System Support for Environmentally Sustainable Computing in Data Centers

Fan Chen

Luddy School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN E-mail: fc7@iu.edu

Abstract-Modern data centers suffer from a growing carbon footprint due to insufficient support for environmental sustainability. While hardware accelerators and renewable energy have been utilized to enhance sustainability, addressing Quality of Service (QoS) degradation caused by renewable energy supply and hardware recycling remains challenging: (1) prior accelerators exhibit significant carbon footprints due to limited reconfigurability and inability to adapt to renewable energy fluctuations; (2) integrating recycled NAND flash chips in data centers poses challenges due to their short lifetime, increasing energy consumption; (3) the absence of a sustainability estimator impedes data centers and users in evaluating and improving their environmental impact. This study aims to improve system support for environmentally sustainable data centers by proposing a reconfigurable hardware accelerator for intensive computing primitives and developing a fractional NAND flash cell to extend the lifetime of recycled flash chips while supporting graceful capacity degradation. We also introduce a sustainability estimator to evaluate user task energy consumption and promote sustainable practices. We present our preliminary results and recognize this as an ongoing initiative with significant potential to advance environmentally sustainable computing in data centers and stimulate further exploration in this critical research domain.

Index Terms—Environmental sustainability, embodied carbon, operational carbon, hardware accelerators, storage systems.

I. Introduction

Modern data centers contribute significantly to an unsustainable carbon footprint, from both operational and embodied sources. Operational emissions arise from routine hardware usage, while *embodied* emissions result from manufacturing and construction processes. Global data center operational energy is projected to reach 4.5% of global demand by 2025 [1]. Additionally, Over the past decade, global data center infrastructure capacity, reflecting embodied carbon footprint, has surged $6 \times [2]$. Renewable energy and hardware recycling are two critical methods to enhance the environmental sustainability of future data centers. Several cloud companies [3] have collectively invested in 22 GW of renewable energy generation to reduce the operational carbon footprint. Meanwhile, recent work considers reusing obsolete hardware, e.g., mobile CPUs [4], mobile DRAMs [4], and FPGAs [5], to reduce the embodied carbon footprint in data centers. However, mitigating Quality of Service (QoS) degradation in future sustainable data centers caused by renewable energy supply and hardware recycling poses a complex challenge.

On one side, renewable energy supply significantly degrades QoS due to its intermittent nature. Solar and wind generate 47% and 34% of the renewable energy [6], leading to hourly and seasonal fluctuations in energy supply. While large energy

storage [3], [7] can mitigate these fluctuations, their addition significantly increases the embodied carbon footprint of data centers. Consequently, servers may need to be slowed down or shut down during low energy supply periods. Software-based techniques such as carbon-aware scheduling [3] and virtual transient servers [7] are proposed to schedule non-critical tasks among CPUs without significant QoS degradation in the face of energy supply fluctuations. However, emerging workloads such as big data, deep learning, and cryptography heavily use accelerators [8], [9] and large data storage [10]. There is a near total lack of hardware (particularly accelerator and storage) support for maintaining the QoS in data centers powered by renewable energy supply.

On the other side, while hardware recycling reduces the embodied carbon footprint, it compromises data center QoS due to the limited performance and reliability of obsolete hardware. Recent attempts to integrate outdated computing components like mobile processors [4] and DRAMs [4] fall short in handling demanding tasks in emerging workloads. Thus, future data centers will require reconfigurable application-specific hardware acceleration with low embodied carbon footprints. Although few works [11] explore reusing obsolete NAND flash chips in data center storage systems, utilizing aging flash chips with about-to-worn-out blocks poses challenges in supporting long-term workloads in future data centers.

Below, we provide a detailed overview of these challenges across the computing and storage layers of system stacks:

- Hardware Acceleration: Previous GPU-, FPGA- [5], [12], ASIC- [8] and PIM-based [9] accelerators have been utilized to mitigate performance loss caused by hardware recycling. However, none effectively addresses energy supply fluctuations or maintains a low embodied carbon footprint. Specifically, GPU- and application-specific-integratedcircuit (ASIC)-based accelerators lack the ability for rapid wake-up or shutdown in response to energy supply fluctuations. While FPGA- [13], [14] and PIM-based [9], [15] accelerators adopt nonvolatile memory arrays for data retention, critical components such as SRAM-based switches in FPGAs and analog-to-digital converters (ADCs) in PIMs still lose information without power, hindering prompt workload resumption after power loss. ASIC-based accelerators [8] focusing on accelerating only one type of computing kernels suffer from high embodied carbon footprint, due to their lack of reconfigurability.
- NAND Flash: To reduce embodied carbon footprint, old NAND flash chips will be adopted by future sustainable data centers. However, flash suffers from short cell en-

durance [16], where *about-to-worn-out* blocks limit chip lifetime. Despite error correcting code (ECC) [17], [18] and wear leveling [19] assistance, no prior technique significantly extends the lifetime of about-to-worn-out blocks, hindering widespread adoption of recycled NAND flash chips in future data centers. Moreover, due to fluctuations in renewable energy supply, DRAM snapshots are frequently stored in and loaded from storage systems. Unfortunately, most recycled flash chips use multi-level cells (MLCs) [20], suffering from long program and read operations.

• Sustainability Estimator: Without an environmental sustainability estimator, data centers struggle to assess the operational and embodied energy of user workloads, leading to a lack of visibility regarding the sustainability costs of user actions. Additionally, the absence of such metrics inhibits the development of flexible billing policies that could reward users who opt for recycled hardware and encourage broader adoption of environmentally friendly practices.

II. APPROACHES

This project aims to develop system support for environmentally sustainable data centers, with three main objectives: (1) Facilitate energy-efficient hardware acceleration for memory-intensive tasks such as big data, deep learning, and cryptography, ensuring steady progress even in the face of renewable energy fluctuations; (2) Extend the lifetime of about-to-worn-out blocks in recycled NAND flash chips, thereby mitigating the embodied energy consumption; and (3) Provide an accurate evaluation of both operational and embodied energy consumption for data center user tasks and incentivize users to make their computing more environmentally sustainable. The primary approach entails constructing an environmentally sustainable computing system. This system encompasses a nonvolatile reconfigurable PIM accelerator, a storage system based on recycled NAND flash, and an accurate environmental sustainability estimator.

A. Amoeba - A Reconfigurable Nonvolatile Accelerator

Existing PIMs in data centers encounter substantial manufacturing costs and operational inefficiencies, particularly when faced with intermittent renewable energy sources, resulting in significant embodied and operational carbon emissions. We propose Amoeba, depicted in Figure 1, as a reconfigurable FeFET-based PIM architecture aimed at addressing these challenges. Our approach involves profiling computeand resource-intensive computational kernels in data centers, including NTT for lattice-based cryptography, SHA3 for blockchains, and convolution for machine learning, leveraging insights from prior studies. Categorizing basic operations and supporting computation primitives with three types of processing engines (PE) based on the FeFET crossbar structure, Amoeba offers fine-grained PE-level reconfiguration to markedly minimize embodied carbon emissions. Furthermore, serving as a fully nonvolatile accelerator, it ensures consistent forward progress even amidst renewable energy fluctuations, thereby contributing to a notable reduction in the operational carbon footprint.

PE-Level Reconfigurability for Reducing Embodied Carbon Footprint. Crossbars in Amoeba can be logically configured into different modes: (1) Associative PE (APE) for LUT and ADD. Each crossbar acts as an embedded CAM

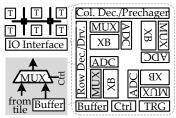
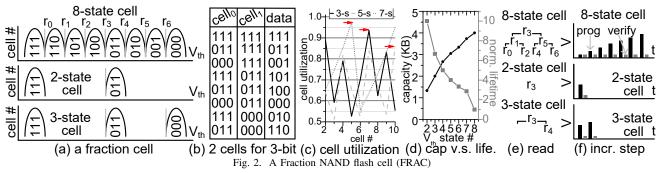


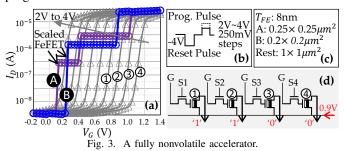
Fig. 1. The Amoeba architecture.

block for associative computing. As in previous work [21], each unit consists of complementary cells (H and L) for 1' and 0', respectively. The search word is encoded by complementary voltages (V/0 for 1' and 0/V for 0') and broadcasted onto vertical search lines. After a parallel search, the horizontal match line identifies matches and outputs the vector for postprocessing. LUT is directly implemented. For ADD, bitwise search-based additions are cascaded for multi-bit addition, allowing for parallel execution. (2) Multiplicaiton PE (MPE) for MVM and SHIFT. The crossbar-based MVM unit, as in [22], [23], maps dataflow in a weight-stationary manner. Weights are pre-coded onto columns, and input feature maps are converted to read voltages and applied to horizontal wordlines. The ADC aggregates current at vertical bitlines, representing dot-products. SHIFT operations comprise >40% of NTT workloads, but they aren't inherently MVM operations and can't be directly mapped to crossbars. We propose transferring SHIFT to MVM by pre-coding the transformation matrix onto the crossbar and applying the input as a multiplicand. This scheme, based on linear algebra, can be generalized to support any cyclic permutation. (3) Computing PE (CPE) for LOGIC operations. We perform Logic operations like AND and XOR within the crossbar array without using CMOS ALUs. In contrast to prior methods [24], [25], which necessitate a larger crossbar array, our approach only requires a 2×N crossbar, where N represents the input width. Take ADD as an example, we adjust ADC sensing levels to output a logical '1' only when both cells are in a low-resistance state. (4) Combining APE and MPE for MUL. Prior algorithms [26], [27] broke an N-bit MUL to SHIFT and ADD of partial products. [28] implemented such MUL with bitwise operations based ADD and implicit data selection based SHIFT. Instead, we combine the APE and MPE to implement MUL. The benefits are twofold: first, the search-based ADD reduces expensive read/write operations in [28]; second, the MVM based explicit SHIFT reduces the complex control for memory block selection in [28].

Fully Nonvolatile Accelerator for Reducing Operational Carbon Footprint. To reduce the *operational* carbon footprint, we propose a fully nonvolatile accelerator with FeFET-based True Random Number Generator (TRG) and ultralow power FeFET-based ADC. (1) **FeFET-based TRG**. We leverage the stochastic switching behavior in scaled FeFET devices [29] as an on-chip entropy source for TRG designs, same as prior work [30]. Our baseline TRG employs a $0.2\mu m \times 0.2\mu m$ FeFET device with a read voltage set at 0.2V. Initial findings indicate a bias towards '0's in the output bits. To mitigate this bias, we propose a tracking scheme involving



an 8-bit counter to record output probabilities in consecutive 256-bit segments. The counter's output then acts as the control signal for adjusting write voltages in generating subsequent segments. (2) Low-Power ADC with Tunable Precision. We develop a precision-scalable, low-power ADC using FeFET technology for faster analog-to-digital conversion. Figure 3(d) illustrates a four-bit ADC. We utilize four partially polarized $1\mu m \times 1\mu m$ FeFET devices ((1)~(4)). The write line toggles between 0V and V_{DD} for enabling/disabling. Inputs are applied to the data line. Due to variations in V_{th} , read currents differ. For instance, with a 0.9V input in Figure 3(d), higherthan-threshold current is detected on (1) and (2), while lower current is observed on (3) and (4). This results in a 4-bit output "1100" with an appropriately set current threshold. The ADC's precision can be easily adjusted by disabling (1) and (3). Threshold currents for each device can be dynamically programmed.



B. FRAC - Recycling Used NAND Flash Chips

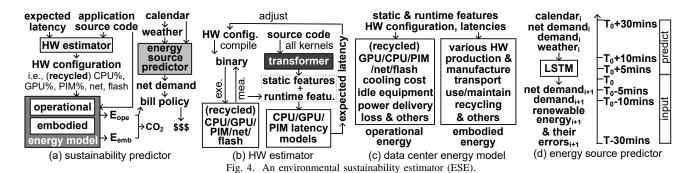
To reduce embodied energy, repurposing outdated NAND flash chips in sustainable data centers is crucial. These chips, manufactured using older process technologies, may have undergone varying levels of write operations before recycling, leading to differing remaining lifetimes. While existing work [31]–[33] salvages already-worn-out blocks, the remaining lifetime of recycled chips is predominantly influenced by about-to-worn-out blocks. We propose FRAC to improve the lifetime of about-to-worn-out blocks by exploring the trade-off between chip capacity and cell lifetime. FRAC supports graceful storage capacity degradation by gradually decreasing the number of V_{th} states in a cell.

FRAC: A NAND Flash Fraction Cell. A conventional flash cell (e.g., SLC, MLC, TLC, or QLC) uses $2^n V_{th}$ states to store n bits respectively, with $n \in [1,4]$. Figure 2(a) illustrates a TLC cell. In contrast, FRAC uses $m V_{th}$ states to store data, where $m \in [2,2^n]$, e.g., m=3. However, a fraction cell having 3 V_{th} states can represent only 1 bit, wasting 1 V_{th} state. To use

more V_{th} states in a cell, we propose to use α FRAC cells, each of which has m V_{th} states, to store $\lfloor \log_2(m^\alpha) \rfloor$ bits. For instance, two 3-state cells can store 3 bits. The truth table of two 3-state cells is shown in Figure 2(b). By looking up this table, the value of two 3-state cells is translated to 3 bits. We define cell utilization as the number of V_{th} states representing data divided by the total number of V_{th} states of FRAC cells. We explore the cell utilization of 3-, 5-, and 7-state cells in Figure 2(c). To achieve the highest cell utilization, we can store 11 bits in seven 3-state cells, 16 bits in ten 5-state cells, and 16 bits in five 7-state cells.

Tradeoff Between Capacity and Lifetime. The endurance of a flash cell (L) has a power-dependence on the number of program/erase (P/E) cyclings (N_{PE}) [34], [35], i.e., $L \propto N_{PE}^{\beta}$, where $\beta \geq 0.3$. Producing more V_{th} states in a flash cell requires more program operations [36], thereby greatly decreasing the cell endurance. Compared to a 3-bit TLC, a 2-state cell shown in Figure 2(a) prolongs the cell endurance by $10 \times [37]$. Although prior work [38] improves the flash lifetime by converting 2-bit MLC to 1-bit SLC, the 50% capacity loss does not maintain a gradual and graceful capacity degradation. In contrast, FRAC can gracefully explore the tradeoff between the chip endurance and capacity, as shown in Figure 2(d). By gradually reducing the number of V_{th} states from 8 to 2, FRAC gracefully degrades the capacity of a page from 4KB to 1.3KB and prolongs the cell endurance from $1 \times$ to $10 \times$.

Read and Write. To sense data from a TLC, as Figure 2(e) exhibits, totally $log_2(8) = 3$ iterations are required. The reading reference of this iteration is decided by the result of the previous iteration. For example, after comparing against r_3 , if the V_{th} state of the cell is larger than r_3 , r_5 will be selected in the next iteration. Otherwise, r_1 will be chosen. The same as conventional MLC, TLC, and QLC, a read operation on a mstate faction cell also requires $\lceil \log_2(m) \rceil$ iterations. After the data are sensed out of the cells in a page, the data in multiple cells is translated to one value. After an erase operation, to program a TLC, the incremental step pulse scheme [39] issues multiple programming pulses, each of which is stronger than the previous one, as shown in Figure 2(f). Between two programming pulses, there is a verify that checks whether the cell reaches the target V_{th} or not. On the contrary, to program an m-state cell, instead of a small pulse, the incremental step pulse scheme can directly start with a larger pulse. In this way, FRAC requires fewer pulses and thus prolongs the cell endurance, when m < 8.



C. ESE - An Environmental Sustainability Estimator

We introduce the Environmental Sustainability Estimator (ESE), an innovative tool facilitating accurate assessment of CO2 emissions linked with user activities. ESE integrates environmental sustainability into data center billing policies, incentivizing users to adopt more sustainable computing practices. Illustrated in Figure 4(a), ESE comprises a hardware estimator, a data center energy model, and an energy source predictor. First, the hardware estimator analyzes user source code and expected Quality of Service (latency), partitions tasks, and distributes them across CPUs, GPUs, and PIMs to ensure latency requirements are met. It also assesses storage capacity and network bandwidth based on task data access patterns. Second, the data center energy model will use the hardware estimator output to produce both operational energy consumption (E_{ope}) and embodied energy consumption (E_{emb}) . Third, the user inputs the task's start time to access weather forecasts. The energy source predictor utilizes this information along with calendar data to determine the net energy demand, representing the disparity between total data center operational energy and current renewable energy generation. Finally, based on the values of E_{ope} , E_{emb} , and net energy demand, the data center uses different billing policies to decide the user charge.

Hardware estimator. The hardware estimator minimizes E_{ope} and E_{emb} for each user task within a latency constraint on a heterogeneous hardware platform. It takes source code and expected latency as inputs. The source code is compiled into binaries for an initial hardware configuration, and static features are extracted directly. Runtime features are collected by executing the binaries on real hardware. A latency model processes both types of features to estimate CPU/GPU/PIM/net/SSD latency. The task's estimated latency is the sum of these values. If the estimated latency exceeds the expected latency, kernels are moved to GPUs or PIMs, and the estimator iterates until the expected latency is achieved. Otherwise, the minimum estimated latency is returned.

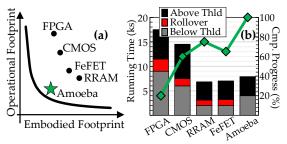
Traditional task partition methods [40], [41] rely on *static features* such as branch count and arithmetic operations, extracted from source code using intermediate representations (IRs). However, these handcrafted features often overlook inter-kernel data flows, treating each kernel independently. Our hardware estimator employs a transformer architecture [42] to directly extract more informative static features from the source code, capturing interactions among all kernels. Leveraging a pre-trained transformer [43] tailored for code gener-

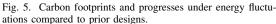
ation eliminates the need for training from scratch. *Runtime features* encompass CPU cache misses, GPU core utilization, PIM computing unit utilization, and host-device data transfers, among others. Measured using performance profiling tools like Intel VTune and NVIDIA visual profiler, these features are assessed after deploying compiled binaries on real hardware.

To expedite latency evaluation, we train a CNN as a latency model utilizing static and runtime features alongside hardware configuration to predict latency. We create a latency dataset by generating varied partition results and measuring latency values on CPUs, GPUs, and PIMs for training. Post-training, partition results won't require execution on real hardware; instead, latency models will directly provide values.

Data Center Energy Model. The energy model comprises an operational and an embodied energy model. The energy model inputs are the static and runtime features, hardware configuration, and hardware latency values, while its outputs are operational energy and embodied energy. We will train a CNN as the operational energy model. Similar to latency models in the hardware estimator, we will randomly generate different partition results and measure the energy consumption on CPUs, GPUs, PIMs, networks, and storage systems to build an energy dataset for training the energy model. Besides the dynamic and leakage energy consumption of various hardware units, we will also consider the cooling energy cost, idle equipment energy, power delivery loss, and other operationrelated energy in the operational energy model. The embodied energy model is a linear equation. We will compute the total embodied energy of a hardware unit by considering the energy consumption during production and manufacture, transport, use and maintenance, recycling, and other stages in the whole hardware lifetime. The embodied energy (E_{emb}) of a user task can be computed as $\sum_{i \in X} TBE_i \times \frac{latency_i}{lifetime_i}$, where Xindicates the hardware units used by the task; TBE_i is the total embodied energy of the hardware unit i; $latency_i$ means the latency of the hardware unit i; and $lifetime_i$ represents the lifetime of the hardware unit i.

Energy Source Predictor. The energy source predictor aims to characterize the distribution of net energy demand forecast errors under various weather conditions and data center workload capacities, as shown in Figure 4(d). The energy source predictor will be built as a long short-term memory (LSTM) network to output simultaneous quantile forecasts of net energy demand and generated renewable energy. The network inputs include an array of calendar data and weather





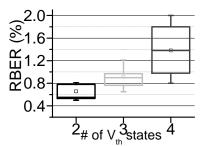


Fig. 6. The Raw Bit Error Rate (RBER) of FRAC-based pages in a recycled flash chip.

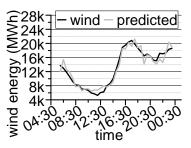


Fig. 7. The prediction on the wind energy generation in the California grid.

information. The network will be trained on historical records and predicts the forecasts and forecast errors of T_0+5 -minute, T_0+10 -minute, and T_0+15 -minute at T_0 . After training, the network will ingest near-past 5-minute forecasts as well as near-past 10-minute and near-past 15-minute historical data available at T_0 . More specifically, the network predicts the conditional quantiles of forecast errors for net energy demand and renewable energy production. We will consider seven target quantiles, i.e., P2.5, P5, P25, P50, P75, P95, and P97.5.

III. RESULTS

We offer preliminary results of our ongoing research. These findings highlight the potential of our work to advance environmentally sustainable computing in data centers.

Amoeba Results. We evaluated Amoebaby implementing a simple prototype. We compare Amoeba against prior accelerators designs, i.e., FPGA [44], CMOS [45], RRAM [46], and FeFET [47]. We use three benchmarks including a 32k NTT using Montgomery reduction with a fixed q = 12289 [28], SHA3 with a 1088 blocksize and 1600-bit state sizes, and AlexNet [45]. For embodied carbon emissions, we mainly focused on manufacturing costs and used the total cost for three different domain accelerators as the overall footprint. We consider California grid [48] as the renewable power supply and adopt its historical data, taking into account dynamic intermittency and fluctuations. We adopt the recently proposed Carbon Explorer framework [48] to study the Pareto Frontier analysis and report the design solution for each scheme in Figure 5(left). In general, the Amoeba achieves state-of-theart carbon minimization. Figure 5(right) compares the breakdown of simulation run times and computational progress for AlexNet inference at different times of a week under California grid [48] supply. For all architectures, Amoeba achieves the highest forward progress since the fully nonvolatile accelerator can work below threshold power (i.e., below Thld) with reduced operational energy consumption. In addition, we can also see that the fluctuation of the power supply imposes large rollover penalties for nonvolatile processors and CMOS circuitries in existing RRAM and FeFET accelerators.

FRAC Results. We evaluated the viability of FRAC using a prototype built upon a Zynq FPGA [49] with a NAND flash daughter broad. Recycled flash chips can be plugged into the flash daughter board, while we implement the control logic for FRAC read, program, and erase operations on the FPGA that can access those chips. Our prototype is a minimal

storage system built upon recycled flash chips and can be used to study the lifetime improvement achieved by FRAC. Our prototype primarily translates high-level read/write I/O requests to device-level NAND flash commands. We tested the RBER of FRAC cells with different numbers of V_{th} states using our prototype. Figure 6 highlights the RBER of FRAC-based pages subjected to 6k programming and erasure cycles in an aged flash chip. A higher RBER indicates a page nearing the end of its lifetime [10]. When the cells in the recycled flash chip have only two V_{th} states, the RBER is only 0.6% on average. In contrast, three V_{th} states in a cell increase the RBER to 0.9% on average. Four V_{th} -state-cells further enlarge the RBER of the same page to 1.4% on average. These data suggest that a full-fledged FRAC design can fully explore the trade-off between cell endurance and chip capacity.

ESE Results. We evaluated ESE using a simplified prototype of the energy source predictor. Our prototype predicts the average value of renewable energy generated by wind in the California grid every 30 minutes, utilizing a 2-layer LSTM network with forget, input, and output gates. Historical wind energy generation data from the California grid served as ground truth, sourced from California ISO [50], while wind data from Apple Weather [51] provided input. Training utilized 70% of the data, with 10% for validation and the remaining 20% for testing. Figure 7 illustrates the average wind energy generation prediction. While our prototype's predictions capture the general trend of the ground truth data, the 30-minute prediction interval results in larger fluctuations compared to the ground truth, suggesting the need for shorter intervals (e.g., $5 \sim 15$ minutes). These findings imply that a comprehensive ESE could accurately evaluate operational and embodied energy consumption in data centers.

IV. CONCLUSION

Modern data centers face significant environmental sustainability challenges throughout their lifecycle. The project aims to tackle these challenges by developing novel technologies including a reconfigurable hardware accelerator architecture, a fraction NAND flash cell and its system integration, and an environmental sustainability estimator. These advancements lay a solid foundation for improving environmental sustainability in data centers, enabling both users and data centers to conduct data-intensive applications in a more sustainable and scalable manner. This not only enhances computing capabilities but also maximizes the societal benefits derived from computing.

REFERENCES

- [1] Y. Liu *et al.*, "Energy consumption and emission mitigation prediction based on data center traffic and pue for global data centers," *Global Energy Interconnection*, vol. 3, no. 3, pp. 272–282, 2020.
- [2] E. Masanet et al., "Recalibrating global data center energy-use estimates," Science, vol. 367, no. 6481, pp. 984–986, 2020.
- [3] B. Acun et al., "Carbon explorer: A holistic framework for designing carbon aware datacenters," in ASPLOS, pp. 118–132, 2023.
- [4] J. Switzer et al., "Junkyard computing: Repurposing discarded smartphones to minimize carbon," in ASPLOS, pp. 400—412, 2023.
- [5] H. Dogan et al., "Aging analysis for recycled fpga detection," in IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, pp. 171–176, 2014.
- [6] S. Nalley et al., "Annual energy outlook 2022," Energy Information Agency, p. 23, 2022.
- [7] R. Singh et al., "Yank: Enabling green data centers to pull the plug," in 10th USENIX Conference on Networked Systems Design and Implementation, (USA), pp. 143—156, USENIX Association, 2013.
- [8] Y. Chen et al., "Dadiannao: A machine-learning supercomputer," in *International Symposium on Microarchitecture*, 2014.
- [9] A. Shafiee et al., "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," in ACM/IEEE International Symposium on Computer Architecture, pp. 14—26, 2016.
- [10] L. M. Grupp et al., "The bleak future of nand flash memory," in USENIX Conference on File and Storage Technologies, 2012.
- [11] S. Boyd *et al.*, "Life-cycle assessment of nand flash memory," *IEEE Transactions on Semiconductor Manufacturing*, 2011.
- [12] Y. Isaka et al., "Systematic unsupervised recycled field-programmable gate array detection," *IEEE Transactions on Device and Materials Reliability*, vol. 22, no. 2, pp. 154–163, 2022.
- [13] Y. Y. Liauw et al., "Nonvolatile 3d-fpga with monolithically stacked rram-based configuration memory," in *IEEE International Solid-State Circuits Conference*, pp. 406–408, 2012.
- [14] Y. Chen et al., "3d-nonfar: Three-dimensional non-volatile fpga architecture using phase change memory," in ACM/IEEE International Symposium on Low Power Electronics and Design, pp. 55—60, 2010.
- [15] K. Qiu et al., "Resirca: A resilient energy harvesting reram crossbarbased accelerator for intelligent embedded processors," in *IEEE International Symposium on High Performance Computer Architecture*, pp. 315–327, 2020.
- [16] Y. Cai et al., "Error patterns in mlc nand flash memory: Measurement, characterization, and analysis," in *Design, Automation & Test in Europe Conference & Exhibition*, pp. 521–526, 2012.
- [17] T. Nakamura et al., "Adaptive artificial neural network-coupled ldpc ecc as universal solution for 3-d and 2-d, charge-trap and floating-gate nand flash memories," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 3, pp. 745–754, 2019.
- [18] T. Tanzawa *et al.*, "A compact on-chip ecc for low cost flash memories," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 5, pp. 662–669, 1997.
- [19] X. Jimenez et al., "Wear unleveling: Improving NAND flash lifetime by balancing page endurance," in USENIX Conference on File and Storage Technologies, pp. 47–59, 2014.
- [20] Y. Cai et al., "Program interference in mlc nand flash memory: Characterization, modeling, and mitigation," in *IEEE International Conference on Computer Design*, pp. 123–130, 2013.
- [21] F. Chen et al., "Parc: A processing-in-cam architecture for genomic long read pairwise alignment using reram," in IEEE Asia and South Pacific Design Automation Conference, pp. 175–180, 2020.
- [22] F. Chen et al., "Regan: A pipelined reram-based accelerator for generative adversarial networks," in IEEE Asia and South Pacific Design Automation Conference, pp. 178–183, 2018.
- [23] F. Chen et al., "Zara: A novel zero-free dataflow accelerator for generative adversarial networks in 3d reram," in *Design Automation Conference*, p. 133, 2019.
- [24] Y. Zha et al., "Reconfigurable in-memory computing with resistive memory crossbar," in *International Conference on Computer-Aided Design*, 2016.
- [25] Y. Zha et al., "Liquid silicon: A data-centric reconfigurable architecture enabled by rram technology," in ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, 2018.
- [26] A. Haj-Ali et al., "Efficient algorithms for in-memory fixed point multiplication using magic," in IEEE International Symposium on Circuits and Systems, 2018.

- [27] A. Haj-Ali et al., "Imaging: In-memory algorithms for image processing," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 65, no. 12, pp. 4258–4271, 2018.
- [28] H. Nejatollahi et al., "Cryptopim: In-memory acceleration for lattice-based cryptographic hardware," in Design Automation Conference, 2020.
- [29] S. Deng et al., "A comprehensive model for ferroelectric fet capturing the key behaviors: Scalability, variation, stochasticity, and accumulation," in IEEE symposium on VLSI technology, pp. 1–2, 2020.
- [30] F. Chen, "Puffin: An efficient dnn training accelerator for direct feed-back alignment in fefet," in *International Symposium on Low Power Electronics and Design*, pp. 1–6, 2021.
- [31] C. Wang et al., "Extending the lifetime of nand flash memory by salvaging bad blocks," in Design, Automation & Test in Europe Conference & Exhibition, pp. 260–263, 2012.
- [32] J.-W. Hsieh et al., "Implementation strategy for downgraded flash-memory storage devices," ACM Transaction Embedded Computing System, vol. 12, mar 2013.
- [33] H.-Y. Lin et al., "Revive bad flash-memory pages by hlc scheme," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 5, pp. 860–873, 2019.
- [34] H. Yang et al., "Reliability issues and models of sub-90nm nand flash memory cells," in *International Conference on Solid-State and Integrated Circuit Technology*, pp. 760–762, 2006.
- [35] V. Mohan et al., "How i learned to stop worrying and love flash endurance," in USENIX Conference on Hot Topics in Storage and File Systems, p. 3, 2010.
- [36] D.-H. Kim et al., "A 1tb 4b/cell nand flash memory with tprog=2ms, tr=110µs and 1.2gb/s high-speed io rate," in *IEEE International Solid-State Circuits Conference*, pp. 218–220, 2020.
- [37] B. Gu et al., "Dynamic file cache optimization for hybrid ssds with high-density and low-cost flash memory," in *IEEE 39th International Conference on Computer Design*, pp. 170–173, 2021.
- [38] X. Jimenez *et al.*, "Phoenix: Reviving mlc blocks as slc to extend nand flash devices lifetime," in *Design, Automation & Test in Europe Conference & Exhibition*, pp. 226–229, 2013.
- [39] K.-D. Suh et al., "A 3.3 v 32 mb nand flash memory with incremental step pulse programming scheme," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 11, pp. 1149–1156, 1995.
- [40] Z. Wang et al., "Partitioning streaming parallelism for multi-cores: a machine learning based approach," in *IEEE International Conference on Parallel Architectures and Compilation Techniques*, pp. 307–318, 2010.
- [41] Y. Wen *et al.*, "Smart multi-task scheduling for opencl programs on cpu/gpu heterogeneous platforms," in *IEEE International conference on high performance computing*, pp. 1–10, 2014.
- [42] A. Svyatkovskiy et al., "Intellicode compose: Code generation using transformer," in ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2020.
- [43] Z. Feng et al., "CodeBERT: A pre-trained model for programming and natural languages," in Findings of the Association for Computational Linguistics: EMNLP 2020, 2020.
- [44] C. Zhang et al., "Optimizing fpga-based accelerator design for deep convolutional neural networks," in ACM/SIGDA international symposium on field-programmable gate arrays, 2015.
- [45] Y.-H. Chen et al., "Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks," in ACM/IEEE International Symposium on Computer Architecture, 2016.
- [46] F. Wang et al., "Reram-based processing-in-memory architecture for blockchain platforms," in IEEE Asia and South Pacific Design Automation Conference, 2019.
- [47] R. Agrawal *et al.*, "Security and privacy of blockchain-based single-bit cache memory architecture for iot systems," *IEEE Access*, 2022.
- [48] B. Acun et al., "Carbon explorer: A holistic framework for designing carbon aware datacenters," in ACM International Conference on Architectural Support for Programming Languages and Operating Systems, 2023.
- [49] eeNews Europe, "Fpga-based nvme data storage solution." https://www.eenewseurope.com/en/fpga-based-nvme-data-storage-solution/, 2019.
- [50] C. I. S. Operator, "California iso open access same-time information system." http://oasis.caiso.com/mrioasis/logon.do, 2021.
- [51] D. Sky, "Dark sky and apple weather." https://darksky.net/apple-weather, 2021.