acceleration cores (e.g., GPUs, DFEs), consists of specialized islands of computation in architectures where different executions models can be used to exploit the peculiarities of the resources. However heterogeneity has a price too; chips have limited resources and all units on the chip compete for the shared resources, thus receiving only a limited share of load, so being under- To reach an optimal solution, it is up to the architect to distribute the resources among the different units by taking into account the efficiency of these units as well as the workload. SAVE will address this issue by developing a customizable and adaptable computing capability that manages the available resources, through virtualization, to decouple the relation between the applications and the complex underlying heterogeneous architecture.

References

1. Dennard, R., Gaensslen, F., Rideout, V., Bassous, E., LeBlanc, A.: Design of ion-implanted MOSFET's with very small physical dimensions. IEEE Journal of Solid-State Circuits 9(5), 256–268 (1974)
2. EU FP7 project SAVE, http://www.fp7-save.eu
3. KALRAY, http://www.kalray.eu
4. Lattner, C., Adve, V.: LLVM: A compilation framework for lifelong program analysis & transformation. In: Proc. Int. Symp. Code Generation and Optimization, pp. 75–86 (2004)
5. STMicroelectronics and CEA: Platform 2012: A many-core programmable accelerator for Ultra-Efficient Embedded Computing in Nanometer Technology. In: Research Workshop on STMicroelectronics Platform 2012 (2010)