# FPGA-based Near Data Processing Platform Selection Using Fast Performance Modeling (WiP Paper)

Nazanin Farahpour
nazanin@cs.ucla.edu
University of California, Los Angeles

Zhenman Fang
zhenman@sfu.ca
Simon Fraser University, Canada

Glenn Reinman
reinman@cs.ucla.edu
University of California, Los Angeles

## Abstract

With the trend of adopting FPGAs in data centers, various FPGA acceleration platforms have been developed in recent years. Each server could incorporate one or many of these FPGAs at different compute hierarchy levels to match its workload intensity. FPGAs could either be used as IO-attached accelerators or be closely integrated with CPU as on-chip co-processors. For a more data-centric approach, an FPGA could be moved closer to the data medium (RAM or disk) and serve as a near-memory or near-storage accelerator.

In this work, we present a quantitative model and in-depth analysis of application characteristics to determine when an application is more suitable for each acceleration hierarchy.

**ACM Reference Format:**
Nazanin Farahpour, Zhenman Fang, and Glenn Reinman. 2020. FPGA-based Near Data Processing Platform Selection Using Fast Performance Modeling (WiP Paper). In *Proceedings of the 21st ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES '20), June 16, 2020, London, United Kingdom.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3372799.3394373

## 1 Introduction

The increasing demand on power-efficient computing in today's data centers, has sparked a growing number of CPU-FPGA acceleration platforms that can be reconfigured to accelerate a broad class of applications with orders-of-magnitude performance/watt gains. Attaching FPGAs as an IO-attached accelerator (Section 2.1.1) is the most common way to deploy accelerators in the systems, especially for compute-intensive applications. While the performance and energy gains of such platforms are promising, their applicability is constrained by the working set size that cannot exceed the accelerator's private DRAM capacity.

*LCTES '20, June 16, 2020, London, United Kingdom*
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7094-3/20/06…$15.00
https://doi.org/10.1145/3372799.3394373

A typical server workload could comprise a diverse set of compute- and communication-bound applications [19]. A number of these applications are presented in Figure 3 to 5. They often interact with in-memory or in-storage datasets, and their throughput is crucial to the user experience. But today's commodity hardware solutions often follow a compute-centric design, which lacks adequate data access bandwidth for many of these use-cases and leads to substantial data transfer latency and energy consumption.

The potential benefits of data-centric computing have led to renewed research interest in near data processing (NDP) architectures in recent years. A large number of near-memory [8, 11, 22, 24, 25] and near-storage [10, 13, 14, 20] accelerator models have been proposed and studied (Section 2.2). We expect to see heterogeneous server racks with one or more types of these FPGA resources, being available for server workloads in near future. Thus, it is crucial to analyze the suitability of each FPGA resource for accelerating commonly used data center applications.

In this work, we present an in-depth analysis of applications' qualitative and quantitative attributes to derive a first-order performance model that could answer the following questions: (1) Could all applications dealing with in-storage/in-memory datasets benefit from a data-centric acceleration? (2) How would the performance change for each application, as we change the accelerator type and system configuration? To the best of our knowledge, this is the first work to evaluate all four categories of FPGA-based accelerators for data center applications.

## 2 Background and Related Work

### 2.1 Compute-Centric Acceleration

**2.1.1 PCIe-Attached Private-Memory Accelerators.** The most common integration is to connect an FPGA board equipped with private memory to a CPU via PCIe (Figure 1a). Amazon EC2 F1 instances [3] and Microsoft Catapult boards [9] use such integration due to its flexibility and easy plug-in. They are usually limited by the effective PCIe bandwidth and latency between the host CPU and FPGA. Thus, they are best suitable for coarse-grained tasks that have an initial large payload transfer to the private device memory, followed by high data reuse. If a dataset can fit in on-chip memory or the on-board DRAM (e.g., 32GB or 64GB DDR4) of a PCIe-attached FPGA, it could benefit from this platform. However, many datasets exceed the device memory capacity.

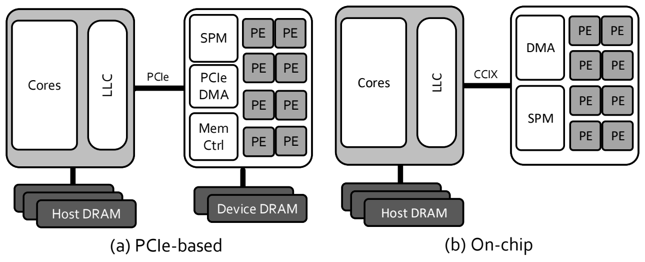

**Figure 1.** Compute-centric FPGA acceleration platforms

#### 2.1.2 Shared-Memory On-Chip Accelerators.

Unlike FPGA DRAM, the host-side DRAM capacity could be much larger than 64GB. To leverage the higher memory bandwidth and capacity, a closer server-FPGA integration has been proposed, where it couples the CPU and FPGA into a single package and provides a shared memory, cache-coherent interface to allow seamless data access to CPU cache or memory (Figure 1b). Recently, Intel has introduced two such on-chip FPGA platforms (AgileX [6], XeonSP [4]) that connect the Xeon processor and FPGA fabric in the same package through a cache-coherent interface. Xilinx has also introduced Versal ACAP platform [5] that connects the FPGA to the host CPU using CCIX (Cache Coherent Interconnect for Accelerators) link [1]. Moreover, with the capability of swapping partial bitstreams in less than a millisecond, the Versal ACAP could be utilized by multiple real-time applications simultaneously. However, once the working set exceeds the on-chip cache capacity, the acceleration is limited by the off-chip memory access latency and bandwidth.

### 2.2 Data-Centric Acceleration

#### 2.2.1 Near-Memory Accelerators.

The advancement of emerging memory and 3D stacking technologies is considered as the true enabler of processing close to memory. The stacking of the logic die and memory using through-silicon via (TSV) allows lower memory access latency and higher bandwidth. High bandwidth Memory (HBM) [18] from AMD and Hynix, and Samsung's Wide I/O [15] are the memory industries' competing 3D memory products. The logic die contains a dedicated memory controller, and could encompass simple SIMD cores or an embedded FPGA chip for data analysis. However, WideIO is used for mobile SoC systems and HBM is costly to populate the server memory and replace conventional DDR4.

Thus, we focus on near-memory accelerators for conventional DRAM architecture. Today's high-end servers have a limited number of memory channels per socket. Multiple DIMMs often share the same memory channel, which limits the overall bandwidth to the CPU. Near DRAM accelerators help achieve a lower latency and a higher bandwidth. For instance, Copacobana [17] builds FPGA modules directly into DIMMs. AIM [11] places FPGA modules between each DIMM and the memory bus, making the design noninvasive to the existing memory controller, memory bus and DIMMs.

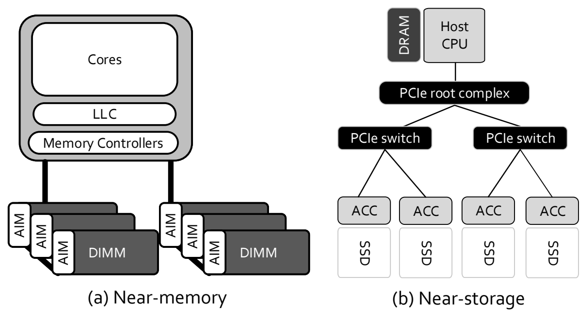

**Figure 2.** Data-centric FPGA acceleration platforms

Contutto [25] prototypes such idea by plugging accelerators in DIMM slots in a POWER8 machine and shows acceleration with end-to-end experiments. Our platform setup is most similar to AIM (Figure 2a).

#### 2.2.2 Near-Storage Accelerators.

With a revolution of non-volatile memory (NVM) technologies and more powerful embedded processors, the idea of near-storage acceleration has recently redrawn considerable attentions [10, 13, 14, 16]. This is mainly because an NVM-based SSD normally has a very high internal bandwidth, exceeding its external bandwidth to host by factors of 2x to 4x [10]. Therefore, processing data near-storage could achieve higher performance and save more energy than transferring data all the way to the host CPU. Several studies in NVM-based near-storage accelerators have been reported recently in academia and industry. Projects such as Samsung SmartSSD [13], IBM Netezza [23], Mobiveil [2], Willow [21] and BlueDBM [14] place FPGA units between the flash controller and the host IO interface. Our near-storage accelerator follows a similar architecture (Fig 2b). Flash chips have to be accessed in data-block granularity (4KB or 8KB). So, our near-storage accelerator features a standalone 1GB DRAM buffer, in order to hide the access granularity problem.

## 3 Workload Characterization

We evaluate 18 widely deployed accelerators from the Xilinx Vitis Library [7] and a few of our own extended from the PARADE simulation suite [12], which come from five diverse domains: search, security, database, vision, and finance. To minimize and amortize the data movement cost for a long period of time, there are a few application-specific and kernel-specific characteristics that help choose the best acceleration platform. From here, we refer to our four acceleration platforms by compute level: PCIe, on-chip, near-memory and near-storage levels.

### 3.1 Application-Specific Characteristics

There could be multiple FPGA kernels designed for a given application. However, there are certain attributes that are more dependent on the use-case of an application rather than the FPGA kernel itself, which play a role in selecting or ruling out a compute level.

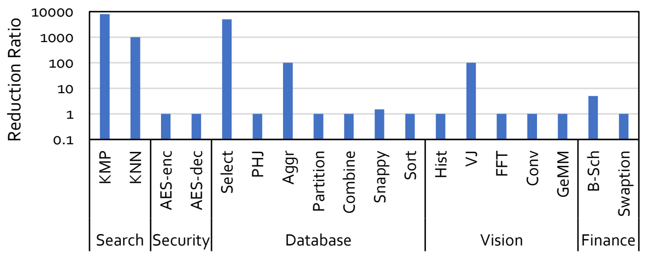

**Figure 3.** The ratio ($\gamma$) of the kernel's input to output size. For abbreviation in the figure, PHJ: Partition Hash Join; Aggr: Aggregate; Hist: EqualizeHist; VJ: Viola-Jones; B-Sch: Black-Scholes.

1) *Memory Capacity Requirement*: The total data size that is potentially accessed, modified or generated by the same application's threads over a long period of time.

2) *Working Set Size (WSS)*: We define WSS as the subset of the memory capacity that is required to process a single application call. For instance, a query processing engine equipped with Join, Aggregate and Partition FPGA kernels, might receive consecutive queries that target dozens of SQL tables, while working on a single table (or two) for each query. We consider the entire database size as the memory capacity and the size of the requested database tables per query as the working set size.

3) *Input/Output Data Flow*: If an application data flow never requires access to the storage level, then we would certainly rule out a near-storage acceleration. Similarly, if an application requires an in-situ manipulation of data (read-modify-write), then we would favor the near-storage or near-memory accelerators.

4) *Data Reduction Ratio ($\gamma$)*: The main motivation of near-data processing is to leverage the higher internal bandwidth. So, if an application input size is the same as or higher than its output size ($\gamma >= 1$), the application speed-up will be bound to the interconnection bandwidth for input. Figure 3 shows the reduction ratio $\gamma$ of our applications. The higher $\gamma$ is, the more chance near-data acceleration has to win.

### 3.2 Kernel-Specific Characteristics

For a given FPGA kernel, there is plenty of useful information that can be extracted from the kernel and its high-level synthesis, which helps us derive a performance model.

1) *Kernel Frequency (Freq), Initiation Interval (II) and input/output bit-width (datawidth)*: These help us compute the approximate computing time and maximum analytical bandwidth $bw_{peak} = datawidth \times Freq/II$. II is the number of cycles that the FPGA kernel takes to process one input of *datawidth* size, which shows the computational intensity of a kernel. Based on $bw_{peak}$, one can determine if a kernel would ever saturate the bandwidth of the attached memory module. Figure 4 shows the peak bandwidth per FPGA kernel. Applications like AES-enc and Viola-Jones have a low peak bandwidth and will not benefit from running near storage.

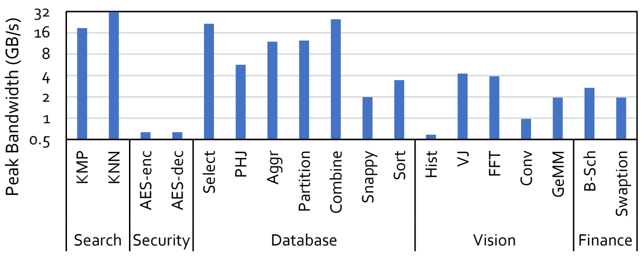

**Figure 4.** The analytical required bandwidth of the kernel

2) *Attached memory access pattern*: This helps us estimate the effective bandwidth utilization of the attached memory. Sequential access with large burst size result in higher effective bandwidth than scatter or gather patterns. The effective bandwidth of a tiled access depends on the tile row size.

3) *Multi-pass over the data* ($\alpha$): Maximum analytical bandwidth does not take into account the number of passes over the input data. Realistically, we have to multiply the data access time by $\alpha$.

4) *Intermediate data size ratio ($\beta$)*: Similar to multiple passes over the input data, another contributing factor to the execution time is the generated intermediate data size.

## 4 Performance Model

To guide the platform selection, we present a first-order performance model based on features from Section 3, assuming initial datasets are in storage. All of our FPGA-based accelerators have an on-chip scratchpad memory (SPM) and use the Xilinx AXI stream interface to communicate with the attached memory level. The AXI stream interface supports a maximum bus bit-width of 4,096. We adopt a three-stage accelerator execution model (load-compute-store) and double-buffering in SPM. The three stages can be scheduled to run concurrently. Therefore, the execution time of each compute level $l$ follows this equation:

$$T(l) = T_{init}(l) + max(T_{load}(l), T_{comp}(l), T_{store}(l)) \quad (1)$$

The compute time follows this equation:

$$T_{comp}(l) = \frac{(\alpha + \beta)D_{in}}{datawidth} \times \frac{II}{Freq_l} \times \frac{1}{\#PE_l} \quad (2)$$

where $D_{in}$ is the size of input data, and $\#PE_l$ is the number of concurrent processing elements (PEs).

**1) For near-storage (ns) FPGA**: The near-storage FPGA uses its own DRAM buffer as caching for intermediate variables ($\beta D_{in}$). $T_{init}(ns) = 0$ as there is no initial data loading.

$$T_{load}(ns) = \frac{\alpha D_{in}}{bw_{nvm}} + \frac{\beta D_{in}}{bw_{ddr}} \quad (3)$$

$$T_{store}(ns) = \frac{D_{out}}{bw_{nvm}} = \frac{D_{in}}{\gamma \times bw_{nvm}} \quad (4)$$

where $bw_{nvm}$ and $bw_{ddr}$ are the collective bandwidths of the NVM channels and local DRAM buffers.

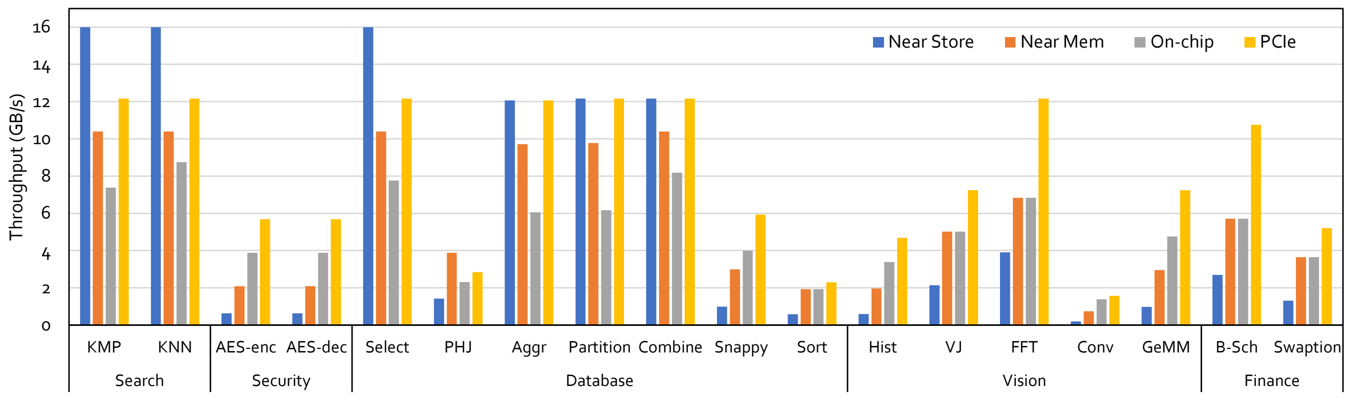

**Figure 5.** Application throughput for in-storage datasets based on four acceleration platforms

**2) For PCIe-attached FPGA**: We assume a direct PCIe connection between the FPGA and the host storage, and thus $T_{init}(pcie) = 0$.

$$T_{load}(pcie) = \frac{\alpha D_{in}}{bw_{hostIO}} + \frac{\beta D_{in}}{bw_{ddr}} \tag{5}$$

$$T_{store}(pcie) = \frac{D_{in}}{\gamma \times bw_{hostIO}} \tag{6}$$

where $bw_{hostIO}$ is the effective bandwidth of the PCIe.

**3) For near-memory (nm) and on-chip (oc) FPGAs**: Both on-chip and near-memory FPGAs incur a one-time data transfer time from storage to DDR: $T_{init}(oc) = T_{init}(nm) = D_{in}/bw_{hostIO}$.

Their load and store time follows the equations below.

$$T_{load}(nm) = \frac{(\alpha+\beta)D_{in}}{bw_{ddr}(nm)}, \quad T_{store}(nm) = \frac{D_{in}}{\gamma \times bw_{ddr}(nm)} \tag{7}$$

$$T_{load}(oc) = \frac{\alpha D_{in}}{bw_{ddr}(oc)} + \frac{\beta D_{in}}{bw_{llc}}, \quad T_{store}(oc) = \frac{D_{in}}{\gamma \times bw_{ddr}(oc)} \tag{8}$$

where the collective memory bandwidth ($bw_{ddr}(nm)$) of near-memory FPGAs depends on the number of DIMMs, while the ($bw_{ddr}(oc)$) for on-chip FPGAs depends on the number of memory channels. Thus, $bw_{ddr}(nm)$ is much higher than $bw_{ddr}(oc)$. $bw_{llc}$ is the collective bandwidth of the last-level cache.

### 4.1 Model Evaluation and Validation

We analyze a system that consists of one on-chip FPGA, one PCIe-attached FPGA, four DDR4 DIMMs and four NVMe SSDs (each DIMM/SSD connected to one FPGA). As shown in Figure 5, we find that simple kernels with no data reuse and a high reduction ratio—e.g., many applications from the search and database domains—benefit the most from mapping to near-storage accelerators. Second, for compute-bound applications such as in security, finance and vision domains and some from database domain, it helps to stream the data to a more powerful PCIe-attached accelerator. Moreover, for communication-bound applications, the throughput is ultimately bound by the near-storage accelerator's performance.

For model validation, we extend the cycle-accurate simulator PARADE [12] to model the four categories of accelerators and their attached memory modules. We convert six of our applications into simulator-compatible versions and plug in their parameters. Table 1 compares the best performing platform for each benchmark according to our model and simulation results: four out of six applications have a similar selection. Meanwhile, the simulation shows that the performance gap between on-chip, near-memory and PCIe-attached accelerator are much smaller than what the model suggests. We realize that many of the applications (e.g. GeMM, Swaption) consist of consecutive accelerator kernel calls and each accelerator call works on part of the input data. So, the $T_{init}(oc)$ and $T_{init}(nm)$ are partially concealed by the kernel executions. For applications with multiple kernel calls, we could include $T_{init}$ as part of the $T_{load}$ for a more fair comparison.

**Table 1.** Preliminary model validation

| | Conv | GeMM | KNN | VJ | B-Sch | Swaption |
|---|---|---|---|---|---|---|
| Modeling | PCIe | PCIe | NS | PCIe | PCIe | PCIe |
| Simulation | OC/PCIe | NM | NS | PCIe | PCIe | NM/OC |
| Matching | ✓ | × | ✓ | ✓ | ✓ | × |

## 5 Conclusion

In this work, we present an in-depth analysis of applications' qualitative and quantitative attributes and derive a first-order performance model for near-data FPGA platform selection. In future work, we plan to validate more benchmarks and address other limitations of our performance model.

## Acknowledgements

We acknowledge the support from Center for Domain-Specific Computing; NSERC Discovery Grant RGPIN-2019-04613 and DGECR-2019-00120; Simon Fraser University New Faculty Start-up Grant; Samsung, Huawei, and Xilinx.

## References

[1] 2017. An Introduction to CCIX. https://www.synopsys.com/designware-ip/technical-bulletin/introduction-ccix-2017q3.html.

[2] 2017. Mobiveil Announces FPGA-Based SSD Platform. https://www.globenewswire.com/news-release/2017/08/03/1215428/0/en/Mobiveil-Announces-FPGA-Based-SSD-Platform-for-3D-NAND-Flash-Devices-Upgrades-NVMe-PCI-Express-Controllers-to-Support-Latest-Specifications.html.

[3] 2018. Amazon EC2 F1 Instance. https://aws.amazon.com/ec2/instance-types/f1/.

[4] 2018. Intel Xeon Scalable Processor 6138p. https://www.eejournal.com/article/intel-delivers-xeon-scalable-processor-6138p-with-arria-10-gx-1150-fpga/.

[5] 2019. Versal: The First Adaptive Compute Acceleration Platform (ACAP). https://www.xilinx.com/support/documentation/white_papers/wp505-versal-acap.pdf.

[6] 2019. With AgileX Intel gets a coherent FPGA strategy. https://www.nextplatform.com/2019/04/02/with-agilex-intel-gets-a-coherent-fpga-strategy.

[7] 2019. Xilinx Vitis Accelerated Libraries. https://www.xilinx.com/products/design-tools/vitis/vitis-libraries.html.

[8] Ping Chi, Shuangchen Li, Cong Xu, Tao Zhang, Jishen Zhao, Yongpan Liu, Yu Wang, and Yuan Xie. 2016. PRIME: A Novel Processing-in-Memory Architecture for Neural Network Computation in ReRAM-Based Main Memory. In *Proceedings of the 43rd International Symposium on Computer Architecture* (Seoul, Republic of Korea) *(ISCA '16)*. IEEE Press, 27–39.

[9] Derek Chiou. 2017. The microsoft catapult project. In *2017 IEEE International Symposium on Workload Characterization (IISWC)*. 124–124.

[10] Sangyeun Cho, Chanik Park, Hyunok Oh, Sungchan Kim, Youngmin Yi, and Gregory R Ganger. 2013. Active Disk Meets Flash: A Case for Intelligent SSDs. In *Proceedings of the 27th International ACM Conference on International Conference on Supercomputing* (Eugene, Oregon, USA) *(ICS '13)*. Association for Computing Machinery, New York, NY, USA, 91–102.

[11] Jason Cong, Zhenman Fang, Michael Gill, Farnoosh Javadi, and Glenn Reinman. 2017. AIM: accelerating computational genomics through scalable and noninvasive accelerator-interposed memory. In *Proceedings of the International Symposium on Memory Systems* (Alexandria, Virginia) *(MEMSYS '17)*. Association for Computing Machinery, New York, NY, USA, 3–14.

[12] Jason Cong, Zhenman Fang, Michael Gill, and Glenn Reinman. 2015. PARADE: A cycle-accurate full-system simulation Platform for Accelerator-Rich Architectural Design and Exploration. In *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 380–387.

[13] Jaeyoung Do, Yang-Suk Kee, Jignesh M. Patel, Chanik Park, Kwanghyun Park, and David J. DeWitt. 2013. Query Processing on Smart SSDs: Opportunities and Challenges. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data* (New York, New York, USA) *(SIGMOD '13)*. Association for Computing Machinery, New York, NY, USA, 1221–1230.

[14] Sang-Woo Jun, Ming Liu, Sungjin Lee, Jamey Hicks, John Ankcorn, Myron King, Shuotao Xu, and Arvind. 2015. Bluedbm: An appliance for big data analytics. In *Proceedings of the 42nd Annual International Symposium on Computer Architecture* (Portland, Oregon) *(ISCA '15)*. Association for Computing Machinery, New York, NY, USA, 1–13.

[15] J. Kim, C. S. Oh, H. Lee, D. Lee, H. Hwang, S. Hwang, B. Na, J. Moon, J. Kim, H. Park, J. Ryu, K. Park, S. Kang, S. Kim, H. Kim, J. Bang, H. Cho, M. Jang, C. Han, J. Lee, K. Kyung, J. Choi, and Y. Jun. 2011. A 1.2V 12.8GB/s 2Gb mobile Wide-I/O DRAM with 4×128 I/Os using TSV-based stacking. (2011), 496–498.

[16] 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.

[17] Sandeep Kumar, Christof Paar, Jan Pelzl, Gerd Pfeiffer, and Manfred Schimmler. 2006. Breaking Ciphers with COPACOBANA –a Cost-Optimized Parallel Code Breaker. In *Proceedings of the 8th International Conference on Cryptographic Hardware and Embedded Systems* (Yokohama, Japan) *(CHES'06)*. Springer-Verlag, Berlin, Heidelberg, 101–118.

[18] D. U. Lee, K. W. Kim, K. W. Kim, H. Kim, J. Y. Kim, Y. J. Park, J. H. Kim, D. S. Kim, H. B. Park, J. W. Shin, J. H. Cho, K. H. Kwon, M. J. Kim, J. Lee, K. W. Park, B. Chung, and S. Hong. 2014. 25.2 A 1.2V 8Gb 8-channel 128GB/s high-bandwidth memory (HBM) stacked DRAM with effective microbump I/O test methods using 29nm process and TSV. In *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*. 432–433.

[19] Asit K. Mishra, Joseph L. Hellerstein, Walfredo Cirne, and Chita R. Das. 2010. Towards Characterizing Cloud Backend Workloads: Insights from Google Compute Clusters. *SIGMETRICS Perform. Eval. Rev.* 37, 4 (March 2010), 34–41.

[20] 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 2019, Renton, WA, USA, July 10-12, 2019*. USENIX Association, 379–394.

[21] 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.

[22] Ali Shafiee, Anirban Nag, Naveen Muralimanohar, Rajeev Balasubramonian, John Paul Strachan, Miao Hu, R. Stanley Williams, and Vivek Srikumar. 2016. ISAAC: A Convolutional Neural Network Accelerator with in-Situ Analog Arithmetic in Crossbars. In *Proceedings of the 43rd International Symposium on Computer Architecture* (Seoul, Republic of Korea) *(ISCA '16)*. IEEE Press, 14–26.

[23] Malcolm Singh and Ben Leonhardi. 2011. Introduction to the IBM Netezza warehouse appliance. In *Proceedings of the 2011 Conference of the Center for Advanced Studies on Collaborative Research*. IBM Corp., 385–386.

[24] Linghao Song, Xuehai Qian, Hai Li, and Yiran Chen. 2017. PipeLayer: A Pipelined ReRAM-Based Accelerator for Deep Learning. In *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. 541–552.

[25] Bharat Sukhwani, Thomas Roewer, Charles L. Haymes, Kyu-Hyoun Kim, Adam J. McPadden, Daniel M. Dreps, Dean Sanner, Jan Van Lunteren, and Sameh Asaad. 2017. Contutto: A Novel FPGA-Based Prototyping Platform Enabling Innovation in the Memory Subsystem of a Server Class Processor. 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, 15–26.