When Application-Specific ISA Meets FPGAs: A Multi-Layer Virtualization Framework for Heterogeneous Cloud FPGAs

Extended Abstract

Yue Zha and Jing Li
University of Pennsylvania
{zhayue, janeli}@seas.upenn.edu

1. Motivation

Field-Programmable Gate Arrays (FPGAs) have been widely deployed into private/public cloud platforms to enhance cloud computing. While delivering superior performance for diverse applications (e.g., machine learning, data analysis and graph processing), the high programming complexity largely limits the adoption of FPGA acceleration. Moreover, the inability to manage cloud FPGA resources in an elastic and scalable manner reduces the runtime management efficiency in the dynamic cloud environment. The fundamental reason is that cloud FPGA resources are not fully virtualized. It requires an efficient virtualization framework to exploit the potential of cloud FPGAs.

2. Limitations of the State of the Art

Existing FPGA virtualization mechanisms partially address aforementioned challenges. *Application-Specific* (*AS*) ISA is one promising virtualization mechanism [4, 11, 12, 13, 5, 6, 10, 14] that provides a nice abstraction to substantially reduce the programming complexity of cloud FPGAs (the first challenge). It inherits the advantages of the general-purpose ISA (e.g., x86, PowerPC, ARM and RISC-V) and decouples the software development from the underlying ISA-based FPGA accelerators, thereby providing a simple software programming environment/flow and making FPGA acceleration accessible by the mainstream application software developers. Different from general-purpose ISAs, *AS* ISA fully exploits the customization opportunities from the application itself and provides a customized instruction set to reduce storage/control overhead by generating more compact code. However, as ISA is not originally designed for the *spatial* hardware, the current *AS* ISA-based virtualization solutions only allow to manage the pool of FPGA resources at a per-device granularity by *statically* allocating one FPGA device to one or multiple ISA-based accelerators, resulting in an inefficient runtime management (low elasticity). This static resource management also faces a *NP-hard* bin packing problem due to the diversity of both ISA-based accelerators and FPGAs, which reduces the resource management efficiency. Moreover, *AS* ISA itself lacks native support for scale-out acceleration. ISA-based accelerators need to be *manually* partitioned across the physical FPGA boundary to exploit scale-out acceleration, which could be a time-consuming and error-prone process.

Alternatively, *hardware-specific* (*HS*) abstractions [1, 2, 3, 7, 8, 9, 15, 16] have been proposed to spatially share one FPGA device among multiple FPGA hardware designs at a sub-FPGA granularity, thereby dramatically improving the runtime resource management efficiency (the second challenge). Moreover, some *HS* abstractions [16] also provide essential system support so that FPGA hardware designs can be dynamically deployed into multiple physical FPGAs at runtime without time-consuming recompilation to exploit scale-out acceleration. However, these *HS* abstractions lack a high-level programming model and still require the hardware programming environment/flow with a high programming complexity. Moreover, existing *HS* abstractions typically target *homogeneous* FPGA clusters and it is not trivial to extend them to virtualizing the *heterogeneous* FPGA clusters. Specifically, since FPGA hardware designs describe physical circuits wired together under given spatial resource constraints, *HS* abstractions need to capture these *FPGA-specific* constraints to avoid undesired degradation in the compilation quality. Therefore, *HS* abstractions cannot provide a homogeneous view for the heterogeneous FPGA clusters, which dramatically increases the runtime management complexity and potentially reduces the system performance (e.g., resource utilization).

3. Key Insights

In this paper, we propose to combine *AS* ISAs and *HS* abstractions to efficiently virtualize the *heterogeneous* cloud FPGAs by taking the best of both worlds. Despite its promise, this is not a trivial task mainly due to two reasons. At first, trivially combining these two types of abstractions cannot enable an efficient runtime resource management for heterogeneous FPGA clusters due to their inherent limitations. Moreover, a trivial combination neither exploits the opportunity of improving the runtime performance by leveraging the application-specific customization in *AS* ISAs. This is because the *HS* abstractions are used in runtime management and the application-specific information is not preserved when mapping ISA-based FPGA accelerators onto the *application-agnostic HS* abstractions.

Our main insight is that an new system abstraction that serves as the indirection layer between *AS* ISAs and *HS* abstractions is required to enable an efficient virtualization for heterogeneous cloud FPGAs. Specifically, this new abstraction is designed to hide the *FPGA-specific* constraints and provide a homogeneous view for the heterogeneous cloud FPGAs. It is used for the runtime management to reduce the management complexity, while *HS* abstractions are applied for offline compilation to ensure the mapping quality. This effectively *decouples* the conflicting requirements in the runtime management and compilation. Moreover, this new system abstraction is also designed to effectively capture the *application-specific* customization in *AS* ISAs, which is then leveraged during the runtime management to improve the system performance.

4. Main Artifacts

The main artifact of this paper is a multi-layer virtualization framework that comprises a new system abstraction, a set of custom tools for application mapping, and a runtime management system. Specifically, the proposed system abstraction comprises a pool of soft blocks that adopt a multi-level tree structure to efficiently represent the parallel patterns extracted from the ISA-based accelerators. This is the key differentiator between the proposed system abstraction and previous abstractions. While parallel patterns have been widely used in previous works to simplify parallel programming complexity, they are leveraged in the proposed abstraction to (1) effectively capture the application-specific customization in *AS* ISAs, and (2) reduce the complexity of decomposing ISA-based FPGA accelerators onto the proposed system abstraction. The spatial resource constraints (e.g., type and capacity) of one soft block can be be arbitrarily chosen to abstract away the FPGA-specific hardware details in heterogeneous FPGAs. One soft block is offline mapped into different *HS* abstractions and dynamically deployed into one physical FPGA at runtime.

While the system designers might be able to manually decompose small ISA-based accelerators onto the proposed system abstraction, we develop a custom tool to automate this decomposing process for large and complex accelerators. The compilation tool provided in *HS* abstraction-based solutions is reused to map the decomposed ISA-based accelerator onto *HS* abstractions. The decomposing process could generate a large number of soft blocks for one ISA-based accelerator, increasing the timing complexity of the mapping process. To address this issue, we provide another tool to prune the decomposing results and only map few soft blocks onto *HS* abstractions. This prune process slightly reduces the runtime management flexibility and only leads to a negligible degradation in the runtime system performance.

We then provide a runtime management system that dynamically allocates physical FPGAs for the mapped ISA-based accelerators. It then send requests to the system controller provided by the *HS* abstraction-based solutions for the low-level management/configuration. We also incorporate an optimization technique in the management system that improves the performance of scale-out acceleration by leveraging the application-specific parallel patterns captured in soft blocks.

We evaluate the effectiveness of the proposed multi-layer virtualization framework on a custom-built heterogeneous FPGA cluster. This virtualization framework is not limited to specific *AS* ISA or *HS* abstraction. We use a representative *AS* ISA similar to the one proposed in Microsoft BrainWave [4] and a recent *HS* abstraction [16] as the case study to evaluate its performance.

5. Key Results and Contributions

- We propose to combine *AS* ISAs with *HS* abstractions to efficiently virtualize *heterogeneous* cloud FPGAs. We then identify the two major challenges when combining them and propose to include a new indirection layer in between to fully address these challenges.
- We propose a multi-layer virtualization framework to implement the aforementioned mechanism. It provides a new system abstraction that serves as the indirection layer between *AS* ISAs and *HS* abstractions, and a custom compilation tool to automates the mapping process. The new system abstraction leverages parallel patterns to capture the application-specific customization in ISA-based accelerators, which is then utilized by the runtime management system to improve the performance of scale-out acceleration.
- We evaluate the effectiveness of the proposed virtualization mechanism on a custom-built heterogeneous FPGA cluster. We use a representative *AS* ISA similar to the one proposed in Microsoft BrainWave [4] and a recent *HS* abstraction [16] in our case study. Compared with a system that only uses *AS* ISA, the proposed framework can improve the aggregated system throughput by 2.54× on average with a marginal latency increase.

6. Why ASPLOS

Virtualizing cloud FPGA resources needs both architecture and system support to minimize the virtualization overhead. Therefore, this paper fits well within the scope of ASPLOS in architecture and system (broadly defined).

7. Citation for Most Influential Paper Award

The paper is the first to propose a new virtualization mechanism that can natively support scale-out acceleration across *heterogeneous* cloud FPGAs and demonstrate its effectiveness on a custom FPGA cluster with heterogeneous FPGA resources. Cloud FPGA virtualization mainly targets homogeneous FPGAs in the previous works. The key contribution of this work is to introduce a new system abstraction that can serve as an indirection layer to bridge high-level Application-specific (AS) ISA and low-level Hardware-specific (HS) abstraction. A set of custom tools is developed to implement the proposed virtualization mechanism with high mapping quality. Additional optimization technique and system support are also provided to maximally hide the inter-FPGA communication latency when exploiting scale-out acceleration using multiple heterogeneous FPGAs.

References

[1] Mikhail Asiatici, Nithin George, Kizheppatt Vipin, Suhaib A Fahmy, and Paolo Ienne. Virtualized Execution Runtime for FPGA Accelerators in the Cloud. *IEEE Access*, 5:1900–1910, 2017.

[2] Stuart Byma, J Gregory Steffan, Hadi Bannazadeh, Alberto Leon Garcia, and Paul Chow. FPGAs in the Cloud: Booting Virtualized Hardware Accelerators with OpenStack. In *2014 IEEE 22nd Annual International Symposium on Field-Programmable Custom Computing Machines*, pages 109–116. IEEE, 2014.

[3] Fei Chen, Yi Shan, Yu Zhang, Yu Wang, Hubertus Franke, Xiaotao Chang, and Kun Wang. Enabling FPGAs in the Cloud. In *Proceedings of the 11th ACM Conference on Computing Frontiers*, pages 1–10, 2014.

[4] Jeremy Fowers, Kalin Ovtcharov, Michael Papamichael, Todd Massengill, Ming Liu, Daniel Lo, Shlomi Alkalay, Michael Haselman, Logan Adams, Mahdi Ghandi, et al. A Configurable Cloud-Scale DNN Processor for Real-Time AI. In *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*, pages 1–14. IEEE, 2018.

[5] Kaiyuan Guo, Lingzhi Sui, Jiantao Qiu, Jincheng Yu, Junbin Wang, Song Yao, Song Han, Yu Wang, and Huazhong Yang. Angel-Eye: A Complete Design Flow for Mapping CNN onto Embedded FPGA. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(1):35–47, 2017.

[6] Nachiket Kapre. Custom FPGA-based Soft-Processors for Sparse Graph Acceleration. In *2015 IEEE 26th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, pages 9–16. IEEE, 2015.

[7] Ahmed Khawaja, Joshua Landgraf, Rohith Prakash, Michael Wei, Eric Schkufza, and Christopher J Rossbach. Sharing, Protection, and Compatibility for Reconfigurable Fabric with AmorphOS. In *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*, pages 107–127, 2018.

[8] Oliver Knodel, Paul R Genssler, and Rainer G Spallek. Virtualizing Reconfigurable Hardware to Provide Scalability in Cloud Architectures. In *International Conference on Advances in Circuits, Electronics and Micro-electronics (CENICS)*, 2017.

[9] Oliver Knodel and Rainer G Spallek. RC3E: Provision and Management of Reconfigurable Hardware Accelerators in a Cloud Environment. *arXiv preprint arXiv:1508.06843*, 2015.

[10] Rui Ma, Jia-Ching Hsu, Tian Tan, Eriko Nurvitadhi, David Sheffield, Rob Pelt, Martin Langhammer, Jaewoong Sim, Aravind Dasu, and Derek Chiou. Specializing FGPU for Persistent Deep Learning. In *2019 29th International Conference on Field Programmable Logic and Applications (FPL)*, pages 326–333. IEEE, 2019.

[11] Jian Ouyang et al. XPU: A Programmable FPGA Accelerator for Diverse Workloads. In *2017 IEEE Hot Chips 29 Symposium*, 2017.

[12] Andrew Putnam, Adrian M Caulfield, Eric S Chung, Derek Chiou, Kypros Constantinides, John Demme, Hadi Esmaeilzadeh, Jeremy Fowers, Gopi Prashanth Gopal, Jan Gray, et al. A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services. In *2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, pages 13–24. IEEE, 2014.

[13] Aaron Severance and Guy GF Lemieux. Embedded Supercomputing in FPGAs with the VectorBlox MXP Matrix Processor. In *2013 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ ISSS)*, pages 1–10. IEEE, 2013.

[14] Chao Wang, Lei Gong, Fahui Jia, and Zhou Xuehai. An FPGA based Accelerator for Ubiquitous Clustering Applications with Custom Instructions. *IEEE Transactions on Computers*, 2020.

[15] Jagath Weerasinghe, Francois Abel, Christoph Hagleitner, and Andreas Herkersdorf. Enabling FPGAs in Hyperscale Data Centers. In *2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom)*, pages 1078–1086. IEEE, 2015.

[16] Yue Zha and Jing Li. Virtualizing FPGAs in the Cloud. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 845–858, 2020.