FPGA as a RESTful Service: Release Your Accelerator From the PCIe Cage!

Fabio Maschi
Systems Group, Dept. of Computer Science
ETH Zurich, Switzerland

Gustavo Alonso
Systems Group, Dept. of Computer Science
ETH Zurich, Switzerland

ABSTRACT

FPGAs have proven to be an efficient execution node to accelerate compute-intense tasks in distributed system. However, keeping them as a primary PCIe-attached device remains a limiting factor for their broader adoption: (i) the tight coupling of technology stacks increases development and maintenance costs; (ii) PCIe communication pattern reduces the potential for low-latency and high-bandwidth acceleration; and (iii) cloud offerings are restricted to few settings. In this work, we present Strega, an HTTP server-side stack for FPGA kernels, making them available as a RESTful micro-service directly over the network. The enhanced hardware abstraction facilitates software integration, while offering deterministic, orders of magnitude higher performance.

1 INTRODUCTION

The PCIe bus has been the traditional interconnect for accelerators, such as FPGAs, GPUs and TPUs. It exposes the hardware lanes to the host CPU and operates with PCIe transactions (i.e., data read/write, interruptions, configuration). To issue a full kernel invocation, the host needs to (i) allocate memory in the device; (ii) transfer the input data to the device's memory; and (iii) trigger the start of the kernel. The accelerator then (iv) fetches the input data; (v) executes it; (vi) writes the result data back into its memory; and (vii) notifies the host of completion, when it can finally (viii) fetch the result data to its local memory. This long sequence of synchronisation, transfers, and messages reduces the potential of acceleration for latency-sensitive functions, very often requiring batch processing to achieve high throughput, at the cost of high latency. Furthermore, the low-level communication of the PCIe architecture imposes vendor-specific drivers support, and restricts the number of processes able to interact with the device.

Other than software integration costs, FPGA-based applications suffer from current reduced cloud offerings. Since the PCIe bus is the *status quo*, cloud providers offer powerful FPGA devices attached to a relatively small CPU (e.g., 8 vCPU per FPGA device on AWS). To maximise server utilisation, applications can be co-located in the host CPU, but encounter a scalability problem: as long as the application running on only 8 vCPUs cannot generate enough workload for the FPGA kernel, the accelerator is under-utilised. Similarly, if the FPGA instance needs to scale out, another CPU instance will also be allocated. In fact, disaggregation is one of the key principles allowing high scalability in distributed systems, and the imposed provisioning of FPGAs tight to CPUs makes scalability a complex equation to solve.

2 SOLUTION

Modern FPGA devices support direct network connection, i.e., without the intervention of the host CPU. To leverage this feature and tackle integration issues of PCIe-based kernels, we present Strega, an HTTP server-side implementation built on hardware, which allows to offload the entire network stack to an FPGA board. In this design, the FPGA becomes a first-class computing node in a distributed system, abstracted away thanks to the HTTP interface. Kernel invocation is turned into a RESTful call over the network, just like a standard CPU micro-service call in a distributed system.

To evaluate the proposed architecture, we performed 5000 function invocation calls from the CPU to: (α) a PCIe-attached FPGA kernel; (β) an HTTP-interfaced kernel on the FPGA; and (δ) a commercial CPU HTTP server baseline, nginx. β shows a consistent lower latency compared to α, having about an order of magnitude faster latency, as low as 16 µs compared to 108 µs. δ distribution ranges from 15 µs up to 85.8 ms. An interesting point to highlight comes from the sample variance, or lack of: only 4 out of 5000 measurements in β have latency higher than 30 µs, and the gap between the minimum and the 95th percentile is only 3 µs, while for α and β this gap represents 23 µs and 85 µs respectively.

In terms of number of function invocations per second (with 512 bit payload), α reaches up to 348 `Kips`. δ suffers from PCIe back and forth transaction synchronisations and reaches only 66 `Kips`, while β can sustain a global throughput of 1.7 `Mips`, aggregating workload generated by twelve CPU HTTP benchmark clients without performance loss.

By increasing the hardware abstraction of RESTful calls over the network in the communication between client applications and kernels, FPGAs can be more easily adopted as micro-services, while preserving all the performance advantages of heterogeneous computing. The opportunity of such standardised interface should help making FPGAs first-class processing units in the cloud, removing the requirement of necessarily instantiating a CPU for each FPGA board.