Rapid Prototyping and Design Space Exploration Methodologies for Many-Accelerator Systems

Efstathios Sotiropoulos, Sotirios Xydis, Konstantinos Siozios, George Economakis and Dimitrios Soudris
School of Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece
E-mail: {stasot, sxydis,ksiop, geconom, dsoudris}@microlab.ntua.gr

Abstract—The ever-growing design complexity of modern embedded systems and the need for lower energy consumption have lead to design techniques which target to bridge the gap between the designer’s productivity and the design complexity. In particular, Virtual Prototyping enables the system modeling and simulation in multiple abstraction levels, while the automated Design Space Exploration (DSE) targets to find optimized design solutions in a reasonable time. However, there is the need for more efficient techniques for prototyping and co-simulation, as the rapid simulation has become a stringent requirement.

In addition, as emerging heterogeneous architectures expose even higher design complexity, typical DSE techniques may not achieve high-quality design solutions. Towards this direction, the proposed design flow introduces (a) a set of prototyping techniques which target to faster but accurate simulation, also enabling faster system evaluations, thus enabling the investment of the evaluation time gains on achieving better-optimized solutions. Additionally, DSE on both computation and communication aims to achieve higher-quality solutions (i.e. solutions of the exact Pareto curve of the design space) without including architectural details (e.g. fully modeled register file, branch prediction, pipeline, ALU, etc.) which are mandatory in early design stages, the CPU models are modeled as a VP, according to the system specifications. VP modeling utilizes a source of off-the-shelf CPU models. The first step of virtual prototyping is the application profiling, in order to find the computationally-intensive kernels, and the mapping of algorithmic kernels onto specialized architectures. The latter leads to heterogeneous Systems-on-Chip (SoC) comprising numerous processing elements of different types, which can achieve higher throughput and lower power consumption, thus better covering the increased requirements of modern embedded systems.

The viability of this selection relies on the Moore’s law [1], according to which the number of transistors in a specific area of the chip doubles every two years approximately. However, this law also implies that the design complexity highly increases, so that the full exploitation of the increased technological capabilities becomes difficult, in both system level and physical implementation. Towards this issue, there is the need for efficient design techniques [2], like (a) High-Level Synthesis (HLS) to synthesize a C/C++ source code to RTL description; (b) Virtual Platforms (VP), i.e. software representations of the system-under-design that allow for faster simulation and verification in higher abstraction levels; and (c) Design Space Exploration (DSE) which aims to cover large design spaces (i.e. sets of design solutions) by performing only a limited number of evaluations.

However, the ever-increasing requirements of higher throughput and lower power consumption has lead to large-scale heterogeneous SoCs, like many-accelerator systems and high-complexity Network-on-Chip (NoC) architectures, which impose an exponential increase in the design complexity. Finally, the fast simulation and the high accuracy are two requirements of utmost importance. To alleviate the aforementioned problems, the contributions of this paper are organized into two axes: (a) Virtual Prototyping techniques for the fast but accurate system evaluation, and (b) DSE methodologies which enable the efficient exploration of exponentially-grown design spaces. The proposed methodologies are integrated into a systematic design flow for many-accelerator systems, while they are accompanied by a set of experimental results.

II. PROPOSED DESIGN FLOW

Figure 1 depicts an overview of the proposed design flow. The flow comprises three main stages: (a) Virtual prototyping with SystemC, (b) DSE for computation and (c) DSE for communication. The presented virtual prototyping techniques enable faster system evaluations, thus enabling the investment of the evaluation time gains on achieving better-optimized solutions. Additionally, DSE on both computation and communication aims to achieve higher-quality solutions (i.e. solutions which are closer to the exact Pareto curve of the design space) for many-accelerator SoCs.

During virtual prototyping, a N-threaded application is modeled as a VP, according to the system specifications. VP modeling utilizes a source of off-the-shelf CPU models. The first step of virtual prototyping is the application profiling in order to find the computationally-intensive kernels, and HW/SW partitioning in order to create the N-threaded software (CPU) and the hardware part (HW accelerators) [3].

To provide high-accuracy power estimation for the software part, the VP should include power-annotated cycle-accurate CPU models. However, in early design stages, the CPU models do not include architectural details (e.g. fully modeled register file, branch prediction, pipeline, ALU, etc.) which are mandatory for the power characterization. For this flow, we developed a power annotation technique for SystemC-based CPU models in higher abstraction levels, where every instruction of the target CPU is executed onto real hardware, in order to measure its power consumption. This measurement includes the power consumption of multiple CPU parts, e.g. ALU, cache misses,
register file, instruction decoding, etc. Thus, there is no need for analytically annotating every single CPU part. Additionally, from such measurements, an estimation of the pipeline, the cache misses and the static power are extracted as well.

One of the goals in hardware prototyping of the heterogeneous system is the rapid and accurate evaluation of the accelerators configuration. In the proposed flow, this is achieved by avoiding long non-productive simulation phases, e.g. the VP and software initialization and system warm-up. To implement such a functionality, we developed a VP framework, which separates a VP into two O/S processes, one for the “static” (including CPU, memory, etc.) and one for the “dynamic” part (including the accelerators). Hence, if during DSE there is a change only on the accelerators, then only the dynamic part is restarted, instead of the whole simulation. The resulted time gains are exploited for sake of metrics accuracy, by annotating the hardware kernels through HLS, in order to find optimized timing, area and power metrics.

Apart from HW/SW parts running on the VP, there is a possibility for the VP to be co-simulated with real hardware devices and/or external software of the host machine. Thus, we developed a set of high-performance Inter-Process Communication (IPC) mechanisms for establishing communication between the host machine and the VP, as well as between the VP and a hardware board [4]. These mechanisms include an interface which is inserted as a VP module and enables the VP software to recognize the external environments. Also, we provide the appropriate libraries for both the host software and the hardware device. The communication mechanisms can be used in any SystemC/TLM-based platform and require the least possible programming effort.

After virtual prototyping, a N-instance many-accelerator VP is deployed. Each instance includes a specific accelerators group, where an application thread is running. Afterwards, DSE is performed in both computation and communication.

DSE for computation focuses on the design of processing elements, targeting to many-accelerator systems. To achieve a fully-optimized system, each accelerators group is customly-configured, according to its exact computational requirements, thus leading to significant area and power savings. However, this results in an exponential increase in the design space. To alleviate this issue, we propose a DSE flow which commits an application characterization stage with the use of analytical models, in order to significantly reduce the design space (typically 8 to 11 orders of magnitude). In addition, we developed a technique for performing massively parallel evaluations of multiple design points in a single VP run.

DSE for communication targets to find optimized interconnection schemes, focusing on the following NoC architectures:

A. Hierarchical NoCs (HNoC) are networks where each node is a sub-network. HNoCs are very efficient interconnection schemes for many-accelerator systems, as each instance is mapped onto a separate sub-NoC and thus can be isolated from the activity of other instances. Also, each sub-NoC can be customized according to the exact communication requirements, to achieve significant timing and power savings. However, the design space of such a network is exponentially increased. Thus, we developed a DSE procedure which utilizes communication characterization stages to prune design solutions of both the sub-NoCs and the top-level NoC that are considered a priori as non-optimal.

B. Heterogeneous 3D NoCs are 3D networks2 which comprise both 2D and 3D routers: The former provide only horizontal connections to neighbors, while the latter also provide vertical connections to the upper/lower layer. Hence, only the most required vertical connections are implemented, which leads to a more efficient NoC design. In this flow, we developed a system-level exploration which searches for optimized combinations of 2D and 3D routers, based on the NoC congestion and packet hops [5]. Afterwards, we propose a 3D implementation flow for the NoC which finds the most efficient mapping of each NoC component on each layer of the 3D stack.

III. EXPERIMENTAL RESULTS

The proposed methodologies of the presented design flow have resulted in faster and more accurate VP simulations, as well as in higher-quality design solutions. Concerning virtual prototyping, the proposed CPU power characterization technique achieves low estimation error ranging from -3% up to +4%, as compared to real hardware measurements, with an average absolute estimation error reaching 2%. The division of a VP into different O/S processes leads to significant time gains reaching 34% on average [6], while the additional use of HLS leads to 77% higher accuracy [7] as compared to manual characterization techniques. The proposed co-simulation mechanisms enable 5.6× and 1.4× faster communication with host and real hardware respectively, as compared to typical communication techniques.

Concerning DSE, the proposed DSE flow for the computational resources of many-accelerator systems achieves better coverage of the design space, resulting to higher-quality solutions. Also, the execution of parallel evaluations in a single VP reduces the DSE time by 5× to 10× as compared to the full-serial DSE. As for communication, the presented DSE approach for HNoCs achieved significant throughput and power gains from 1.7× up to 2× and 1.26× up to 1.48× respectively. Finally, the design flow for 3D NoCs resulted to heterogeneous networks with 25% higher operational frequency and 39% lower power consumption, as compared to homogeneous ones.

IV. CONCLUSIONS

This paper presented a design flow for the prototyping and exploration of many-accelerator heterogeneous systems. The prototyping techniques of the flow achieve accurate and time-efficient VP simulation, by avoiding non-productive simulation phases and focusing on the accurate annotation of both the CPUs and the accelerators, even in higher abstraction levels. Additionally, co-simulation techniques enable the VP interaction with host software and/or real hardware. On the other hand, the DSE methodologies focus on finding high-quality design solutions for both computation and communication, by efficiently covering high-complexity design spaces.

Concerning communication, DSE focuses on emerging NoC architectures, i.e. heterogeneous 3D NoCs and HNoCs. For the analyzed steps of the proposed flow, there have been publications to 3 journals and 11 international conferences.

REFERENCES


1A static part contains modules which remain constantly-configured or do not change frequently.

2We refer to physical 3D NoCs implemented with die-stacking.