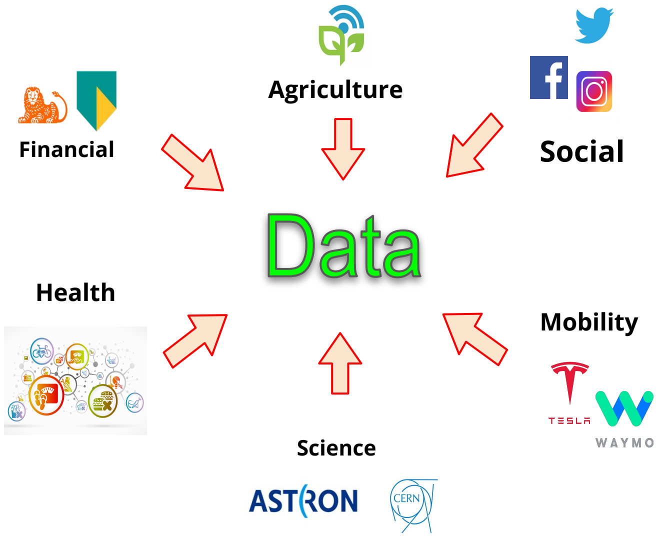
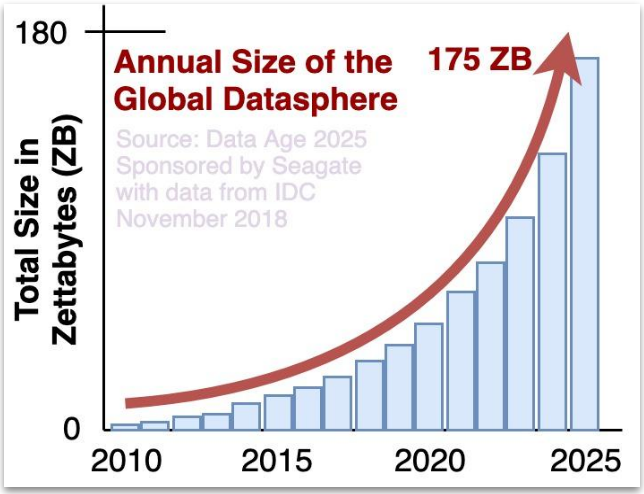
Hyperion: A Unified, Zero-CPU Data-Processing Unit (DPU)

Marco Spaziani Brunella, Marco Bonola and **Animesh Trivedi**

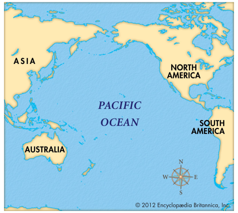
CompSys 2022

The Data Explosion





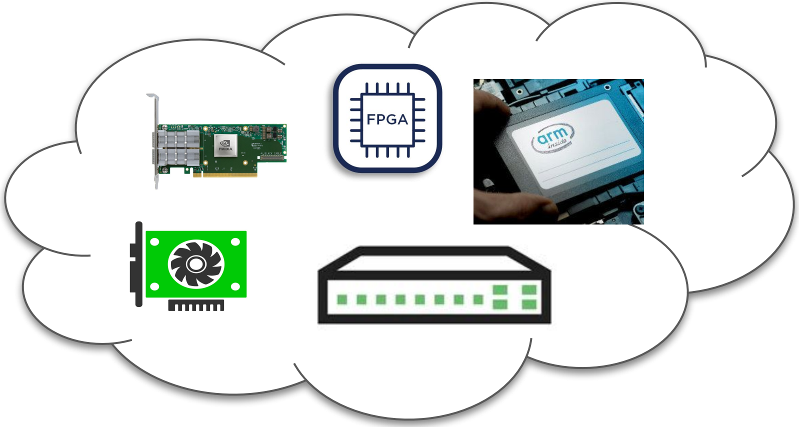
200 Zettabytes *revised



Fills the Pacific Ocean 200x over

https://cybersecurityventures.com/the-world-will-store-200-zettabytes-of-data-by-2025/

CPU - as the Performance Horse

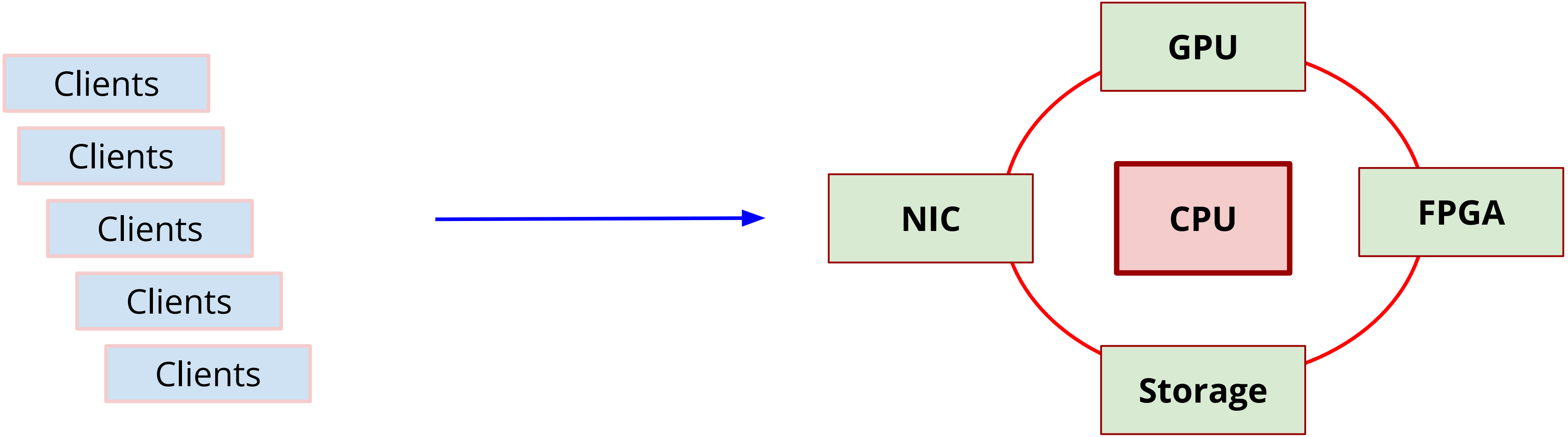


- **Stalling** of Moore's Law and Dennard Scaling
- Turing Tax - **the cost of Generalization**
- **Security** considerations
- **Energy needs**

→ ***Rise of accelerator-centric computing***

John L. Hennessy and David A. Patterson. 2019. A new golden age for computer architecture. Commun. ACM 62, 2 (February 2019), 48–60. DOI: https://doi.org/10.1145/3282307

Imagine this setup



Disaggregated clients ***Network protocols*** ***Interaction among the accelerators***

The Key Challenges with the CPU in the Loop

1. The CPU coordinates the control path and resource allocation
   a. Coordinate control flow among accelerators - which buffers to allocate, pin, DMA
   b. Control the data transfer among accelerators - when to initiate and how to initiate
   c. Done with pair-wise accelerator integrations, but multiple?

2. The CPU dictates the computing abstractions
   a. Shared memory, virtual memory, processes, context switches, files
   b. ***Keeping the memory coherent between the host's view and accelerator view***

3. The CPU limits the innovation and imagination
   a. Active and passive disaggregation
   b. Designing a new interconnect, network discovery protocols
   c. Scalable energy needs

Hyperion: A Zero-CPU Data Processing Unit (DPU)

Hardware:

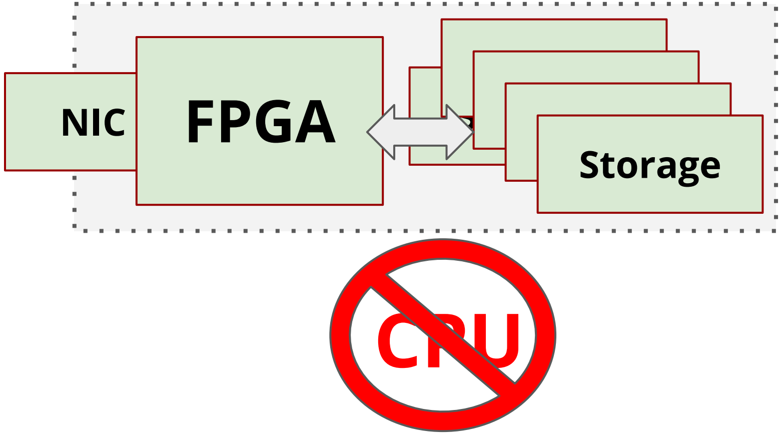
- FPGA + NIC + Storage = DPU

Software:

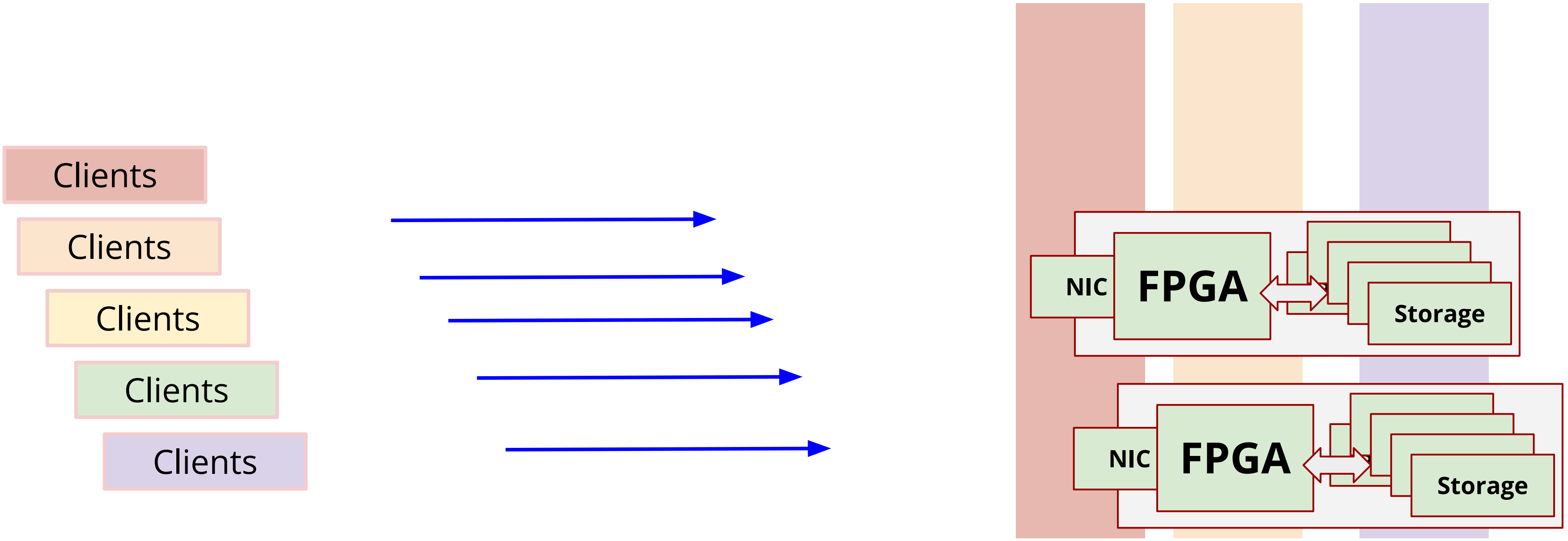
- A new compiler
- eBPF as an **IR** for *__(any)__* hardware

Client:

- *Disaggregated clients*
- Network protocols - NVMoF
- Application-level, KV, NFS, DSes



Disaggregation and Slicing



Innovation in Discovery, reconfiguration, slicing, virtualization, communication etc.

Comments on the Reviews

First of all, thank you :)

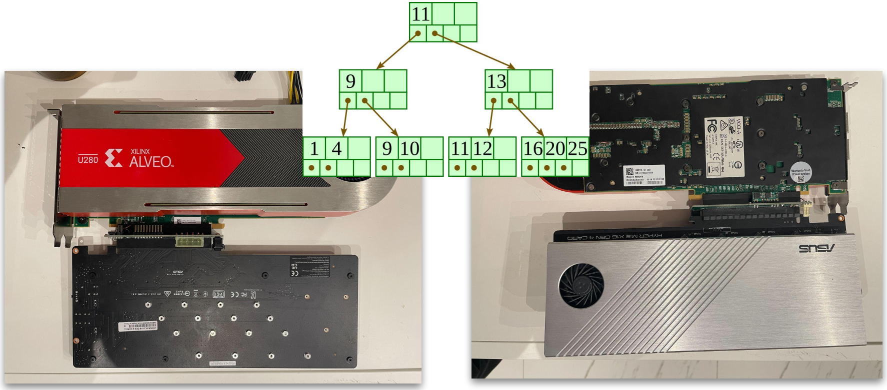
- ***Target application-domain?***
  - Disaggregated, cloud storage and processing
  - Mostly well-defined, requires multi-tenancy and dynamic reconfiguration
- ***Limited FPGA resources, esp. on-chip memories***
  - Needs data staging primitives between SRAM, DRAM, HBM, then NVMe storage
- **Development complexity**
  - Target well-defined data structures as the basic building blocks: B-arr Tree, Hash Tables, Arrays, LSM tree, Heaps, extent-trees, etc.
- **Compiler development:** challenging, but feasible
- ***"I wonder if this approach can really fully eliminate CPUs"***
  - We also do not know. We think it can, but we are open to hear counter arguments

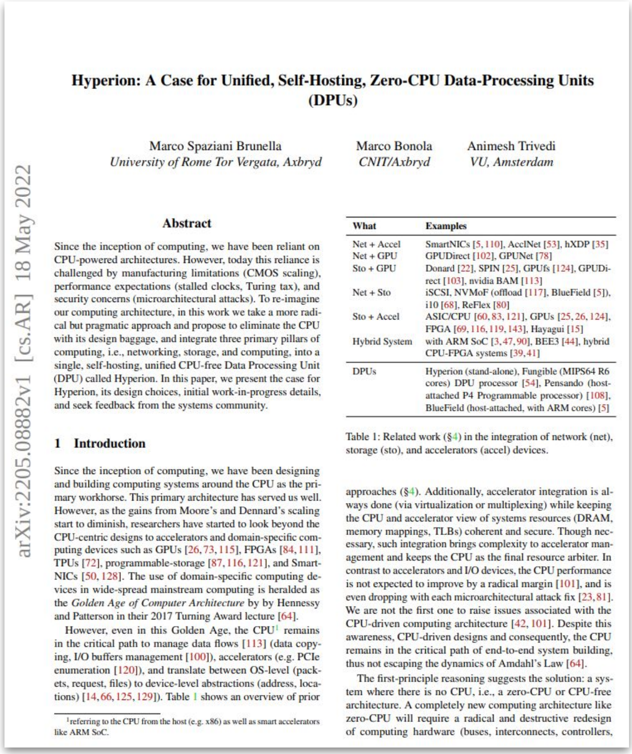
Where are we going from here?

5-page vision:

Hyperion: A Case for Unified, Self-Hosting, Zero-CPU Data-Processing Units (DPUs)

https://arxiv.org/abs/2205.08882





arXiv:2205.08882v1 [cs.AR] 18 May 2022

Hyperion: A Case for Unified, Self-Hosting, Zero-CPU Data-Processing Units (DPUs)

Marco Spaziani Brunella
University of Rome Tor Vergata, Axbryd

Marco Bonola
CNIT/Axbryd

Animesh Trivedi
VU, Amsterdam

Abstract

Since the inception of computing, we have been reliant on CPU-powered architectures. However, today this reliance is challenged by manufacturing limitations (CMOS scaling), performance expectations (stalled clocks, Turing tax), and security concerns (microarchitectural attacks). To re-imagine our computing architecture, in this work we take a more radical but pragmatic approach and propose to eliminate the CPU with its design baggage, and integrate three primary pillars of computing, i.e., networking, storage, and computing, into a single, self-hosting, unified CPU-free Data Processing Unit (DPU) called Hyperion. In this paper, we present the case for Hyperion, its design choices, initial work-in-progress details, and seek feedback from the systems community.

1 Introduction

Since the inception of computing, we have been designing and building computing systems around the CPU as the primary workhorse. This primary architecture has served us well. However, as the gains from Moore's and Dennard's scaling start to diminish, researchers have started to look beyond the CPU-centric designs to accelerators and domain-specific computing devices such as GPUs [26, 73, 115], FPGAs [84, 111], TPUs [72], programmable-storage [87, 116, 121], and SmartNICs [50, 128]. The use of domain-specific computing devices in wide-spread mainstream computing is heralded as the *Golden Age of Computer Architecture* by by Hennessy and Patterson in their 2017 Turning Award lecture [64].

However, even in this Golden Age, the CPU[1] remains in the critical path to manage data flows [113] (data copying, I/O buffers management [100]), accelerators (e.g. PCIe enumeration [120]), and translate between OS-level (packets, request, files) to device-level abstractions (address, locations) [14, 66, 125, 129]). Table 1 shows an overview of prior approaches (§4). Additionally, accelerator integration is always done (via virtualization or multiplexing) while keeping the CPU and accelerator view of systems resources (DRAM, memory mappings, TLBs) coherent and secure. Though necessary, such integration brings complexity to accelerator management and keeps the CPU as the final resource arbiter. In contrast to accelerators and I/O devices, the CPU performance is not expected to improve by a radical margin [101], and is even dropping with each microarchitectural attack fix [23, 81]. We are not the first one to raise issues associated with the CPU-driven computing architecture [42, 101]. Despite this awareness, CPU-driven designs and consequently, the CPU remains in the critical path of end-to-end system building, thus not escaping the dynamics of Amdahl's Law [64].

The first-principle reasoning suggests the solution: a system where there is no CPU, i.e., a zero-CPU or CPU-free architecture. A completely new computing architecture like zero-CPU will require a radical and destructive redesign of computing hardware (buses, interconnects, controllers,

What	Examples
Net + Accel	SmartNICs [5, 110], AcclNet [53], hXDP [35]
Net + GPU	GPUDirect [102], GPUNet [78]
Sto + GPU	Donard [22], SPIN [25], GPUfs [124], GPUDirect [103], nvidia BAM [113]
Net + Sto	iSCSI, NVMoF (offload [117], BlueField [5]), i10 [68], ReFlex [80]
Sto + Accel	ASIC/CPU [60, 83, 121], GPUs [25, 26, 124], FPGA [69, 116, 119, 143], Hayagui [15]
Hybrid System	with ARM SoC [3, 47, 90], BEE3 [44], hybrid CPU-FPGA systems [39, 41]
DPUs	Hyperion (stand-alone), Fungible (MIPS64 R6 cores) DPU processor [54], Pensando (host-attached P4 Programmable processor) [108], BlueField (host-attached, with ARM cores) [5]

Table 1: Related work (§4) in the integration of network (net), storage (sto), and accelerators (accel) devices.

[1] referring to the CPU from the host (e.g. x86) as well as smart accelerators like ARM SoC.

Acknowledgements: NWO XS OCENW.XS3.030, and the Xilinx University Donation Program (XUP)

Call for a Revolution!