TREAD-M3D: Temperature-Aware DNN Accelerators for Monolithic 3D Mobile Systems

Prachi Shukla, *Member, IEEE,* Vasilis F. Pavlidis, *Senior Member, IEEE,* Emre Salman, *Senior Member, IEEE,* and Ayse K. Coskun, *Senior Member, IEEE,*

***Abstract*—Monolithic 3D (MONO3D) integration provides performance and power efficiency benefits over 2D circuits and, thus, is a potent technology for the design of Deep Neural Network (DNN) accelerators with enhanced energy efficiency. However, high IC temperatures are major challenges for the design of MONO3D systems. To this end, this paper focuses on designing temperature-aware MONO3D DNN accelerators. We propose a new automated method, called TREAD-M3D, that provides a near-optimal MONO3D DNN accelerator architecture in terms of systolic array size, SRAM organization, partition across 3D layers, and operating frequency, for a given DNN, optimization goal, and temperature constraint. TREAD-M3D incorporates circuit- and architecture-level models to evaluate the power and performance characteristics of different partitions. Our method reveals valuable insights and enables tradeoff analysis for achieving high energy efficiency in MONO3D systolic arrays. In comparison to recent works that adopt a fixed partition choice to design MONO3D DNN systems, TREAD-M3D yields up to 22% higher energy efficiency. Using TREAD-M3D, we further demonstrate that temperature unawareness not only leads to infeasible configurations due to temperature violations but also over-estimates energy-delay-product benefits by up to 24%.**

***Index Terms*—Monolithic 3D, systolic arrays, temperature optimization, deep neural networks, energy efficiency**

I. INTRODUCTION

Monolithic 3D (MONO3D) technology, out of the several 3D integration technologies such as die-stacked 3D or package-on-package [1], has emerged as a promising technology to enhance the power and performance characteristics of ICs [2]–[4]. In MONO3D, two or more thin device layers (or tiers) are sequentially fabricated with a thin dielectric in-between, achieving dense vertical connections using nanometer-scale monolithic inter-tier vias (MIVs). Fig. 1 shows a chip stack for a two-tier MONO3D IC using flip-chip packaging. Unlike the through silicon vias (TSVs) in die-stacked 3D technology (referred to as TSV3D in this paper), MIVs have a negligible keep-out-zone. These features enable a lower form factor in MONO3D, resulting in chip footprint savings and shorter interconnects. These wirelength savings result in a 9% to 20% power savings at iso-performance [5], [6], thus, improving power efficiency. MONO3D also supports partitioning at diverse granularities (i.e., transistor-, gate-, and block-level) that can be leveraged to design an energy-efficient system [2], [6]. However, even though the thin tiers and dielectric layer enable effective vertical heat dissipation in MONO3D systems, hot spots may still appear across neighboring tiers [7], [8]. Thus, temperature is a critical design issue in MONO3D systems and must be considered while designing these systems to maintain thermal integrity.

Many application domains are expected to benefit from the energy efficiency promise of MONO3D [8]. Deep neural network (DNN) inference one such application with growing significance. Due to the rapid growth of mobile applications that rely on DNN inference (e.g., drones and phones), the demand for mobile DNN accelerators has increased tremendously to achieve low latency and energy efficiency (e.g., [9]–[13]). Due to energy and chip footprint limitations, mobile systems require area- and energy-efficient DNN accelerators. Consequently, for a given chip footprint, MONO3D accelerators can potentially comprise more processing elements (PEs), compared to 2D, thus, improving compute density and inference latency. In addition to the power/area constraints in mobile accelerators, absence of advanced cooling techniques make temperature a major design challenge as well.

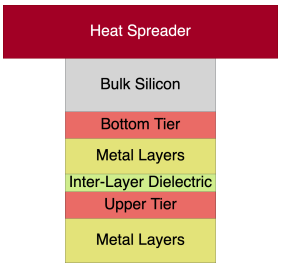


Fig. 1: Flip-chip packaging for a two-tier MONO3D IC.

Systolic arrays are used for DNN inference in mobile systems due to their high throughput without increasing memory bandwidth requirements [14], [15]. As shown in Fig. 2, a systolic array consists of a homogeneous 2D network of PEs, where a PE is a Multiply-and-Accumulate (MAC) unit with internal registers. The left edge PEs in systolic arrays read input feature map (IFMAP). The top edge PEs read filters. At every clock cycle, these inputs are read from their respective SRAMs. This input data is processed, stored in internal registers, and transferred to the neighboring PEs in the next cycle. PEs along the bottom edge of the systolic array write back the output feature map (OFMAP) to OFMAP SRAM. Since the SRAMs are accessed very frequently by the systolic array, interconnect power savings between systolic arrays and SRAMs offered by MONO3D technology can have a substantial impact on overall energy efficiency.

Recent research in mobile DNN accelerators focuses on improving energy efficiency by compressing the inputs or by replacing expensive off-chip DRAM accesses [9], [16]. Circuit-level optimization has also led to low-power mobile MONO3D systems [17], [18]. It is important to note that aside from the power and area constraints in mobile accelerators, absence of advanced cooling techniques make temperature a major design challenge as well. However, none of these works considers temperature. Consequently, we propose a method to design temperature-aware MONO3D systolic DNN accelerators. *Our method optimizes* MONO3D *systolic arrays*

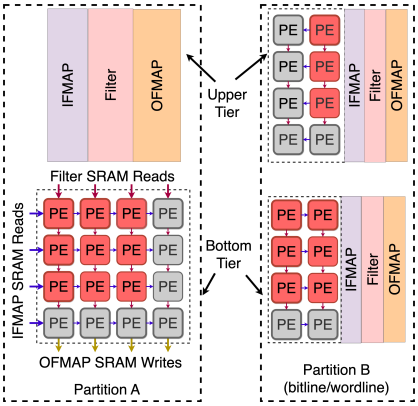


Fig. 2: A sample 16 PE systolic array with its simplified MONO3D layout for *Partition A* (SRAMs on systolic array) and *Partition B* (systolic array and SRAMs split across two layers). Active PEs are red, while idle PEs are gray.

by tuning their size, SRAM organization, MONO3D *partition*[1]*, and operating frequency for a given DNN. The effective control of these knobs is enabled by the cross-layer performance and power models that we develop in this work. This new method is a significant improvement over our prior work that conducts temperature-aware optimization with coarse-grained estimates and, thus, may overlook the benefits offered by* MONO3D *and converge to sub-optimal designs [19].* The main contributions of this work over our prior work [19] are as follows:

- We introduce MONO3D-*specific* cross-layer performance and power models for PEs, SRAMs, and interconnects for various partition options. These models enable evaluation of and comparison among MONO3D and 2D systems.
- We build an automated method to design **T**emperatu**RE-A**ware systolic **D**NN accelerators for **MONO3D**, called TREAD-M3D. TREAD-M3D optimizes a MONO3D systolic array for a desired objective, such as energy efficiency, aiming DNNs used for image recognition. We integrate our MONO3D-specific models in TREAD-M3D to enable navigation through the design space under user-specified performance and thermal constraints.
- Using TREAD-M3D, we demonstrate interesting insights among MONO3D partition choices and comparisons to 2D integration, and also the importance of considering thermal awareness in the design of mobile systolic arrays. We also extend TREAD-M3D to find a single generic accelerator architecture optimized for multiple DNNs studied in this paper for various objectives.

In comparison to recent works that utilize only one partition choice, TREAD-M3D generates configurations with up to 22% higher energy efficiency. On relaxing the thermal constraint across all partition choices, TREAD-M3D efficiently explores the design space and outputs MONO3D configurations with 21% smaller footprint savings and 8% lower energy. Using TREAD-M3D, we also show that temperature unawareness over-estimates MONO3D efficiency benefits by 24% and lead to thermal violations. TREAD-M3D also yields 10% lower power and 50% footprint savings w.r.t. 2D ICs.

[1]In this work, different partition choices describe different systolic array and SRAM organization between MONO3D layers.

Labels	Systolic array with SRAMs
A1	20 × 22, (16, 16, 16) KB SRAMs
A2	40 × 40 (64, 16, 128) KB SRAMs
A3	64 × 64, (32, 32, 512) KB SRAMs
A4	124 × 130, (512, 1024, 1024) KB SRAMs
A5	144 × 120, (512, 1024, 1024) KB SRAMs
A6	182 × 192, (2048, 2048, 2048) KB SRAMs

TABLE I: Systolic arrays for motivational examples.

The rest of the paper starts with a motivation for temperature-aware optimization framework to design efficient MONO3D systolic arrays. Section III presents related work on systolic arrays, MONO3D thermal issues, and MONO3D DNN inference accelerators. Section IV describes TREAD-M3D, followed by results in Section V. Finally, we conclude and discuss future work in Sections VI and VII, respectively.

II. A MOTIVATIONAL EXAMPLE

This section compares 2D and MONO3D configurations to motivate the need for temperature-aware optimization for MONO3D mobile systems. For simplicity, we only focus on one type of MONO3D partition. We introduce other MONO3D partition choices later in Section IV-A. The MONO3D configurations in this section are comprised of SRAMs in the upper tier and a systolic array in the bottom tier (see Fig. 1). In 2D, the SRAM and systolic array are placed adjacently in one tier. Table I shows six configurations, sorted by their footprint, for this motivational example. We select six configurations, listed in Table I, from our design space using a coarse grid sampling for this motivation example. Our complete design space is detailed in Sec. V-A. For this study, we select VGG11 as our target DNN. We use three frequency levels (600, 800, and 1,000 MHz), a performance constraint of 10% loss in inference latency from 60 frames per second (fps) [20], and two thermal constraints (70°C and 80°C) to capture the importance of temperature. We evaluate two objectives: minimizing chip power and energy-delay-area-product (EDAP). Note that the best configuration not only has the least objective value, but also satisfies the performance and temperature constraints. Figs. 3a and 3b show the deviation of VGG11's inference latency from 60 fps and steady state temperatures, respectively, for each Ai at the three frequencies. E.g., $A1$ at 600 and 800 MHz does not satisfy the latency constraint due to small number of PEs, or $A1$ violates 80°C in MONO3D at 1,000 MHz. Missing bars in Fig. 3 imply that the systolic array is unable to operate at that frequency due to long wire lengths. For example, A6 has a missing bar in Figure 3a at 1,000 MHz. However, the design space can comprise hundreds of thousands of configurations due to several control knobs, such as DNN of interest, systolic array size, array aspect ratio, SRAM size, operating frequencies, MONO3D partition choices, etc. Thus, exhaustively identifying configurations that satisfy constraints for each DNN is not feasible.

Figs. 3c, 3d, and 3e show the normalized EDAP, power, and power density with respect to the least values among all six configurations, respectively. Fig. 3d shows that $A1$ at 1,000 MHz has the lowest power and satisfies all constraints in 2D. However, the same configuration results in thermal

violation in MONO3D technology due to 1.9× higher power density resulting from footprint reduction. Similarly, for EDAP efficiency (or, minimum EDAP), Fig. 3c shows that $A3$ at 1,000 MHz is the most efficient in 2D technology. However, under a tight 70°C constraint, $A3$ results in a thermal violation in MONO3D. On the other hand, on relaxing the constraint to 80°C, $A3$ executes safely. Thus, not only 2D and MONO3D may have different optimal configurations, but varying the thermal constraint may lead to a different optimum point in MONO3D. Furthermore, finding optimal configurations is often non-trivial in a vast design space due to (i) continuously emerging state-of-the-art DNNs with competitive accuracies, different topologies and compute/memory requirements, (ii) limited thermal headroom in mobile systems, (iii) desired constraints and objectives for a chip architect/designer, and (iv) numerous possible values for the multiple control knobs discussed above (e.g., frequency, array size, etc.). Also, a fine-grained grid search or exhaustive search may be infeasible and time consuming to find optimized points. Hence, it is important to develop a temperature-aware optimization framework to traverse through the design space efficiently, evaluate performance, power, and thermal characteristics for a subset of the design space for a DNN of interest, and converge to near-optimal configurations for 2D or MONO3D technologies.

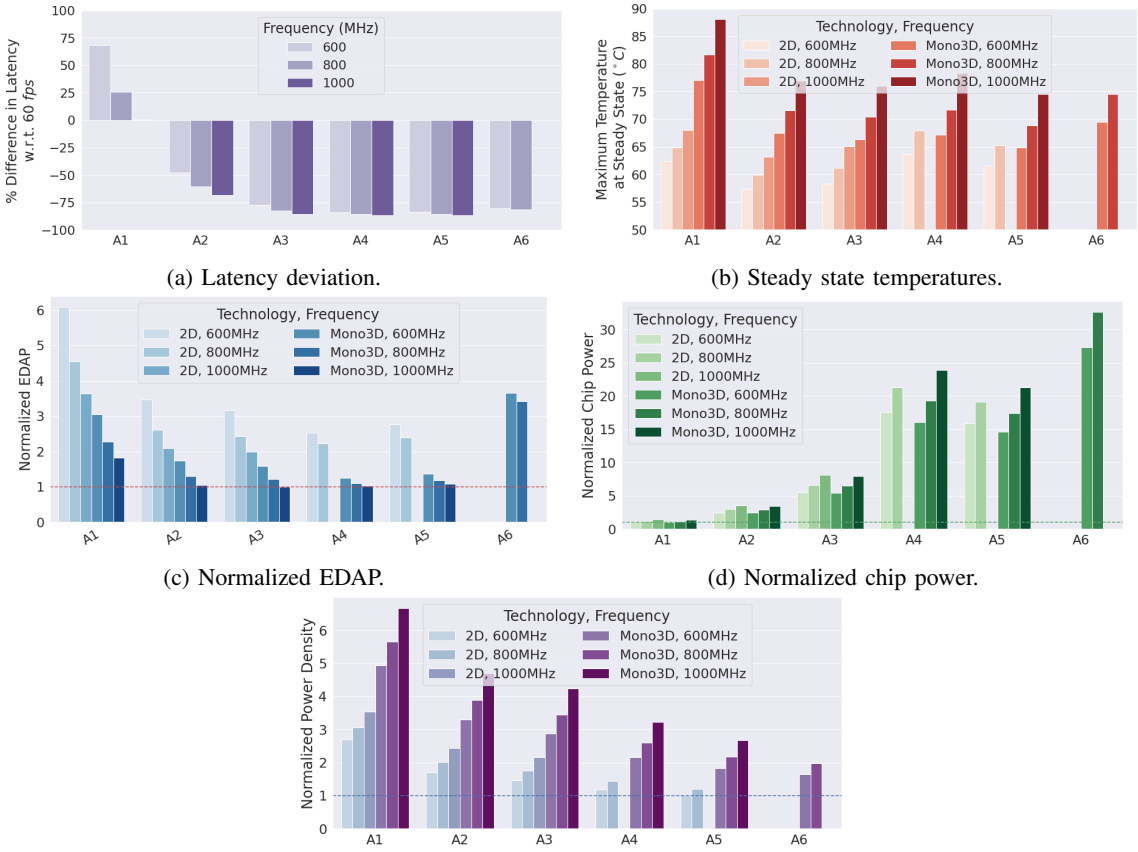
III. RELATED WORK

Systolic DNN accelerators. Several works have proposed techniques for increasing efficiency of systolic arrays due to their growing use for DNN inference. For example, Asgari *et al.* propose pruning methods to reduce memory accesses and achieve higher energy efficiency in sparse DNN inference [21], [22]. Liu *et al.* combine sub-arrays of PEs into larger Tensor-PEs to improve data re-use and achieve 2× power efficiency in mobile systems [23]. Li *et al.* replace off-chip DRAM with emerging on-chip memory technologies to achieve 2× energy efficiency in mobile systolic arrays [9]. There exists another body of work that focuses on DNN and hardware co-design for higher energy efficiency [24], [25]. All these works target 2D arrays. Kung *et al.* introduce tiled systolic architecture vertically interfaced with a memory die using TSVs for high memory bandwidth, and thus, resulting in significant latency improvement over 2D [16]. However, the effect of temperature on such systems has not been assessed. Another work demonstrates a need for thermal awareness in the design of TSV3D systolic array chiplets in a multi-chip module for a multi-DNN workload [26].

Thermal integrity in MONO3D systems. Several works have investigated thermal issues in MONO3D systems and proposed appropriate remedies. For example, optimizing the power delivery network can improve thermal conductivity and lead to reduction in on-chip temperature [27], [28]. Iqbal *et al.* propose the use of nano pillars for extracting heat from selected hot spot regions [29]. Such techniques improve thermal conductivity, which allows effective heat removal via the heat spreader/sink. Lee *et al.* demonstrate MONO3D benefits over 2D for high-performance ICs when using emerging cooling techniques [30]. In another work, Samal *et al.* show that tight inter-tier vertical thermal coupling with negligible lateral flow of heat exists in MONO3D ICs and build a non-linear regression model for temperature estimation of the tiers [31]. None of these works target thermal integrity in mobile systems that are area, power, and thermally constrained.

MONO3D DNN accelerators for inference. On the MONO3D front, power and performance benefits offered by this technology have led to an increasing interest in designing DNN accelerators. Chen *et al.* propose an accelerator architecture with resistive RAMs (ReRAMs), multiple layers of carbon nanotube field-effect transistors (CNFETs) based ADC/DAC for the ReRAMS, and CNFETs-based SRAMs. The CNFETs based logic and memory structures result in a higher power efficiency, in comparison to CMOS, thus improving performance per watt in DNN inference [32]. Yu *et al.* introduce an architecture with ReRAM memory tiers interfaced with an accelerator tier using MIVs. The high density MIVs provide high memory bandwidth resulting in significant energy savings in DNN inference [33]. Chang *et al.* investigate partitioning choices to design a post-layout two-tier MONO3D ASIC (with MAC units and memory blocks) for speech recognition DNN models and show significant performance/power improvements [18]. However, none of these works considers temperature awareness, which can be a major issue especially with multiple layer stacking. Furthermore, these works do not find optimal architectures for DNN workloads. A recent work proposes a variant of output stationary dataflow to utilize the vertical dimension in MONO3D systolic arrays, where each tier has private SRAMs [34]. A 12-tier 3D systolic architecture is shown to have a 9.14× speed up over 2D in high-performance systems such as servers. The 3D systolic arrays are shown to not have major thermal issues because high-performance systems are usually equipped with powerful cooling solutions. The authors, however, have not modeled temperature-dependent leakage, which can be non-negligible due to strong inter-tier thermal coupling in MONO3D systems [7], [35], and thus, have an impact on the thermal behavior of the system. Also, the DNNs considered in their work may not be high-power DNNs. We later demonstrate in this paper that DNNs like VGG11 or Faster R-CNN are limited by thermal headroom. Another work models different options of stacking multiple layers of systolic array and SRAM layers to achieve better performance than 2D systolic arrays [36]. However, they do not have a performance model in place to measure the effect of stacking options on frequencies or wirelengths. Nor do they investigate SRAM partitions or determine optimal and efficient systolic architecture for DNNs of interest. In summary, these works do not provide a systematic method that *i) effectively explores the design space of* MONO3D *systems, ii) considers thermal issues, and iii) evaluates improvements over 2D DNN accelerators. Our proposed method enables these missing features through circuit- and architecture-level power and performance models that are scalable across systolic array sizes and determines optimal* MONO3D *systems for given objectives and thermal constraints.*

Our prior work shows performance versus temperature tradeoffs in systolic arrays for only one MONO3D partition: SRAMs monolithically grown on top of systolic arrays [19].



(a) Latency deviation.

(b) Steady state temperatures.

(c) Normalized EDAP.

(d) Normalized chip power.

(e) Normalized power density.

Fig. 3: A motivational example with VGG11 at three frequencies. Normalization (red dotted lines) is done with respect to the minimum value among the six configurations. (a) %Difference in inference latency with respect to 60 fps. Positive bars: worse latency, negative bars: smaller latency. $A$is with y-axis values $>$10 violate the performance constraint. (b) Steady state temperatures. (c) Normalized system EDAP with respect to $A3$ at 1,000 MHz. Best configurations differ in 2D ($A3$ at 1,000 MHz) and MONO3D ($A5$ at 800 MHz) under a tight thermal budget of 70°C. (d) Normalized total chip power with respect to $A1$ at 600 MHz. Best configurations differ in 2D ($A1$ at 1,000 MHz) and MONO3D ($A2$ at 600 MHz) due to thermal issues even at a relaxed thermal constraint of 80°C. (e) Normalized power density with respect to $A5$ at 600 MHz.

It assumes fixed frequency levels across all systolic arrays and adopts a coarse-grained MONO3D power model common across all DNNs and systolic array architectures, and thus, may lead to sub-optimal choices. Furthermore, it lacks appropriate performance and power models needed for comparison with 2D and other MONO3D partition choices. In this work, we introduce cross-layer models that determine the maximum operating frequency levels for DNN inference on systolic arrays and estimate power for varying DNN topologies, systolic arrays, and MONO3D partition choices. In addition, they also allow comparison of 2D and MONO3D partition choices w.r.t. various optimization goals, unlike our prior work.

IV. TREAD-M3D

This section introduces TREAD-M3D, its cross-layer models, MONO3D-partitioned accelerators, and the optimizer. Fig. 4 shows an overview of TREAD-M3D with its five phases ($P1$-$P5$). TREAD-M3D takes a DNN topology (i.e., layer-wise description of a DNN including #filters, #channels, input/filter size, and strides), MONO3D partition choices, design constraints (e.g., bounds on systolic array), performance/thermal constraints, and optimization goal as inputs, and outputs the optimal configuration in $P5$ after multiple iterations through $P1$-$P4$. An optimization goal is a metric that is minimized: power, energy, energy-delay-product (EDP), etc. Our temperature-aware optimizer starts in $P1$, by randomly selecting an accelerator configuration for initialization (C_i). C_i comprises a systolic array and three SRAMs using a MONO3D partition, and also satisfies the user-defined constraints. The optimizer then evaluates performance, power, and temperature characteristics of C_i using cross-layer models in $P2$-$P4$. In each iteration, the $P2$-$P4$ outputs are sent to $P1$ to select a new configuration for next iteration. After multiple iterations, the optimizer converges to a near-optimal configuration in $P5$.

A configuration C is first evaluated for performance in $P2$ as follows: TREAD-M3D performs (i) architecture-level SRAM optimization to meet the bandwidth requirement for DNN inference on C, (ii) architecture-level analysis to generate compute cycles and other performance metrics, (iii) circuit-level interconnect delay optimization between systolic array and SRAMs, (iv) circuit-level modeling to determine the highest operating frequency $freq_{max}$ for C, and (v) random

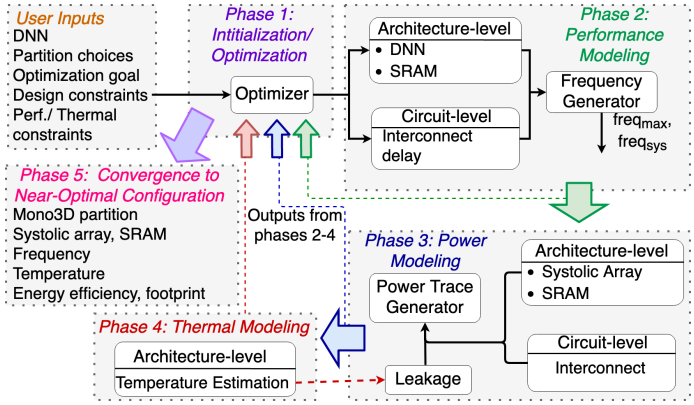


Fig. 4: TREAD-M3D overview.

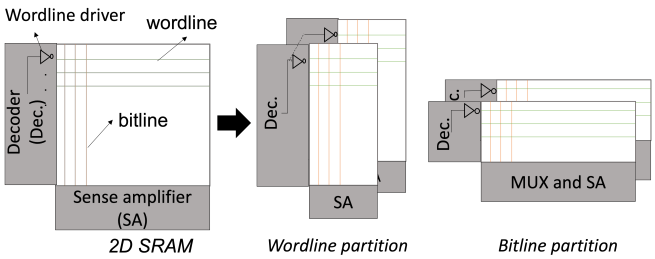


Fig. 5: SRAM partitioning styles: Wordline partition reduces wordline capacitance but duplicates wordline drivers. Bitline partition reduces bitline capacitance but adds muxes and sense amplifiers. Both result in latency reduction.

selection of a frequency $freq_{sys} (\leq freq_{max})$ for the current iteration. The latency is calculated with $freq_{sys}$. A higher $freq_{max}$ implies more frequency choices for faster inference. Note that $freq_{max}$ for given systolic array and SRAM sizes depends on wire length and thus, can vary between different partition choices (explained in Sec. IV-C). $P3$ generates power traces for C at $freq_{sys}$ using (i) architecture-level systolic array and SRAM power estimation, and (ii) circuit-level interconnect power optimization between SRAM and systolic array. $P4$ then computes steady state temperature, and back-propagates the temperatures to $P3$ for leakage estimation. $P3$ and $P4$ operate in a loop until temperature converges (i.e., the difference between consecutive thermal simulations is $\leq$ 1°C for both SRAMs and PEs). The optimizer iterates through $P1$-$P4$ until it can no longer find a better configuration, and converges in $P5$. The rest of the section first describes the partitions and optimizer, followed by our cross-layer models.

A. MONO3D-*partitioned Systolic Arrays*

In this work, we assume a mature MONO3D process in which the characteristics of upper and bottom tiers are similar. This assumption has also been adopted in other MONO3D state-of-the-art works [37], [38]. In addition, specific processes that limit the temperature to sub-600°C for the manufacturing of the upper tiers have been reported to achieve similar characteristics in the top and bottom tiers [39]. Fig. 2 shows the three two-tier (realistic number of tiers due to low thermal budget during fabrication and limited yield) MONO3D partitions that this work investigates: *Partition A*, *Partition* $B_{bitline}$, and *Partition* $B_{wordline}$. *Partition A* is a two-tier block-level partitioned system with 2D SRAMs monolithically grown on top of the systolic array. In this case, the read/write latencies generated by CACTI are the same as that in 2D because there is no change in the SRAM block design, an assumption also used by Guler *et al.* [40]. We note that SRAM latencies become the bottleneck in configurations with large SRAMS ($\geq 1\ MB$). Hence, we also investigate *Partition B*, where the SRAM sub-arrays[2] are partitioned into two tiers either along wordlines (*Partition* $B_{wordline}$) or bitlines (*Partition* $B_{bitline}$), as shown in Fig. 5. *Partition B* reduces SRAM access latency since the latency is limited by wordline/bitline delay [41]. The systolic array in *Partition B* is folded into two tiers from the center (see Fig. 2). Splitting the SRAM peripheral circuitry across two tiers is another approach for partitioning [42], but this approach is not considered in this work as we primarily focus on splitting the data array for low latency.

B. *Phase 1: Optimizer*

1) Multi-start annealing (MSA) overview: MSA is an optimization algorithm extensively used in search problems where an objective function $F(x)$ is minimized subject to some constraints CS. Fig. 6 shows a flow diagram for MSA. Multiple starts execute in parallel, with an annealer running inside each start. Multiple starts increase the probability of reaching the global minima than a single instance of the annealer. Each annealer in MSA operates on the hill-climbing principle but can select a worse configuration to escape a local minima. Each annealer starts with randomly generating a configuration that satisfies CS. At every iteration, the annealer randomly generates a new configuration by tuning its control knobs with uniform probability. If $F(x)$ of the new configuration is lower than the current one, it is accepted, otherwise it is accepted based on Boltzmann probability ($Pr = \exp^{-\Delta F/(\Delta F_{avg} \cdot T_{msa})}$). ΔF is the difference between $F(x)$ of the current and new configurations, ΔF_{avg} is the running average of ΔF for the accepted configurations, and T_{msa} is the annealing temperature[3] in that iteration. An initial probability (P_s) decides the starting annealing temperature T_s

[2]By definiton, SRAM data arrays are organized in sub-arrays [41]

[3]Unitless variable in MSA that decides whether to accept a worse configuration.

that is lowered after a fixed number of steps (N) by a factor α, with a total of num_T distinct annealing temperatures. The algorithm finally converges when T_{msa} is sufficiently low to not accept worse configurations.

2) Temperature-aware optimizer in TREAD-M3D*:* We use an MSA-based optimizer to generate an optimal configuration, for the user-specified optimization function, $F(x)$ (e.g., inference latency or power efficiency). This optimizer takes the following user inputs as constraints: (i) bounds on systolic array, (ii) maximum SRAM size, (iii) chip footprint budget, (iv) latency overhead, and (v) temperature budget. The optimizer uses MONO3D partitions, systolic array size, SRAM organization, and clock frequencies (determined using the performance model presented in Sec. IV-C) as control knobs to find near-optimal architectures. As the annealing temperature decreases, the optimizer does not accept configurations with worse $F(x)$ and finally converges to a near-optimal configuration by exploring a small fraction of the total design space. MSA is also inherently scalable with increasing design space because more starts can be launched in parallel for design space exploration. After initial tuning of the annealing parameters with known good results, it can be used to find optimized configurations for various other DNNs.

C. Phase 2: MONO3D Performance Models

We design a high-performance PE using Synopsys Design Compiler at 65 nm {794 MHz, 1.37 mW, 1028 μm^2, 1.2 V} and scale it down to 22 nm {1 GHz, 0.25 mW, 121 μm^2, 0.8 V} to utilize its latency, power, and area estimates in our analyses [43]. We detail the MONO3D cross-layer performance modeling approach below.

1) Architecture-level models: We use SCALE-Sim, a cycle-accurate DNN simulator for systolic arrays, for architecture-level simulations [44]. Inputs to SCALE-Sim are the DNN topology, systolic array and SRAM sizes, dataflow, and DRAM bandwidth. SCALE-Sim simulates a stall-free feed-forward inference on 8-bit integer data, and outputs compute cycles, DRAM cycles, average array utilization, and DRAM/SRAM bytes transferred. We calculate the total inference latency by adding compute time and DRAM time that does not overlap with compute. We choose output stationary (OS) dataflow for our analysis since SCALE-Sim has been validated against an RTL model for OS [44].

For SRAM architecture-level modeling and optimization, we use a popular SRAM simulator, CACTI-6.5 [41]. Both bitline and wordline partitioning styles decrease the global interconnect length within the SRAM (e.g., distance between the predecoder and local wordline decoder of sub-arrays, or length of select lines for MUXes) due to reduction in chip footprint and lead to latency improvement [45]. We explicitly consider the cost of duplicated blocks such as wordline drivers or sense amplifiers [45] (unlike recent prior work on MONO3D L1 caches partitioned across bitlines/wordlines [46]). For a two-tier SRAM partition across wordlines/bitlines, we divide the wordline/bitline capacitance of each sub-array by 2, which reduces the access latencies since these lie on the critical path. In wordline partition, we add drivers for each wordline in the two tiers. Even though we have twice as many wordline drivers, each driver now drives a smaller load as wordline capacitance becomes half, and can potentially lead to power savings [45]. For bitline partition, we add sense amplifiers to both tiers for faster bitline access. However, this also increases leakage due to the duplication. CACTI internally performs a design space exploration to generate an energy-delay^2 product (ED^{2}P)-optimized SRAM configuration for the desired partition. For instance, a 32KB SRAM may have dissimilar design for the two partition styles (e.g., different number of banks and/or block sizes). Thus, it is not straightforward to determine which partition will be optimal for an iso-capacity SRAM because (a) internal SRAM design may be different, (b) ED^{2}P is a lumped metric, and (c) thermal budget affects the power density of the MONO3D SRAM that can be endured. Similar observations are also cited in another 3D cache work [45]. We ignore the area overhead due to MIVs due to their small size [2], but add MIV's delay (1.83 ps), dynamic energy (4.66e-7 nJ), and leakage (3.87e-5 mW) to the model [46].

TREAD-M3D first determines the minimum bandwidth (bytes transferred per SRAM access) for each SRAM for OS, and then generates an ED^{2}P-optimized design (#banks, block size) for each SRAM. E.g., for a 48×32 (rows×columns) systolic array, the bandwidths we model in CACTI for (IFMAP, Filter, OFMAP) SRAMs are (64, 32, 32) bytes per SRAM access, since SCALE-Sim assumes single-cycle SRAM access. We round them off to the nearest powers of 2 due to SRAM design constraints in CACTI. It is not straightforward to determine which MONO3D SRAM partition (bitline or wordline) gives the lowest latency because a design optimized for ED^{2}P may not be necessarily optimized for latency [45].

2) Circuit-level models and optimization: TREAD-M3D uses HSPICE and, without loss of generality, 22 nm PTM models [47] for wirelength modeling. The interconnect delay between the systolic array and SRAM is determined by the Manhattan distance between them [48].

Since only the edge PEs read/write SRAMs, TREAD-M3D first calculates the longest Manhattan distances between: (i) left edge PEs and IFMAP SRAM, L_I, (ii) top edge PEs and Filter SRAM, L_F, and (iii) bottom edge PEs and OFMAP SRAM, L_O, followed by the longest interconnect length among them, i.e., $L_{max} = max(L_I,\ L_F,\ L_O)$. TREAD-M3D then runs HSPICE to determine the optimal number of repeaters (i.e., CMOS inverter) that are inserted across L_{max} for minimum delay, D_{Lmax} [49], [50]. Note that we include repeater insertion in HSPICE modeling to minimize delay associated with data transfer. CACTI also internally inserts repeaters to optimize the SRAM architecture.

3) Frequency generation: TREAD-M3D calculates the highest possible frequency $freq_{max}$ (assuming that the PEs, SRAM access, and interconnect represent individual pipeline stages), for the configuration, as shown in Eq. (1). It then discretizes the frequencies between $freq_{max}$ and a lower bound in 50 MHz step size, inclusive of the bounds. We assume 100 MHz as a safe lower bound so that all realistic frequencies are considered. Finally, our optimizer selects a discretized frequency ($freq_{sys}$) with uniform probability, calculates inference latency, and proceeds to $P3$. Note that

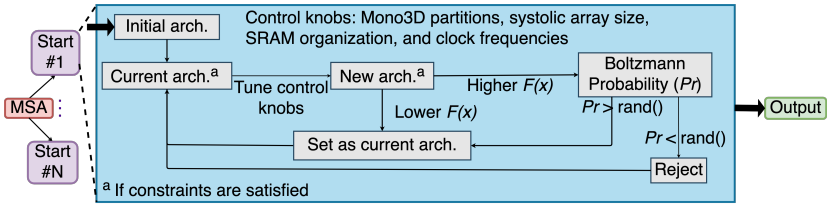


Fig. 6: Flow diagram of MSA.

the upper tiers in MONO3D may suffer from degraded performance, e.g., 10% [17]. This behavior may be captured by the proposed framework by (i) adding a 10% degradation in CACTI-generated latencies and/or MAC latency (depending on which blocks are placed in the upper tier), and (ii) substitute these degraded latencies in Eq. (1) to calculate the new max frequency ($freq_{max,new}$). Upon running the optimizer, it uses the equation to ensure that the frequencies are tuned between 100 MHz and $\text{freq}_{max,new}$.

$$freq_{max} = max(PE\ latency,\ SRAM\ latency,\ D_{Lmax})^{-1} \quad (1)$$

D. Phase 3: MONO3D Power Models

$P2$ outputs are used in $P3$ to generate power traces.

1) Systolic array and SRAM: TREAD-M3D first uses average array utilization (U_{av}) to determine the number of active PEs ($rows \times columns \times U_{av}$). It then calculates PE's dynamic power ($DynP_{PE}$) at $freq_{sys}$, as shown in Eq. (2). TREAD-M3D also uses CACTI, configured using the optimized SRAM parameters from $P2$, to calculate SRAM power. We also fit an exponential PE leakage model using temperature and leakage data points from our synthesized MAC model. For SRAM leakage, in addition to the CACTI's leakage output, we build a linear interpolation model for SRAM leakage at a finer granularity than CACTI's default of 10 degrees [51]. Initial PE and SRAM leakage are determined at ambient temperature[4].

$$DynP_{array} = \#active\ PEs\ \times\ DynP_{PE} \quad (2)$$

2) Systolic array-SRAM interconnect power: Typically either PE latency or SRAM latency dominates in Eq. (1). Therefore, interconnect parameters are reoptimized to save power. Thus, TREAD-M3D again performs repeater insertion ($Opt_{repeater}$) for L_{max} for power minimization such that its delay $\leq freq_{max}^{-1}$. This way, TREAD-M3D finds the corresponding optimal segment length ($Opt\_SegLength$). For calculating interconnect power at $freq_{sys}$, our proposed method first runs HSPICE transient simulations, assuming 50% switching activity at $freq_{sys}$, to compute dynamic power for $Opt\_SegLength$, i.e., $DynP_{Opt\_SegLength}$ (includes wire segment's and repeater's power). It then determines the average Manhattan distances between the active edge PEs (only the edge PEs read/write to the SRAMs) and their respective SRAMs (i.e., $L_{av,IFMAP}$, $L_{av,FMAP}$, and $L_{av,OFMAP}$). Next, it calculates the average dynamic power for each of the average wire lengths ($DynP_{wire,SRAMi}$), where $SRAMi \in$ (IFMAP, Filter, OFMAP), as shown in Eq. (3). Finally, since each PE operates on 8-bit data, the total interconnect power ($IntP$) is calculated, as shown in Eq. (4).

[4]Ambient temperature refers to the temperature inside a device [52].

$$DynP_{wire,SRAMi} = DynP_{Opt\_SegLength} \times \frac{L_{av,SRAMi}}{Opt\_SegLength} \quad (3)$$

$$\begin{aligned} IntP = 8 \times \{&(\#activePEs_{left\_edge} \times DynP_{wire,IFMAP}) + \\ &(\#activePE_{top\_edge} \times DynP_{wire,Filter}) + \\ &(\#activePEs_{bottom\_edge} \times DynP_{wire,OFMAP})\} \end{aligned} \quad (4)$$

E. Phase 4: MONO3D Temperature Models

TREAD-M3D creates compact thermal models (CTMs) for the three partitions (*Partition A*, *Partition* $B_{bitline}$, and *Partition* $B_{wordline}$) in HotSpot-v6.0 [53] for temperature estimation in mobile systems. We adopt the validated MONO3D CTM presented in our prior work, where the CTM has 32 layers [19]. A top view of *Partition A* and *Partition B* is shown in Figure 2. The heat spreader thickness is set to 50 μm while the heat sink is effectively removed by assigning it a negligible thickness. For SRAM leakage estimation, *Partition A* uses the HotSpot-generated temperature, while *Partition B* takes the average across the two tiers for each SRAM. TREAD-M3D updates the power traces with SRAM and array leakage and re-estimates steady state temperature in HotSpot. This loop continues until the temperature difference between consecutive runs is $\leq 1°C$ for SRAMs and PEs. Representative material thicknesses and conductivities are from recent works [7], [31].

V. EXPERIMENTAL RESULTS

This section describes the experimental setup, followed by optimizer's accuracy and runtime analysis. We then analyze the Pareto optimal front for energy versus latency at various thermal constraints. This helps us to capture the effect of temperature on energy, area, power, and frequency of the Pareto optimal configurations. We also present optimization results for various objectives. We then compare TREAD-M3D to 2D and baseline MONO3D architectures. Finally, we demonstrate that TREAD-M3D can be further used to generate a single accelerator architecture to efficiently execute the multiple DNNs studied in this paper.

A. Experimental setup

We study the effects of MONO3D on commonly used DNNs for image processing [54]–[56]. We group them into two types: (i) large DNNs with large input and filter size (i.e., higher number of MAC operations) - VGG19, VGG16, VGG11, Faster R-CNN, and ResNet50; and (ii) small DNNs with fewer convolutional layers, smaller input size and filters (i.e., fewer MAC operations) - MobileNet, DQN, GoogLeNet, and Tiny-YOLO. Table II summarizes systolic array configurations in our complete design space, $\mathbb{D}_{\mathbb{C}}$. We set the lower bound on

clock frequency to 100 MHz and upper bound to the frequency determined by TREAD-M3D. In total, there are 263k unique configurations per DNN in $\mathbb{D}_{\mathbb{C}}$. For DRAM, we use LPDDR2 with a frequency of 400 MHz and energy consumption of 40 pJ/bit [57], [58]. We also explore three MONO3D partitions and use HotSpot's default ambient temperature of 45°C (also a commonly accepted value [52], [59]) and grid mode with grid length = PE length for steady state analysis. To set performance constraints for the DNNs, we set an acceptable loss to 10% of the inference latency for a 60 fps camera [20], i.e., 16.7 ms. Note that the MONO3D partition choices are shown in Fig. 5 and described in Sec. IV-A.

B. Optimizer's accuracy and runtime analysis

We select a small subset of the design space ($\mathbb{D}_{\mathbb{T}}$) to tune our optimizer. $\mathbb{D}_{\mathbb{T}}$ includes all architectures between a smaller aspect ratio range of 0.98 to 1.02, while keeping the other configuration settings the same. In total, $\mathbb{D}_{\mathbb{T}}$ contains 28k unique configurations per DNN, inclusive of the three MONO3D partitions. We evaluate the optimizer's accuracy with two DNNs: VGG11 and GoogLeNet, and for several optimization goals (i.e., chip power, energy, EDP, and EDAP). Note that energy, EDP, and EDAP include both on-chip and DRAM energy. To approach the globally optimal solution in $\mathbb{D}_{\mathbb{T}}$, we tune our optimizer by varying P_s, num_T, α, N for the various optimization goals discussed in Sec. IV-B1. We initiate nine starts in parallel, where each start randomly searches in a subset of the total configurations. We achieve high accuracy with a deviation from global optima by $\leq$ 3.84% by exploring only 20% of $\mathbb{D}_{\mathbb{T}}$. The optimizer rejects worse solutions near termination, hence verifying convergence. The optimal settings of the optimizer are: P_s = 0.5, N = 100, num_T = 6, and α = .84, .87, .83, and .91 for power, energy, EDP, and EDAP optimization, respectively. Using these optimal settings and same design space $\mathbb{D}_{\mathbb{T}}$, we verify our optimizer's accuracy at two more ambient temperatures, i.e., 25°C [60] and 55°C [61], to show that it is robust and not over-tailored to specific experimental settings. The optimizer converges by exploring only 20% of $\mathbb{D}_{\mathbb{T}}$. Furthermore, the deviation from the global optima, across all the four objectives, at 25°C and 55°C is only up to 4.56% and 4.64%, respectively, thus, showing that the optimizer is versatile enough for real-life conditions with fluctuating temperatures.

HSPICE simulations for each configuration take up to two minutes to calculate delay and power with repeater insertion optimization for the three SRAMs. SCALE-Sim and HotSpot take 10-60 and 5-45 mins, respectively, depending on the chip footprint and DNN. Large DNNs have a higher number of MAC operations that lead to higher power dissipation and peak temperatures (more active PEs), which increase temperature-dependent leakage. Thus, these DNNs require up to 4-5 iterations to converge in HotSpot. Small DNNs require up to 2-3 iterations due to fewer MAC operations [62] and lower chip power. Long simulation times are bottlenecks to perform an exhaustive search in our large design space and demonstrate the need for an optimizer. Compared to a brute force search, we expect to see a reduction in search time by 80% because the optimizer only traverses through 20% of the total design space. Specifically, our design space consists of 6,175 unique systolic array configurations between aspect ratio 0.8 to 1.2 and a total of 263k HotSpot simulations (including various frequencies). We perform our simulations on the Massachusetts Green High Performance Computing Center (MGHPCC). We run each simulation on an Intel Xeon E5-2680 v4 CPU node with 128 GB of memory. For instance, one VGG19 simulation (the largest DNN in our work), on average, takes 30 minutes in SCALE-Sim and 15 minutes in HotSpot. With 33 multi starts, our optimizer takes 18 days to converge to a near optimal solution. In contrast, exhaustive search will take 87 days if we are running 33 searches in parallel.

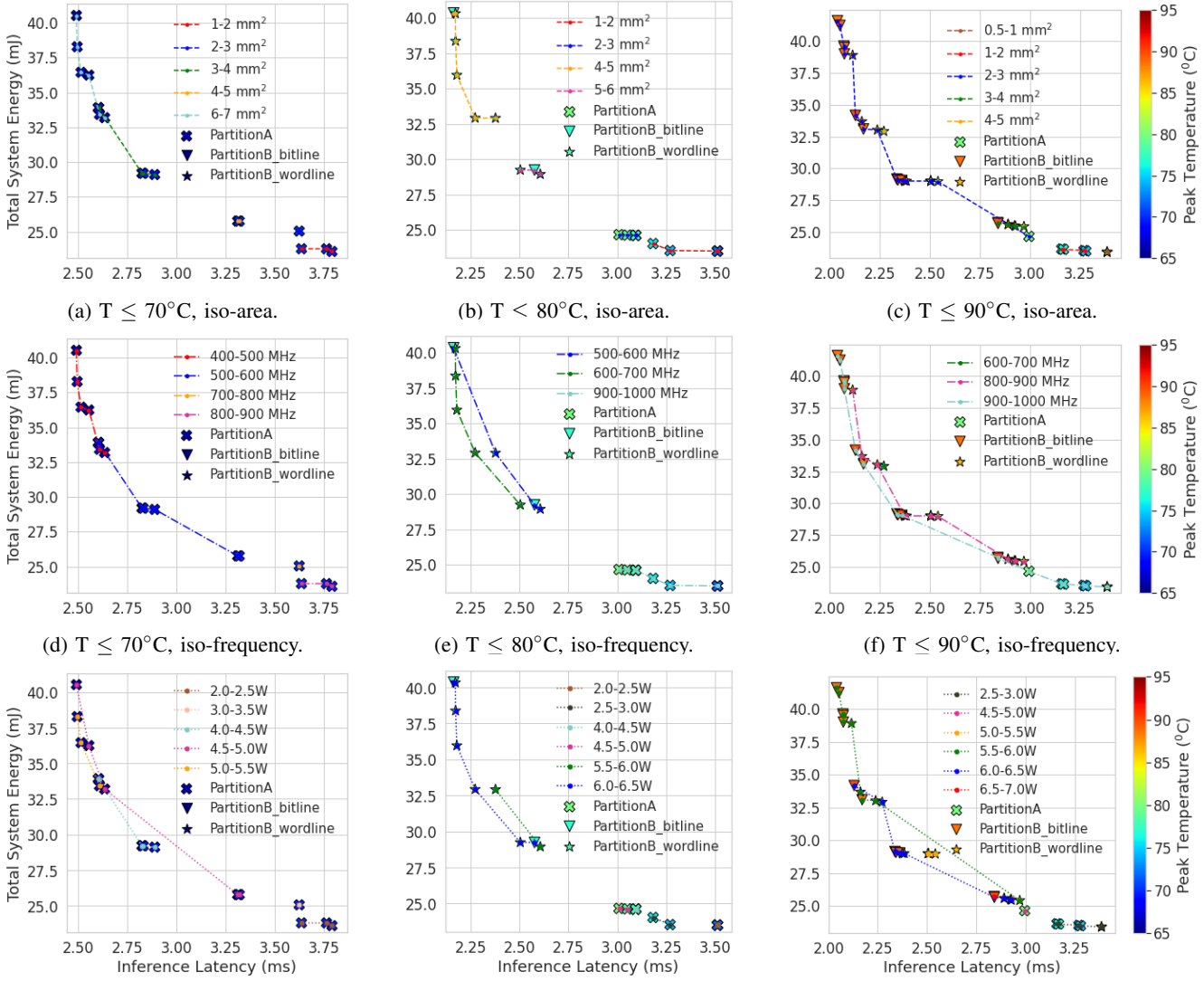
Systolic arrays	16×16 to 256×256
Aspect ratio of systolic arrays	0.8 to 1.2
SRAMs	(8, 16, 32 ... 8192) KB
Frequency bounds	100 MHz to $freq_{max}$ in 50 MHz steps
White space allowed (due to area mismatch between tiers)	0.5%
Partition choices	*Partition A* *Partition* $B_{bitline}$ *Partition* $B_{wordline}$

TABLE II: Complete design space ($\mathbb{D}_{\mathbb{C}}$).

C. Pareto optimal front

We now present interesting insights by running TREAD-M3D under various temperature constraints. We use 80°C as a thermal constraint because it is a commonly accepted constraint in commercial mobile devices [63], [64]. Typically, the primary reason for having strict thermal constraints in mobile applications is the leakage power. Note that we consider temperature-dependent leakage power in this optimization framework. In addition, considering that mobile devices can range from mobile phones to tablets to drones etc. we also present a Pareto optimal analysis for three thermal constraints: 70°C (tight constraint), 80°C (commonly accepted constraint), and 90°C (relaxed constraint). The optimizer searches $\mathbb{D}_{\mathbb{C}}$, including the partition choices (Fig. 2), and outputs the Pareto optimal frontier for a given DNN. Through this Pareto curve, we capture interesting thermal effects on energy, inference latency, systolic array size, and optimal MONO3D partition choice. We initiate 33 parallel starts and increase the number of perturbations to 250 per annealing temperature to achieve convergence, while fixing the other annealing parameters. The optimizer explores 20% of $\mathbb{D}_{\mathbb{C}}$ and converges near termination. Fig. 7 shows the Pareto front obtained on running TREAD-M3D for VGG16 for minimizing system energy at various temperature constraints. Each column in the figure is a Pareto front at a different temperature constraint, while the rows in a column display the same front at iso-area, iso-frequency, and iso-power, respectively. The x and y axes show inference latency and system energy, respectively.

Figs. 7a, 7b, and 7c show that across all temperature constraints, small footprints (< 2 mm^2) contain few PEs and, therefore, demand the least energy. Thus, the optimizer



(a) T $\leq$ 70°C, iso-area. (b) T $<$ 80°C, iso-area. (c) T $\leq$ 90°C, iso-area.

(d) T $\leq$ 70°C, iso-frequency. (e) T $\leq$ 80°C, iso-frequency. (f) T $\leq$ 90°C, iso-frequency.

(g) T $\leq$ 70°C, iso-power. (h) T $\leq$ 80°C, iso-power. (i) T $\leq$ 90°C, iso-power.

Fig. 7: Pareto optimal front for VGG16.

chooses high frequencies $\geq$ 800 MHz (Figs. 7d, 7e, 7f) to meet the latency constraint due to the small array size. Furthermore, at the tighter 70°C constraint, *Partition A* is more energy-efficient (note the absence of *Partition B* designs at 70°C). This is because small footprint in *Partition B* architectures yield thermal violations due to the higher power density caused by logic-on-logic integration. Pareto optimal architectures at 80 and 90°C with lower latencies have larger footprints (i.e., more active PEs) and predominantly follow *Partition B*. This is because the optimizer utilizes the thermal slack and selects SRAM partition to achieve higher frequencies (due to reduction in the critical path latency), hence lower inference latencies. We also observe that relaxing the thermal constraints for VGG16 allows for Pareto optimal designs that are on average 36% smaller (thus, fewer active PEs) with 54% higher frequencies for minimizing system energy. Higher thermal constraints can endure higher power density (Figs. 7h, 7b, 7i, and 7c), hence permitting the DNNs to run on smaller arrays $\approx$ 2 mm^2 with high frequencies $\geq$ 950 MHz at 90°C, resulting in 8% energy reduction. Across all DNNs, relaxing the thermal constraint leads to an average of 21% footprint savings.

D. Optimization Results

We now demonstrate the utility of TREAD-M3D by showing that its optimizer makes meaningful choices within the vast design space ($\mathbb{D}_\mathbb{C}$) and among those selects the best configuration for the desired objective. We run TREAD-M3D at two performance constraints of 5% and 10% loss in latency and 80°C thermal constraint, to optimize for power, EDAP, EDP, and energy. Interestingly, we observe that at such a small latency relaxation, TREAD-M3D finds better configurations with up to 17% lower EDAP across all DNNs, as shown in Fig. 8. The latency slack allows DNN execution on slower and smaller systolic arrays, resulting in a better EDAP. Thus, our optimizer can be effectively used for finding efficient solutions by varying the performance constraint.

Table III lists the near-optimal choices for all nine DNNs, at 80°C and 10% loss in latency constraints. For minimizing chip power, TREAD-M3D selects small systolic arrays with footprint $<$0.5 mm^2. The optimizer selects *Partition A* for

DNN	Chip Power	System Energy	System EDP	System EDAP
DQN	22 × 20 Systolic Array (16, 16, 16) KB *Partition* $B_{bitline}$ 100 MHz	50 × 44 Systolic Array (256, 32, 16) KB *Partition A* 800 MHz	112 × 118 Systolic Array (16, 2048, 8) KB *Partition* $B_{wordline}$ 1000 MHz	38 × 34 Systolic Array (128, 16, 16) KB *Partition* $B_{wordline}$ 700 MHz
Faster R-CNN	22 × 22 Systolic Array (8, 16, 32) KB *Partition* $B_{bitline}$ 450 MHz	96 × 108 Systolic Array (512, 1024, 8) KB *Partition* $B_{wordline}$ 800 MHz	234 × 216 Systolic Array (8192, 64, 8) KB *Partition* $B_{wordline}$ 602 MHz	52 × 44 Systolic Array (256, 64, 8) KB *Partition* $B_{bitline}$ 800 MHz
GoogLeNet	20 × 22 Systolic Array (16, 16, 16) KB *Partition* $B_{bitline}$ 200 MHz	64x54 Systolic Array (256, 256, 8) KB *Partition* $B_{wordline}$ 750 MHz	236 × 212 Systolic Array (8, 8, 8192) KB *Partition* $B_{wordline}$ 602 MHz	52 × 44 Systolic Array (256, 64, 8) KB *Partition* $B_{wordline}$ 800 MHz
MobileNet	22 × 20 Systolic Array (16, 16, 16) KB *Partition* $B_{bitline}$ 150 MHz	64 × 62 Systolic Array (512, 36, 16) KB *Partition A* 950 MHz	114 × 118 Systolic Array (2048, 64, 8) KB *Partition* $B_{wordline}$ 1000 MHz	40 × 40 Systolic Array (128, 64, 16) KB *Partition* $B_{wordline}$ 1000 MHz
ResNet50	22 × 22 Systolic Array (16, 8, 32) KB *Partition A* 500 MHz	68 × 76 Systolic Array (512, 256, 8) KB *Partition A* 750 MHz	158 × 184 Systolic Array (4096, 64, 8) KB *Partition* $B_{wordline}$ 877 MHz	52 × 46 Systolic Array (256, 64, 16) KB *Partition* $B_{bitline}$ 800 MHz
Tiny-YOLO	22 × 20 Systolic Array (16, 16, 16) KB *Partition* $B_{wordline}$ 300 MHz	88 × 102 Systolic Array (1024, 256, 16) KB *Partition* $B_{bitline}$ 950 MHz	168 × 172 Systolic Array (4096, 32, 8) KB *Partition* $B_{bitline}$ 925 MHz	52 × 44 Systolic Array (128, 128, 64) KB *Partition* $B_{bitline}$ 950 MHz
VGG11	28 × 28 Systolic Array (32, 8, 64) KB *Partition A* 600 MHz	96 × 112 Systolic Array (1024, 512, 64) KB *Partition A* 1000 MHz	246 × 208 Systolic Array (8192, 128, 8) KB *Partition* $B_{wordline}$ 602 MHz	64 × 54 Systolic Array (256, 256, 8) KB *Partition* $B_{wordline}$ 950 MHz
VGG16	36 × 34 Systolic Array (64, 16, 64) KB *Partition A* 750 MHz	96 × 112 Systolic Array (1024, 512, 64) KB *Partition A* 900 MHz	248 × 216 Systolic Array (8192, 512, 8) KB *Partition* $B_{wordline}$ 600 MHz	64 × 54 Systolic Array (256, 256, 8) KB *Partition* $B_{wordline}$ 900 MHz
VGG19	34 × 36 Systolic Array (64, 16, 64) KB *Partition A* 1000 MHz	96 × 112 Systolic Array (1024, 512, 64) KB *Partition A* 1000 MHz	252 × 212 Systolic Array (8192, 512, 16) KB *Partition* $B_{wordline}$ 600 MHz	64 × 54 Systolic Array (256, 256, 8) KB *Partition* $B_{wordline}$ 850 MHz

TABLE III: TREAD-M3D near-optimal designs at 80°C: Systolic array, SRAMs (IFMAP, Filter, OFMAP), MONO3D partition, and frequency.

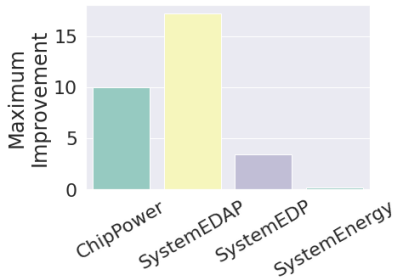


Fig. 8: %Improvement when the latency constraint is minimally relaxed from 5% to 10%. *SystemEnergy* shows negligible improvement.

VGGNets because the corresponding *Partition B* (logic-on-logic) configurations lead to thermal violations due to higher power density from high array utilization. Also, since the VGGNets are the largest out of all nine DNNs, TREAD-M3D selects higher frequencies to meet the latency constraint due to the small array size.

For minimizing energy, TREAD-M3D selects, on average, 12× larger arrays executing at 3× higher frequencies than the power-optimized arrays. The selections for VGGNets predominantly follow *Partition A* due to the thermal and power density issues discussed above. Interestingly, for EDP optimization, it selects *Partition B* for all DNNs that are, on average, 6× larger in size than the energy-optimized selections. This is because: (i) larger arrays reduce compute cycles (more active PEs), and (ii) *Partition B* supports higher operating frequencies than *Partition A* due to SRAM partition, thus resulting in lower latencies. Finally, for EDAP optimization, TREAD-M3D selects small systolic arrays operating at higher frequencies ($\geq$ 700 MHz), as shown in Table III, to minimize area, latency, and energy. Note that since a smaller footprint will have fewer PEs, it selects the higher frequencies to minimize latency.

E. Comparison to 2D

TREAD-M3D not only selects near-optimal designs for various objectives, but also enables the exploration of intricate tradeoffs, as discussed below, which is not supported by our prior work [19]. To this end, we compare the near-optimal designs in MONO3D to those in 2D, all generated by TREAD-M3D, for all nine DNNs in $\mathbb{D}_{\mathbb{C}}$. We set the constraints to 80°C and 10% loss in latency. In our analysis, 2D floorplan resembles *Partition A* with the two tiers placed laterally to form

a 2D system. We observe that power savings resulting from power-optimized near-optimal MONO3D configurations are, on average, ≈10% across all nine DNNs. While MONO3D configurations result in an average of 45% savings in interconnect power due to reduced footprint, a 12°C (average) higher temperature results in 38% greater leakage than in 2D configurations. The effective average power savings reduce to 10% because in MONO3D configurations leakage contributes to 12-18% of the total chip power, while the interconnect power contributes to only 4-10%. This also demonstrates that leakage cannot be ignored in characterizing MONO3D systems. For $EDAP$-optimized configurations, MONO3D configurations are, on average, 55% more efficient. Specifically, MONO3D configurations produce, on average, 6% lower energy, 50% footprint savings with 20% longer latencies than 2D. Since 2D configurations exhibit lower leakage than MONO3D due to small vertical thermal resistance, the 2D configurations are operating at higher frequencies, thus resulting in lower latencies compared to MONO3D.

F. Baseline Comparison

To demonstrate the importance of temperature-aware optimization for MONO3D systems, we compare TREAD-M3D to the following baselines at 80°C and 10% loss in latency: (i) *Partition A* choices only, i.e., SRAM blocks monolithically integrated on systolic array [19], [33], [36], and (ii) temperature-unaware optimization [18], [33], [65]. For the first baseline, we run TREAD-M3D for only *Partition A* (TREAD-M3D$_{PA}$) and compare its near-optimal selections to those listed in Table III. For the second baseline, we run TREAD-M3D without the thermal constraint (TREAD-M3D$_{w/o\ T}$) and compare the near-optimal configurations to those listed in the table.

Unlike some recent works [19], [33], [36] that consider only *Partition A*, TREAD-M3D incorporates performance, power, and thermal models to evaluate SRAM partitions across bitlines and wordlines to gain improvement in SRAM latency and power. Fig. 9 shows the impact of considering various partitions on different objectives. Positive bars denote that TREAD-M3D generates better configurations than TREAD-M3D$_{PA}$. TREAD-M3D outputs have lower chip power by up to 19%. These savings primarily come from 7%-20% lower SRAM power in *Partition B* configurations due to reduction in wordline/bitline capacitance. For system EDP and system EDAP, TREAD-M3D generates near-optimal configurations that are 17% and 22% more efficient, respectively. These savings primarily arise from reduction in latency and wirelength resulting from SRAM partition. However, system energy shows negligible savings because LPDDR2 energy is ≈80% of the total system energy and thus, reduction in energy coming from partition choices is insignificant. Thus, we observe that a comprehensive optimization method should consider various partition choices to optimize for objectives under the user-defined constraints.

Next, we present a comparison with thermally-unaware methodology. To generate near-optimal thermally-unaware configurations, we use our method without the thermal constraint (TREAD-M3D$_{w/o\ T}$). Fig. 10 shows the minimum and

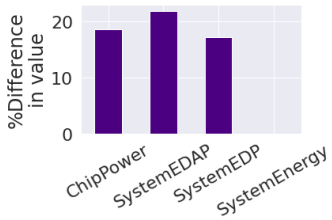


Fig. 9: TREAD-M3D$_{PA}$ versus TREAD-M3D. Positive values indicate better TREAD-M3D configurations. The *SystemEnergy* bar shows negligible difference because it is dominated by DRAM energy.

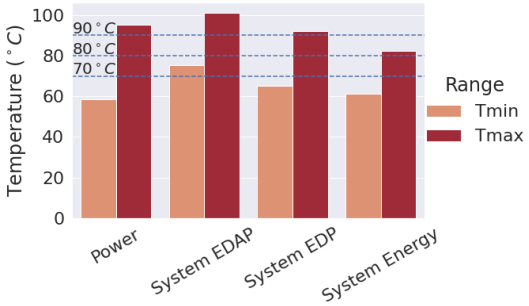


Fig. 10: Range of steady state temperatures obtained by TREAD-M3D$_{w/o\ T}$ across all DNNs. The dashed lines represent thermal constraints.

maximum steady state temperatures of near-optimal configurations generated by TREAD-M3D$_{w/o\ T}$ for various objectives across all DNNs. The figure shows that the thermally-unaware configurations can be infeasible due to thermal violations at various thermal constraints shown using dashed lines. These violations would either lead to system throttling or even shut down at very high temperatures. Thus, ignoring temperature not only may lead to sub-optimal DNN execution but also over-estimation of MONO3D benefits because in reality, the system may be throttled. We discuss this further in Fig. 11, which compares objective values between TREAD-M3D$_{w/o\ T}$ outputs and TREAD-M3D outputs in Table III at various thermal constraints. Negative bars imply over-estimation of MONO3D benefits by TREAD-M3D$_{w/o\ T}$ due to ignoring temperature. Note that both TREAD-M3D and TREAD-M3D$_{w/o\ T}$ selections result in similar system energy values. As discussed in the above section, this is primarily due to LPDDR2's contribution in system energy. Across all objectives, the over-estimation is maximum at 70°C (longest bars). As we relax the thermal constraint, the difference also reduces due to more available thermal headroom. This shows that temperature-aware optimization plays a critical role in MONO3D systems, especially at tight thermal constraints. We make some interesting observations between TREAD-M3D$_{w/o\ T}$ outputs and TREAD-M3D outputs. E.g., for minimizing chip power, TREAD-M3D$_{w/o\ T}$ selects up to 68% smaller footprint configurations because fewer PEs will consume lower power. For EDP minimization, TREAD-M3D$_{w/o\ T}$ selects larger systolic arrays operating at higher frequency levels ($\geq$ 900 MHz) since EDP emphasizes latency. For $EDAP$, TREAD-M3D$_{w/o\ T}$ selects configurations that are smaller in footprint but operating at higher

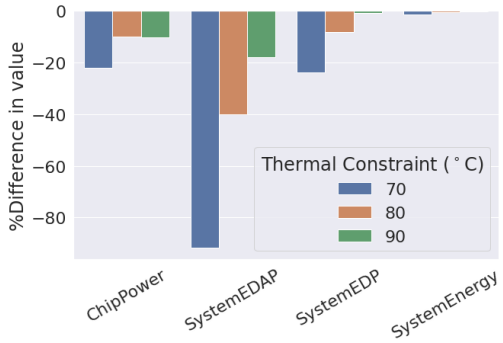


Fig. 11: TREAD-M3D$_{w/o\ T}$ versus TREAD-M3D. Negative values indicate worse TREAD-M3D configurations than TREAD-M3D$_{w/o\ T}$.

frequencies, to minimize both latency and footprint by up to 48% and 63%, respectively. However, several of the near-optimal configurations generated by TREAD-M3D$_{w/o\ T}$ lead to thermal violations, albeit small objective values, and thus, are rejected by TREAD-M3D. Thus, we demonstrate that ignoring temperature in the optimization flow not only leads to infeasible configurations due to thermal violations, but also over-estimates benefits coming from MONO3D systolic arrays.

G. Generic Accelerator for DNN inference

We extend TREAD-M3D to demonstrate that our optimizer is capable of generating a single MONO3D accelerator that can efficiently execute the 9 DNNs studied in this paper. A single accelerator may be desirable if there is a need to run several DNNs on a single mobile system. To this end, we update the objective function in our optimizer to minimize the Euclidean distance (ED) for the desired objective from the near-optimal configurations for each DNN, as shown below in Eq. (5).

$$\textit{Minimize } ED = \sqrt{\sum_{1}^{9}(OptVal_{DNNi} - ConfigVal_{DNNi})^2} \quad (5)$$

, where $OptVal_{DNNi}$ is a near-optimal objective (power, energy, EDP, or $EDAP$) of a *DNNi* and $ConfigVal_{DNNi}$ refers to the same objective when *DNNi* executes on another configuration. Using TREAD-M3D, we get a near-optimal configuration for each objective (Table IV). All of these configurations meet the 10% loss in latency and 80°C thermal constraints. Fig. 12 shows the relative degradation with respect to the near-optimal configurations at 80°C. We observe a maximum of 2.8× degradation in power minimization. This is primarily due to the small DNNs such as MobileNet or DQN. that otherwise run on small array and low frequencies to minimize power (Table III) now execute on larger arrays at higher frequencies because the former leads to thermal/performance violation for the other DNNs (e.g., VGGnets). For the other objectives, a common accelerator shows an average degradation of up to 1.4×. Note that TREAD-M3D can also be used in the absence of information on each DNN's near-optimal configuration. In this case, we can update the optimizer's implementation such that as it explores the design space, it keeps track of the best constraint-satisfying configuration explored so far for each DNN, and substitutes them into $OptVal_{DNNi}$ to calculate the objective function.

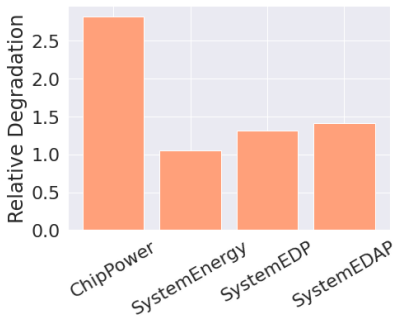


Fig. 12: Average degradation of generic accelerator architectures selected by TREAD-M3D with respect to the individual near-optimal configurations.

Objective Function	Configuration
Chip Power	44 × 52 Systolic Array (64, 128, 128) KB *Partition A* 550 MHz
System Energy	126 × 112 Systolic Array (1024, 1024, 1024) KB *Partition A* 800 MHz
System EDP	246 × 208 Systolic Array (8192, 128, 8) KB *Partition B*$_{wordline}$ 600 MHz
System EDAP	64 × 54 Systolic Array (256, 256, 8) KB *Partition B*$_{wordline}$ 800 MHz

TABLE IV: TREAD-M3D near-optimal selections for a generic accelerator at 80°C: Systolic array, SRAMs (IFMAP, Filter, OFMAP), MONO3D partition, and frequency.

VI. CONCLUSION

We propose TREAD-M3D, a method to determine thermally-safe near-optimal MONO3D configuration for desired objectives and to enable intricate tradeoff analysis in a vast design space. Compared to a baseline that uses a fiixed MONO3D integration of SRAMs on top of systolic arrays, we show, using TREAD-M3D, that other partitioning choices can provide greater benefits in certain cases. For instance, *Partition B* is more suitable for optimizing EDP and EDAP due to latency improvement arising from SRAM partition, while *Partition A* is more suitable for DNNs with high systolic array utilization for power efficiency due to logic-on-memory integration. In addition, *Partition A* configurations are thermally-safe to run DNNs with high array utilization at tight thermal constraints, e.g., 70°C, due to the limited thermal headroom, albeit ≈8% less energy efficient than *Partition B*. Our optimizer is also capable of generating configurations with 17% lower EDAP at a tolerable relaxation in inference latency, which otherwise is not straightforward to determine. Using TREAD-M3D, we also demonstrate that thermally-unaware configurations can over-estimate EDP efficiency benefits in MONO3D by up to 24% at tight thermal constraints. Finally, we extended the utility of TREAD-M3D to find a single accelerator configuration designed to run all the DNNs. Such an accelerator is on average 2.4× less efficient than the near-optimal configurations optimized for each DNN individually.

VII. Discussion and Future Work

This work aims to understand the pros and cons offered by MONO3D technology when applied to DNN systolic arrays. The reported results can help make an informed decision as to where MONO3D is the right technology for such a system. Note that the additional cost incurred by the increased processing steps may be counterbalanced by the increased area that a 2D system will require to achieve comparable performance with MONO3D. For instance, to match the low latency in MONO3D due to higher frequency resulting from SRAM partition, 2D technology may need a larger systolic array to reduce compute cycles. In this case, yield will also decrease for 2D systems, thereby reducing the cost difference between 2D and MONO3D manufacturing.

TREAD-M3D currently contains tools/models for evaluating specific partition choices for systolic arrays. It is sufficiently flexible if the user chooses different tools, provided the tools can evaluate the computational and memory attributes of DNNs executing on systolic arrays. Similarly, TREAD-M3D can be adopted to design different accelerator architectures. However, new performance and power models are needed to support different DNN acceleration architectures in MONO3D that can generate the data needed by our optimizer to converge to near-optimal points, which we leave as future work.

References

[1] X. Hu, D. Stow, and Y. Xie, "Die stacking is happening," *IEEE Micro '18*, vol. 38, no. 1, pp. 22–28, 2018.

[2] P. Batude *et al.*, "3D sequential integration: Application-driven technological achievements and guidelines," in *IEEE International Electron Devices Meeting (IEDM)*, 2017, pp. 3–1.

[3] L. Brunet *et al.*, "First demonstration of a CMOS over CMOS 3D VLSI CoolCube™ integration on 300mm wafers," in *IEEE Symposium on VLSI Tech.*, 2016, pp. 1–2.

[4] V. F. Pavlidis, I. Savidis, and E. G. Friedman, *Three-dimensional integrated circuit design*. Newnes, 2017.

[5] S. Panth, K. Samadi, Y. Du, and S. K. Lim, "Power-performance study of block-level monolithic 3D ICs considering inter-tier performance variations," in *ACM/IEEE DAC*, 2014, pp. 1–6.

[6] C. Yan and E. Salman, "Mono3D: Open source cell library for monolithic 3-D integrated circuits," *IEEE Trans. on CAS I: Regular Papers*, vol. 65, no. 3, pp. 1075–1085, 2018.

[7] P. Shukla, A. K. Coskun, V. F. Pavlidis, and E. Salman, "An overview of thermal challenges and opportunities for monolithic 3D ICs," in *Proc. of Great Lakes Symposium on VLSI*, 2019, pp. 439–444.

[8] K. Dhananjay, P. Shukla, V. F. Pavlidis, A. Coskun, and E. Salman, "Monolithic 3D integrated circuits: Recent trends and future prospects," *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2021.

[9] H. Li, M. Bhargav, P. N. Whatmough, and H.-S. P. Wong, "On-chip memory technology design space explorations for mobile deep neural network accelerators," in *ACM/IEEE DAC*, 2019, pp. 1–6.

[10] Y. S. Shao *et al.*, "Simba: Scaling deep-learning inference with multi-chip-module-based architecture," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, 2019, pp. 14–27.

[11] T. Chen *et al.*, "DianNao: A small-footprint high-throughput accelerator for ubiquitous machine-learning," *ACM SIGARCH Computer Architecture News*, vol. 42, no. 1, pp. 269–284, 2014.

[12] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE journal of solid-state circuits*, vol. 52, no. 1, pp. 127–138, 2016.

[13] P. Shukla, V. F. Pavlidis, E. Salman, and A. K. Coskun, "Temperature-Aware Monolithic 3D DNN Accelerators for Biomedical Applications," *In 2022 Design, Automation & Test in Europe Conference (DATE) Workshop on 3D Integration: Heterogeneous 3D Architectures and Sensors*, 2022, arXiv preprint arXiv:2203.15874.

[14] H.-T. Kung, "Why systolic architectures?" *Computer*, no. 1, pp. 37–46, 1982.

[15] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proc. of the Annual Int. Symp. on Computer Architecture (ISCA)*, 2017, pp. 1–12.

[16] H. T. Kung, B. McDanel, S. Q. Zhang, X. Dong, and C. C. Chen, "Maestro: A memory-on-logic architecture for coordinated parallel use of many systolic arrays," in *IEEE Int. Conf. on Application-specific Systems, Architectures and Processors (ASAP)*, vol. 2160, 2019, pp. 42–50.

[17] S. A. Panth, K. Samadi, Y. Du, and S. K. Lim, "Design and CAD methodologies for low power gate-level monolithic 3D ICs," in *Proc. of ISLPED*, 2014, pp. 171–176.

[18] K. Chang, D. Kadetotad, Y. Cao, J.-s. Seo, and S. K. Lim, "Monolithic 3D IC designs for low-power deep neural networks targeting speech recognition," in *IEEE/ACM ISLPED*, 2017, pp. 1–6.

[19] P. Shukla, S. S. Nemtzow, V. F. Pavlidis, E. Salman, and A. K. Coskun, "Temperature-aware optimization of monolithic 3D deep neural network accelerators," in *IEEE Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2021, pp. 709–714.

[20] C.-J. Wu *et al.*, "Machine learning at facebook: Understanding inference at the edge," in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2019, pp. 331–344.

[21] B. Asgari, R. Hadidi, H. Kim, and S. Yalamanchili, "ERIDANUS: Efficiently running inference of DNNs using systolic arrays," *IEEE Micro '19*, vol. 39, no. 5, pp. 46–54, 2019.

[22] ——, "Lodestar: Creating locally-dense CNNs for efficient inference on systolic arrays," in *IEEE/ACM Design Automation Conference (DAC'19)*, 2019, pp. 1–2.

[23] Z.-G. Liu, P. N. Whatmough, and M. Mattina, "Systolic Tensor Array: An efficient structured-sparse GEMM accelerator for mobile CNN inference," *IEEE Computer Architecture Letters*, vol. 19, no. 1, pp. 34–37, 2020.

[24] M. Zhu, T. Zhang, Z. Gu, and Y. Xie, "Sparse tensor core: Algorithm and hardware co-design for vector-wise sparse neural networks on modern gpus," in *IEEE/ACM Proc. of International Symposium on Microarchitecture*, 2019, pp. 359–371.

[25] A. Ren *et al.*, "Admm-nn: An algorithm-hardware co-design framework of dnns using alternating direction methods of multipliers," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 925–938.

[26] P. Shukla, D. Aguren, T. Burd, A. K. Coskun, and J. Kalamatianos, "Temperature-aware sizing of multi-chip module accelerators for multi-dnn workloads," in *IEEE DATE*, 2023, pp. 1–6.

[27] S. K. Samal, K. Samadi, P. Kamal, Y. Du, and S. K. Lim, "Full chip impact study of power delivery network designs in monolithic 3D ICs," in *IEEE/ACM Proc. of International Conference on Computer-Aided Design (ICCAD)*, 2014, pp. 565–572.

[28] H. Wei *et al.*, "Cooling three-dimensional integrated circuits using power delivery networks," in *IEEE International Electron Devices Meeting*, 2012, pp. 14–2.

[29] M. A. Iqbal and M. Rahman, "New thermal management approach for transistor-level 3-d integtration," in *IEEE Proc. of SOI-3D-Subthreshold Microelectronics Technology Unified Conference (S3S)*, 2017, pp. 1–3.

[30] J. H. Lee *et al.*, "Characterizing the thermal feasibility of monolithic 3d microprocessors," *IEEE Access*, vol. 9, pp. 120 715–120 729, 2021.

[31] S. K. Samal *et al.*, "Adaptive regression-based thermal modeling and optimization for monolithic 3-d ics," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 10, pp. 1707–1720, 2016.

[32] F. Chen, L. Song, H. Li, and Y. Chen, "Marvel: A vertical resistive accelerator for low-power deep learning inference in monolithic 3d," in *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2021, pp. 1240–1245.

[33] Y. Yu and N. K. Jha, "Spring: A sparsity-aware reduced-precision monolithic 3D CNN accelerator architecture for training and inference," *IEEE Transactions on Emerging Topics in Computing (TETC)*, 2020.

[34] J. M. Joseph *et al.*, "Architecture, dataflow and physical design implications of 3d-ics for dnn-accelerators," in *2021 22nd Int. Symp. on Quality Electronic Design (ISQED)*. IEEE, 2021, pp. 60–66.

[35] S. K. Samal *et al.*, "Fast and accurate thermal modeling and optimization for monolithic 3D ICs," in *ACM/IEEE Design Automation Conference (DAC)*, 2014, pp. 1–6.

[36] R. Mathur, A. K. A. Kumar, L. John, and J. P. Kulkarni, "Thermal-aware design space exploration of 3-d systolic ml accelerators," *IEEE Journal*

on Exploratory Solid-State Computational Devices and Circuits, vol. 7, no. 1, pp. 70–78, 2021.
[37] S. Panth, K. Samadi, Y. Du, and S. K. Lim, "Shrunk-2-d: A physical design methodology to build commercial-quality monolithic 3-d ics," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 10, pp. 1716–1724, 2017.
[38] A. Guler and N. K. Jha, "Mcpat-monolithic: An area/power/timing architecture modeling framework for 3-d hybrid monolithic multicore systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 10, pp. 2146–2156, 2020.
[39] P. Batude *et al.*, "GeOI and SOI 3D monolithic cell integrations for high density applications," in *2009 Symposium on VLSI Technology*. IEEE, 2009, pp. 166–167.
[40] A. Guler and N. K. Jha, "Hybrid monolithic 3D IC floorplanner," *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, vol. 26, no. 10, pp. 1868–1880, 2018.
[41] S. Thoziyoor, N. Muralimanohar, J. Ahn, and N. Jouppi, "CACTI 6.5," *hpl.hp.com*, 2009.
[42] Z. Zhang, X. Si, S. Srinivasa, A. K. Ramanathan, and M.-F. Chang, "Recent advances in compute-in-memory support for sram using monolithic 3-d integration," *IEEE Micro*, vol. 39, no. 6, pp. 28–37, 2019.
[43] A. Stillmaker and B. Baas, "Scaling equations for the accurate prediction of CMOS device performance from 180 nm to 7 nm," *Integration*, vol. 58, pp. 74–81, 2017.
[44] A. Samajdar *et al.*, "A systematic methodology for characterizing scalability of DNN accelerators using scale-sim," in *IEEE Int. Symp. on Performance Analysis of Systems and Software (ISPASS)*, 2020, pp. 58–68.
[45] Y.-F. Tsai, F. Wang, Y. Xie, N. Vijaykrishnan, and M. J. Irwin, "Design space exploration for 3D cache," *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, vol. 16, no. 4, pp. 444–455, 2008.
[46] Y.-H. Gong, J. Kong, and S. W. Chung, "Quantifying the impact of monolithic 3D (M3D) integration on L1 caches," *IEEE TETC*, 2019.
[47] "Predictive technology model," http://ptm.asu.edu/.
[48] D. H. Kim, R. O. Topaloglu, and S. K. Lim, "Block-level 3D IC design with through-silicon-via planning," in *IEEE ASP-DAC*, 2012, pp. 335–340.
[49] C. Sitik, W. Liu, B. Taskin, and E. Salman, "Design methodology for voltage-scaled clock distribution networks," *IEEE TVLSI*, vol. 24, no. 10, 2016.
[50] I. Ciofi *et al.*, "Impact of wire geometry on interconnect rc and circuit delay," *IEEE Trans. on Electron Devices*, vol. 63, no. 6, pp. 2488–2496, 2016.
[51] Y. Liu, R. P. Dick, L. Shang, and H. Yang, "Accurate temperature-dependent integrated circuit leakage power estimation is easy," in *IEEE Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2007, pp. 1–6.
[52] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-aware microarchitecture," *ACM SIGARCH Computer Architecture News*, vol. 31, no. 2, pp. 2–13, 2003.
[53] W. Huang *et al.*, "HotSpot: A compact thermal modeling methodology for early-stage VLSI design," *IEEE TVLSI*, vol. 14, no. 5, pp. 501–513, 2006.
[54] V. J. Reddi *et al.*, "Mlperf inference benchmark," in *ACM/IEEE ISCA*, 2020, pp. 446–459.
[55] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *NIPS*, 2015, pp. 91–99.
[56] S. Krishnan, B. Borojerdian, W. Fu, A. Faust, and V. J. Reddi, "Air learning: An AI research platform for algorithm-hardware benchmarking of autonomous aerial robots," *arXiv preprint arXiv:1906.00421*, 2019.
[57] "Micron Technology Inc. Embedded LPDDR2 SDRAM," https://www.micron.com/products/dram/lpdram/part-catalog/edb8132b4pb-8d-f.
[58] K. T. Malladi *et al.*, "Towards energy-proportional datacenter memory with mobile dram," in *2012 39th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2012, pp. 37–48.
[59] R. Z. Ayoub and T. S. Rosing, "Predict and act: dynamic thermal management for multi-core processors," in *Proceedings of the 2009 ACM/IEEE international symposium on Low power electronics and design*, 2009, pp. 99–104.
[60] W. He *et al.*, "Optimal thermal management of server cooling system based cooling tower under different ambient temperatures," *Applied Thermal Engineering*, vol. 207, p. 118176, 2022.
[61] H. B. Jang *et al.*, "Exploiting application/system-dependent ambient temperature for accurate microarchitectural simulation," *IEEE Transactions on Computers*, vol. 62, no. 4, pp. 705–715, 2012.
[62] S. Bianco, R. Cadene, L. Celona, and P. Napoletano, "Benchmark analysis of representative deep neural network architectures," *IEEE Access*, vol. 6, pp. 64 270–64 277, 2018.
[63] A. Prakash, H. Amrouch, M. Shafique, T. Mitra, and J. Henkel, "Improving mobile gaming performance through cooperative cpu-gpu thermal management," in *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 2016, pp. 1–6.
[64] O. Sahin, L. Thiele, and A. K. Coskun, "Maestro: Autonomous qos management for mobile applications under thermal constraints," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 8, pp. 1557–1570, 2018.
[65] M. Gao, J. Pu, X. Yang, M. Horowitz, and C. Kozyrakis, "Tetris: Scalable and efficient neural network acceleration with 3D memory," in *Proc. of International Conference on Architectural Support for Programming Languages and Operating Systems*, 2017, pp. 751–764.

Prachi Shukla (M'20) received her B.S. degree in information systems from BITS-Pilani Goa, India, in 2012, and the M.S. and Ph.D degrees in computer engineering from Columbia University, NY, USA, and Boston University, MA, USA, in 2015 and 2023, respectively. She is currently an MTS Silicon Design Engineer at Advanced Micro Devices. Her current research interests include computer architecture, energy-efficient computing, 3D-integrated DNN systems, and EDA tool development.

Ayse K. Coskun (M'06–SM'16) received the M.S. and Ph.D. degrees in computer science and engineering from the University of California at San Diego, CA, USA. She is a Professor with the Electrical and Computer Engineering Department, Boston University, MA, USA. Her research interests include energy and temperature awareness in computing systems, novel computer architectures, and management of cloud and HPC data centers. Prof. Coskun was a recipient of the IEEE CEDA Early Career Award. She currently serves as a Deputy Editor-in-Chief for the IEEE Transactions on CAD and previously served as an Associate Editor for IEEE Transactions on Computers.

Emre Salman (S'03-M'10-SM'17) received the M.S. and Ph.D. degrees in electrical engineering from the University of Rochester, NY, USA, in 2006 and 2009, respectively. He is an Associate Professor with the Electrical and Computer Engineering Department, Stony Brook University (SUNY), NY, USA. His research interests include analysis, modeling, and design methodologies for integrated circuits and VLSI systems with applications to low power and secure computing, and implantable devices. He currently serves as the America's Regional Editor for the Journal of Circuits, Systems and Computers and on the Editorial Board of IEEE Transactions on Emerging Topics in Computing. He previously served as the Chair for the VLSI Systems and Applications Technical Committee (VSA-TC) of the IEEE Circuits and Systems Society.

Vasilis F. Pavlidis (M'03-SM'18) holds a Ph.D. and M.Sc. degree in Electrical and Computer Engineering from the University of Rochester, NY, USA. He has been an Associate Professor with the Advanced Processor Technologies Group, Department of Computer Science at the University of Manchester, Manchester, U.K. He servers as an Associate Editor for the editorial board of the IEEE Transactions on VLSI and IEEE Transactions on CAD. He has also served in the Ex Com of the IEEE Council on EDA. His current research interests include the area of interconnect modeling and design, 2.5D/3D integration, spintronics, and related design issues in VLSI. He is the leading author of the title Three Dimensional Integrated Circuit Design, 1st and 2nd edition.