CPU-free Computing: A Vision with a Blueprint

Animesh Trivedi
Vrije Universiteit Amsterdam

Marco Spaziani Brunella
Hyperable

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 from data center computing. We 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. The elimination of the CPU from computing necessitates re-thinking our computing, networking, and storage abstractions, and tackle the associated challenges which we sketch in this paper. We share the blueprint of our work-in-progress, Hyperion's hardware and software stack, and seek feedback.

CCS CONCEPTS

• **Hardware** → **Hardware accelerators**; **Reconfigurable logic and FPGAs**; *Hardware description languages and compilation*; • **Software and its engineering** → *General programming languages*; **Operating systems**; **Secondary storage**; **Secondary storage**.

KEYWORDS

CPU-free computing, Accelerators, Programming, Data storage, Data Processing

ACM Reference Format:
Animesh Trivedi and Marco Spaziani Brunella. 2023. CPU-free Computing: A Vision with a Blueprint. In *Workshop on Hot Topics in Operating Systems (HotOS '23), June 22–24, 2023, Providence, RI, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3593856.3595906

HotOS '23, June 22–24, 2023, Providence, RI, USA

ACM ISBN 979-8-4007-0195-5/23/06.
https://doi.org/10.1145/3593856.3595906

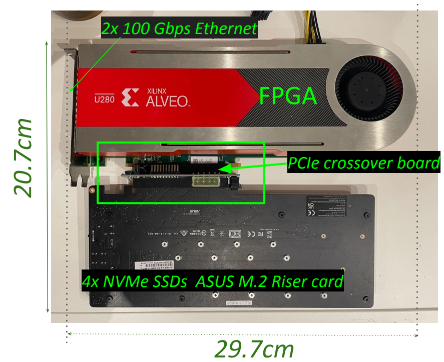

Figure 1: A CPU-free Hyperion prototype with a unified 100 Gbps network, U280 FPGA, and NVMe SSDs.

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 for the CPU start to diminish (Turing tax, complexity, security challenges [36, 56, 57, 70, 97]), researchers have started to look beyond the CPU-centric design to domain-specific accelerators such as GPUs [27, 86, 140], TPUs [83], computational storage devices (CSDs, NVMe TP-4091) [105, 141, 146], SmartNICs [55, 157], Reconfigurable Architectures (CGRAs [166] and FPGAs [100, 135]). The use of *specialized* domain-specific hardware in mainstream computing is heralded as the *Golden Age of Computer Architecture* by Hennessy and Patterson in their Turing Award lecture [71].

The CPU Problem: However, even in this Golden Age, the CPU[1] remains in the critical path to manage data flows [137] (data copying, I/O buffers management [122]), accelerators (e.g. complex PCIe enumerations [145]), and translate between OS-level (packets, processes, files) to device-level abstractions (memory and block addresses) [17, 73, 153, 158]). Much of the current state-of-the-art efforts are still focused on minimizing the CPU involvement in control and data paths between the accelerators (see Table 1). 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 an integration brings

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

GPU-with-network [93, 125]	Does not have or consider any storage integration
GPU-with-storage [23, 26, 124, 137, 151]	CPU-assisted storage translation, no or limited networking support
FPGA/ARM SoC-with-Network [37, 54, 58, 113, 134, 135]	Does not have or consider storage integration
Storage-with-Network [75, 95, 109, 126, 142]	Block-level protocols only (iSCSI, NVMoF), no support for file systems
Storage-with-accelerator [27, 67, 80, 87, 99, 141, 144, 146]	CPU does the file system/translations, no/limited network support
Commercial domain-specialized DPUs [59, 126, 131]	DPU designed around specialized CPU cores (P4, ARM, MIPS64)

Table 1: Overview of the state-of-the-art efforts in decreasing the CPU involvement in computing *while* maintaining CPU-centric memory and storage abstractions when doing *pair-wise* accelerator interactions.

complexity in the accelerator management and keeps the CPU as the final resource arbiter. We are not the first ones to raise issues associated with the CPU-centric computing or the CPU overheads with accelerators [16, 47, 123, 150] (see Related work in §3). However, unlike the previous efforts, we are making a case that it is not only the CPU, but the CPU-associated, CPU-centric abstractions (processes, virtual memory, coherency, sharing, caches, storage-memory hierarchy) that need to be reconsidered as well. The design of these abstractions predate the emergence of accelerators. Without re-imagining, the CPU and consequently, the CPU-centric computing abstractions remain in the critical path of end-to-end system building, thus not escaping the dynamics of Amdahl's Law.

A case for CPU-free computing: In this work, we enquire a more foundational and far-reaching question: *how would computing look today if we were only given accelerators to design a computing system from scratch?* Our position for CPU-free computing is inspired from the challenges around:

(1) *Shared and coherent virtual memory:* A direct consequence of keeping a CPU-centric design is to inherit its choices of memory addressing, translation, and protection mechanisms. When an accelerator such as an FPGA[2] is attached to a CPU as an external device [41] or as a co-processor [44], there is a temptation to provide/port the familiar memory abstractions like unified virtual memory [100] and/or shared memory [115]. Virtual memory is a deceptively simple idea [50] whose implementation on modern CPUs with multicore, caches, nested page tables, prefetchers, virtualization, TLBs, IOMMUs have been known to be a source of major complexity, overheads, security vulnerabilities, and energy inefficiencies [22, 45, 65, 154]. When an accelerator (GPUs, FPGAs, CSDs) is integrated with a CPU-centric host system, it multiplies this complexity in an attempt to keep the CPU's view of the system coherent [100, 115]. ETH's Enzian system, which is a hybrid CPU-FPGA dual socket system, reports the heroic engineering effort it took to re-design all systems components to integrate an FPGA as a co-processor *with a CPU* [44]. Not only the virtual memory, but the CPU's view a flat, physical memory is also outdated in presence of multiple accelerators [9].

(2) *Persistence and storage hierarchy:* Due to the two-level of storage/memory hierarchy, both DRAM (memory) and storage use different storage abstractions. Examples of storage abstractions are block-oriented formats (e.g., file formats ORC, Parquet [159]), files, directories, data structures (B+/LSM trees), etc. DRAM uses ephemeral pointer-based data structures in virtual memory. As a result, there is an abstraction translation process done by the CPU (and systems software) whenever storage is accessed by an accelerator to translate storage abstraction (*data-at-rest*) to memory abstraction (*data-in-motion*). Recent works in persistent memory heaps, file systems, and OS designs grapple with the translation complexity, and question even if there is a need for such a translation [31, 43, 74, 85, 110]. No significant improvements are expected in the process of abstraction translations as very modest CPU performance improvements are projected in the future [1, 78].

(3) *Disaggregation:* The CPU-centric design encourages the *active* resources disaggregation where resources remain attached to a host CPU that manages the disaggregation logic. This design results in a coarser disaggregation granularity with complex and bloated software [60] and a tight integration of processor/memory [68, 147]. To achieve the vision painted by Han et al. in their seminal HotNet'13 paper [69], there is a renewed push for *passive* disaggregation where the disaggregation logic/smartness lies with clients, and a remote resource only serves fast datapath requests [13, 38, 68, 147, 160]. Passive disaggregation promotes a *network-attached model*, where self-hosting memory, storage, DPUs, and ASICs are directly connected to a network. Such a design encourages innovations in: (i) discovery and configuration network protocols (e.g., Catapult fabric [135]); (ii) work division between clients and remote servers for distributed resource allocation, and access (e.g., Clio [68], DUA [149]); and (iii) offload-friendly abstractions with isolation, multiplexing mechanisms (e.g., group offloading and memory re-assignments [13, 92, 112]).

[2] Using an FPGA as a canonical example for broader non-CPU devices.

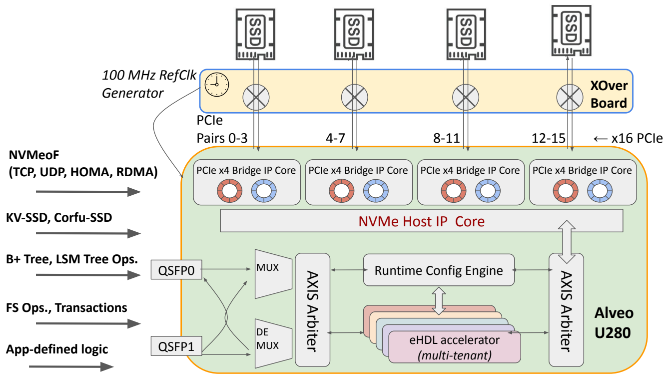

Figure 2: Schematic diagram of the Hyperion.

To summarize: The first-principle reasoning suggests a solution to tackle the aforementioned challenges: a system where there is no CPU, i.e., a CPU-free architecture. The CPU-centric design has its merits (with challenges), and its elimination is *not recommended* for every workload in general computing. Here we focus on specialized, accelerator-amenable data center workloads for which we aim to explore the CPU-free design space (see §2.4).

2 DESIGNING A CPU-FREE SYSTEM

In order to explore the CPU-free computing design space, in this work, we design a CPU-free DPU called Hyperion. The DPU is built around a Xilinx Alveo U280 board with 2x100 Gbps Ethernet QSFP [2], a PCIe cross overboard [51] to attach 4x NVMe devices to the U280 with power (figure 1). Commercially, NICs and storage devices are sold as separate PCIe devices. Communication between the two requires control coordination with P2P DMA from the CPU (if supported, e.g., NVMe Controller Memory Buffers (CMBs) [24]) via the PCIe root complex. To make the DPU self-hosting, Hyperion runs a PCIe root complex with an NVMe controller on the FPGA board. The FPGA (x16) PCIe lanes are connected to off-the-shelf NVMe storage devices via a PCIe bifurcation. Hence, all access to the storage is funneled through the FPGA. Hyperion follows the directly network-attached model that has been used before [80, 135, 149, 164], but extends it with storage integration. With such a design, Hyperion now has an *end-to-end hardware path* from network to FPGA to storage devices without any CPU. The end-to-end hardware path can be *specialized* with workload-specific abstractions with an application-defined network transport (TCP, UDP, RDMA, HOMA [127]), storage API (NVMoF [142], KV [28], ZNS [32]), and optimizations [75, 76, 95, 121].

The DPU boots in a *stand-alone mode* without any CPU when power is applied and FPGA JTAG self-tests are passed. The DPU is currently attached to a host-system via USB for programming. We are in the process of developing an OS-shell and control path over the network that can program the FPGA without a CPU, leveraging Partial Dynamic Reconfiguration through the Internal Configuration Access Port (ICAP) of the FPGA. In comparison to a conventional 1U rack-mounted server like SuperMicro X12, Hyperion is 5-10× more compact in volume, and 4-8× more energy efficient with the maximum TDP energy specifications (approx. 230 Watts vs 1,600 Watts). Figure 2 shows the overall schematic diagram of the different components and how they are connected on the hardware level.

In comparison to past such efforts, Hyperion presents a complete, self-hosted, network-attached system that does not rely on any host CPU/OS services to support workloads, thus reducing the integration complexity and overheads. Hyperion can be used as a (low-cost) research platform to explore various CPU-free hardware and software techniques.

The choice of FPGA to explore the CPU-free model is governed by its three strengths: (1) *Application-specific reconfigurability:* The use of FPGA allows us to reconfigure hardware (deep pipelines, unrolled loops, data parallelism, large caches, memories: SRAM, DRAM, or HBM) to the best possible implementation for a wide variety of application-specific logics [8, 12, 40, 96, 161]. (2) *Improved FPGA systems software support:* With the availability of high-quality DSLs [19, 79, 98, 143], OS-shells [89, 100], HDL DSL compilers [37], and debuggers [116, 167], it has become more affordable to generate high-quality HDL codes. (3) *Predictable performance with energy efficiency:* FPGAs excel in coarse-grained spatial multiplexing with longer time-scales (10-100 msecs, partial reconfiguration) [89, 100]. This sharing model helps with building a highly *predictable execution pipeline* where once an associated bitstream has been sent to the FPGA, the circuit runs a certain clock frequency without any outside interference, thus delivering energy efficient [37, 136, 141] and predictable performance [81, 108]. As the availability of open-source EDA processes and projects improve, in the future we can explore workload-specific ASIC-centric IP designs as well [3, 5].

2.1 Memory and Storage Model

Not having any host-attached CPU resources makes the integration of the memory and storage model less complex. In Hyperion, we leverage a segmentation-based, single-level unified storage-memory addressing with 128-bits objects (inspired from Twizzler [31]). In the current implementation, we statically divide FPGA AXI-streaming bus address ranges to map to FPGA DRAM addresses, and others to NVMe PCIe BAR addresses. Hence, the total addressable capacity is DRAM plus NVMe storage capacities. The segment location translation is done using a segment translation table that maps a segment id (128 bits) to their bus addresses and to their location, DRAM or NVMe. The current allocation of segments to their locations (DRAM or NVMe) is static. It is done based on their bus addresses. However, we expect hints-based allocation should also be possible where temporary and/or performance-critical objects are allocated or eventually promoted to DRAM or HBM. One can treat all segments as ephemeral and use NVMe just as a large capacity location. When durability is required, all durable segments must also be allocated on NVMe addresses. The segment translation table is periodically persisted on a pre-selected control/boot NVMe area. The unique aspect of segmentation-based location translation is that it is coarser (object-based) than virtual memory (page-based), thus reducing overheads associated with the virtual memory translation [10].

Segmentation-based addressing is already in use with FPGAs with heterogeneous memory locations [11, 89]. Single-level data stores like MULTICS and Atlas [46, 91] pioneered the idea of hiding an object's location and are precursor to the virtual memory idea. Due to the ephemeral nature of memories, the addressing mechanism with memories were never integrated with the storage to make them durable. IBM AS/400 [77] and EROS [148] are closest to our approach with single-level, segmentation-based persistent objects. In the future, as more accelerator are integrated in Hyperion, we consider leveraging the CXL protocol to support application-specific coherence, if required [64, 84].

2.2 Programming the Hyperion DPU

Inspired by the LLVM project, in this work, we argue that FPGA programming needs to decouple the frontend (application logic) and backend (HDL codes) with an accelerator-independent, intermediate representation (IR) language. The IR can be used to reason about correctness and safety properties of the program, with compiler-assisted transformations for pointer swizzling and privilege calls. We make a case that the extended Berkeley Packet Filter (eBPF) [42, 120] language is a suitable match for such an IR for three key reasons. First, eBPF is not tied to a specific application-domain and it is used in networking [7, 72], tracing [66], caching [62], security [88], and (very successfully) storage [21, 29, 101, 114, 173]. It is also supported by healthy, growing communities (Cilium, the eBPF foundation), thus establishing expertise and a knowledge base. Second, due to the simplified nature of the eBPF instruction set, it is possible to verify and reason about its execution. The Linux kernel already ships with an eBPF verifier [156] (with simplified symbolic execution checks). Lastly, eBPF supports efficiently generating codes for multiple hardware devices such as x86, ARM, or FPGAs, thus solidifying its position as an accelerator-independent unifying IR [90].

Bear in mind, here we take a broader position regarding eBPF where the Linux kernel implementation is one of many possible implementations of an eBPF execution environment. For example, there are userspace BPF VMs [6], checkers [61], and application-specific ISA extensions [37]. Hyperion can use any eBPF-supporting programming language as a frontend. It then uses `clang/LLVM` to generate eBPF IR from the frontend. We are developing a code generation pipeline from eBPF-to-HDL using a set of open-source compilers for parallelism extraction, and then eBPF instructions specific HDL code generation, fusion, and wrapping in hardware [35, 37, 139]. Apart from eBPF, we also consider P4, another popular programming language for in-network acceleration (NICs and switches). However, P4 programs are designed around packet processing and network abstractions. In restricted capabilities (with only filtering and forwarding), there are P4 to eBPF compilers available, though the generality of P4 for general data processing is yet to be explored.

We expect to leverage the already established slot-style spatial slicing of FPGA resources [89, 100] or a compiler-assisted workload partitioning for multi-FPGA deployments [170]. Hyperion can run a privileged configuration kernel that can receive authorized, encrypted FPGA bitstreams over a certain control network port and assign slices to it.

2.3 Storage Abstractions: Files and Objects

Beyond supporting block-level offloaded accesses to storage (NVMoF), in this section we explore how Hyperion can support higher-level familiar abstractions like file systems and workloads-level data objects without any CPU support. The key challenge here is how to resolve a higher-level object such as a columnar data to its storage location where multiple storage layers (formats, file systems) do abstraction translation. Inspired by the Internet that decouples packet processing from its well-defined packet formats (e.g., the TCP/IP formats), we propose to decouple data formats from their accessing explicitly using an annotation-based, domain-specific language (DSL). Starting at the file-system level, prior research from Sun et al. show that such a file-system layout annotation can be generated efficiently for ext4 and F2FS files systems [155]. The availability of annotation enables us to generate file system layout and metadata access codes (in C/C++), thus accessing directories and files directly. These annotated codes can further be translated to HDL codes using the Hyperion compiler. As a next step, we target well-defined application-level object formats Parquet (on storage) and Arrow (in-memory) that are used in a variety of data processing pipelines [14, 15] with FPGA support for their formats [103, 130]. With the file system access annotation, we expect to build an end-to-end Parquet/Arrow object access pipeline in hardware with end-to-end optimizations [129]. With such capabilities, Hyperion can access and process data that is stored in Arrow/Parquet format, on the F2FS/ext4 file system on NVMe storage without any host-side, or client-side CPU involvement. A file-, object-, or datastructure-based interface to storage can co-exist in Hyperion.

2.4 Client Interface and Workloads

There are two classes of workloads here: (C1) mixed distributed workloads where a mix of CPU servers and CPU-free Hyperion DPUs run in a distributed networked; and (C2) pure Hyperion workloads where an application runs completely in a CPU-free mode over one or multiple DPUs. Currently we are focusing on developing the latter (C2) for a single DPU with three applications. First, high data volume network middleware applications such as fail2Ban [4] or load-balancers [169]. These applications have traffic-flow proportional states that either need to be persisted (in case of fail2Ban that needs to log network traffic data persistently) or require large temporary data storage (e.g., Tiara offloads load-balancing state from FPGAs to x86 servers [169]). These network middleware applications can run in a pure, stand-alone mode on Hyperion with attached SSDs. Second, a latency-sensitive application such as network pointer-chasing. In a disaggregated storage, pointer chasing over B+ trees, extent trees, LSM trees (used in many databases, file systems, and key-value stores [133]) results in multiple network RTTs with significant performance degradation [101, 104]. These latency-sensitive applications can now be deployed in the FPGA even if they access higher-level data objects. Lastly, network-attached SSDs that can export application-defined, high-level, fault-tolerant data structures and abstractions (pioneered by Boxwood [117]) such as trees, lookup-tables [28], distributed/shared ordered logs [20, 165], atomic writes [128] with transactional interfaces (left side of figure 2).

For mixed, distributed workloads, we take inspiration from the flexible RPC interface pioneered by Willow [146]. The RPC interface can be *specialized* end-to-end with network, storage, and application-level protocols. For example, we can build network-attached SSDs that can support Corfu consensus protocol [20, 165], or remote file system access acceleration with DPUs using `virtio-fs` [63, 132], or RDMA acceleration for a flash file system [158], or the bump-in-the-wire/near-data execution of application-provided codes (B+/LSM tree search, compaction and insertions, file system walks, transactions) [141, 171]. Here, we can leverage client-driven request routing [111] with a shared-nothing, run-to-completion data path [25] for performance. Further possibilities exist with network transport and scheduling optimizations [75, 76, 121, 127]. As more applications get developed, there will be a need to manage end-to-end CPU-free design patterns efficiently, with possibilities of sharing and composibility among different workloads even in presence of failures [162].

2.5 Compilers as the new operating system?

Due to the absence of the conventional CPU, doing the classical resource arbitration with elevated privileges to mediate accesses to a shared resource in Hyperion would be challenging. There is no clear abstraction of different protection-levels as envisioned by the classical UNIX-style OS designs. Hence, we must re-negotiate the work division among the hardware, a compiler, and an application with the compiler taking a leading role to deliver translation, multiplexing, and isolation properties — the traditional roles of an OS. With this compiler-centric approach, we run the risk of repeating the failure of the VLIW processors[3]. We argue that there are two fundamental shifts that work in our favor. First, domain/workload-specific architectures are common, and associated languages and compilers are used extensively as the norm. There are significant research and commercial interests in co-designing domain- or workload-specific hardware/software. Second, unlike VLIW processors, a DPU (specifically FPGA driven) is not aimed at delivering performance for *all or any* workloads, hence restricting the optimization design space. The role of compiler is not unusual here. It supports many OS-level roles already for services like Unikernels with specialization via compilation, end-to-end optimizations of various components [129], and a reduction of attack surfaces by dynamic recompilation [102]. Even with hardware/FPGAs it has been shown that the compilers can assist with traditional OS roles such as context switching [52, 106, 118], memory virtualization [154], single-level memory/storage [31, 74], isolation [107], extraction of parallelism [37], virtualization and multi-tenancy [89, 170].

3 RELATED WORK

Nider and Fedorova also question the utility of "the Last CPU" in the system and investigate the design of a *system management bus* (hardware) with autonomous devices to take over the OS/CPU responsibilities [123]. However, they still rely on shared, virtual memory (with IOMMU) as the core abstraction around which smart devices are integrated, and data identification, access, and processing happens. The CPU is removed from its managerial role, while the CPU-centric abstraction of a shared, coherent virtual memory is kept, hence inheriting and perpetuating the complexity with (virtual) memory management with accelerators [9, 22, 45, 65, 109, 154]. Furthermore, in comparison to their proposal which requires changes to the current architecture in hardware (a new management bus, and new types of devices), Hyperion is more pragmatic that can be

[3]VLIW compilers were left responsible for parallelism extraction in general workloads, which lead Donald Knuth to comment that *"…the "Itanium" approach that was supposed to be so terrific—until it turned out that the wished-for compilers were basically impossible to write"* [30].

realized today with the help of compilers. Omni-X system proposes to "exterminate the CPU" for inter-accelerator communication, and use P2P DMA for control/data coordination while still relying on CPU-centric abstractions in a multikernel/unikernel setting (coherent, shared virtual memory, file descriptors, sockets) [150]. M^3X system does not need shared coherent memory for inter-device communication and with the OS services, but requires explicit hardware support for network-on-chip communication endpoints between accelerators [16]. The MSR BEE3 system (used for emulation) is an early example of a complete non-CPU-centric system design [49].

Table 1 shows past research for *pair-wise device* interactions efforts where the role of the CPU is minimized or eliminated such as GPU-with-storage [23, 26, 27, 137, 151], GPU-with-network [48, 93, 125], accelerator-to/from-storage [16, 18, 109], SmartNICs [55, 134, 157], and networked storage accesses [94, 142]. FPGAs are explored with (1) networks [37, 58, 163, 174]; and (2) storage [141, 144, 146]. (e)BPF offloading to NIC/FPGA for processing is done with Endance DAG cards [53], Netronome [82], Combo6 [119], but mostly limited to monitoring and traffic shaping. FPGA-assisted KV stores have considered a close integration of network and KV processing (in-memory) [33, 39, 80, 108] and selective integration of NAND flash (e.g., BlueDB and Xilinx-KV [34, 168]). Project Alkali combines FPGA SoC with Cortex CPUs to design a computational storage device (CSD) designed to accelerate TensorFlow ML workloads from NVMe storage [87]. It uses eBPF (on the CPU with uBPF) with TensorFlow Lite with a possibility of ML operator specialization in FPGA with Chisel and TVM/VTA frameworks. However, even with that it is the CPU that drives the NVMe FPGA data orchestration logic for workflow execution. In comparison to these efforts, Hyperion targets a broader design space, where we explore new types of system design with a unified reconfigurable hardware (FPGA), network transport (100 Gbps Ethernet), and storage (NVMe flash). This unification offers end-to-end exploration of novel full-system CPU-free abstractions and with specializations to support emerging workloads like serverless and disaggregation.

4 DISCUSSION AND FEEDBACK

Hyperion is still in its early prototyping phase. From the systems community, we seek feedback on issues like:

(1) Is eliminating the CPU a worthy pursuit? In this paper, we made an ambitious case for removing the CPU. We believe that with the recent hardware/software advancements it is the right time to re-evaluate the role of the CPU and its design baggage. However, we are interested in hearing counter-arguments. We realize that engineering, development, and prototyping complexities might put limits to the realization of this idea. At what levels of performance, energy, and packaging efficiency gains from a CPU-free design will make the idea worth while? The elimination of the CPU-side mediation also necessitates a bigger supporting role from the FPGA toolchains, languages, and compilers, a role which was previously split between the host CPU and OS. Are FPGA toolchains ready?

(2) What are the killer workloads? Currently, our focus is on developing a familar set of reusable core storage abstractions such as trees (B+, LSM), hash tables, and graphs that can leverage the heterogeneous computing, memory, and storage resources available in Hyperion. Using these core data structures, we can build various key-value stores, network services (load balancers, packet loggers), and end-to-end workload-specialized DPUs, e.g., analytics (TPC, OLTP), LDBC Graphalytics with graph database, Bioinformatics (genome assembly), or climate modeling — all workloads which are data-intensive and have been shown to benefit from FPGA acceleration [152, 161]. Here we seek feedback on identifying specific high-priority workloads that can help us explore the abstraction design space.

(3) What is the right client-interface to build *distributed* Hyperion applications? Looking beyond a single DPU, what kind of application-level abstractions are required for building *distributed* CPU-free applications that can be executed over multiple DPUs? How should one build CPU-free distributed applications and composable service ecosystems of such standalone, passively disaggregated DPUs? Can such CPU-free ecosystems exist, or is a mixed CPU and CPU-free Hyperion setup a more realistic model?

(4) Complexity in multi-tenant clouds? In data centers, hardware and software fail. Tenants are untrusted. The costs of inefficiency and downtime are high. Hence, how to ensure that Hyperion can offer a secure, multi-tenant execution over multiple FPGAs [172]? How to reduce microarchitectural attacks with Hyperion? Can or should the micro-architectural resources of Hyperion be managed explicitly with tenants to ensure sufficient isolation with Hyperion DPUs [138]?

ACKNOWLEDGMENTS

This work is supported by the Dutch Research Council (NWO) grants OCENW.XS3.030 and OCENW.KLEIN.561, and Xilinx University Donation Program. The authors thank Marco Bonola, Giulia Frascaria, Corne Lukken, Kaveh Razavi, Herbert Bos, Tiziano De Matteis, Ana-Lucia Varbanescu, Alexandru Iosup, and the HotOS'23 reviewers for their constructive feedback. We would like to express our special gratitude to Jonas Pfefferle (IBM Research) who presented this work on the behalf of the authors at HotOS'23.

REFERENCES

[1] 2021. Intel’s Process Roadmap to 2025: with 4nm, 3nm, 20A and 18A?! https://www.anandtech.com/show/16823/intel-accelerated-offensive-process-roadmap-updates-to-10nm-7nm-4nm-3nm-20a-18a-packaging-foundry-emib-foveros. Accessed: 2023-Feb-02.

[2] 2023. Alveo U280 Data Center Accelerator Card. https://www.xilinx.com/products/boards-and-kits/alveo/u280.html. Accessed: 2023-Feb-20.

[3] 2023. CHIPS (Common Hardware for Interfaces, Processors and Systems) Alliance. https://chipsalliance.org/. Accessed: 2023-Jan-30.

[4] 2023. Fail2ban. https://www.fail2ban.org/wiki/index.php/Main_Page. Accessed: 2023-Jan-30.

[5] 2023. The OpenRoad Project, Democratizing Hardware Design. https://theopenroadproject.org/. Accessed: 2023-Jan-30.

[6] 2023. Userspace eBPF VM. Accessed: 2023-Feb-02, https://github.com/iovisor/ubpf.

[7] 2023. XDP: eXpress Data Path. https://www.iovisor.org/technology/xdp.

[8] Daniel Abadi, Anastasia Ailamaki, David Andersen, Peter Bailis, Magdalena Balazinska, Philip A. Bernstein, Peter Boncz, Surajit Chaudhuri, Alvin Cheung, Anhai Doan, Luna Dong, Michael J. Franklin, Juliana Freire, Alon Halevy, Joseph M. Hellerstein, Stratos Idreos, Donald Kossmann, Tim Kraska, Sailesh Krishnamurthy, Volker Markl, Sergey Melnik, Tova Milo, C. Mohan, Thomas Neumann, Beng Chin Ooi, Fatma Ozcan, Jignesh Patel, Andrew Pavlo, Raluca Popa, Raghu Ramakrishnan, Christopher Re, Michael Stonebraker, and Dan Suciu. 2022. The Seattle Report on Database Research. *Commun. ACM* 65, 8 (jul 2022), 72–79. https://doi.org/10.1145/3524284

[9] Reto Achermann. 2020. *On Memory Addressing*. PhD dissertation. ETH Zurich.

[10] Reto Achermann, Ashish Panwar, Abhishek Bhattacharjee, Timothy Roscoe, and Jayneel Gandhi. 2020. Mitosis: Transparently Self-Replicating Page-Tables for Large-Memory Machines. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS ’20)*. Association for Computing Machinery, New York, NY, USA, 283–300. https://doi.org/10.1145/3373376.3378468

[11] Michael Adler, Kermin E. Fleming, Angshuman Parashar, Michael Pellauer, and Joel Emer. 2011. Leap Scratchpads: Automatic Memory and Cache Management for Reconfigurable Logic. In *Proceedings of the 19th ACM/SIGDA International Symposium on Field Programmable Gate Arrays* (Monterey, CA, USA) *(FPGA ’11)*. Association for Computing Machinery, New York, NY, USA, 25–28. https://doi.org/10.1145/1950413.1950421

[12] Gustavo Alonso. 2018. FPGAs in Data Centers: FPGAs Are Slowly Leaving the Niche Space They Have Occupied for Decades. *Queue* 16, 2 (apr 2018), 52–57. https://doi.org/10.1145/3212477.3231573

[13] Sebastian Angel, Mihir Nanavati, and Siddhartha Sen. 2020. Disaggregation and the Application. In *12th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 20)*. USENIX Association. https://www.usenix.org/conference/hotcloud20/presentation/angel

[14] Apache. 2023. Apache Arrow: A cross-language development platform for in-memory data. https://arrow.apache.org/. Accessed: 2023-05-20.

[15] Apache. 2023. Apache Parquet: Columnar storage format. https://parquet.apache.org/. Accessed: 2023-05-20.

[16] Nils Asmussen, Michael Roitzsch, and Hermann Härtig. 2019. M3X: Autonomous Accelerators via Context-Enabled Fast-Path Communication. In *Proceedings of the 2019 USENIX Conference on Usenix Annual Technical Conference* (Renton, WA, USA) *(USENIX ATC ’19)*. USENIX Association, USA, 617–631.

[17] Vaggelis Atlidakis, Jeremy Andrus, Roxana Geambasu, Dimitris Mitropoulos, and Jason Nieh. 2016. POSIX Abstractions in Modern Operating Systems: The Old, the New, and the Missing. In *Proceedings of the Eleventh European Conference on Computer Systems* (London, United Kingdom) *(EuroSys ’16)*. Association for Computing Machinery, New York, NY, USA, Article 19, 17 pages. https://doi.org/10.1145/2901318.2901350

[18] Shinichi Awamoto, Erich Focht, and Michio Honda. 2020. Designing a Storage Software Stack for Accelerators. In *12th USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage 20)*. USENIX Association. https://www.usenix.org/conference/hotstorage20/presentation/awamoto

[19] Jonathan Bachrach, Huy Vo, Brian C. Richards, Yunsup Lee, Andrew Waterman, Rimas Avizienis, John Wawrzynek, and Krste Asanovic. 2012. Chisel: constructing hardware in a Scala embedded language. In *The 49th Annual Design Automation Conference 2012, DAC ’12, San Francisco, CA, USA, June 3-7, 2012*, Patrick Groeneveld, Donatella Sciuto, and Soha Hassoun (Eds.). ACM, 1216–1225. https://doi.org/10.1145/2228360.2228584

[20] Mahesh Balakrishnan, Dahlia Malkhi, Vijayan Prabhakaran, Ted Wobbler, Michael Wei, and John D. Davis. 2012. CORFU: A Shared Log Design for Flash Clusters. In *9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*. USENIX Association, San Jose, CA, 1–14. https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/balakrishnan

[21] Antonio Barbalace, Martin Decky, Javier Picorel, and Pramod Bhatotia. 2020. BlockNDP: Block-Storage Near Data Processing. In *Proceedings of the 21st International Middleware Conference Industrial Track* (Delft, Netherlands) *(Middleware ’20)*. Association for Computing Machinery, New York, NY, USA, 8–15. https://doi.org/10.1145/3429357.3430519

[22] Arkaprava Basu, Mark D. Hill, and Michael M. Swift. 2012. Reducing Memory Reference Energy with Opportunistic Virtual Caching. In *Proceedings of the 39th Annual International Symposium on Computer Architecture* (Portland, Oregon) *(ISCA ’12)*. IEEE Computer Society, USA, 297–308.

[23] Stephen Bates. 2015. Project Donard: NVM Express for Peer-2-Peer between SSDs and other PCIe Devices. https://www.snia.org/sites/default/files/SDC15_presentations/nvme_fab/StephenBates_Donard_NVM_Express_Peer-2_Peer.pdf. Accessed: 2023-Jan-30.

[24] Stephen Bates. 2018. Enabling the NVMe™ CMB and PMR Ecosystem. https://nvmexpress.org/wp-content/uploads/Session-2-Enabling-the-NVMe-CMB-and-PMR-Ecosystem-Eideticom-and-Mell....pdf. Accessed: 2023-Jan-30.

[25] Adam Belay, George Prekas, Ana Klimovic, Samuel Grossman, Christos Kozyrakis, and Edouard Bugnion. 2014. IX: A Protected Dataplane Operating System for High Throughput and Low Latency. In *11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14)*. USENIX Association, Broomfield, CO, 49–65. https://www.usenix.org/conference/osdi14/technical-sessions/presentation/belay

[26] Shai Bergman, Tanya Brokhman, Tzachi Cohen, and Mark Silberstein. 2019. SPIN: Seamless Operating System Integration of Peer-to-Peer DMA Between SSDs and GPUs. *ACM Trans. Comput. Syst.* 36, 2, Article 5 (apr 2019), 26 pages. https://doi.org/10.1145/3309987

[27] Pramod Bhatotia, Rodrigo Rodrigues, and Akshat Verma. 2012. Shredder: GPU-Accelerated Incremental Storage and Computation. In *Proceedings of the 10th USENIX Conference on File and Storage Technologies* (San Jose, CA) *(FAST’12)*. USENIX Association, USA, 14.

[28] Janki Bhimani, Jingpei Yang, Ningfang Mi, Changho Choi, Manoj Saha, and Adnan Maruf. 2021. Fine-Grained Control of Concurrency

within KV-SSDs. In *Proceedings of the 14th ACM International Conference on Systems and Storage* (Haifa, Israel) *(SYSTOR '21)*. Association for Computing Machinery, New York, NY, USA, Article 4, 12 pages. https://doi.org/10.1145/3456727.3463777

[29] Ashish Bijlani and Umakishore Ramachandran. 2019. Extension framework for file systems in user space. In *2019 {USENIX} Annual Technical Conference ({USENIX}{ATC} 19)*. 121–134.

[30] Andrew Binstock and Donald Knuth. 2008. Interview with Donald Knuth. https://www.informit.com/articles/article.aspx?p=1193856. Accessed: 2023-Jan-30.

[31] Daniel Bittman, Peter Alvaro, Pankaj Mehra, Darrell D. E. Long, and Ethan L. Miller. 2020. Twizzler: a Data-Centric OS for Non-Volatile Memory. In *2020 USENIX Annual Technical Conference (USENIX ATC 20)*. USENIX Association, 65–80. https://www.usenix.org/conference/atc20/presentation/bittman

[32] Matias Bjørling, Abutalib Aghayev, Hans Holmberg, Aravind Ramesh, Damien Le Moal, Gregory R. Ganger, and George Amvrosiadis. 2021. ZNS: Avoiding the Block Interface Tax for Flash-based SSDs. In *2021 USENIX Annual Technical Conference, USENIX ATC 2021, July 14-16, 2021*, Irina Calciu and Geoff Kuenning (Eds.). USENIX Association, 689–703. https://www.usenix.org/conference/atc21/presentation/bjorling

[33] Michaela Blott, Kimon Karras, Ling Liu, Kees Vissers, Jeremia Bär, and Zsolt István. 2013. Achieving 10Gbps Line-rate Key-value Stores with FPGAs. In *5th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 13)*. USENIX Association, San Jose, CA. https://www.usenix.org/conference/hotcloud13/workshop-program/presentations/blott

[34] Michaela Blott, Ling Liu, Kimon Karras, and Kees Vissers. 2015. Scaling Out to a Single-Node 80Gbps Memcached Server with 40Terabytes of Memory. In *7th USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage 15)*. USENIX Association, Santa Clara, CA. https://www.usenix.org/conference/hotstorage15/workshop-program/presentation/blott

[35] Marco Bonola, Giacomo Belocchi, Angelo Tulumello, Marco Spaziani Brunella, Giuseppe Siracusano, Giuseppe Bianchi, and Roberto Bifulco. 2022. Faster Software Packet Processing on FPGA NICs with eBPF Program Warping. In *2022 USENIX Annual Technical Conference (USENIX ATC 22)*. USENIX Association, Carlsbad, CA, 987–1004. https://www.usenix.org/conference/atc22/presentation/bonola

[36] Shekhar Borkar and Andrew A. Chien. 2011. The Future of Microprocessors. *Commun. ACM* 54, 5 (may 2011), 67–77. https://doi.org/10.1145/1941487.1941507

[37] Marco Spaziani Brunella, Giacomo Belocchi, Marco Bonola, Salvatore Pontarelli, Giuseppe Siracusano, Giuseppe Bianchi, Aniello Cammarano, Alessandro Palumbo, Luca Petrucci, and Roberto Bifulco. 2020. hXDP: Efficient Software Packet Processing on FPGA NICs. In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*. USENIX Association, 973–990. https://www.usenix.org/conference/osdi20/presentation/brunella

[38] Irina Calciu, M. Talha Imran, Ivan Puddu, Sanidhya Kashyap, Hasan Al Maruf, Onur Mutlu, and Aasheesh Kolli. 2021. *Rethinking Software Runtimes for Disaggregated Memory*. Association for Computing Machinery, New York, NY, USA, 79–92. https://doi.org/10.1145/3445814.3446713

[39] Sai Rahul Chalamalasetti, Kevin Lim, Mitch Wright, Alvin AuYoung, Parthasarathy Ranganathan, and Martin Margala. 2013. An FPGA Memcached Appliance. In *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays* (Monterey, California, USA) *(FPGA '13)*. Association for Computing Machinery, New York, NY, USA, 245–254. https://doi.org/10.1145/2435264.2435306

[40] Deming Chen. 2019. FPGAs in Supercomputers: Opportunity or Folly?. In *Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays* (Seaside, CA, USA) *(FPGA '19)*. Association for Computing Machinery, New York, NY, USA, 201. https://doi.org/10.1145/3289602.3293929

[41] Young-Kyu Choi, Jason Cong, Zhenman Fang, Yuchen Hao, Glenn Reinman, and Peng Wei. 2019. In-Depth Analysis on Microarchitectures of Modern Heterogeneous CPU-FPGA Platforms. *ACM Trans. Reconfigurable Technol. Syst.* 12, 1, Article 4 (feb 2019), 20 pages. https://doi.org/10.1145/3294054

[42] Cilium. 2023. https://ebpf.io/. Accessed: 2023-Jan-30.

[43] Joel Coburn, Adrian M. Caulfield, Ameen Akel, Laura M. Grupp, Rajesh K. Gupta, Ranjit Jhala, and Steven Swanson. 2011. NV-Heaps: Making Persistent Objects Fast and Safe with next-Generation, Non-Volatile Memories. In *Proceedings of the Sixteenth International Conference on Architectural Support for Programming Languages and Operating Systems* (Newport Beach, California, USA) *(ASPLOS XVI)*. Association for Computing Machinery, New York, NY, USA, 105–118. https://doi.org/10.1145/1950365.1950380

[44] David Cock, Abishek Ramdas, Daniel Schwyn, Michael Giardino, Adam Turowski, Zhenhao He, Nora Hossle, Dario Korolija, Melissa Licciardello, Kristina Martsenko, Reto Achermann, Gustavo Alonso, and Timothy Roscoe. 2022. Enzian: An Open, General, CPU/FPGA Platform for Systems Software Research. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS 2022)*. Association for Computing Machinery, New York, NY, USA, 434–451. https://doi.org/10.1145/3503222.3507742

[45] Guilherme Cox and Abhishek Bhattacharjee. 2017. Efficient Address Translation for Architectures with Multiple Page Sizes. In *Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems* (Xi'an, China) *(ASPLOS '17)*. Association for Computing Machinery, New York, NY, USA, 435–448. https://doi.org/10.1145/3037697.3037704

[46] Robert C. Daley and Jack B. Dennis. 1968. Virtual Memory, Processes, and Sharing in MULTICS. *Commun. ACM* 11, 5 (may 1968), 306–312. https://doi.org/10.1145/363095.363139

[47] William J. Dally, Yatish Turakhia, and Song Han. 2020. Domain-Specific Hardware Accelerators. *Commun. ACM* 63, 7 (jun 2020), 48–57. https://doi.org/10.1145/3361682

[48] Feras Daoud, Amir Watad, and Mark Silberstein. 2016. GPUrdma: GPU-Side Library for High Performance Networking from GPU Kernels. In *Proceedings of the 6th International Workshop on Runtime and Operating Systems for Supercomputers* (Kyoto, Japan) *(ROSS '16)*. Association for Computing Machinery, New York, NY, USA, Article 6, 8 pages. https://doi.org/10.1145/2931088.2931091

[49] John Davis, Chuck Thacker, and Chen Chang. 2009. BEE3: Revitalizing Computer Architecture Research (MSR-TR-2009-45). https://www.microsoft.com/en-us/research/wp-content/uploads/2009/04/BEE3_TechReport.pdf.

[50] Peter J. Denning. 1970. Virtual Memory. *ACM Comput. Surv.* 2, 3 (sep 1970), 153–189. https://doi.org/10.1145/356571.356573

[51] Design Gateway. 2023. PCIe x16 Lanes Crossover adapter board for NVMe-IP evaluation. Accessed: 2023-Feb-02, https://nl.mouser.com/ProductDetail/Design-Gateway/AB18-PCIeX16?qs=T3oQrply3y9MKpPjG7SUNQ%3D%3D.

[52] Stephen Dolan, Servesh Muralidharan, and David Gregg. 2013. Compiler Support for Lightweight Context Switching. *ACM Trans. Archit. Code Optim.* 9, 4, Article 36 (jan 2013), 25 pages. https://doi.org/10.1145/2400682.2400695

[53] Endace. 2020. Endace DAG Packet Capture Cards: Part 1. https://tryingtokeepitsecure.bz/index.php/8-network-

engineering/14-endace-dag-packet-capture-cards. Accessed: 2023-Jan-30.

[54] Haggai Eran, Maxim Fudim, Gabi Malka, Gal Shalom, Noam Cohen, Amit Hermony, Dotan Levi, Liran Liss, and Mark Silberstein. 2022. FlexDriver: A Network Driver for Your Accelerator. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS '22)*. Association for Computing Machinery, New York, NY, USA, 1115–1129. https://doi.org/10.1145/3503222.3507776

[55] Haggai Eran, Lior Zeno, Maroun Tork, Gabi Malka, and Mark Silberstein. 2019. NICA: An Infrastructure for Inline Acceleration of Network Applications. In *2019 USENIX Annual Technical Conference (USENIX ATC 19)*. USENIX Association, Renton, WA, 345–362. https://www.usenix.org/conference/atc19/presentation/eran

[56] Hadi Esmaeilzadeh, Emily Blem, Renee St. Amant, Karthikeyan Sankaralingam, and Doug Burger. 2011. Dark Silicon and the End of Multicore Scaling. In *Proceedings of the 38th Annual International Symposium on Computer Architecture* (San Jose, California, USA) *(ISCA '11)*. Association for Computing Machinery, New York, NY, USA, 365–376. https://doi.org/10.1145/2000064.2000108

[57] Michael Ferdman, Almutaz Adileh, Onur Kocberber, Stavros Volos, Mohammad Alisafaee, Djordje Jevdjic, Cansu Kaynak, Adrian Daniel Popescu, Anastasia Ailamaki, and Babak Falsafi. 2012. Clearing the Clouds: A Study of Emerging Scale-out Workloads on Modern Hardware. In *Proceedings of the Seventeenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS XVII)*. ACM, London, England, UK, 37–48. https://doi.org/10.1145/2150976.2150982

[58] Daniel Firestone, Andrew Putnam, Sambhrama Mundkur, Derek Chiou, Alireza Dabagh, Mike Andrewartha, Hari Angepat, Vivek Bhanu, Adrian Caulfield, Eric Chung, Harish Kumar Chandrappa, Somesh Chaturmohta, Matt Humphrey, Jack Lavier, Norman Lam, Fengfen Liu, Kalin Ovtcharov, Jitu Padhye, Gautham Popuri, Shachar Raindel, Tejas Sapre, Mark Shaw, Gabriel Silva, Madhan Sivakumar, Nisheeth Srivastava, Anshuman Verma, Qasim Zuhair, Deepak Bansal, Doug Burger, Kushagra Vaid, David A. Maltz, and Albert Greenberg. 2018. Azure Accelerated Networking: SmartNICs in the Public Cloud. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. USENIX Association, Renton, WA, 51–66. https://www.usenix.org/conference/nsdi18/presentation/firestone

[59] Fungible. 2023. Fungible F1 Data Processing Unit - Acquired by Microsoft. https://blogs.microsoft.com/blog/2023/01/09/microsoft-announces-acquisition-of-fungible-to-accelerate-datacenter-innovation/. Accessed: 2023-Jan-30.

[60] Peter X. Gao, Akshay Narayan, Sagar Karandikar, Joao Carreira, Sangjin Han, Rachit Agarwal, Sylvia Ratnasamy, and Scott Shenker. 2016. Network Requirements for Resource Disaggregation. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. USENIX Association, Savannah, GA, 249–264. https://www.usenix.org/conference/osdi16/technical-sessions/presentation/gao

[61] Elazar Gershuni, Nadav Amit, Arie Gurfinkel, Nina Narodytska, Jorge A. Navas, Noam Rinetzky, Leonid Ryzhyk, and Mooly Sagiv. 2019. Simple and Precise Static Analysis of Untrusted Linux Kernel Extensions. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation* (Phoenix, AZ, USA) *(PLDI 2019)*. Association for Computing Machinery, New York, NY, USA, 1069–1084. https://doi.org/10.1145/3314221.3314590

[62] Yoann Ghigoff, Julien Sopena, Kahina Lazri, Antoine Blin, and Gilles Muller. 2021. BMC: Accelerating Memcached using Safe In-kernel Caching and Pre-stack Processing. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. USENIX Association, 487–501. https://www.usenix.org/conference/nsdi21/presentation/ghigoff

[63] Peter-Jan Gootzen, Jonas Pfefferle, Radu Stoica, and Animesh Trivedi. 2023. DPFS: DPU-Powered File System Virtualization. In *Proceedings of the 16th ACM International Conference on Systems and Storage* (Haifa, Israel) *(SYSTOR '23)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3579370.3594769

[64] Donghyun Gouk, Sangwon Lee, Miryeong Kwon, and Myoungsoo Jung. 2022. Direct Access, High-Performance Memory Disaggregation with DirectCXL. In *2022 USENIX Annual Technical Conference (USENIX ATC 22)*. USENIX Association, Carlsbad, CA, 287–294. https://www.usenix.org/conference/atc22/presentation/gouk

[65] Ben Gras, Kaveh Razavi, Herbert Bos, and Cristiano Giuffrida. 2018. Translation Leak-aside Buffer: Defeating Cache Side-channel Protections with TLB Attacks. In *27th USENIX Security Symposium (USENIX Security 18)*. Baltimore, MD, 955–972. https://www.usenix.org/conference/usenixsecurity18/presentation/gras

[66] Brendan D. Gregg. 2023. Linux Enhanced BPF (eBPF) Tracing Tools. Accessed: 2023-Feb-02, http://www.brendangregg.com/ebpf.html.

[67] Boncheol Gu, Andre S. Yoon, Duck-Ho Bae, Insoon Jo, Jinyoung Lee, Jonghyun Yoon, Jeong-Uk Kang, Moonsang Kwon, Chanho Yoon, Sangyeun Cho, Jaeheon Jeong, and Duckhyun Chang. 2016. Biscuit: A Framework for near-Data Processing of Big Data Workloads. In *Proceedings of the 43rd International Symposium on Computer Architecture* (Seoul, Republic of Korea) *(ISCA '16)*. IEEE Press, 153–165. https://doi.org/10.1109/ISCA.2016.23

[68] Zhiyuan Guo, Yizhou Shan, Xuhao Luo, Yutong Huang, and Yiying Zhang. 2022. Clio: A Hardware-Software Co-Designed Disaggregated Memory System. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS 2022)*. Association for Computing Machinery, New York, NY, USA, 417–433. https://doi.org/10.1145/3503222.3507762

[69] Sangjin Han, Norbert Egi, Aurojit Panda, Sylvia Ratnasamy, Guangyu Shi, and Scott Shenker. 2013. Network Support for Resource Disaggregation in Next-Generation Datacenters. In *Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks* (College Park, Maryland) *(HotNets-XII)*. Association for Computing Machinery, New York, NY, USA, Article 10, 7 pages. https://doi.org/10.1145/2535771.2535778

[70] Nikos Hardavellas, Michael Ferdman, Babak Falsafi, and Anastasia Ailamaki. 2011. Toward Dark Silicon in Servers. *IEEE Micro* 31, 4 (2011), 6–15. https://doi.org/10.1109/MM.2011.77

[71] John L. Hennessy and David A. Patterson. 2019. A New Golden Age for Computer Architecture. *Commun. ACM* 62, 2 (Jan. 2019), 48–60. https://doi.org/10.1145/3282307

[72] Toke Høiland-Jørgensen, Jesper Dangaard Brouer, Daniel Borkmann, John Fastabend, Tom Herbert, David Ahern, and David Miller. 2018. The EXpress Data Path: Fast Programmable Packet Processing in the Operating System Kernel. In *Proceedings of the 14th International Conference on Emerging Networking EXperiments and Technologies* (Heraklion, Greece) *(CoNEXT '18)*. Association for Computing Machinery, New York, NY, USA, 54–66. https://doi.org/10.1145/3281411.3281443

[73] Michio Honda. 2021. Packets as Persistent In-Memory Data Structures. In *Proceedings of the Twentieth ACM Workshop on Hot Topics in Networks* (Virtual Event, United Kingdom) *(HotNets '21)*. Association for Computing Machinery, New York, NY, USA, 31–37. https://doi.org/10.1145/3484266.3487386

[74] Morteza Hoseinzadeh and Steven Swanson. 2021. Corundum: Statically-Enforced Persistent Memory Safety. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Virtual, USA) *(ASPLOS '21)*. Association for Computing Machinery, New York, NY, USA,

429–442. https://doi.org/10.1145/3445814.3446710

[75] Jaehyun Hwang, Qizhe Cai, Ao Tang, and Rachit Agarwal. 2020. TCP == RDMA: CPU-efficient Remote Storage Access with i10. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*. USENIX Association, Santa Clara, CA, 127–140. https://www.usenix.org/conference/nsdi20/presentation/hwang

[76] Jaehyun Hwang, Midhul Vuppalapati, Simon Peter, and Rachit Agarwal. 2021. Rearchitecting Linux Storage Stack for μs Latency and High Throughput. In *15th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2021, July 14-16, 2021*, Angela Demke Brown and Jay R. Lorch (Eds.). USENIX Association, 113–128. https://www.usenix.org/conference/osdi21/presentation/hwang

[77] IBM Corporation. 1998. AS/400 Machine Internal Functional Reference, Number SC41-5810-01.

[78] Intel. 2022. Intel Technology Roadmaps and Milestones. https://www.intel.com/content/www/us/en/newsroom/news/intel-technology-roadmaps-milestones.html. Accessed: 2023-Feb-02.

[79] Intel. 2023. oneAPI: A New Era of Heterogeneous Computing. https://www.intel.com/content/www/us/en/developer/tools/oneapi/overview.html. Accessed: 2023-Feb-02.

[80] Zsolt István, David Sidler, and Gustavo Alonso. 2017. Caribou: Intelligent Distributed Storage. *Proc. VLDB Endow.* 10, 11 (aug 2017), 1202–1213. https://doi.org/10.14778/3137628.3137632

[81] Zsolt István, David Sidler, Gustavo Alonso, and Marko Vukolic. 2016. Consensus in a Box: Inexpensive Coordination in Hardware. In *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*. USENIX Association, Santa Clara, CA, 425–438. https://www.usenix.org/conference/nsdi16/technical-sessions/presentation/istvan

[82] Jakub Kicinski, Nicolaas Viljoen. 2016. Netronome Systems, eBPF Hardware Offload to SmartNICs: cls bpf and XDP. https://www.netronome.com/media/documents/eBPF_HW_OFFLOAD_HNiMne8_2_.pdf. Accessed: 2023-Jan-30.

[83] Norman P. Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, Rick Boyle, Pierre-luc Cantin, Clifford Chao, Chris Clark, Jeremy Coriell, Mike Daley, Matt Dau, Jeffrey Dean, Ben Gelb, Tara Vazir Ghaemmaghami, Rajendra Gottipati, William Gulland, Robert Hagmann, C. Richard Ho, Doug Hogberg, John Hu, Robert Hundt, Dan Hurt, Julian Ibarz, Aaron Jaffey, Alek Jaworski, Alexander Kaplan, Harshit Khaitan, Daniel Killebrew, Andy Koch, Naveen Kumar, Steve Lacy, James Laudon, James Law, Diemthu Le, Chris Leary, Zhuyuan Liu, Kyle Lucke, Alan Lundin, Gordon MacKean, Adriana Maggiore, Maire Mahony, Kieran Miller, Rahul Nagarajan, Ravi Narayanaswami, Ray Ni, Kathy Nix, Thomas Norrie, Mark Omernick, Narayana Penukonda, Andy Phelps, Jonathan Ross, Matt Ross, Amir Salek, Emad Samadiani, Chris Severn, Gregory Sizikov, Matthew Snelham, Jed Souter, Dan Steinberg, Andy Swing, Mercedes Tan, Gregory Thorson, Bo Tian, Horia Toma, Erick Tuttle, Vijay Vasudevan, Richard Walter, Walter Wang, Eric Wilcox, and Doe Hyun Yoon. 2017. In-Datacenter Performance Analysis of a Tensor Processing Unit. In *Proceedings of the 44th Annual International Symposium on Computer Architecture* (Toronto, ON, Canada) *(ISCA '17)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3079856.3080246

[84] Myoungsoo Jung. 2022. Hello Bytes, Bye Blocks: PCIe Storage Meets Compute Express Link for Memory Expansion (CXL-SSD). In *Proceedings of the 14th ACM Workshop on Hot Topics in Storage and File Systems* (Virtual Event) *(HotStorage '22)*. Association for Computing Machinery, New York, NY, USA, 45–51. https://doi.org/10.1145/3538643.3539745

[85] Rohan Kadekodi, Se Kwon Lee, Sanidhya Kashyap, Taesoo Kim, Aasheesh Kolli, and Vijay Chidambaram. 2019. SplitFS: Reducing Software Overhead in File Systems for Persistent Memory. In *Proceedings of the 27th ACM Symposium on Operating Systems Principles* (Huntsville, Ontario, Canada) *(SOSP '19)*. Association for Computing Machinery, New York, NY, USA, 494–508. https://doi.org/10.1145/3341301.3359631

[86] Anuj Kalia, Dong Zhou, Michael Kaminsky, and David G. Andersen. 2015. Raising the Bar for Using GPUs in Software Packet Processing. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. USENIX Association, Oakland, CA, 409–423. https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/kalia

[87] Karol Gugala. 2022. Open source FPGA NVMe accelerator platform for BPF driven ML processing with Linux/Zephyr. https://lpc.events/event/16/contributions/1245/. Accessed: 2023-Jan-30.

[88] Michael Kerrisk. 2015. Using seccomp to limit the kernel attack surface. Linux Plumbers Conference. Accessed: 2023-Feb-02, https://man7.org/conf/lpc2015/limiting_kernel_attack_surface_with_seccomp-LPC_2015-Kerrisk.pdf.

[89] Ahmed Khawaja, Joshua Landgraf, Rohith Prakash, Michael Wei, Eric Schkufza, and Christopher J. Rossbach. 2018. Sharing, Protection, and Compatibility for Reconfigurable Fabric with AmorphOS. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*. USENIX Association, Carlsbad, CA, 107–127. http://www.usenix.org/conference/osdi18/presentation/khawaja

[90] Jakub Kicinski. 2018. Using eBPF as a heterogeneous processing ABI. Linux Plumbers Conference. Accessed: 2023-Feb-02, https://lpc.events/event/2/contributions/120/.

[91] T. Kilburn, D. B. G. Edwards, M. J. Lanigan, and F. H. Sumner. 1962. One-Level Storage System. *IRE Transactions on Electronic Computers* EC-11, 2 (1962), 223–235. https://doi.org/10.1109/TEC.1962.5219356

[92] Daehyeok Kim, Amirsaman Memaripour, Anirudh Badam, Yibo Zhu, Hongqiang Harry Liu, Jitu Padhye, Shachar Raindel, Steven Swanson, Vyas Sekar, and Srinivasan Seshan. 2018. Hyperloop: Group-based NIC-offloading to Accelerate Replicated Transactions in Multi-tenant Storage Systems. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '18)*. ACM, Budapest, Hungary, 297–312. https://doi.org/10.1145/3230543.3230572

[93] Sangman Kim, Seonggu Huh, Xinya Zhang, Yige Hu, Amir Wated, Emmett Witchel, and Mark Silberstein. 2014. GPUnet: Networking Abstractions for GPU Programs. In *11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14)*. USENIX Association, Broomfield, CO, 201–216. https://www.usenix.org/conference/osdi14/technical-sessions/presentation/kim

[94] Ana Klimovic, Christos Kozyrakis, Eno Thereska, Binu John, and Sanjeev Kumar. 2016. Flash Storage Disaggregation. In *Proceedings of the Eleventh European Conference on Computer Systems* (London, United Kingdom) *(EuroSys '16)*. Association for Computing Machinery, New York, NY, USA, Article 29, 15 pages. https://doi.org/10.1145/2901318.2901337

[95] Ana Klimovic, Heiner Litz, and Christos Kozyrakis. 2017. ReFlex: Remote Flash = Local Flash. In *Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '17)*. ACM, Xi'an, China, 345–359. https://doi.org/10.1145/3037697.3037732

[96] Ryohei Kobayashi, Yuma Oobata, Norihisa Fujita, Yoshiki Yamaguchi, and Taisuke Boku. 2018. OpenCL-Ready High Speed FPGA Network for Reconfigurable High Performance Computing. In *Proceedings of the International Conference on High Performance Computing in Asia-Pacific Region* (Chiyoda, Tokyo, Japan) *(HPC Asia 2018)*. Association

for Computing Machinery, New York, NY, USA, 192–201. https://doi.org/10.1145/3149457.3149479

[97] Paul Kocher, Jann Horn, Anders Fogh, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom. 2020. Spectre Attacks: Exploiting Speculative Execution. *Commun. ACM* 63, 7 (jun 2020), 93–101. https://doi.org/10.1145/3399742

[98] David Koeplinger, Matthew Feldman, Raghu Prabhakar, Yaqi Zhang, Stefan Hadjis, Ruben Fiszel, Tian Zhao, Luigi Nardi, Ardavan Pedram, Christos Kozyrakis, and Kunle Olukotun. 2018. Spatial: A Language and Compiler for Application Accelerators. In *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation* (Philadelphia, PA, USA) *(PLDI 2018)*. Association for Computing Machinery, New York, NY, USA, 296–311. https://doi.org/10.1145/3192366.3192379

[99] Gunjae Koo, Kiran Kumar Matam, Te I, H. V. Krishna Giri Narra, Jing Li, Hung-Wei Tseng, Steven Swanson, and Murali Annavaram. 2017. Summarizer: Trading Communication with Computing near Storage. In *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture* (Cambridge, Massachusetts) *(MICRO-50 '17)*. Association for Computing Machinery, New York, NY, USA, 219–231. https://doi.org/10.1145/3123939.3124553

[100] Dario Korolija, Timothy Roscoe, and Gustavo Alonso. 2020. Do OS abstractions make sense on FPGAs?. In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*. USENIX Association, 991–1010. https://www.usenix.org/conference/osdi20/presentation/roscoe

[101] Kornilios Kourtis, Animesh Trivedi, and Nikolas Ioannou. 2020. Safe and Efficient Remote Application Code Execution on Disaggregated NVM Storage with eBPF. *CoRR* abs/2002.11528 (2020). arXiv:2002.11528 https://arxiv.org/abs/2002.11528

[102] Taddeus Kroes, Anil Altinay, Joseph Nash, Yeoul Na, Stijn Volckaert, Herbert Bos, Michael Franz, and Cristiano Giuffrida. 2018. BinRec: Attack Surface Reduction Through Dynamic Binary Recovery. In *Proceedings of the 2018 Workshop on Forming an Ecosystem Around Software Transformation* (Toronto, Canada) *(FEAST '18)*. Association for Computing Machinery, New York, NY, USA, 8–13. https://doi.org/10.1145/3273045.3273050

[103] Lucas Kuhring, Eva Garcia, and Zsolt István. 2019. Specialize in Moderation—Building Application-aware Storage Services using FPGAs in the Datacenter. In *11th USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage 19)*. USENIX Association, Renton, WA. https://www.usenix.org/conference/hotstorage19/presentation/kuhring

[104] Chinmay Kulkarni, Sara Moore, Mazhar Naqvi, Tian Zhang, Robert Ricci, and Ryan Stutsman. 2018. Splinter: Bare-Metal Extensions for Multi-Tenant Low-Latency Storage. In *Proceedings of the 13th USENIX Conference on Operating Systems Design and Implementation* (Carlsbad, CA, USA) *(OSDI'18)*. USENIX Association, USA, 627–643.

[105] Dongup Kwon, Dongryeong Kim, Junehyuk Boo, Wonsik Lee, and Jangwoo Kim. 2021. A Fast and Flexible Hardware-based Virtualization Mechanism for Computational Storage Devices. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*. USENIX Association, 729–743. https://www.usenix.org/conference/atc21/presentation/kwon

[106] Joshua Landgraf, Tiffany Yang, Will Lin, Christopher J. Rossbach, and Eric Schkufza. 2021. *Compiler-Driven FPGA Virtualization with SYNERGY*. Association for Computing Machinery, New York, NY, USA, 818–831. https://doi.org/10.1145/3445814.3446755

[107] Hugo Lefeuvre, Vlad-Andrei Bădoiu, Alexander Jung, Stefan Lucian Teodorescu, Sebastian Rauch, Felipe Huici, Costin Raiciu, and Pierre Olivier. 2022. FlexOS: Towards Flexible OS Isolation. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS '22)*. Association for Computing Machinery, New York, NY, USA, 467–482. https://doi.org/10.1145/3503222.3507759

[108] Bojie Li, Zhenyuan Ruan, Wencong Xiao, Yuanwei Lu, Yongqiang Xiong, Andrew Putnam, Enhong Chen, and Lintao Zhang. 2017. KV-Direct: High-Performance In-Memory Key-Value Store with Programmable NIC. In *Proceedings of the 26th Symposium on Operating Systems Principles (SOSP '17)*. ACM, Shanghai, China, 137–152. https://doi.org/10.1145/3132747.3132756

[109] Huaicheng Li, Mingzhe Hao, Stanko Novakovic, Vaibhav Gogte, Sriram Govindan, Dan R. K. Ports, Irene Zhang, Ricardo Bianchini, Haryadi S. Gunawi, and Anirudh Badam. 2020. *LeapIO: Efficient and Portable Virtual NVMe Storage on ARM SoCs*. Association for Computing Machinery, New York, NY, USA, 591–605. https://doi.org/10.1145/3373376.3378531

[110] Ruibin Li, Xiang Ren, Xu Zhao, Siwei He, Michael Stumm, and Ding Yuan. 2022. ctFS: Replacing File Indexing with Hardware Memory Translation through Contiguous File Allocation for Persistent Memory. In *20th USENIX Conference on File and Storage Technologies (FAST 22)*. USENIX Association, Santa Clara, CA, 35–50. https://www.usenix.org/conference/fast22/presentation/li

[111] Hyeontaek Lim, Dongsu Han, David G. Andersen, and Michael Kaminsky. 2014. MICA: A Holistic Approach to Fast In-memory Key-value Storage. In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation (NSDI'14)*. Seattle, WA, 429–444. http://dl.acm.org/citation.cfm?id=2616448.2616488

[112] Ming Liu, Tianyi Cui, Henry Schuh, Arvind Krishnamurthy, Simon Peter, and Karan Gupta. 2019. Offloading Distributed Applications onto SmartNICs Using IPipe. In *Proceedings of the ACM Special Interest Group on Data Communication* (Beijing, China) *(SIGCOMM '19)*. Association for Computing Machinery, New York, NY, USA, 318–333. https://doi.org/10.1145/3341302.3342079

[113] John W. Lockwood, Nick McKeown, Greg Watson, Glen Gibb, Paul Hartke, Jad Naous, Ramanan Raghuraman, and Jianying Luo. 2007. NetFPGA–An Open Platform for Gigabit-Rate Network Switching and Routing. In *Proceedings of the 2007 IEEE International Conference on Microelectronic Systems Education (MSE '07)*. IEEE Computer Society, USA, 160–161. https://doi.org/10.1109/MSE.2007.69

[114] Corne Lukken, Giulia Frascaria, and Animesh Trivedi. 2021. ZCSD: a Computational Storage Device over Zoned Namespaces (ZNS) SSDs. https://doi.org/10.48550/ARXIV.2112.00142

[115] Jiacheng Ma, Gefei Zuo, Kevin Loughlin, Xiaohe Cheng, Yanqiang Liu, Abel Mulugeta Eneyew, Zhengwei Qi, and Baris Kasikci. 2020. *A Hypervisor for Shared-Memory FPGA Platforms*. Association for Computing Machinery, New York, NY, USA, 827–844. https://doi.org/10.1145/3373376.3378482

[116] Jiacheng Ma, Gefei Zuo, Kevin Loughlin, Haoyang Zhang, Andrew Quinn, and Baris Kasikci. 2022. Debugging in the Brave New World of Reconfigurable Hardware. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS 2022)*. Association for Computing Machinery, New York, NY, USA, 946–962. https://doi.org/10.1145/3503222.3507701

[117] John MacCormick, Nick Murphy, Marc Najork, Chandramohan A. Thekkath, and Lidong Zhou. 2004. Boxwood: Abstractions as the Foundation for Storage Infrastructure. In *6th Symposium on Operating Systems Design & Implementation (OSDI 04)*. USENIX Association, San Francisco, CA. https://www.usenix.org/conference/osdi-04/boxwood-abstractions-foundation-storage-infrastructure

[118] Kiwan Maeng and Brandon Lucia. 2018. Adaptive Dynamic Checkpointing for Safe Efficient Intermittent Computing. In *13th*

USENIX Symposium on Operating Systems Design and Implementation (OSDI 18). USENIX Association, Carlsbad, CA, 129–144. https://www.usenix.org/conference/osdi18/presentation/maeng

[119] Evangelos Markatos, Ji Y, Michalis Polychronakis, Vladimir Smotlacha, and Sven Ubik. 2004. SCAMPI - A Scaleable Monitoring Platform for the Internet, https://publications.ics.forth.gr/_publications/SCAMPI_coppens_ips2004.pdf. (05 2004).

[120] Steven McCanne and Van Jacobson. 1993. The BSD Packet Filter: A New Architecture for User-Level Packet Capture. In *Proceedings of the USENIX Winter 1993 Conference Proceedings on USENIX Winter 1993 Conference Proceedings* (San Diego, California) *(USENIX'93)*. USENIX Association, USA, 2.

[121] Jaehong Min, Ming Liu, Tapan Chugh, Chenxingyu Zhao, Andrew Wei, In Hwan Doh, and Arvind Krishnamurthy. 2021. Gimbal: Enabling Multi-Tenant Storage Disaggregation on SmartNIC JBOFs. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference* (Virtual Event, USA) *(SIGCOMM '21)*. Association for Computing Machinery, New York, NY, USA, 106–122. https://doi.org/10.1145/3452296.3472940

[122] Ryo Nakamura, Yohei Kuga, and Kunio Akashi. 2020. How Beneficial is Peer-to-Peer DMA?. In *Proceedings of the 11th ACM SIGOPS Asia-Pacific Workshop on Systems* (Tsukuba, Japan) *(APSys '20)*. Association for Computing Machinery, New York, NY, USA, 25–32. https://doi.org/10.1145/3409963.3410491

[123] Joel Nider and Alexandra (Sasha) Fedorova. 2021. The Last CPU. In *Proceedings of the Workshop on Hot Topics in Operating Systems* (Ann Arbor, Michigan) *(HotOS '21)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3458336.3465291

[124] NVIDIA. 2019. GPUDirect Storage: A Direct Path Between Storage and GPU Memory. https://developer.nvidia.com/blog/gpudirect-storage/. Accessed: 2023-Feb-02.

[125] NVIDIA. 2023. Developing a Linux Kernel Module using GPUDirect RDMA. https://docs.nvidia.com/cuda/gpudirect-rdma/index.html. Accessed: 2023-Feb-02.

[126] Nvidia. 2023. Mellanox BlueField SmartNIC for Ethernet. https://www.mellanox.com/files/doc-2020/pb-bluefield-smart-nic.pdf. Accessed: 2023-Jan-30.

[127] John Ousterhout. 2021. A Linux Kernel Implementation of the Homa Transport Protocol. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*. USENIX Association, 99–115. https://www.usenix.org/conference/atc21/presentation/ousterhout

[128] Xiangyong Ouyang, David W. Nellans, Robert Wipfel, David Flynn, and Dhabaleswar K. Panda. 2011. Beyond block I/O: Rethinking traditional storage primitives. In *17th International Conference on High-Performance Computer Architecture (HPCA-17 2011), February 12-16 2011, San Antonio, Texas, USA*. IEEE Computer Society, 301–311. https://doi.org/10.1109/HPCA.2011.5749738

[129] Shoumik Palkar, James Thomas, Deepak Narayanan, Pratiksha Thaker, Rahul Palamuttam, Parimajan Negi, Anil Shanbhag, Malte Schwarzkopf, Holger Pirk, Saman Amarasinghe, Samuel Madden, and Matei Zaharia. 2018. Evaluating End-to-End Optimization for Data Analytics Applications in Weld. *Proc. VLDB Endow.* 11, 9 (may 2018), 1002–1015. https://doi.org/10.14778/3213880.3213890

[130] Johan Peltenburg, Lars T. J. van Leeuwen, Joost Hoozemans, Jian Fang, Zaid Al-Ars, and H. Peter Hofstee. 2020. Battling the CPU Bottleneck in Apache Parquet to Arrow Conversion Using FPGA. In *International Conference on Field-Programmable Technology, (IC)FPT 2020, Maui, HI, USA, December 9-11, 2020*. IEEE, 281–286. https://doi.org/10.1109/ICFPT51103.2020.00048

[131] Pensando. 2023. (An AMD company) The Pensando Distributed Services Card (DSC). https://www.amd.com/en/accelerators/pensando. Accessed: 2023-Jan-30.

[132] Peter-Jan Gootzen. 2023. Filesystem Virtualization using DPUs. https://github.com/IBM/dpu-virtio-fs. Accessed: 2023-Feb-02.

[133] Alex Petrov. 2018. Algorithms Behind Modern Storage Systems: Different Uses for Read-Optimized B-Trees and Write-Optimized LSM-Trees. *Queue* 16, 2 (apr 2018), 31–51. https://doi.org/10.1145/3212477.3220266

[134] Boris Pismenny, Haggai Eran, Aviad Yehezkel, Liran Liss, Adam Morrison, and Dan Tsafrir. 2021. Autonomous NIC Offloads. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Virtual, USA) *(ASPLOS 2021)*. Association for Computing Machinery, New York, NY, USA, 18–35. https://doi.org/10.1145/3445814.3446732

[135] Andrew Putnam, Adrian M. Caulfield, Eric S. Chung, Derek Chiou, Kypros Constantinides, John Demme, Hadi Esmaeilzadeh, Jeremy Fowers, Gopi Prashanth Gopal, Jan Gray, Michael Haselman, Scott Hauck, Stephen Heil, Amir Hormati, Joo-Young Kim, Sitaram Lanka, James Larus, Eric Peterson, Simon Pope, Aaron Smith, Jason Thong, Phillip Yi Xiao, and Doug Burger. 2014. A Reconfigurable Fabric for Accelerating Large-scale Datacenter Services. In *Proceeding of the 41st Annual International Symposium on Computer Architecuture (ISCA '14)*. IEEE Press, Minneapolis, Minnesota, USA, 13–24. http://dl.acm.org/citation.cfm?id=2665671.2665678

[136] Murad Qasaimeh, Kristof Denolf, Jack Lo, Kees A. Vissers, Joseph Zambreno, and Phillip H. Jones. 2019. Comparing Energy Efficiency of CPU, GPU and FPGA Implementations for Vision Kernels. In *15th IEEE International Conference on Embedded Software and Systems, ICESS 2019, Las Vegas, NV, USA, June 2-3, 2019*. IEEE, 1–8. https://doi.org/10.1109/ICESS.2019.8782524

[137] Zaid Qureshi, Vikram Sharma Mailthody, Isaac Gelado, Seung Won Min, Amna Masood, Jeongmin Park, Jinjun Xiong, CJ Newburn, Dmitri Vainbrand, I Chung, et al. 2022. BaM: A Case for Enabling Fine-grain High Throughput GPU-Orchestrated Access to Storage. *arXiv preprint arXiv:2203.04910* (2022).

[138] Kaveh Razavi and Animesh Trivedi. 2020. Stratus: Clouds with Microarchitectural Resource Management. In *12th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 20)*. USENIX Association. https://www.usenix.org/conference/hotcloud20/presentation/razavi

[139] Alessandro Rivitti, Roberto Bifulco, Angelo Tulumello, Marco Bonola, and Salvatore Pontarelli. 2023. EHDL: Turning EBPF/XDP Programs into Hardware Designs for the NIC. In *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3* (Vancouver, BC, Canada) *(ASPLOS 2023)*. Association for Computing Machinery, New York, NY, USA, 208–223. https://doi.org/10.1145/3582016.3582035

[140] Christopher J. Rossbach, Jon Currey, Mark Silberstein, Baishakhi Ray, and Emmett Witchel. 2011. PTask: Operating System Abstractions to Manage GPUs As Compute Devices. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles (SOSP '11)*. ACM, Cascais, Portugal, 233–248. https://doi.org/10.1145/2043556.2043579

[141] Zhenyuan Ruan, Tong He, and Jason Cong. 2019. INSIDER: Designing In-Storage Computing System for Emerging High-Performance Drive. In *2019 USENIX Annual Technical Conference (USENIX ATC 19)*. Renton, WA, 379–394. https://www.usenix.org/conference/atc19/presentation/ruan

[142] Deboleena Sakalley. 2022. Using FPGAs to accelerate NVMe-oF based Storage Networks. Accessed: 2023-Feb-02, https://www.flashmemorysummit.com/English/Collaterals/Proceedings/2017/20170810_FW32_Sakalley.pdf.

[143] Eric Schkufza, Michael Wei, and Christopher J. Rossbach. 2019. Just-In-Time Compilation for Verilog: A New Technique for Improving the

FPGA Programming Experience. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems* (Providence, RI, USA) *(ASPLOS '19)*. Association for Computing Machinery, New York, NY, USA, 271–286. https://doi.org/10.1145/3297858.3304010

[144] Robert Schmid, Max Plauth, Lukas Wenzel, Felix Eberhardt, and Andreas Polze. 2020. Accessible Near-Storage Computing with FPGAs. In *Proceedings of the Fifteenth European Conference on Computer Systems* (Heraklion, Greece) *(EuroSys '20)*. Association for Computing Machinery, New York, NY, USA, Article 28, 12 pages. https://doi.org/10.1145/3342195.3387557

[145] Adrian Schüpbach, Andrew Baumann, Timothy Roscoe, and Simon Peter. 2011. A Declarative Language Approach to Device Configuration. In *Proceedings of the Sixteenth International Conference on Architectural Support for Programming Languages and Operating Systems* (Newport Beach, California, USA) *(ASPLOS XVI)*. Association for Computing Machinery, New York, NY, USA, 119–132. https://doi.org/10.1145/1950365.1950382

[146] Sudharsan Seshadri, Mark Gahagan, Sundaram Bhaskaran, Trevor Bunker, Arup De, Yanqin Jin, Yang Liu, and Steven Swanson. 2014. Willow: A User-Programmable SSD. In *Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation* (Broomfield, CO) *(OSDI'14)*. USENIX Association, USA, 67–80.

[147] Yizhou Shan, Yutong Huang, Yilun Chen, and Yiying Zhang. 2018. LegoOS: A Disseminated, Distributed OS for Hardware Resource Disaggregation. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*. Carlsbad, CA, 69–87. https://www.usenix.org/conference/osdi18/presentation/shan

[148] Jonathan S. Shapiro and Jonathan Adams. 2002. Design Evolution of the EROS Single-Level Store. In *2002 USENIX Annual Technical Conference (USENIX ATC 02)*. USENIX Association, Monterey, CA. https://www.usenix.org/conference/2002-usenix-annual-technical-conference/design-evolution-eros-single-level-store

[149] Ran Shu, Peng Cheng, Guo Chen, Zhiyuan Guo, Lei Qu, Yongqiang Xiong, Derek Chiou, and Thomas Moscibroda. 2019. Direct Universal Access: Making Data Center Resources Available to FPGA. In *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*. Boston, MA, 127–140. https://www.usenix.org/conference/nsdi19/presentation/shu

[150] Mark Silberstein. 2017. OmniX: An Accelerator-Centric OS for Omni-Programmable Systems. In *Proceedings of the 16th Workshop on Hot Topics in Operating Systems* (Whistler, BC, Canada) *(HotOS '17)*. Association for Computing Machinery, New York, NY, USA, 69–75. https://doi.org/10.1145/3102980.3102992

[151] Mark Silberstein, Bryan Ford, Idit Keidar, and Emmett Witchel. 2013. GPUfs: Integrating a File System with GPUs. In *Proceedings of the Eighteenth International Conference on Architectural Support for Programming Languages and Operating Systems* (Houston, Texas, USA) *(ASPLOS '13)*. Association for Computing Machinery, New York, NY, USA, 485–498. https://doi.org/10.1145/2451116.2451169

[152] Gagandeep Singh, Mohammed Alser, Damla Senol Cali, Dionysios Diamantopoulos, Juan Gómez-Luna, Henk Corporaal, and Onur Mutlu. 2021. FPGA-Based Near-Memory Acceleration of Modern Data-Intensive Applications. *IEEE Micro* 41, 4 (2021), 39–48. https://doi.org/10.1109/MM.2021.3088396

[153] Theano Stavrinos, Daniel S. Berger, Ethan Katz-Bassett, and Wyatt Lloyd. 2021. Don't Be a Blockhead: Zoned Namespaces Make Work on Conventional SSDs Obsolete. In *Proceedings of the Workshop on Hot Topics in Operating Systems* (Ann Arbor, Michigan) *(HotOS '21)*. Association for Computing Machinery, New York, NY, USA, 144–151. https://doi.org/10.1145/3458336.3465300

[154] Brian Suchy, Souradip Ghosh, Drew Kersnar, Siyuan Chai, Zhen Huang, Aaron Nelson, Michael Cuevas, Alex Bernat, Gaurav Chaudhary, Nikos Hardavellas, Simone Campanoni, and Peter Dinda. 2022. CARAT CAKE: Replacing Paging via Compiler/Kernel Cooperation. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS 2022)*. Association for Computing Machinery, New York, NY, USA, 98–114. https://doi.org/10.1145/3503222.3507771

[155] Kuei Sun, Daniel Fryer, Joseph Chu, Matthew Lakier, Angela Demke Brown, and Ashvin Goel. 2018. Spiffy: Enabling File-System Aware Storage Applications. In *16th USENIX Conference on File and Storage Technologies (FAST 18)*. USENIX Association, Oakland, CA, 91–104. https://www.usenix.org/conference/fast18/presentation/sun

[156] Daniel Thompson and Leo Yan. 2018. Kernel analysis using eBPF. Accessed: 2023-Feb-02, https://elinux.org/images/d/dc/Kernel-Analysis-Using-eBPF-Daniel-Thompson-Linaro.pdf.

[157] Maroun Tork, Lina Maudlej, and Mark Silberstein. 2020. Lynx: A SmartNIC-Driven Accelerator-Centric Architecture for Network Servers. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS '20)*. Association for Computing Machinery, New York, NY, USA, 117–131. https://doi.org/10.1145/3373376.3378528

[158] Animesh Trivedi, Nikolas Ioannou, Bernard Metzler, Patrick Stuedi, Jonas Pfefferle, Ioannis Koltsidas, Kornilios Kourtis, and Thomas R. Gross. 2017. FlashNet: Flash/Network Stack Co-Design. In *Proceedings of the 10th ACM International Systems and Storage Conference* (Haifa, Israel) *(SYSTOR '17)*. Association for Computing Machinery, New York, NY, USA, Article 15, 14 pages. https://doi.org/10.1145/3078468.3078477

[159] Animesh Trivedi, Patrick Stuedi, Jonas Pfefferle, Adrian Schuepbach, and Bernard Metzler. 2018. Albis: High-Performance File Format for Big Data Systems. In *USENIX Annual Technical Conference (ATC)*. 615–630.

[160] Shin-Yeh Tsai, Yizhou Shan, and Yiying Zhang. 2020. Disaggregating Persistent Memory and Controlling Them Remotely: An Exploration of Passive Disaggregated Key-Value Stores. In *2020 USENIX Annual Technical Conference (USENIX ATC 20)*. USENIX Association, 33–48. https://www.usenix.org/conference/atc20/presentation/tsai

[161] Yatish Turakhia, Gill Bejerano, and William J. Dally. 2018. *Darwin: A Genomics Co-Processor Provides up to 15,000X Acceleration on Long Read Assembly*. Association for Computing Machinery, New York, NY, USA, 199–213. https://doi.org/10.1145/3173162.3173193

[162] Lluís Vilanova, Lina Maudlej, Shai Bergman, Till Miemietz, Matthias Hille, Nils Asmussen, Michael Roitzsch, Hermann Härtig, and Mark Silberstein. 2022. Slashing the Disaggregation Tax in Heterogeneous Data Centers with FractOS. In *Proceedings of the Seventeenth European Conference on Computer Systems* (Rennes, France) *(EuroSys '22)*. Association for Computing Machinery, New York, NY, USA, 352–367. https://doi.org/10.1145/3492321.3519569

[163] Han Wang, Robert Soulé, Huynh Tu Dang, Ki Suh Lee, Vishal Shrivastav, Nate Foster, and Hakim Weatherspoon. 2017. P4FPGA: A Rapid Prototyping Framework for P4. In *Proceedings of the Symposium on SDN Research* (Santa Clara, CA, USA) *(SOSR '17)*. Association for Computing Machinery, New York, NY, USA, 122–135. https://doi.org/10.1145/3050220.3050234

[164] Jagath Weerasinghe, Raphael Polig, François Abel, and Christoph Hagleitner. 2016. Network-attached FPGAs for data center applications. In *2016 International Conference on Field-Programmable Technology, FPT 2016, Xi'an, China, December 7-9, 2016*, Yuchen Song, Shaojun Wang, Brent Nelson, Junbao Li, and Yu Peng (Eds.). IEEE, 36–43.

https://doi.org/10.1109/FPT.2016.7929186

[165] Michael Wei, John D. Davis, Ted Wobber, Mahesh Balakrishnan, and Dahlia Malkhi. 2013. Beyond Block I/O: Implementing a Distributed Shared Log in Hardware. In *Proceedings of the 6th International Systems and Storage Conference* (Haifa, Israel) *(SYSTOR '13)*. Association for Computing Machinery, New York, NY, USA, Article 21, 11 pages. https://doi.org/10.1145/2485732.2485739

[166] M. Wijtvliet, L. Waeijen, and H. Corporaal. 2017. Coarse grained reconfigurable architectures in the past 25 years: overview and classification. In *Proceedings - 2016 16th International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation, SAMOS 2016*. Institute of Electrical and Electronics Engineers, United States, 235–244. https://doi.org/10.1109/SAMOS.2016.7818353 16th International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS 2016), 18-21 July 2016, Samos, Greece, SAMOS2016 ; Conference date: 18-07-2016 Through 21-07-2016.

[167] Yuanlong Xiao, Eric Micallef, Andrew Butt, Matthew Hofmann, Marc Alston, Matthew Goldsmith, Andrew Merczynski-Hait, and André DeHon. 2022. PLD: Fast FPGA Compilation to Make Reconfigurable Acceleration Compatible with Modern Incremental Refinement Software Development. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS 2022)*. Association for Computing Machinery, New York, NY, USA, 933–945. https://doi.org/10.1145/3503222.3507740

[168] Shuotao Xu, Sungjin Lee, Sang-Woo Jun, Ming Liu, Jamey Hicks, and Arvind. 2016. Bluecache: A Scalable Distributed Flash-Based Key-Value Store. *Proc. VLDB Endow.* 10, 4 (nov 2016), 301–312. https://doi.org/10.14778/3025111.3025113

[169] Chaoliang Zeng, Layong Luo, Teng Zhang, Zilong Wang, Luyang Li, Wenchen Han, Nan Chen, Lebing Wan, Lichao Liu, Zhipeng Ding, Xiongfei Geng, Tao Feng, Feng Ning, Kai Chen, and Chuanxiong Guo. 2022. Tiara: A Scalable and Efficient Hardware Acceleration Architecture for Stateful Layer-4 Load Balancing. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. USENIX Association, Renton, WA, 1345–1358. https://www.usenix.org/conference/nsdi22/presentation/zeng

[170] Yue Zha and Jing Li. 2020. Virtualizing FPGAs in the Cloud. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*. Association for Computing Machinery, New York, NY, USA, 845–858. https://doi.org/10.1145/3373376.3378491

[171] Teng Zhang, Jianying Wang, Xuntao Cheng, Hao Xu, Nanlong Yu, Gui Huang, Tieying Zhang, Dengcheng He, Feifei Li, Wei Cao, Zhongdong Huang, and Jianling Sun. 2020. FPGA-Accelerated Compactions for LSM-based Key-Value Store. In *18th USENIX Conference on File and Storage Technologies (FAST 20)*. USENIX Association, Santa Clara, CA, 225–237. https://www.usenix.org/conference/fast20/presentation/zhang-teng

[172] Mark Zhao, Mingyu Gao, and Christos Kozyrakis. 2022. ShEF: Shielded Enclaves for Cloud FPGAs. In *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems* (Lausanne, Switzerland) *(ASPLOS 2022)*. Association for Computing Machinery, New York, NY, USA, 1070–1085. https://doi.org/10.1145/3503222.3507733

[173] Yuhong Zhong, Haoyu Li, Yu Jian Wu, Ioannis Zarkadas, Jeffrey Tao, Evan Mesterhazy, Michael Makris, Junfeng Yang, Amy Tai, Ryan Stutsman, and Asaf Cidon. 2022. XRP: In-Kernel Storage Functions with eBPF. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*. USENIX Association, Carlsbad, CA, 375–393. https://www.usenix.org/conference/osdi22/presentation/zhong

[174] Noa Zilberman, Yury Audzevich, Georgina Kalogeridou, Neelakandan Manihatty-Bojan, Jingyun Zhang, and Andrew Moore. 2015. NetFPGA: Rapid Prototyping of Networking Devices in Open Source. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication* (London, United Kingdom) *(SIGCOMM '15)*. Association for Computing Machinery, New York, NY, USA, 363–364. https://doi.org/10.1145/2785956.2790029