2025 IEEE International Symposium on High Performance Computer Architecture (HPCA)

CORDOBA: Carbon-Efficient Optimization Framework for Computing Systems

Mariam Elgamal∗†, Doug Carmean‡, Elnaz Ansari‡, Okay Zed‡, Ramesh Peri‡, Srilatha Manne‡, Udit Gupta‡, Gu-Yeon Wei†, David Brooks†, Gage Hills†, and Carole-Jean Wu‡
Harvard University†, Meta‡
mariamelgamal@g.harvard.edu, ghills@g.harvard.edu, carolejeanwu@meta.com

***Abstract*—The world's push toward an environmentally sustainable society is highly dependent on the semiconductor industry. Despite existing carbon *modeling* efforts to quantify carbon footprint of computing systems, *optimizing* carbon footprint in large design spaces—while also considering trade-offs in power, performance, and area—is especially challenging. To address this need, we present CORDOBA, a carbon-aware optimization framework that optimizes *carbon efficiency*. We quantify carbon efficiency using the *total Carbon Delay Product* metric (tCDP): the product of total carbon and application execution time. We justify why tCDP is an effective metric for quantifying carbon efficiency. We use CORDOBA to explore the large design space for carbon-efficient specialized hardware, and identify distinct carbon-efficient optimal designs across operational use (eliminating up to 98% of the design space) despite uncertainty in carbon footprint parameters. We quantify opportunities to improve tCDP for real system case studies: (a) optimizing hardware provisioning from 8 to 4 cores in real system CPUs improves tCDP by 1.25×; and (b) leveraging advanced three-dimensional (3D) integration techniques (3D stacking of separately-fabricated logic and memory chips) improves tCDP by 6.9× versus conventional systems.**

I. Introduction

The carbon footprint of the Information and Communication Technology (ICT) sector in 2020 already accounted for 2.1-3.9% of global carbon emissions (quantified in units of carbon dioxide equivalent: CO_2e) [1], [16]. It is expected to grow due to an ever-increasing demand for computing hardware. For example, between 2010 and 2018, the global datacenter compute instances increased by 550% [32]. With the explosive growth of Artificial Intelligence (AI), the design of next-generation computing infrastructures requires more compute, memory, and bandwidth [53].

Computer architects and designers have an opportunity to improve computing's carbon footprint, by exploring carbon-aware design choices across process technologies, logic/memory devices, architectures, integration and packaging, and more. For decades, designers have conventionally targeted *energy-efficient* computing systems, where energy efficiency can be quantified by metrics such as energy-delay product (EDP) [12], [19]. In this paper, we explore techniques for designing *carbon-efficient* computing systems.

For quantifying the total carbon footprint of a computing system (tC, in units of CO_2e), over its entire lifetime, it is essential to consider both *operational carbon*: emissions from

∗ Part of this work was done during internship at FAIR, Meta.
‡ Authors have contributed to this work while at Meta.

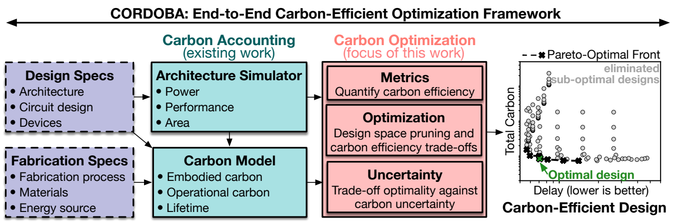

Fig. 1. Summary of our CORDOBA framework for optimizing carbon efficiency of computing systems. We distinguish between *carbon accounting* efforts and *carbon optimization* efforts in carbon-aware design. Carbon accounting focuses on quantifying carbon emissions due to hardware manufacturing and use. Carbon optimization – the focus of this work – leverages carbon accounting, in conjunction with optimization techniques, to determine optimal trade-offs in power, performance, area, and total carbon footprint.

day-to-day operational use, and *embodied carbon*: emissions due to hardware manufacturing [22], [27], [50]. While operational carbon is directly proportional to a system's energy consumption during operation, quantifying embodied carbon requires knowledge of parameters such as material sourcing, energy used in fabrication, and other factors that we describe in detail in this paper [7], [9], [22], [39].

Despite the importance of improving carbon efficiency, reducing carbon footprint in large design spaces—while also considering trade-offs in power, performance, and area—is especially challenging. First, carbon footprint quantification is imprecise, due to both **(a)** *lack of transparency*: designers may not have access to detailed carbon emissions data from manufacturing; and **(b)** *inherent quantification uncertainties*: carbon emissions depend on parameters that can vary significantly, such as variations in the carbon intensity (CI, in units of CO_2e per kWh) of renewable/non-renewable energy sources, which changes over time. Second, design guidelines for carbon-aware optimization remain nascent. Hardware system design communities are still learning which parameters are the most important for improving carbon footprint, and how these improvements affect energy efficiency, performance, and area. Fortunately, useful tools for modeling tC of integrated circuits (ICs), memory, and solid state drives are starting to become available [22], [27]. To further improve carbon-aware design, especially in the presence of inherent quantification uncertainties, researchers can benefit from optimization techniques that handle these uncertainties directly, instead of waiting for carbon accounting accuracy to improve.

We distinguish between *carbon accounting* and *carbon optimization* in carbon-aware design. Carbon accounting focuses on quantifying carbon emissions due to hardware manufacturing and use. Such efforts involve companies who are best positioned to report and minimize the environmental impact of their supply chains (e.g., imec's SSTS [25]). On the other hand, carbon optimization – the focus of this work (see Figure 1) – aims to develop methodologies, techniques, and metrics to *optimize* carbon efficiency, by combining existing carbon quantification efforts with mathematical optimization techniques. This distinction between carbon accounting and carbon optimization is analogous to the complementary roles of process design kits (PDKs) and electronic design automation (EDA) tools in IC design. PDKs enable accurate modeling of electronic device performance (such as transistors) in computing systems. Their role is analogous to carbon accounting and quantification. EDA tools leverage PDKs to optimize the overall performance of computing systems. Their role is analogous to carbon optimization. Both are essential, and can be developed in parallel.

In this paper, we present CORDOBA – a carbon efficiency optimization framework for computer system design. CORDOBA focuses on *carbon optimization*, while leveraging carbon accounting tools to understand trade-offs in power, performance, area, and total carbon footprint. Since carbon accounting might never be fully accurate, CORDOBA includes techniques for optimizing carbon efficiency in the presence of uncertainty in carbon accounting. Our key contributions are:

① Justification that tCDP is an effective metric for quantifying carbon efficiency of computing systems (§III).

② CORDOBA, an end-to-end cross-stack carbon-efficient optimization framework. CORDOBA builds on existing carbon accounting efforts (e.g. ACT [22], GreenChip [27]) and enables carbon-efficient optimization (§IV-A) despite uncertainty in quantifying tC (§IV-B).

③ Techniques to optimize tCDP of specialized hardware accelerators, considering trade-offs in power, performance, area, and total carbon. Our results show that we can eliminate 98% of accelerators in the design space as being sub-optimal for any value of operational lifetime (§VI-B). We show how to select carbon-efficient designs even when there is uncertainty in embodied and operational carbon quantification (§VI-C).

④ Case studies for improving tCDP of real computing systems. We show: (a) hardware over-provisioning is a key parameter for carbon-efficient design; for a multi-core CPU in production Virtual Reality (VR) headsets, optimizing number of cores (from 8 to 4) improves tCDP by 1.25× (§VI-D). (b) Leveraging 3D stacking of separately-fabricated logic and memory chips improves tCDP by 1.1× to 6.9× (§VI-E).

II. Related Work

Existing efforts have characterized the rising environmental impact of various computing systems [10], [16], [18], [21], [26], [35], [36], [39]. Designing hardware for emerging applications now requires environmental sustainability to be a core consideration [24], [37]. Toward tackling computing's carbon footprint challenge, a variety of carbon modeling tools and methodologies have been proposed to quantify computing systems' carbon footprint [7], [22], [27], [38]. These models enable embodied and operational carbon accounting and consider sustainability as a first-order design metric. Across the proposed solutions, sustainability-aware carbon optimization results in distinct designs compared to optimizing for power, performance, and area alone. For example, one study shows that targeting energy efficiency alone may result in large amounts of dark silicon, which diminishes the holistic sustainability of computing systems [9].

Prior work, GreenChip [27], presents a predictive tool for quantifying embodied and operational carbon footprint of systems. It integrates with architecture simulators, and enables designers to evaluate trade-offs in the environmental impact of competing system architectures, depending on their hardware usage and utilization, using indifference point analysis. Another architectural carbon modeling tool, ACT [22], leverages fabrication data and industry data to predict carbon footprint of systems beyond the 28 nm technology node. To overcome uncertainties in carbon data, FOCAL [13], proposes a first-order model that leverages proxies for embodied and operational carbon to provide early-stage insights for building environmentally-sustainable processors. Given the complex dependence of carbon footprint on parameters across the computing stack, designers will likely require multiple of these tools in tandem to enable effective carbon-aware optimization in high-dimensional design spaces.

Furthermore, a variety of metrics have been proposed to quantify carbon efficiency [14], [22], [50]. However, additional efforts are required to determine which metrics should be used for which scenarios. This is the focus of Section III.

III. Metrics for Carbon-Aware Optimization

The "total Carbon Delay Product" (tCDP) of a computing system is the product of its total carbon footprint (over its lifetime), and the time to execute a desired computing task (delay). In this section, we justify why tCDP is an effective metric to target for designing carbon-efficient computing systems. We also describe the pitfalls of targeting alternative metrics, including total carbon alone (not considering execution time), and Computational Carbon Intensity (CCI), where CCI is the carbon emissions per computing task (in units of CO_2e per computing task) [50]. CCI of a computing system is its total carbon footprint divided by the number of times it executes a target computing task in its lifetime ($\text{CCI} = \text{tC}/\text{N}_{\text{task}}$). In this section, we consider the computing task to be inference, though our analysis applies to other computing tasks as well.

A. Foundations in Energy-Efficient Design: the EDP Metric

There are parallels between energy-efficient design and carbon-efficient design. We digress momentarily to note that there are multiple interpretations of the phrase "energy efficiency", and the physical units used to quantify it. In some cases, energy efficiency is quantified solely in terms of the energy required to execute a computing task, no matter how

TABLE I

IC "A" ENABLES 1,000 INFERENCES PER SECOND (INF/S) WITH LOWEST POWER CONSUMPTION, DESPITE HAVING THE WORST PERFORMANCE. IC "D" ENABLES THE HIGHEST INFERENCE THROUGHPUT FOR A FIXED ENERGY BUDGET, SINCE IT HAS THE BEST (LOWEST) EDP.

		Candidate IC					
		"A"	**"B"**	**"C"**	**"D"**	**"E"**	**"F"**
[1]: given	clock frequency (GHz)	0.02	0.20	0.40	0.80	1.6	3.2
[2]: given	energy per cycle (nJ)	1.9	2	2.5	4	10	50
[3]: given	clock cycles per inf	100 million					
[4] = [1]/[3]	inf throughput (inf/s)	0.2	2	4	8	16	32
[5] = 1000/[4]	# ICs in parallel to meet 1000 inf/s	5000	500	250	125	62.5	31.25
[6] = [1]·[2]	power of each IC (W)	0.038	0.4	1	3.2	16	160
[7] = [5]·[6]	overall power of all ICs in parallel (W)	190	200	250	400	1000	5000
[8] = [2]·[3]	energy per inference (J/inf)	0.19	0.2	0.25	0.4	1	5
[9] = 9.5/[8]	# ICs in parallel given E budget: 9.5 J	50	47.5	38	23.75	9.5	1.9
[10] = [9]·[4]	inference throughput for all ICs in parallel (inf/s)	10	95	152	190	152	60.8
[11] = [8]/[4]	**EDP** (J/Hz)	0.950	0.100	0.063	0.050	0.063	0.156

much execution time is required. The units of energy efficiency are then *Joules per task*. As an example, consider two ICs "A" and "B" (Table I). IC "A" requires $\sim$5% less energy than IC "B", but is 10$\times$ slower. Under this definition, IC "A" is considered more energy-efficient than IC "B", despite its 10$\times$ longer execution time. Alternatively, energy efficiency is often quantified by the product of energy and execution time (EDP) in units of *Joules per Hz* [19], i.e., the energy required to enable task execution at a high rate. Designs are penalized for having long execution time. As such, IC "B" is more energy-efficient than IC "A", since it enables 10$\times$ faster execution time with only $\sim$5% more energy.

Yet another metric for quantifying energy efficiency is $E \cdot D^2$ in units of Joules per Hertz^2 (or $J \cdot s^2$). ED^2 has historically been effective in cases where the desired figure of merit is independent of the supply voltage (V_{DD}) [29]. One such scenario is evaluating designs used for Dynamic Voltage and Frequency Scaling (DVFS), in which it is convenient to have a metric whose value does not change when the system is operating in the low V_{DD} + low f_{CLK} mode, or in the high V_{DD} + high f_{CLK} mode (f_{CLK} is clock frequency). It can be shown that ED^2 is V_{DD}-independent under the antiquated assumptions that MOSFET on-current follows the ideal square law MOSFET model [42], [45], that MOSFET threshold voltage (V_T) is 0 V, and that energy is proportional only to $C_{load} \cdot {V_{DD}}^2$ (C_{load} is the capacitive load of digital logic gates, as in Dennard Scaling [12]) with no contributions from leakage power. However, these assumptions no longer hold for today's short-channel MOSFETs. Thus, ED^2 is no longer useful as a V_{DD}-independent metric to quantify energy efficiency.

Contrasting definitions of energy efficiency each have their place, and there are certainly application scenarios in which each is a suitable metric. In this paper, we quantify energy efficiency in Joules per Hz, which we justify in this section. Correspondingly, we quantify carbon efficiency in units of CO_2e per Hz and describe why it is an effective metric.

For energy-efficient design, energy is the resource we are consuming to improve computing performance. For carbon-efficient design, total carbon is the resource we are "consuming" (CO_2e emissions) to improve computing performance. Metrics that designers often target for energy-aware design are energy consumption (E_{task}: the energy required to execute a computing task), and EDP: the product of energy and execution time [19]. Since energy and carbon are both *resources*, E_{task} and CCI are corresponding *metrics* for energy-aware design versus carbon-aware design (where $E = N_{task} \cdot E_{task}$ and $tC = N_{task} \cdot CCI$). Similarly, EDP and tCDP are corresponding metrics, with EDP quantified in Joules per Hz, and tCDP quantified in CO_2e per Hz. Given these parallels, we first summarize the consequences of optimizing E_{task} versus optimizing EDP for energy-aware computing. We then build upon this discussion to describe the consequences of optimizing CCI versus optimizing tCDP for carbon-aware computing.

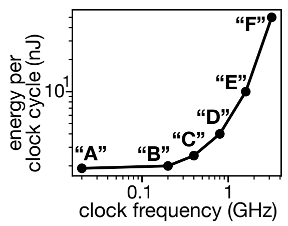

Fig. 2. Trade-offs in energy versus clock frequency for six candidate ICs ("A" through "F") targeting energy-efficient computing.

For energy-aware computing, a well-known pitfall of optimizing solely for E_{task} is that the resulting "optimal" design tends to run slowly (long execution time), which is undesirable. As a simple, yet illustrative example, consider the six different designs of ICs in Figure 2 (the same ICs as in Table I), each with different average energy per clock cycle (y-axis) and clock frequency (x-axis). The scenario is as follows. A designer is deciding which IC to use to design a parallel computing system that meets an overall application constraint: performing 1,000 inferences per second while minimizing total power consumption. For the sake of a concrete example, assume that each inference requires 100 million clock cycles, and that multiple ICs can be used to perform inferences in parallel to increase the overall throughput (inferences per second). Table I (above the bold line) shows the number of inferences per second enabled by each IC, and the resulting number of ICs needed to meet the throughput requirement.

While IC "A" optimizes the target objective, its execution time is by far the slowest (it has the slowest clock frequency). Notably, the optimization dictates that leveraging 5,000 instances of IC "A" is preferable to leveraging 500 instances of IC "B" (to compensate for the slow execution time of IC "A"), just to reduce power by 5%. A designer would likely intervene with this optimization result, and instead choose a design with higher performance, with relatively small cost in energy consumption (e.g., IC "B", "C", or "D"). Of course, manual intervention based on intuition is undesirable, especially in large design spaces. Here, it can be avoided by introducing a measure of performance into the objective function, i.e., targeting a different metric. Thus, while E_{task} may be a useful metric for providing general guidance, optimizing for it blindly leads to poor design decisions.

Optimizing for EDP avoids this pitfall. It enables designers to automatically select values for underlying design parameters that have competing impacts on energy consumption and delay. As classic examples, decreasing circuit supply voltage (V_{DD}), increasing transistor threshold voltage (V_T), and decreasing

TABLE II

IC "E" ENABLES THE HIGHEST INFERENCE THROUGHPUT FOR A FIXED CARBON BUDGET, SINCE IT HAS THE BEST (LOWEST) tCDP. IC "A" HAS THE LOWEST TOTAL CARBON (AND CCI) BUT RUNS VERY SLOWLY.

		Candidate IC					
		"A"	**"B"**	**"C"**	**"D"**	**"E"**	**"F"**
[1]: given	clock frequency (GHz)	0.020	0.200	0.400	0.800	1.6	3.2
[2]: given	energy per cycle (nJ)	1.9	2	2.5	4	10	50
[3]: given	clock cycles per inf	100 million					
[4] = [3]/[1]	time per inf (s)	5	0.5	0.25	0.125	0.0625	0.03125
[5]: given	CI_{use} (gCO_2e/kWh)	380					
[6]: given	$CI_{embodied}$ per IC (gCO_2e)	3000					
[7]: given	lifetime in seconds (s)	1.05×10^7					
[C1]: given constraint	$t_{service}$ (s)	0.100					
[C2]: given constraint	E_{budget} in $t_{service}$ (J)	9.5					
[C3] = [C2]/3.6×10^6	E_{budget} in $t_{service}$ (kWh)	2.639×10^{-6}					
[C4] = [5]·[C3]	tC_{budget} in $t_{service}$ (gCO_2e)	1.003×10^{-3}					
[10] = [9]/[C1]	# infs per lifetime	1.05×10^8					
[11] = [2]·[3]	E per inf (J/inf)	0.19	0.2	0.25	0.4	1	5
[12] = [11]/3.6×10^6	E per inf ($\times10^{-8}$ kWh)	5.28	5.56	6.94	11.1	27.8	139
[13] = [5]·[12]	$CCI_{operation}$($\times10^{-5}$ gCO_2e/inf)	2.01	2.11	2.64	4.22	1.06	5.28
[14] = [6]/[10]	$CCI_{embodied}$(gCO_2e/inf)	2.85×10^{-5}					
[15] = [13]+[14]	CCI ($\times10^{-5}$ gCO_2e/inf)	4.86	4.96	5.49	7.08	13.4	55.6
[16] = [C4]/[15]	# ICs in parallel given tC_{budget}	21	21	19	15	8	2
[17] = [16]/[4]	throughput per $t_{service}$	**4.1**	**40.4**	**73.0**	**113.4**	**119.7**	**57.7**
[18] = [10]·[13]+[6]	tC	5108	5219	5774	7438	14096	58480
[19] = [18]·[4]	tCDP	25541.2	2609.6	1443.5	929.8	881.0	1827.5

transistor widths all tend to improve energy consumption at the cost of degrading delay [12]. Thus, optimizing purely for energy consumption typically results in low V_{DD}, high V_T, and minimum width transistors, at the cost of undesirably slow delay. Instead, optimizing for EDP automatically selects (without manual intervention) values for V_{DD}, V_T, and transistor widths that correspond to optimal trade-offs in energy and delay.

While EDP has somewhat unintuitive units of *Joule seconds* (or more intuitively: Joules per Hertz), optimizing for EDP corresponds mathematically to solving practical real-world scenarios. Consider again the same six IC designs in Figure 2. The updated scenario is as follows. Given a fixed energy budget (e.g., 9.5 J), choose the IC that maximizes throughput (inferences per second). Table I (bottom) shows IC "D" has the best (lowest) EDP, which corresponds mathematically to it being the optimal choice for this scenario. I.e., the relative EDP of IC "D" versus other ICs exactly quantifies its relative improvement in inference throughput.

B. From Energy Efficiency to Carbon Efficiency: EDP to tCDP

We follow a similar approach to assess the effectiveness of CCI and tCDP for carbon-aware design, but with one key distinction: effective metrics for carbon-efficient design should inform designers how trade-offs in energy and delay impact embodied carbon as well. For the same six ICs, consider a parallel scenario as in §III-A, but with CO_2e as the resource instead of energy. Convert the fixed energy budget of 9.5 J to a corresponding fixed carbon budget of 1.003×10^{-3} gCO_2e (for $CI_{use} = 380$ gCO_2e/kWh of energy consumption).

There is also an embodied carbon associated with fabricating each IC, which penalizes computing systems that instantiate more ICs. To determine this amortization factor, we define a "service time interval" (e.g., servicing requests to perform inference come in every 100 ms), where the fixed carbon budget (1.003×10^{-3} gCO_2e) is allocated at each time interval to maximize the inference throughput and calculate hardware lifetime. Table II shows that design "E" is the best choice for this scenario: its inference throughput is the highest in each service period for the fixed carbon budget. If instead we optimize tC (instead of tCDP), we would choose IC "A",

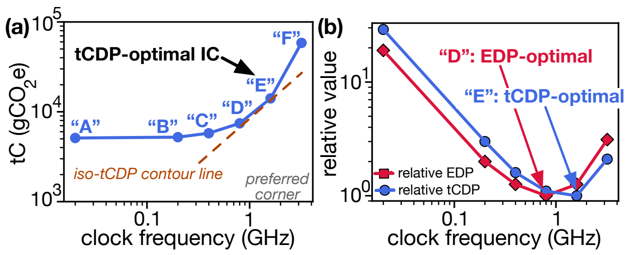

Fig. 3. Optimizing tCDP. (a) tC versus clock frequency. (b) tCDP-optimal versus EDP-optimal ICs.

the slowest design, just as in §III-A when optimizing E_{task} instead of EDP. Table II also shows exactly how to compute the inference throughput for each IC (following the equations in the left-most column). The key take-away is that relative inference throughput enabled by each IC is precisely quantified by its relative tCDP, i.e., *throughput* $\propto$ $tCDP^{-1}$. This is why multiplying throughput and tCDP is a constant (the product of row 17 and row 19 in Table II). Figure 3 shows the relative clock frequency, EDP, and tCDP of each IC. Compared to the EDP-optimal design (IC "D"), the tCDP-optimal design (IC "E") is less-energy efficient (higher EDP), but incurs less embodied carbon.

Analogous to EDP (which has units of Joules per Hz), tCDP has units of CO_2e per Hz. Whereas optimizing EDP balances energy and delay (energy efficiency), optimizing tCDP balances energy efficiency and embodied carbon (carbon efficiency). In §VII, we return to a broader discussion on the implications of key design knobs that affect trade-offs in energy efficiency and embodied carbon, such as technology node scaling and hardware lifetime.

C. Target Metrics for Carbon-Aware Optimization

In this subsection, we highlight the subtle distinction between metrics of carbon efficiency (tCDP), and metrics to target for carbon-aware optimization of computing systems. Specifically, we recognize that optimizing tCDP may not always be the objective function in all application scenarios. To illustrate this point, we again draw parallels to metrics of energy efficiency (EDP) and energy-aware optimization. Even though EDP is an effective measure of energy efficiency, it is not always the optimization target.

Common scenarios for energy-aware optimization include: (a) *minimizing energy given a performance constraint*. For example, a specific application has a maximum latency constraint that eliminates any of the ICs "A" through "F" with a clock frequency less than 250 MHz. Thus, ICs "A" and "B" are eliminated. Even though IC "D" is the EDP-optimal design, it is not selected since its clock frequency far exceeds the application performance constraint. Instead, IC "C" is chosen since it meets the performance constraint with lower energy, even though its energy *efficiency* is worse (i.e., it requires higher energy per Hz). (b) *Minimizing energy consumed to execute a given task, regardless of execution time*. For example, for a performance-agnostic wearable device (e.g., a heart-rate monitor), where the primary objective is to maximize

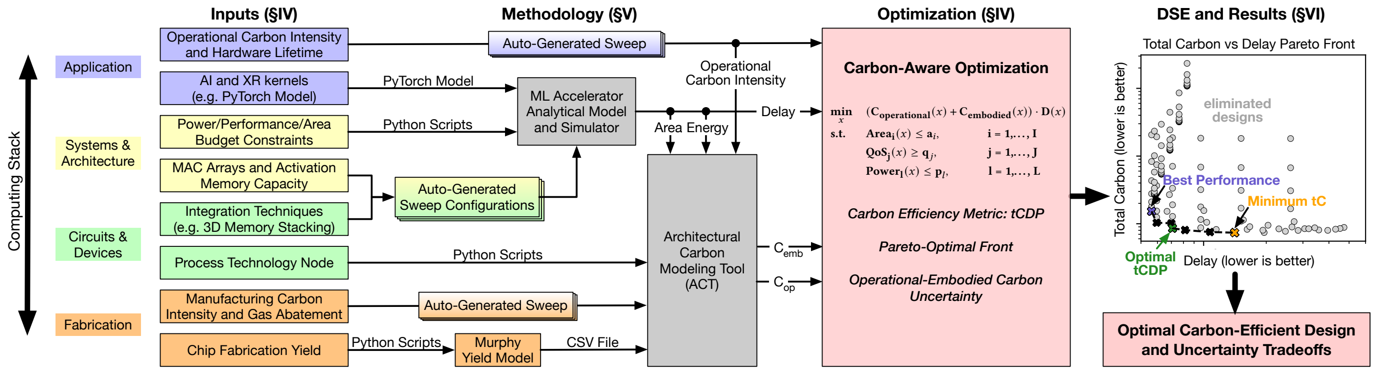
Inputs (§IV)
Methodology (§V)
Optimization (§IV)
DSE and Results (§VI)
Computing Stack
Application
Systems & Architecture
Circuits & Devices
Fabrication
Operational Carbon Intensity and Hardware Lifetime
AI and XR kernels (e.g. PyTorch Model)
Power/Performance/Area Budget Constraints
MAC Arrays and Activation Memory Capacity
Integration Techniques (e.g. 3D Memory Stacking)
Process Technology Node
Manufacturing Carbon Intensity and Gas Abatement
Chip Fabrication Yield
Auto-Generated Sweep
PyTorch Model
Python Scripts
ML Accelerator Analytical Model and Simulator
Auto-Generated Sweep Configurations
Python Scripts
Auto-Generated Sweep
Python Scripts
Murphy Yield Model
CSV File
Operational Carbon Intensity
Delay
Area Energy
Architectural Carbon Modeling Tool (ACT)
C_{emb}
C_{op}
Carbon-Aware Optimization
min (C_operational(x) + C_embodied(x)) · D(x)
s.t. Area_i(x) ≤ a_i, i = 1,…, I
QoS_j(x) ≥ q_j, j = 1,…, J
Power_l(x) ≤ p_l, l = 1,…, L
Carbon Efficiency Metric: tCDP
Pareto-Optimal Front
Operational-Embodied Carbon Uncertainty
Total Carbon vs Delay Pareto Front
Total Carbon (lower is better)
Delay (lower is better)
eliminated designs
Best Performance
Minimum tC
Optimal tCDP
Optimal Carbon-Efficient Design and Uncertainty Tradeoffs

Fig. 4. CORDOBA, our design optimization framework for carbon efficiency across multiple layers of the computing stack. DSE is "Design Space Exploration".

battery life (minimize energy disregarding speed). Application requirements are satisfied whether the IC requires 1 ms or 100 ms to complete a computing task (e.g., computing heart rate from sensor data). Importantly, in both these cases, a designer is sacrificing (degrading) energy efficiency based on the application requirement. In scenario (a), EDP is degraded to meet a performance constraint with lower energy consumption. In scenario (b), EDP is degraded to improve overall energy consumption. The subtle distinction that we reiterate, is that even though EDP quantifies energy efficiency, it is not always the optimization target.

Similarly, even though tCDP is an effective metric to quantify carbon efficiency, there are application scenarios where a designer may be forced to degrade tCDP to meet an application target. Our framework in §IV-A is designed to account for these scenarios. In any case, a key takeaway is that target metrics for carbon-aware design should be derived precisely from application scenarios. The choice of target metrics should not be driven by preconceived notions of how much to value carbon versus delay, i.e., deciding beforehand whether to optimize CCI, tCDP, or tCD^{2}P. For example, a designer might be under the impression that they value delay slightly more than carbon, and as a result, choose to optimize tCD^{2}P instead of tCDP. A scenario that would potentially motivate the use of tCD^{2}P is analogous to the use of ED^{2}P for DVFS-based designs, in which a designer may value a V$_{DD}$-independent figure of merit. However, tCD^{2}P being V$_{DD}$-independent corresponds to MOSFETs following the ideal square law, energy impacted only by $C_{load} \cdot V_{DD}^2$, and tC impacted only by operational carbon, i.e., embodied carbon being 0. Since this is no longer a realistic scenario, tCD^{2}P is not an effective metric to target for V$_{DD}$-independent carbon-aware optimization of DVFS systems, even if a designer might think that they value delay more than carbon. In contrast, we showed in §III-B that optimizing tCDP corresponds mathematically to optimizing real-world scenarios. The next section describes how to optimize tCDP at a system level.

IV. Carbon-Aware Optimization Framework

Here, we detail the mathematical formulation and design parameters of CORDOBA's carbon-efficient design optimization (§IV-A), and show how designers can optimize for carbon efficiency despite uncertainty in total carbon by using a Lagrange multiplier (details in §IV-B).

Our framework takes input parameters across multiple layers of the computing stack, including materials (used in IC fabrication), process technology node, die area, integration techniques, and application workloads, as shown in Figure 4. Fabrication-level inputs are based on detailed foundry characterization data [18], circuit- and system-level parameters are based on hardware configurations set by the designer, and application data are based on software profiling (refer to §V). CORDOBA's inputs are fed into an architecture simulator, into an architectural carbon modeling tool, and into carbon-aware optimization. This enables carbon-aware design space exploration while accounting for uncertainty in operational and embodied carbon data. The outputs of CORDOBA are the Pareto-optimal designs of total carbon versus delay, including the carbon-efficient tCDP optimal design and uncertainty trade-offs (results are in §VI-B and §VI-C).

A. Optimizing Carbon Efficiency

We formulate the design optimization of carbon-efficient systems considering total carbon as shown in equation IV.1. x is the vector of parameters defining the computing system (see Table III), including area, process technology node, energy source, and fabrication facility specifications. Parameters a_i, q_j, and p_l are area, quality of service (QoS), and power optimization constraints, respectively.

$$\begin{array}{lll} \underset{x}{\text{minimize}} & (\mathrm{C}_{\text{operational}}(x)+\mathrm{C}_{\text{embodied}}(x))\cdot \mathrm{D}(x) & \\ \text{subject to} & \mathrm{Area}_i(x)\le a_i, & i=1,\dots,I \\ & \mathrm{QoS}_j(x)\ge q_j, & j=1,\dots,J \\ & \mathrm{Power}_l(x)\le p_l, & l=1,\dots,L \end{array} \tag{IV.1}$$

We define a task T as a set of kernels K required to run a target application. Each task can comprise a single kernel, or multiple kernels running in parallel, depending on the number of kernel calls per task ($N_{T,K}$). For example, an Extended Reality (XR) gaming *task* can include eye-tracking, motion-tracking, and UnityEngine gaming *kernels*. A zero value of $N_{T,K}$ indicates that a kernel K is not part of task T. Equations IV.2-IV.6 allow designers to optimize for multiple kernels and tasks while meeting design constraints. Note that,

TABLE III
PARAMETERS IN OUR CARBON-AWARE OPTIMIZATION FRAMEWORK, CORDOBA.

Parameter	Description	Units	Example Range (§VI-D)
K	Kernels or workloads software (e.g., a program written in Python or C++)	from software profiling	IOT Tracking, UnityEngine, etc.
N	Number of kernel calls per task	1 or more	multiple calls per kernel
T	Task is a set of workloads or an application defined by number of kernel calls	from software profiling	M-1
D	Task execution time	seconds	40 s
x	Hardware target system	from hardware design	VR headset CPU
E	Operational energy consumption	J per task	332 J per task
P_{leak}	Hardware leakage power	W	-
P_{dyn}	Hardware dynamic power	W	-
P_{total}	Total power = P_{leak} + P_{dyn}	W	8.3 W
HW resources	Number of CPU cores, GPU, on-chip SRAM, and DRAM in system	count	4-8 CPU cores
CI_{use}	Use-phase carbon intensity	g CO_2e per kWh	380 g CO_2e/kWh
CI_{fab}	Carbon intensity of the energy source used by chip fabrication facility	g CO_2e per kWh	820 g CO_2e/kWh
EPA	Energy Per die Area consumed by the fabrication facility during manufacturing	kWh per cm^2	2.15 kWh/cm^2
MPA	Materials Per die Area is carbon footprint of procured materials used in fabrication	g CO_2e per mm^2	500 g CO_2e/cm^2
GPA	Gases Per die Area are direct gases emitted at fabrication facility during manufacturing	g CO_2e per cm^2	300 g CO_2e/cm^2
Y	Yield (e.g., estimated using yield models such as Murphy Yield [34])	0-1	0.98
A	Die area of components in system x	cm^2	2.25 cm^2
$C_{operational}$	Operational carbon	g CO_2e	3.154 g CO_2e per hour of use
$C_{embodied,\, x_m}$	Embodied carbon per component m (e.g. m can be a CPU core), in system x	g CO_2e per component	895.89 g CO_2e per gold core
t_{life}	Overall hardware lifetime	1-10 years	5 years
D_{off}	Time when the system is not in use throughout the system's lifetime	hours	22 hours per day for 5 years
Operational time	Active execution time of the system, i.e. when system is consuming energy	hours	2 hours per day for 5 years
$C_{embodied}$	Embodied carbon of a system dependent on operational time = $t_{life} - D_{off}$	g CO_2e per system	5375.33 g CO_2e

1 is a vector where every element is equal to 1 (e.g., $\mathbf{1}^\mathsf{T}\mathbf{E}$ is the sum of all elements in vector **E**). Table III describes each parameter.

$$\mathbf{D} = \begin{bmatrix} D_{T_1} \\ : \\ D_{T_t} \end{bmatrix} = \begin{bmatrix} N_{T_1,K_1} & .. & N_{T_1,K_k} \\ : & : & : \\ N_{T_t,K_1} & .. & N_{T_t,K_k} \end{bmatrix} \begin{bmatrix} D_{K_1} \\ : \\ D_{K_k} \end{bmatrix} \qquad \text{(IV.2)}$$

$$C_{embodied} = \frac{\mathbf{1}^\mathsf{T}\mathbf{D}}{t_{life} - D_{off}} \cdot \begin{bmatrix} C_{embodied,\, x_1} \\ : \\ C_{embodied,\, x_m} \end{bmatrix}^\mathsf{T} \begin{bmatrix} 0 \\ : \\ 1 \end{bmatrix} \qquad \text{(IV.3)}$$

$$\mathbf{E} = \begin{bmatrix} E_{T_1} \\ : \\ E_{T_t} \end{bmatrix} = \begin{bmatrix} N_{T_1,K_1} & .. & N_{T_1,K_k} \\ : & : & : \\ N_{T_t,K_1} & .. & N_{T_t,K_k} \end{bmatrix} \begin{bmatrix} P_{dyn,K_1} D_{K_1} \\ : \\ P_{dyn,K_k} D_{K_k} \end{bmatrix} + P_{leak} \begin{bmatrix} D_{T_1} \\ : \\ D_{T_t} \end{bmatrix} \qquad \text{(IV.4)}$$

$$C_{embodied,\, x_m} = (CI_{fab} \cdot EPA + MPA + GPA) \cdot \frac{A}{Y} \qquad \text{(IV.5)}$$

$$C_{operational} = CI_{use} \cdot \mathbf{1}^\mathsf{T}\mathbf{E} \qquad \text{(IV.6)}$$

Task delay. (D_T) is the product of $N_{T,K}$ and kernel execution time (D_K) (equation IV.2). We compute total task delay by taking the sum of all elements in the task delay vector **D**. Designers can also compute **D** using alternative time-based performance metrics, e.g. reciprocal of SPEC scores [47] or reciprocal of frames-per-second (i.e. $\frac{1}{FPS}$). Note that, delay should correspond to the overall execution time of the entire system, e.g., if the performance of a workload is limited by the delays of multiple hardware.

Total life cycle carbon. $tC = C_{operational} + C_{embodied}$.

Embodied carbon. Embodied carbon footprint of an IC (equation IV.5) depends on the fabrication facility and process technology node [22], [39] (among other dependencies in Table III). To compute tC, we multiply the vector $C_{embodied,\, x}$ (equation IV.3) with a vector whose elements are either 1 or 0; "1" indicates a component is included as part of a computing system (and "0" otherwise), e.g., there is a 1/0 element for each CPU core, GPU, DRAM, and on-chip SRAM that can be included in system x. This formulation enables hardware provisioning as a design parameter (§VI-D), where designers can compare different system configurations of their hardware to optimize tCDP.

Operational time. We amortize embodied carbon over *operational time*, which is not necessarily the same as the system's total lifetime in years (equation IV.3). Operational time is the execution time of the system including light background processes and idle time, i.e. whenever the system is consuming energy (including energy due to off-state leakage power), versus total lifetime which accounts for use time whether the system is consuming energy or not. Operational time ensures that embodied carbon is correctly accounted for when the system is not in active use.

Operational carbon. Energy consumption depends on $N_{T,K}$, dynamic and leakage power, and delay (equation IV.4); these power and delay parameters are extracted using electronic design automation (EDA) tools. Note that, idle time translates to additional energy consumption (thus, operational carbon) due to off-state leakage power. $C_{operational}$ (equation IV.6) is the total energy (sum of elements in the task energy vector **E**) multiplied by the "use-phase carbon intensity" (CI_{use}). CI_{use} is dependent on the energy source, e.g., renewable versus non-renewable sources, and the power grid.

B. *Optimizing Carbon Efficiency Despite Uncertain tC*

Optimizing tCDP can be especially challenging when there is uncertainty in quantifying total carbon footprint. Practical challenges include: *transparency* (designers not having full access to carbon footprint data from manufacturing) and *varying energy sources* (with varying carbon footprint) over a system's lifetime. For example, CI_{use} (Table III) may change dramatically from year-to-year (as renewable energy sources become more prevalent), or depending on the time of day (e.g., depending on the availability of renewable energy sources such as solar). Thus, it is essential to develop techniques for designers to improve carbon efficiency—*even without precise quantification of total carbon footprint.*

Here, we demonstrate that even when $\mathrm{CI_{use}}$ is unknown or changing over time, a designer can still make informed design decisions about optimizing carbon efficiency. Specifically, given a set of hardware targets: $X = \{x_1, x_2, ..., x_n\}$, we show how to identify hardware targets that are not tCDP-optimal for *any* value of $\mathrm{CI_{use}}$ over time. Such hardware targets can thus be *eliminated* from the set of candidate hardware targets, *even when CI_{use} is unknown.*

To justify this claim, let $\mathrm{CI_{use}}(t)$ be the value of $\mathrm{CI_{use}}$ versus time over a hardware target's operational lifetime. Similarly, let $\mathrm{P}(t)$ be the operational power consumption versus time. We assume that $\mathrm{P}(t)$ is fixed and known for a given hardware target (e.g., it can be accurately predicted using mature EDA tools); a short discussion of the benefits and limitations of this assumption is at the end of this section. Using these definitions, equation IV.6 becomes:

$$\mathrm{C_{operational}} = \int_0^{t_{\text{life}}} \mathrm{CI_{use}}(t)\mathrm{P}(t)dt \tag{IV.7}$$

Additionally, the objective function in equation 2.1 becomes:

$$\mathrm{C_{embodied}} \cdot \mathrm{D} + \left(\int_0^{t_{\text{life}}} \mathrm{CI_{use}}(t)\mathrm{P}(t)dt\right) \mathrm{D} \tag{IV.8}$$

However, if $\mathrm{CI_{use}}(t)$ is unknown, then a designer cannot evaluate (or optimize) this objective function. Instead, we leverage mathematical techniques to eliminate designs that cannot be tCDP-optimal, even when we cannot find the tCDP-optimal design. Using a Lagrange multiplier is a common technique used in ill-posed optimization problems when the relative importance (weight) of two objective functions is unknown [8]. It allows us to recast the objective function from equation IV.1 as:

$$\mathrm{C_{embodied}} \cdot \mathrm{D} + \beta \cdot \mathrm{E} \cdot \mathrm{D}, \tag{IV.9}$$

where E is the known total operational energy consumption over the entire lifetime, and β is the Lagrange multiplier. We argue that there exists *some value of* β in the range $[0, \infty)$, such that the objective functions are *exactly equal* (equations IV.1 and IV.9). Call this value β'. Both $\mathrm{CI_{use}}(t)$ and E are always non-negative (i.e., computation always *consumes* energy, which always *increases* carbon emissions), and thus, β' must be non-negative as well.

Next, we can optimize the objective in equation IV.9 for all values of β in $[0, \infty)$, which must include β'. For each value of β, the optimization can result in a different hardware target being tCDP-optimal. We call this set of hardware targets X^*: the set of designs that are tCDP-optimal for some value of β. Note that, if we plot $\mathrm{E} \cdot \mathrm{D}$ versus $\mathrm{C_{embodied}} \cdot \mathrm{D}$ for all designs in X, then the designs in X^* define the *Pareto-optimal curve* for $\mathrm{E} \cdot \mathrm{D}$ versus $\mathrm{C_{embodied}} \cdot \mathrm{D}$. In other words, for all hardware targets in X^*, there is no other hardware target in X that has better $\mathrm{E} \cdot \mathrm{D}$ *and* better $\mathrm{C_{embodied}} \cdot \mathrm{D}$ *simultaneously* (see Figure 12 for an example). *Importantly, X^* contains the tCDP-optimal design* (i.e., tCDP-optimal is on the Pareto-optimal curve of $\mathrm{E} \cdot \mathrm{D}$ versus $\mathrm{C_{embodied}} \cdot \mathrm{D}$). Thus, even if $\mathrm{CI_{use}}(t)$ is unknown, designers can eliminate all hardware targets in X that are not in X^*.

As an example, Figure 12 (§VI-E) shows $\mathrm{E} \cdot \mathrm{D}$ versus $\mathrm{C_{embodied}} \cdot \mathrm{D}$ for a set of seven different hardware targets (details in §VI-E); only two of these seven hardware targets are on the Pareto-optimal curve (i.e., in X^*). Thus, even without knowing $\mathrm{CI_{use}}(t)$, five of the seven hardware targets are ensured not to be tCDP-optimal. Since precise quantification of total carbon (including both operational carbon and embodied carbon) may not always be fully accessible to designers, practical design techniques (such as this) are essential to guide designers toward carbon-efficient design decisions. While we relied on various assumptions to formulate this result (e.g., our limitation that $\mathrm{P}(t)$ is fixed and known), we encourage designers to continue developing optimization techniques that are robust to uncertainty in quantifying carbon emissions. For example, designers can further leverage Lagrange multipliers when parameters for *embodied carbon* are unknown at design time, such as carbon intensity of fabrication ($\mathrm{CI_{fab}}$). Identifying additional scenarios will aid in carbon-aware optimization that is more robust to uncertainties.

TABLE IV
SUMMARY OF KERNELS IN EACH TASK. THE "ALL KERNELS" TASK INCLUDES ALL FIFTEEN KERNELS.

XR (10 kernels)	3D-Agg; ET; JLP; HRN; UNet; E-FAN; DN; SR (256×256); SR (512×512); SR (1024×1024)
AI (10 kernels)	RN-18; RN-50; RN-152; GN; MN2; 3D-Agg; ET; UNet; JLP; HRN;
XR (5 kernels)	3D-Agg; HRN; DN; SR (512×512); SR (1024×1024)
AI (5 kernels)	RN-18; RN-50; RN-152; GN; MN2

V. EVALUATION METHODOLOGY

We evaluate CORDOBA's carbon-aware optimization on a variety of workloads and hardware platforms detailed below.

Workloads. We characterize five computing tasks comprising a variety of fifteen AI and XR kernels. The kernels include ResNets (RN-18, RN-50, RN-152) [23], GoogleNet (GN) [51], MobileNet-V2 (MN2) [43], eye tracking (ET) [4], depth estimation (3D-Agg, HRN) [30], [49], emotion detection (E-FAN) [52], hand tracking (JLP) [33], image denoising (UNet, DN) [40], [55] and super-resolution (SR) [5]. The XR and AI kernels have different activation memory requirements, and accordingly different memory accesses. Kernels that suffer from high activation memory requirements include depth estimation, image denoising and super-resolution. For instance, as super-resolution kernels scale up in resolution requirements, such as SR (1024×1024), their memory and bandwidth requirements grow beyond the typical LPDDR4 DRAM 16 GB/s peak bandwidth. Therefore, increasing the activation SRAM memory from 2 MB to 32 MB, decreases the bandwidth requirements by 89.6× within acceptable ranges [54].

In Table IV, we categorize the kernels into five tasks: "All kernels", "XR 10 kernels", "AI 10 kernels", "XR 5 kernels", and "AI 5 kernels". The "All kernels" task includes running both activation memory heavy and non-activation heavy tasks. The XR tasks are focused on optimizing accelerators

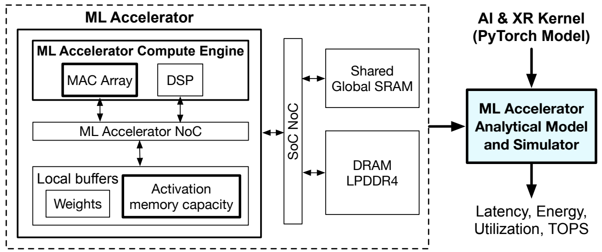
ML Accelerator
ML Accelerator Compute Engine
MAC Array
DSP
ML Accelerator NoC
Local buffers
Weights
Activation memory capacity
SoC NoC
Shared Global SRAM
DRAM LPDDR4
AI & XR Kernel (PyTorch Model)
ML Accelerator Analytical Model and Simulator
Latency, Energy, Utilization, TOPS

Fig. 5. HW accelerator simulator based on [48].

targeting high activation memory kernels, while AI tasks are focused on optimizing more general AI kernels that are less memory bound.

Performance & power simulation for HW design. Figure 5 shows the architecture accelerator simulator we use, which is similar to the one described in references [48] and [44]. To characterize each AI and XR kernel, a PyTorch neural network model is input into the accelerator simulator. The simulator outputs latency and energy consumption for the AI/XR kernel running on a user-specified accelerator architecture. We use equations IV.2-IV.6 to compute tCDP for a range of accelerator architectures by varying the number of Multiply-Accumulate units (MACs) and size of activation memory. Furthermore, we model 3D integration techniques (§VI-E) using a similar approach as reference [54], in which we assume conservative latency for a 3D design (similar to conventional designs), and leverage energy values reported from place-and-route using a 7 nm PDK [54].

Carbon data. We leverage an architectural carbon modeling tool, ACT, based on industry environmental reports and detailed fabrication facility characterization [22]. We update ACT to include the most recent fabrication facility characterization data [39] and incorporate additional models for die placement and yield, such as the Murphy yield model (as an example) for embodied carbon calculations [11], [34].

Carbon model and architecture simulator fidelity. In CORDOBA, carbon estimation accuracy is bounded by two major factors: (1) uncertainty in carbon footprint data, and (2) Machine Learning (ML) accelerator simulator accuracy. The first is a pressing challenge; unlike power and performance, carbon cannot be measured by end-users. Therefore, it is important to distinguish between carbon accounting versus optimization. We present methods to trade-off carbon efficiency optimality against uncertainty in §VI-C. The ML accelerator simulator is verified using cycle-accurate simulations and validated against board-level implementation on a Xilinx Field-Programmable Gate Array (FPGA) with a 2.98% performance error [56]. The time required to run CORDOBA's DSE is on the order of hours (end-to-end) to analyze 121 distinct accelerator configurations.

Production-level VR system HW. To profile production-level VR applications in real-time, we use Android Debug Bridge (adb) software to connect to a Meta Quest 2 SoC. The Qualcomm Snapdragon XR2 SoC is manufactured in a 7 nm process technology node and has an Octa-core CPU. Using Simpleperf [46], a native CPU profiling tool, we collect the

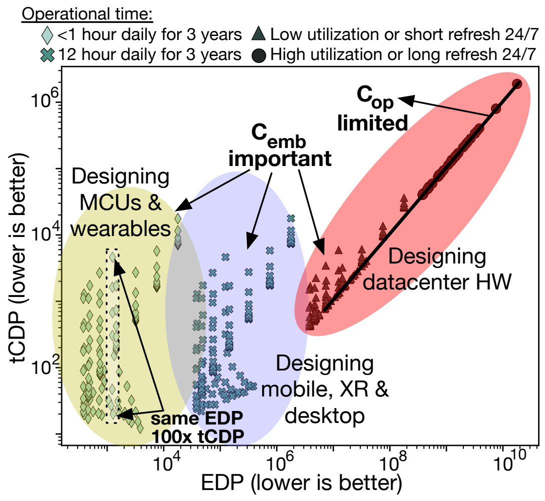

Fig. 6. Optimizing energy efficiency (quantified by EDP) and carbon efficiency (quantified by tCDP) lead to distinct optimal designs, since the relationship between embodied and operational carbon varies across computing domains. E.g., embodied carbon accounts for 95%, 72%, and 50% of total carbon for microcontrollers (MCUs) and wearables [3], mobile [2], and servers [21], respectively. Thus, carbon-efficient optimization must consider *total carbon*, based on parameters such as operational time subject to power, performance, and area constraints. In other words, the only case in which optimizing EDP corresponds to optimizing tCDP (i.e., either optimization selects the same design), is when the design is entirely dominated by operational carbon.

frame rate (FPS), and the number of instructions and clock cycles for each core configuration to compute Instructions Per Cycle (IPC) for each application. To profile the Thread-Level Parallelism (TLP) of the top 10 applications running on Meta Quest 2, we use Perfetto UI [20], an open-source system-wide profiling and application tracing software for Android.

VI. Design Space Exploration: Optimization Results and Case Studies

In this section, we first quantify the trade-offs between carbon efficiency and energy efficiency (§VI-A). Second, we present an in-depth analysis of our design space exploration of carbon-efficient hardware accelerators (§VI-B), across five AI and XR tasks and a continuum of operational time. Third, we analyse design trade-offs between the optimal carbon-efficient accelerators and discuss how to tackle uncertainty in carbon accounting at the optimization stage (§VI-C). Fourth, we conduct real system case studies using CORDOBA to quantify the dependence of tCDP on hardware provisioning in VR headsets (§VI-D), and on 3D integration (§VI-E).

A. The Need for Carbon-Aware Optimization

In Figure 6, we highlight example design spaces for wearables (yellow), mobile (blue) and datacenter (red) domains. Here, we demonstrate that targeting *energy efficiency* as an optimization metric (e.g., EDP) is insufficient for optimizing carbon efficiency. As shown in Figure 6, carbon efficiency (quantified by tCDP) and energy efficiency (quantified by EDP) are not perfectly correlated as depicted across all domains, including datacenters. For systems such as wearables

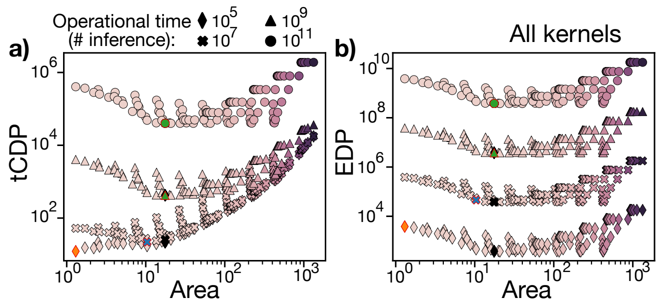

Fig. 7. tCDP versus die area (a) shows that optimal tCDP designs (red points) are not equivalent to designs with minimum area. EDP versus die area (b), shows the EDP-optimal designs (black points) do not change as we design for operational time; since EDP does not account for embodied carbon. Thus, neither EDP nor area are sufficient indicators for carbon efficiency.

and mobile, there is weak correlation between a system's energy efficiency and carbon efficiency. Also, the tCDP-optimal designs are not equivalent to the EDP-optimal designs. This is because the total carbon of the designs is dominated by embodied carbon, which is not accounted for in the EDP metric. Furthermore, Figure 6 shows cases in which two EDP-equivalent designs exhibit 100× difference in tCDP. Such results demonstrate that energy efficiency is an insufficient indicator for carbon efficiency.

When designing high performance and high power systems, such as datacenter systems with high utilization or long refresh cycles, tCDP is more strongly correlated with EDP (the relationship between tCDP and EDP approaches a straight line in Figure 6). This is because designs that consume substantial amounts of energy tend to be operational carbon dominant. This behavior is supported mathematically in equations IV.1 and IV.6; when $\mathrm{C}_{\text{operational}}(x) >> \mathrm{C}_{\text{embodied}}(x)$, then tCDP $\approx \mathrm{C}_{\text{operational}}(x) \cdot \mathrm{D}(x) \approx \mathrm{CI}_{\text{use}}(x) \cdot \mathbf{E}(x) \cdot \mathrm{D}(x) \approx \mathrm{CI}_{\text{use}}(x) \cdot$ EDP(x). Given that the rising prevalence of renewable energy and computing systems' carbon footprint is shifting toward becoming more embodied carbon dominant [21], we predict this correlation between EDP and tCDP is likely to decrease, even for high performance datacenter systems. Furthermore, it has been shown that datacenters can incur >50% embodied carbon footprint depending on utilization and refresh cycles [21], [27]. Accordingly, optimizing for carbon efficiency, using a metric such as tCDP, is of growing importance across application domains.

Figure 7(a) shows the relationship between tCDP and area, demonstrating that they are largely uncorrelated, especially with varying operational time. The tCDP-optimal designs (red points) are not necessarily equivalent to designs with minimum area. Naturally, larger die area incurs higher embodied carbon, and thus, designs with large die area tend to have higher tCDP, i.e., worse carbon efficiency. However, reducing die area by too much can also lead to worse delay and operational carbon, also degrading carbon efficiency. Figure 7(b) shows the relationship between EDP and area. Notably, the EDP-optimal design is unaffected by operational use, unlike the tCDP-optimal design, which accounts for both operational and embodied carbon across the system's lifetime. Again, these observations support the conclusion that neither EDP nor area alone is a sufficient indicator of a system's carbon efficiency. It is essential to account for operational time to determine total carbon and carbon efficiency.

B. *Optimizing tCDP Across Workloads and Operational Time*

Figure 8(a)-(e) presents the carbon efficiency results of the hardware design space exploration for five different tasks comprising a variety of AI and XR kernels (detailed in §V); All kernels, XR (10 kernels), AI (10 kernels), XR (5 kernels), and AI (5 kernels). The x-axis quantifies operational time in terms of number of inferences. The y-axis quantifies carbon efficiency as tCDP^{-1} (higher values correspond to better carbon efficiency). Using our framework, we tailor-design 121 hardware accelerators comprising various combinations of Multiply-Accumulate (MAC) arrays and on-chip SRAM capacity configurations.

Carbon efficiency versus operational time. A characteristic of plotting tCDP versus operational time, is being able to sweep all possible different ratios of embodied carbon and operational carbon to total carbon. At short operational times, embodied carbon is the dominant source of the system's total carbon. As operational time increases and the system does more work (i.e., more inference), operational carbon increases and dominates total carbon of the system.

The benefits of this formulation are two-fold. First, we are able to *narrow down the design space, eliminating a large number of designs that are not tCDP-optimal for any possible operational times of a task, over all possible ratios of embodied carbon to total carbon*. Therefore, *designers can identify that the tCDP-optimal design must be one of a small number of potential deigns, even when there is uncertainty in embodied or operational carbon footprint quantification*. For example, Figure 8(a) shows that only four accelerator designs can be tCDP-optimal depending on the operational time: **a1, a37, a38, and a48**. This allows us to: (i) eliminate 96.7% of the design space (as begin sub-optimal for any operational time) for "All kernels" and "AI 10 kernels" (Figure 8(a) and (c)); (ii) eliminate 98.3% of the design space for "XR 10 kernels" and "XR 5 kernels" (Figure 8(b) and (d)); and (iii) eliminate 97.5% of the design space for "AI 5 kernels" (Figure 8(e)). The second benefit, which we investigate in §VI-C, is that we can improve robustness to uncertainty in carbon accounting by trading off optimality in tCDP (i.e., we can degrade tCDP to improve robustness to uncertainty).

Optimal carbon-efficient accelerators differ architecturally depending on activation memory requirements and operational time. As an illustration, "XR 10 kernels" and "XR 5 kernels" have high activation memory requirements (Figure 8(b) and (d)), and thus, the tCDP-optimal accelerators **a48** and **a58** have higher activation memory (4-8 MB) compared to "AI 5 kernels" tCDP-optimal accelerators **a1, a12, and a23** with 1 MB SRAM (Figure 8(e)). In Figure 8(a) "All kernels", the tCDP-optimal accelerator **a1** (designed for short operational use) has relatively fewer MAC arrays and relatively smaller SRAM capacity to minimize the dominant embodied carbon.

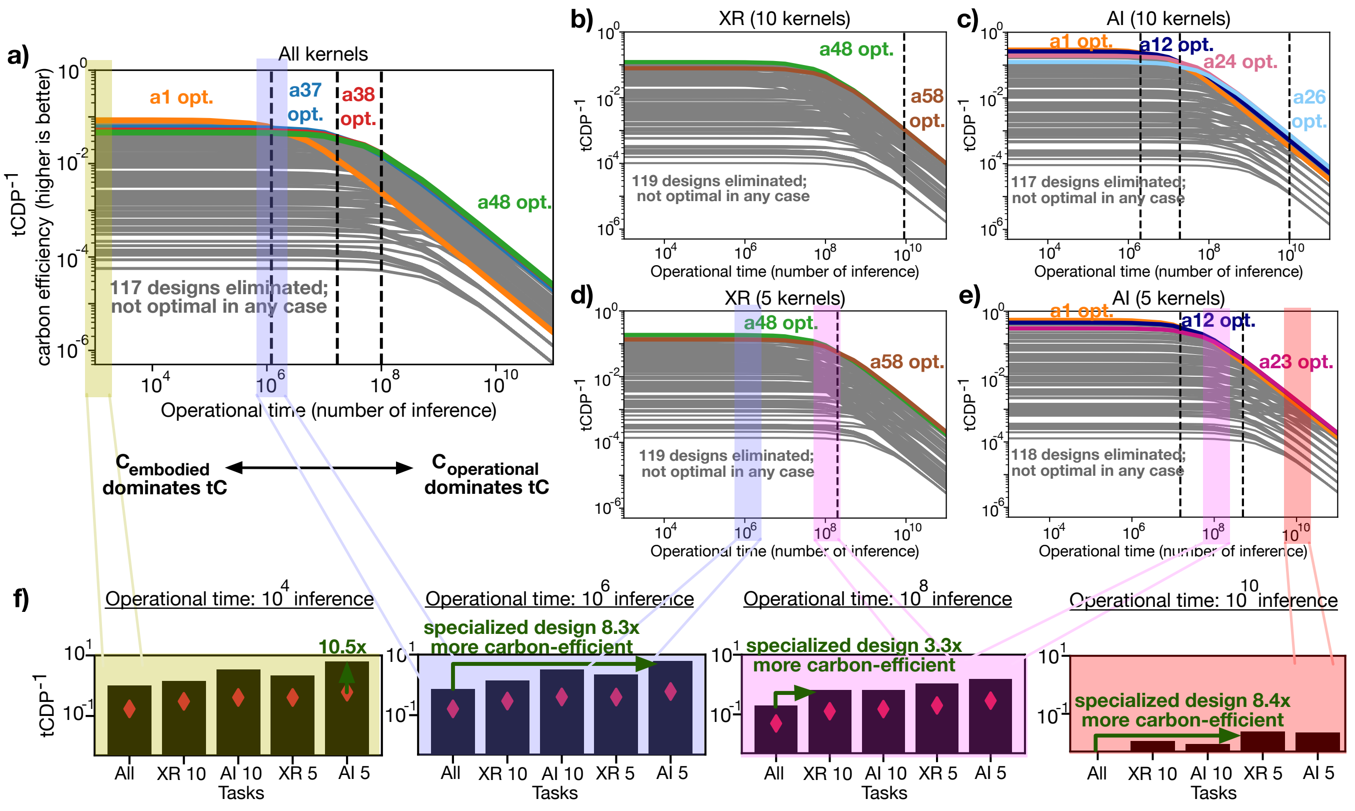

Fig. 8. Carbon efficiency trends of our design space exploration of 121 distinct MAC units+on-chip SRAM accelerator configurations. We account for carbon footprint uncertainties by designing for different tasks (All kernels, XR 10 kernels, AI 10 kernels, XR 5 kernels, AI 5 kernels) and across operational time measured in number of inference. By designing and optimizing for tCDP across operational time, we eliminate up to 98% of the design space. Note that, this analysis can also be adjusted to account for varying workloads over the system's lifetime.

As operational time increases, the tCDP-optimal accelerators **a38** and **a48** are more carbon-efficient, with 8-16 MAC units and 8-16 MB SRAM activation memory.

Carbon efficiency of general versus specialized designs. Figure 8(f) shows carbon efficiency trends of the five tasks at specified operational times, from 10^4 to 10^{10} inferences. The x-axis represents the five AI and XR tasks that our system hardware is tailor-designed for. The bars represent the tCDP of the optimal hardware accelerator configurations. At the same hardware operational time and utilization, when embodied carbon is the dominant source of total carbon, specializing system hardware for workloads of similar characteristics leads to hardware with better carbon efficiency. As an example, for an operational time of 10^6 inferences, optimizing for the specialized "AI 5 kernels" is up to 8.3× more carbon-efficient than optimizing for the more general "All kernels" task. Similarly, when operational carbon dominates at operational time of 10^{10} inferences, optimizing for the specialized "XR 5 kernels" is up to 8.4× more carbon-efficient than optimizing for the more general "All kernels" task. The takeaway is that *specialized designs are more carbon-efficient than general designs. This applies at all operational times, regardless of whether embodied carbon or operational carbon dominates total carbon*. This highlights that specializing hardware for workloads with similar characteristics can lead to more carbon-efficient systems.

Average versus optimal carbon efficiency. The opportunities for carbon efficiency improvement are significant. Red diamonds in Figure 8(f) indicate the average carbon efficiency of all designs at an operational time for each task. At 10^4 inferences, tCDP of the most carbon-efficient hardware accelerator configuration **a1** is 8× and 10.5× higher than the average tCDP of accelerators for "AI 10 kernels" and "AI 5 kernels", respectively. The minimum tCDP benefit between the optimal and the average carbon-efficient design across all tasks and operational times is 2.3× (corresponding to "AI 5 kernels" at 10^{10} inferences).

C. Trading off Carbon Efficiency Against Uncertainty

In addition to carbon footprint quantification, user behavior and system usage can be major sources of uncertainty for designers when trying to determine the workload specifications and system lifetime *a priori* (i.e., before the hardware is actually being used). We have already demonstrated that designers can narrow down the design space to a small number of designs across operational time, by eliminating most of the designs as being tCDP sub-optimal (§VI-B). To choose between the remaining designs for a specific task, designers can trade-off tCDP optimality and uncertainty in determining operational time, using information shown in Figure 9.

Figure 9 shows tCDP normalized to the tCDP-optimal design at each operational time, i.e., a value of 1.0 (on

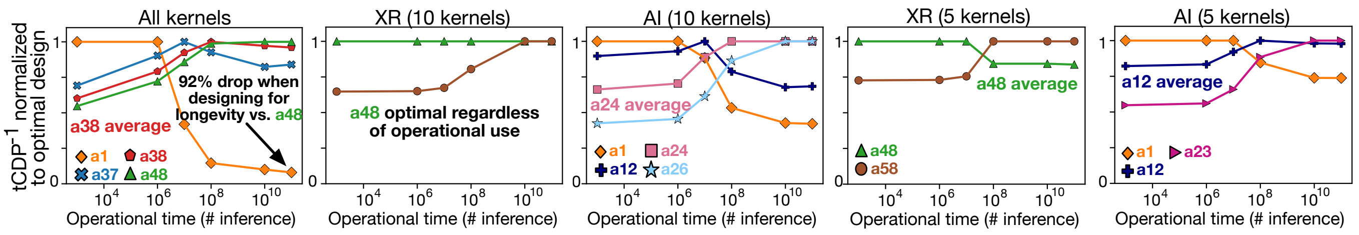

Fig. 9. Carbon efficiency (y-axis) normalized to the optimal design versus operational time (x-axis). To improve robustness against variations and uncertainty in estimating hardware usage, designers can trade-off tCDP optimality at a given operational time with another design that has a better average tCDP across a wide range of operational times.

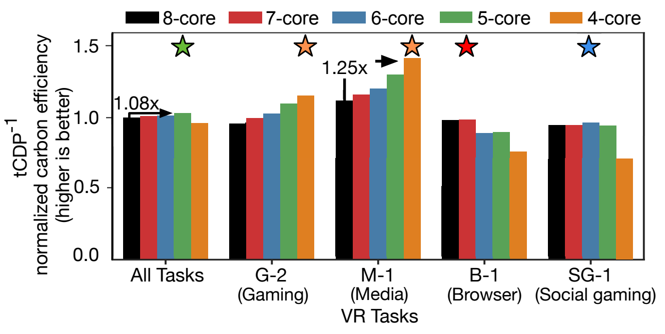

Fig. 10. Carbon efficiency of VR Tasks on Meta Quest 2 VR headset versus CPU core count (stars indicate optimal configuration). Optimal hardware provisioning improves tCDP by up to 1.25×.

the y-axis) represents optimal carbon efficiency. For "All kernels", **a1** is the tCDP-optimal design at low operational times, e.g., for frequently replaced consumer devices when embodied carbon dominates system's total carbon. However, in cases where hardware longevity is desired, operational carbon dominates total carbon, and **a1** is up to 12.5× worse than the tCDP-optimal accelerator (**a48**) at longer operational time (10^{11} inferences). To improve robustness against uncertainty in predicting user behavior or hardware usage, designers can trade-off tCDP optimality at a given operational time with another design. For example, a designer can instead choose the **a38** accelerator for "All kernels", which has better average tCDP across operational time.

The "XR 10 kernels" subplot in Figure 9 shows that the **a48** accelerator (16 MACs, 8 MB SRAM) is tCDP-optimal across all ratios of embodied carbon to total carbon. Therefore, it may not be worth manufacturing accelerator **a58** (32 MACs, 4 MB SRAM) even if the designer is optimizing for an operational carbon dominant use case, to improve robustness against uncertainties and variations as much as possible. Additionally, out of the total nine distinct tCDP-optimal accelerator designs identified in our design space exploration for all tasks and operational times, there are four accelerator designs—**a38** (8 MACs, 16 MB SRAM) for "All kernels", **a48** (16 MACs, 8 MB SRAM) for "XR 10 kernels" and "XR 5 kernels", **a23** (4 MACs, 1 MB SRAM) for "AI 10 kernels", and **a12** (2 MACs, 1 MB SRAM) for "AI 5 kernels". These accelerators have better average tCDP across variations in operational time. This type of analysis highlights the importance of investigating the trade-offs between tCDP-optimal designs across operational time, and the trade-offs of tCDP optimality against uncertainty.

TABLE V
VR SoC PARAMETERS BEFORE AND AFTER CARBON-EFFICIENT OPTIMIZATION FOR M-1 APPLICATION, WITH NUMBER OF CORES AND AREA HAVING HIGH IMPACT ON CARBON EFFICIENCY.

Parameter	Before Optimization	After Optimization	Improvement
P_{total}	8.3 W	8.3 W	-
E	332 J	332 J	-
A	2.25 cm^2	1.35 cm^2	1.67×
$C_{operational}$	3.15 g CO_2e per hour	3.15 g CO_2e per hour	-
CPU cores	4 gold + 4 silver cores	2 gold + 2 silver cores	reduced 4 cores
$C_{embodied}$	5375.33 g CO_2e	2687.67 g CO_2e	2×
C_{total}	12273 g CO_2e	9696 g CO_2e	1.27×
D	1.0 normalized FPS	0.98 normalized FPS	0.98×
EDP	1	1.02	0.98×
tCDP	1	0.8	1.25×

D. Hardware Over-Provisioning to Improve tCDP

In this section, we apply CORDOBA's carbon efficiency framework to explore emerging applications and technologies. Here, we quantify the impact of hardware over-provisioning on the carbon efficiency of a VR headset System-on-Chip (SoC). We identify hardware provisioning as a key design parameter for carbon-efficient computing. We analyze the top 100 tasks running on 8-core CPUs inside deployed production-level Meta Quest 2 VR headset devices. We group the tasks into four categories: general gaming (G), social gaming (SG), browser and virtual desktop (B), and media (M). The *top 10 tasks account for over 85% of overall computation time*, and we quantify the benefits of optimal hardware provisioning (refer §V) for four of those top-10 tasks (labeled G-2, M-1, B-1, and SG-1).

We quantify the degree of thread-level parallelism (TLP) to indicate the level of hardware over-provisioning on the 8-core CPU of Snapdragon XR2 SoC in Quest 2. TLP is computed as the number of cores activated concurrently divided by the total task execution time [6], [15], [17]. For the four VR tasks, TLP ranges from 3.52 to 4.15, indicating there are over three unused cores on average. Thus, we can remove unused cores to improve total carbon (lowering die area) without significant performance degradation, improving tCDP.

Figure 10 shows that reducing number of cores from 8 to 4 improves tCDP by 1.25× for the M-1 task (media category), with parameter values before and after optimization quantified in Table V. Note that, B-1 (browser) and SG-1 (social gaming) tasks suffer degraded tCDP for the 4-core configuration, due to relatively higher TLP and frame-rate requirements. However, even for the "All Tasks" category, reducing cores from 8 to 5 improves tCDP by 1.08×. Thus, designers should carefully optimize hardware provisioning (based on application require-

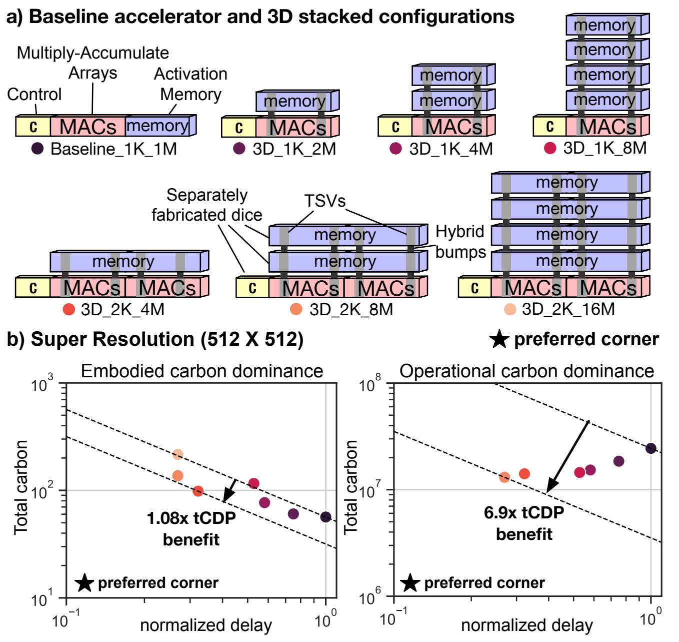

Fig. 11. (a) Baseline accelerator (derived from [48]) and six 3D stacked configurations [54]. 1K/2K is number of MACs (1K = 1,000), 2M/4M/8M/16M is SRAM capacity (in MegaBytes). (b) For SR (512×512) kernel, 3D stacking improves tCDP by 1.08× (left) to 6.9× (right) versus baseline

ments) to develop carbon-efficient systems.

For reference, equations VI.10-VI.12 illustrate how we optimize hardware provisioning using CORDOBA in §IV-A. "MT" is a Motion Tracking kernel, "A" is an Audio kernel, and the "4-c" subscript indicates computing tCDP for the 4-core configuration. We use similar equations to compute tCDP for 5-core, 6-core, 7-core, and 8-core configurations, which correspond to the results in Figure 10.

$$D_{\text{M-1, 4-c}} = \begin{bmatrix} N_{\text{M-1,MT}} & \cdot\cdot & N_{\text{M-1,A}} \end{bmatrix} \begin{bmatrix} D_{\text{MT, 4-c}} \\ : \\ D_{\text{A, 4-c}} \end{bmatrix} \quad \text{(VI.10)}$$

$$E_{\text{M-1, 4-c}} = \begin{bmatrix} N_{\text{M-1,MT}} & \cdot\cdot & N_{\text{M-1,A}} \end{bmatrix} \begin{bmatrix} P_{\text{dyn,MT}} \cdot D_{\text{MT, 4-c}} \\ : \\ P_{\text{dyn,A}} \cdot D_{\text{A, 4-c}} \end{bmatrix} + P_{\text{leak, 4-c}} \begin{bmatrix} D_{\text{M-1, 4-c}} \end{bmatrix} \quad \text{(VI.11)}$$

$$C_{\text{embodied, overall}} = \begin{bmatrix} C_{\text{embodied, silver core 1}} \\ C_{\text{embodied, silver core 2}} \\ C_{\text{embodied, silver core 3}} \\ C_{\text{embodied, silver core 4}} \\ C_{\text{embodied, gold core 1}} \\ C_{\text{embodied, gold core 2}} \\ C_{\text{embodied, gold core 3}} \\ C_{\text{embodied, prime gold core}} \end{bmatrix}^{\top} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad \text{(VI.12)}$$

E. Example tCDP Benefits of 3D Integration

We explore advanced packaging techniques, such as 3D stacking, and optimize tCDP despite uncertain carbon footprint data. State-of-the-art technologies are now using 3D integration techniques to increase connectivity between processor and memory, which can address *system-level bottlenecks* such as the "memory wall" (excessive time and energy overheads

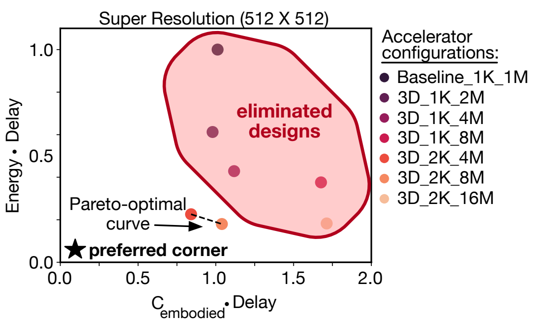

Fig. 12. E · D versus $C_{\text{embodied}} \cdot D$ for the various accelerator configurations running SR (512×512) (same data as in Figure 11(b)). The Pareto-optimal curve indicates the only configurations that can be tCDP-optimal for any possible value of $CI_{\text{use}}(t)$, as described in §IV-B.

for transferring data back-and-forth between processors and memory [41]). Here, we evaluate the carbon efficiency of *3D stacking*: connecting separately-fabricated dice in 3D (using hybrid bumps to connect through silicon vias (TSVs) of vertically-adjacent dice, as described in [54]). Figure 11(a) shows the baseline accelerator memory architecture compared to a variety of 3D-stacked configurations. The 3D-stacked configurations include various combinations of computing resources (1,000 MAC units or 2,000 MAC units), and activation memory (in MegaBytes: 2 MB, 4 MB, 8 MB, or 16 MB). The activation memory *per memory die* is 2 MB for configurations with 1,000 (1K) MACs, and 4 MB for configurations with 2,000 (2K) MACs.

To demonstrate the tCDP benefits of 3D stacking, we analyze the accelerators running a super-resolution kernel (SR), as an example. This SR kernel is used on XR systems to improve low-resolution image quality [5], and "inference" refers to the SR kernel running on a single image. We quantify tCDP in two cases: for an "embodied carbon dominant" case, and an "operational carbon dominant" case. Both cases run the same task (SR 512×512), but operate for different operational times. The embodied carbon dominant case has shorter operational time (fewer inferences) such that the embodied carbon accounts for 80% of total carbon (average across all seven configurations), and operational carbon accounts for the remaining 20%. In contrast, the operational carbon dominant has longer operational time, such that embodied carbon accounts for 8% of total carbon, and operational carbon accounts the remaining 92% (on average).

Figure 11(b) shows: for the embodied carbon dominant case (left side), configuration "3D_2K_4M" improves tCDP by 1.08× versus the baseline, and for the operational carbon dominant case (right side), configuration "3D_2K_8M" improves tCDP by 6.9× versus the baseline. In both cases, 3D stacking improves tCDP. When operational carbon exceeds embodied carbon, the energy efficiency benefits of 3D stacking (more computing resources, higher processor to memory bandwidth) provide significantly larger tCDP benefit. The tCDP benefit is relatively smaller for the embodied carbon case, which is dominated by the physical manufacturing of the compute and memory components; our analysis currently accounts for the

TABLE VI
DESIGN KNOBS TO TRADE-OFF ENERGY AND DELAY FOR ENERGY EFFICIENCY VERSUS DESIGN KNOBS TO TRADE-OFF ENERGY EFFICIENCY AND EMBODIED CARBON FOR TOTAL CARBON EFFICIENT DESIGNS.

	Design knob	Effect on E	Effect on D	Effect on C_{emb}
Trade-off energy and delay	$V_{DD}\downarrow$	↓✓	↑✗	negligible
	$V_T\uparrow$	↓✓	↑✗	negligible
	FET width ($\propto$ Area) ↓	↓✓	↑✗	↓✓
Trade-off energy efficiency and C_{emb}	Lifetime ↓	↓✓	↓✓	↑✗
	Tech. node ↓	↓✓	↓✓	↑✗

area overhead of including TSVs on each die, and can be extended to incorporate future packaging models that account for embodied carbon of physically connecting multiple dice.

Finally, Figure 12 illustrates the relationship between $E \cdot D$ versus $C_{embodied} \cdot D$ for the same data in Figure 11(b), where E corresponds to normalized total energy consumption, in this case of fixed-work analysis E is "Energy per inference" (which is the same for both the embodied carbon dominant and operational carbon dominant cases). As described in §IV-B, any configuration that is not on the Pareto-optimal curve cannot be tCDP-optimal for any values of $CI_{use}(t)$. Thus, even when $CI_{use}(t)$ is unknown or changing over time, five of the seven configurations can be eliminated: Baseline_1K_1M, 3D_1K_2M, 3D_1K_4M, 3D_1K_8M, and 3D_2K_16M. Instead, either of 3D_2K_4M and 3D_2K_8M will be tCDP-optimal depending on $CI_{use}(t)$; indeed, 3D_2K_4M is tCDP-optimal for the embodied carbon dominant case in Figure 11(b), and 3D_2K_8M is tCDP-optimal for the operational carbon dominant case. Note that, the change in *lifetime* (between the embodied carbon dominant and operational carbon dominant cases) has the same effect as a change in $CI_{use}(t)$ *for this particular analysis*—both have the effect of changing the scaling factor from E to $C_{operational}$ (as in equation IV.6). Thus, we can leverage the Lagrange multiplier technique, and demonstrate the ability to make informed design decisions even when there is uncertainty in quantifying tC.

VII. Discussion on Broader Implications of tCDP

In this section, we discuss the design knobs conventionally used to trade-off energy and delay for energy efficiency, and the additional design knobs needed to trade-off energy efficiency and embodied carbon for carbon efficiency. For energy-efficient computing, energy and delay are inherently linked through underlying design parameters, including V_{DD}, V_T, and transistor widths (described in §III). As shown in Table VI, adjusting each of these "design knobs" results in fundamental trade-offs in energy and delay. Thus, EDP has been an effective metric for driving energy-efficient computing, since optimizing EDP dictates how to choose precise values for V_{DD}, V_T, and transistor widths that optimally balance energy and delay.

There are also design knobs that improve *both* energy and delay. For example, advancing to the next technology node, i.e., transistor scaling. Even though continued scaling suffers from diminishing returns, and comes with higher economical cost, scaling has always improved energy efficiency (EDP).

For carbon-efficient computing, on the other hand, technology node is a design knob that results in *trade-offs* between energy efficiency and carbon footprint (Table VI). In other words, leveraging state-of-the-art technology nodes does not always result in the most carbon-efficient design, just as minimizing V_{DD} does not always result in the most energy-efficient design. This is because the embodied carbon footprint of wafer fabrication is increasing as fabrication processes are becoming more intensive, including more power-intensive lithography (Extreme Ultra-Violet versus Deep Ultra-Violet), increasing number of metal layers, and heterogeneous integration (e.g., complex systems including multiple chiplets on interposers [28]). Thus, in many cases the embodied carbon overhead of advancing to the next technology node outweighs the corresponding energy efficiency benefit [9], [18], [22].

Another similar design knob is hardware lifetime. Decreased hardware lifetime with frequent hardware upgrades improves energy efficiency because of better optimized systems for new workloads and/or higher energy efficiency of computer systems using more advanced semiconductor technology nodes. For carbon-efficient computing, however, hardware lifetime results in trade-offs between energy efficiency and carbon footprint [22], [31]. Despite the energy efficiency benefits of hardware refresh, it incurs higher embodied carbon due to more hardware manufacturing. tCDP is an effective metric for carbon-efficient design because it captures all these subtleties, by accounting for embodied carbon (part of total carbon), the ratio between operational and embodied carbon (total carbon across lifetime), and energy efficiency.

VIII. Conclusion

We present CORDOBA, a carbon-aware design framework to optimize parameters at multiple layers of the computing stack (across manufacturing, design, and operational use) for carbon-efficient computing system design. We justify tCDP as an effective metric for carbon efficiency and provide techniques for optimizing carbon-efficient systems in presence of carbon modeling uncertainties. We conduct case studies on real-world applications and systems and quantify carbon efficiency improvements for hardware provisioning and 3D integration, to guide designers in developing future carbon-efficient computing systems. CORDOBA lays key foundations for carbon-efficient optimization, which can be extended to include additional models (e.g. yield, cost, and more) as communities continue to adopt carbon-aware design.

Acknowledgements

We thank Abdulrahman Mahmoud, Bilge Acun, Ramya Raghavendra, Jordan Tse, Noah VanGorder, Lita Yang, and Edith Beigne for their valuable discussions and feedback. The authors at Harvard University acknowledge valuable support from the Salata Institute for Climate and Sustainability, NSF Expeditions in Computing (NSF EiC: 2326605), and NSF CAREER: 2443303.

References

[1] A. S. G. Andrae and T. Edler, "On global electricity usage of communication technology: Trends to 2030," *Challenges*, vol. 6, no. 1, pp. 117–157, 2015. [Online]. Available: https://www.mdpi.com/2078-1547/6/1/117

[2] Apple, "Product environmental report 16-inch macbook pro," 2023. [Online]. Available: https://www.apple.com/environment/pdf/products/notebooks/16-inch_MacBook_Pro_PER_Oct2023.pdf

[3] Apple, "Product environmental report apple watch series 9 — carbon neutral," 2023. [Online]. Available: https://www.apple.com/environment/pdf/products/watch/Carbon_Neutral_Apple_Watch_Series_9_PER_Sept2023.pdf

[4] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[5] G. Bhat, M. Danelljan, L. Van Gool, and R. Timofte, "Deep burst super-resolution," 2021. [Online]. Available: https://arxiv.org/abs/2101.10997

[6] G. Blake, R. G. Dreslinski, T. Mudge, and K. Flautner, "Evolution of thread-level parallelism in desktop applications," in *Proceedings of the 37th Annual International Symposium on Computer Architecture*, ser. ISCA '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 302–313. [Online]. Available: https://doi.org/10.1145/1815961.1816000

[7] S. B. Boyd, *Life-cycle assessment of semiconductors*. Springer Science & Business Media, 2011.

[8] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[9] E. Brunvand, D. Kline, and A. K. Jones, "Dark silicon considered harmful: A case for truly green computing," in *2018 Ninth International Green and Sustainable Computing Conference (IGSC)*, 2018, pp. 1–8.

[10] J. Chang, J. Meza, P. Ranganathan, A. Shah, R. Shih, and C. Bash, "Totally Green: Evaluating and designing servers for lifecycle environmental impact," *SIGPLAN Not.*, vol. 47, no. 4, p. 25–36, mar 2012. [Online]. Available: https://doi.org/10.1145/2248487.2150980

[11] D. de Vries, "Investigation of gross die per wafer formulas," *IEEE Transactions on Semiconductor Manufacturing*, vol. 18, no. 1, pp. 136–139, 2005.

[12] R. H. Dennard, F. H. Gaensslen, H.-N. Yu, V. L. Rideout, E. Bassous, and A. R. LeBlanc, "Design of ion-implanted MOSFET's with very small physical dimensions," *IEEE Journal of solid-state circuits*, vol. 9, no. 5, pp. 256–268, 1974.

[13] L. Eeckhout, "FOCAL: A first-order carbon model to assess processor sustainability," in *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, ser. ASPLOS '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 401–415. [Online]. Available: https://doi.org/10.1145/3620665.3640415

[14] M. Elgamal, D. Carmean, E. Ansari, O. Zed, R. Peri, S. Manne, U. Gupta, G.-Y. Wei, D. Brooks, G. Hills, and C.-J. Wu, "Design space exploration and optimization for carbon-efficient extended reality systems," 2023.

[15] K. Flautner, R. Uhlig, S. Reinhardt, and T. Mudge, "Thread-level parallelism and interactive performance of desktop applications," in *Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS IX. New York, NY, USA: Association for Computing Machinery, 2000, p. 129–138. [Online]. Available: https://doi.org/10.1145/378993.379233

[16] C. Freitag, M. Berners-Lee, K. Widdicks, B. Knowles, G. Blair, and A. Friday, "The climate impact of ICT: A review of estimates, trends and regulations," 2021. [Online]. Available: https://arxiv.org/abs/2102.02622

[17] C. Gao, A. Gutierrez, M. Rajan, R. G. Dreslinski, T. Mudge, and C.-J. Wu, "A study of mobile device utilization," in *2015 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2015, pp. 225–234.

[18] M. Garcia Bardon, P. Wuytens, L.- Ragnarsson, G. Mirabelli, D. Jang, G. Willems, A. Mallik, A. Spessot, J. Ryckaert, and B. Parvais, "DTCO including sustainability: Power-performance-area-cost-environmental score (PPACE) analysis for logic technologies," in *2020 IEEE International Electron Devices Meeting (IEDM)*, 2020, pp. 41.4.1–41.4.4.

[19] R. Gonzalez and M. Horowitz, "Energy dissipation in general purpose microprocessors," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 9, pp. 1277–1284, 1996.

[20] Google, "Perfetto," 2022. [Online]. Available: https://perfetto.dev/

[21] U. Gupta, Y. Kim, S. Lee, J. Tse, H. S. Lee, G. Wei, D. Brooks, and C. Wu, "Chasing Carbon: The elusive environmental footprint of computing," in *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. Los Alamitos, CA, USA: IEEE Computer Society, mar 2021, pp. 854–867. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/HPCA51647.2021.00076

[22] U. Gupta, M. Elgamal, G. Hills, G.-Y. Wei, H.-H. S. Lee, D. Brooks, and C.-J. Wu, "ACT: Designing sustainable computer systems with an architectural carbon modeling tool," in *Proceedings of the 49th Annual International Symposium on Computer Architecture*. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3470496.3527408

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[24] X. Hou, T. Xu, C. Li, C. Xu, J. Liu, Y. Hu, J. Zhao, J. Leng, K.-T. Cheng, and M. Guo, "A tale of two domains: Exploring efficient architecture design for truly autonomous things," in *2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA)*, 2024, pp. 167–181.

[25] imec, "The green transition of the ic industry," 2022. [Online]. Available: https://www.imec-int.com/sites/default/files/2022-07/Whitepaper_SSTS_FINAL.pdf

[26] R. Jain and J. Wullert, "Challenges: Environmental design for pervasive computing systems," in *Proceedings of the 8th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '02. New York, NY, USA: Association for Computing Machinery, 2002, p. 263–270. [Online]. Available: https://doi.org/10.1145/570645.570678

[27] D. Kline, N. Parshook, X. Ge, E. Brunvand, R. Melhem, P. K. Chrysanthis, and A. K. Jones, "GreenChip: A tool for evaluating holistic sustainability of modern computing systems," *Sustainable Computing: Informatics and Systems*, vol. 22, pp. 322–332, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2210537917300823

[28] J. H. Lau, "Recent advances and trends in advanced packaging," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 12, no. 2, pp. 228–252, 2022.

[29] Y. Li, D. Brooks, Z. Hu, K. Skadron, and P. Bose, "Understanding the energy efficiency of simultaneous multithreading," in *Proceedings of the 2004 International Symposium on Low Power Electronics and Design*, ser. ISLPED '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 44–49. [Online]. Available: https://doi.org/10.1145/1013235.1013251

[30] Z. Li, W. Ye, D. Wang, F. X. Creighton, R. H. Taylor, G. Venkatesh, and M. Unberath, "Temporally consistent online depth estimation in dynamic scenes," 2021. [Online]. Available: https://arxiv.org/abs/2111.09337

[31] S. Longofono, D. Kline, R. Melhem, and A. K. Jones, "Toward secure, reliable, and energy efficient phase-change main memory with MACE," in *2019 Tenth International Green and Sustainable Computing Conference (IGSC)*, 2019, pp. 1–8.

[32] E. Masanet, A. Shehabi, N. Lei, S. Smith, and J. Koomey, "Recalibrating global data center energy-use estimates," *Science*, vol. 367, no. 6481, pp. 984–986, 2020. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.aba3758

[33] F. Mueller, D. Mehta, O. Sotnychenko, S. Sridhar, D. Casas, and C. Theobalt, "Real-time hand tracking under occlusion from an egocentric RGB-d sensor," in *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, oct 2017. [Online]. Available: https://doi.org/10.1109%2Ficcv.2017.131

[34] B. Murphy, "Cost-size optima of monolithic integrated circuits," *Proceedings of the IEEE*, vol. 52, no. 12, pp. 1537–1545, 1964.

[35] S. Ollivier, S. Li, Y. Tang, S. Cahoon, R. Caginalp, C. Chaudhuri, P. Zhou, X. Tang, J. Hu, and A. K. Jones, "Sustainable ai processing at the edge," *IEEE Micro*, vol. 43, no. 1, pp. 19–28, 2023.

[36] T. Pirson, T. P. Delhaye, A. G. Pip, G. Le Brun, J.-P. Raskin, and D. Bol, "The environmental footprint of ic production: Review, analysis, and lessons from historical trends," *IEEE Transactions on Semiconductor Manufacturing*, vol. 36, no. 1, pp. 56–67, 2023.

[37] S. Prakash, M. Stewart, C. Banbury, M. Mazumder, P. Warden, B. Plancher, and V. J. Reddi, "Is TinyML sustainable?" *Commun.*

ACM, vol. 66, no. 11, p. 68–77, oct 2023. [Online]. Available: https://doi.org/10.1145/3608473

[38] E. Quisbert-Trujillo, T. Ernst, K. E. Samuel, E. Cor, and E. Monnier, "Lifecycle modeling for the eco design of the internet of things," *Procedia CIRP*, vol. 90, pp. 97–101, 2020, 27th CIRP Life Cycle Engineering Conference (LCE2020)Advancing Life Cycle Engineering : from technological eco-efficiency to technology that supports a world that meets the development goals and the absolute sustainability. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212827120302535

[39] L.- Ragnarsson, M. G. Bardon, P. Wuytens, G. Mirabelli, D. Jang, G. Willems, A. Mallik, A. Spessot, J. Ryckaert, and B. Parvais, "Environmental impact of cmos logic technologies," in *2022 6th IEEE Electron Devices Technology & Manufacturing Conference (EDTM)*, 2022, pp. 82–84.

[40] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015. [Online]. Available: https://arxiv.org/abs/1505.04597

[41] M. M. Sabry Aly, T. F. Wu, A. Bartolo, Y. H. Malviya, W. Hwang, G. Hills, I. Markov, M. Wootters, M. M. Shulaker, H.-S. Philip Wong, and S. Mitra, "The N3XT approach to energy-efficient abundant-data computing," *Proceedings of the IEEE*, vol. 107, no. 1, pp. 19–48, 2019.

[42] T. Sakurai and A. R. Newton, "Alpha-power law mosfet model and its applications to cmos inverter delay and other formulas," *IEEE Journal of solid-state circuits*, vol. 25, no. 2, pp. 584–594, 1990.

[43] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[44] Y. S. Shao, J. Clemons, R. Venkatesan, B. Zimmer, M. Fojtik, N. Jiang, B. Keller, A. Klinefelter, N. Pinckney, P. Raina, S. G. Tell, Y. Zhang, W. J. Dally, J. Emer, C. T. Gray, B. Khailany, and S. W. Keckler, "Simba: Scaling deep-learning inference with multi-chip-module-based architecture," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO '52. New York, NY, USA: Association for Computing Machinery, 2019, p. 14–27. [Online]. Available: https://doi.org/10.1145/3352460.3358302

[45] W. Shockley, "A unipolar" field-effect" transistor," *Proceedings of the IRE*, vol. 40, no. 11, pp. 1365–1376, 1952.

[46] A. G. Source, "Simpleperf," 2022. [Online]. Available: https://android.googlesource.com/platform/system/extras/+/master/simpleperf/doc/README.md

[47] Standard Performance Evaluation Corporation, "Spec cpu2017 results," 2023. [Online]. Available: http://www.spec.org/cpu2017/results/index.html

[48] H. E. Sumbul, T. F. Wu, Y. Li, S. S. Sarwar, W. Koven, E. Murphy-Trotzky, X. Cai, E. Ansari, D. H. Morris, H. Liu, D. Kim, E. Beigne, R. Labs, and Meta, "System-level design and integration of a prototype ar/vr hardware featuring a custom low-power DNN accelerator chip in 7nm technology for codec avatars," in *2022 IEEE Custom Integrated Circuits Conference (CICC)*, 2022, pp. 01–08.

[49] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5686–5696.

[50] J. Switzer, G. Marcano, R. Kastner, and P. Pannuto, "Junkyard computing: Repurposing discarded smartphones to minimize carbon," in *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, ser. ASPLOS 2023. New York, NY, USA: Association for Computing Machinery, 2023, p. 400–412. [Online]. Available: https://doi.org/10.1145/3575693.3575710

[51] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[52] A. Toisoul, J. Kossaifi, A. Bulat, G. Tzimiropoulos, and M. Pantic, "Estimation of continuous valence and arousal levels from faces in naturalistic conditions," in *Nature Machine Intelligence*, 2021.

[53] C.-J. Wu, R. Raghavendra, U. Gupta, B. Acun, N. Ardalani, K. Maeng, G. Chang, F. Aga, J. Huang, C. Bai, M. Gschwind, A. Gupta, M. Ott, A. Melnikov, S. Candido, D. Brooks, G. Chauhan, B. Lee, H.-H. Lee, B. Akyildiz, M. Balandat, J. Spisak, R. Jain, M. Rabbat, and K. Hazelwood, "Sustainable AI: Environmental implications, challenges and opportunities," in *Proceedings of Machine Learning and Systems*, vol. 4, 2022, pp. 795–813.

[54] L. Yang, R. M. Radway, Y.-H. Chen, T. F. Wu, H. Liu, E. Ansari, V. Chandra, S. Mitra, and E. Beigné, "Three-dimensional stacked neural network accelerator architectures for AR/VR applications," *IEEE Micro*, vol. 42, no. 6, pp. 116–124, 2022.

[55] L. D. Young, F. A. Reda, R. Ranjan, J. Morton, J. Hu, Y. Ling, X. Xiang, D. Liu, and V. Chandra, "Feature-align network with knowledge distillation for efficient denoising," in *2022 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, 2022, pp. 709–718.

[56] X. Zhang, D. Wang, P. Chuang, S. Ma, D. Chen, and Y. Li, "F-CAD: A framework to explore hardware accelerators for codec avatar decoding," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, 2021, pp. 763–768.