# Temperature-Aware Optimization of Monolithic 3D Deep Neural Network Accelerators

# Temperature-Aware Optimization of Monolithic 3D Deep Neural Network Accelerators

## ABSTRACT

We propose a design automation methodology to help design of energy-efficient Mono3D DNN accelerators with safe on-chip temperatures for mobile systems. We introduce an optimizer capable of investigating the impact of different aspect ratios of the chip and chip footprint specifications, and selecting energy-efficient accelerators under user-specified thermal and performance constraints. We also demonstrate that using our optimizer we can reduce energy consumption by 1.6× and area by 2× with a maximum of 9.5% increase in latency compared to a Mono3D DNN accelerator optimized only for performance.

## 1 INTRODUCTION

Deep Neural Networks (DNNs) are extremely popular for numerous machine learning applications, such as image classification or object detection [1]. There is an increasing demand for DNNs in mobile systems, such as IoT devices, autonomous drones, tablets, etc. To satisfy the performance demands of these devices, accelerators for DNNs are actively being developed [2]. However, the high energy demand of DNNs (due to their heavy computation and data movement) is a major design issue. In addition, mobile systems have tight area and power/thermal budgets (e.g., due to the absence of heat sinks and fans) that add to the constraints associated with designing energy-efficient mobile DNN accelerators.

A systolic array-based DNN accelerator comprises a two dimensional (2D) array of simple processing elements (PEs), with on-chip scratchpad memories for input feature map (IFMAP), filter weights (Filter), and output feature map (OFMAP), as shown in Fig. 1 [3]. Each PE consists of a Multiply-and-Accumulate (MAC) unit along with internal registers to store the inputs and partial sums. In a systolic architecture, data flows into the array from the PEs along the top and left edges in Fig. 1 and is passed onto their neighboring PEs every clock cycle. This data flow is uni-directional. Straightforward design and high compute density make systolic arrays a popular choice for DNN accelerators in mobile systems [2].

With technology scaling slowing down, improving performance under energy, power, and thermal constraints is increasingly more challenging. Monolithic 3D (Mono3D) is a three-dimensional (3D) integration technology that can overcome 2D scaling bottlenecks by achieving small chip footprint, dense integration, wire length savings, power savings, and high bandwidth [4]. These properties make Mono3D attractive for designing DNN accelerators in mobile systems. However, 3D architectures have significant thermal challenges due to high power densities and vertical thermal resistance [5]. In addition, Mono3D systems have thin device layers, which result in limited lateral heat flow and high inter-tier thermal coupling (unlike through silicon via based 3D stacking), thus exacerbating thermal problems in mobile systems [6]. Consequently, temperature becomes an indispensable part of the methodologies and tools used to architect Mono3D systems.

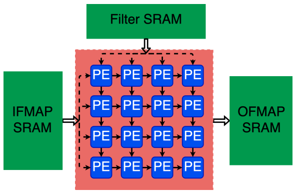


**Figure 1: 4×4 systolic array.**

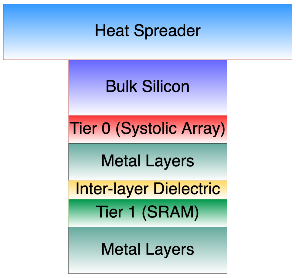


**Figure 2: Mono3D stack.**

This work focuses on designing energy-efficient architectures based on systolic arrays fabricated with Mono3D technologies, enabling DNN inference tasks in mobile systems. The primary contributions of this paper are as follows:

- We develop a design automation methodology to investigate the performance, power, temperature, and energy trends in Mono3D DNN accelerators for a wide range of DNNs commonly used for mobile inference tasks.
- We integrate a DNN performance model and Mono3D power and thermal models to construct a comprehensive optimization flow. We also provide validation for our Mono3D thermal model.
- A performance versus temperature tradeoff exploration with our optimizer reports up to 2× and 1.6× savings in chip footprint and energy, respectively, while staying within the thermal budget compared to a Mono3D DNN accelerator that is only performance-optimized, while increasing latency by 9.5%.

## 2 RELATED WORK

**DNN accelerators.** Energy efficiency is a major design objective for DNN accelerators. Recent works target energy efficiency in systolic array-based accelerators by adjusting DRAM design parameters, such as supply voltage and access latency [7], replacing off-chip DRAM with non volatile memories [2], or designing dataflow mechanisms to improve data re-use and reduce SRAM accesses [8]. Prior works have also focused on co-designing DNN models and their corresponding hardware accelerators (e.g., [9]). These works focus on 2D accelerators without considering temperature. Another work achieves DNN energy efficiency and latency improvement by stacking memory-on-logic using through-silicon vias (TSVs) [10].

**Mono3D.** Mono3D is an emerging 3D integration technology where multiple tiers (or device layers) are fabricated sequentially, separated by thin dielectrics, even though current Mono3D fabrication challenges limit the number of tiers to two [4]. The vertical connections between the tiers are achieved using nano-scale intertier vias (MIVs) [4]. The thin tiers and MIVs can overcome 2D scaling limitations and provide greater interconnect density, wire length reduction, power savings, and denser integration than traditional TSV-based 3D ICs. There are three types of partitions possible in Mono3D: block-, gate-, and transistor-level. While there are several works in gate- and transistor-level partition [11–13], we focus on a

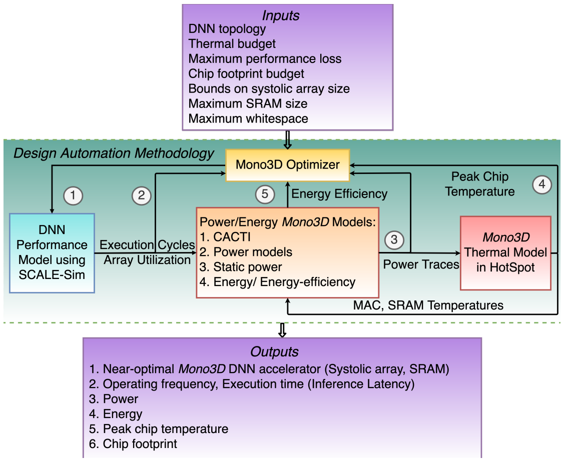


**Figure 3: Flow diagram of the optimization process.**

two-tier block-level partition in this paper, in which 2D IP blocks can be used in the design process of Mono3D.

Prior works have focused on designing DNN accelerators in Mono3D but have not considered thermal awareness [14–17]. Yu *et al.* design a block-level Mono3D architecture with an FPGA-based accelerator and several resistive RAM tiers to improve performance, power, and energy compared to a 2D baseline with an off-chip DRAM [14, 15]. Chang *et al.* implement a single accelerator (with MAC units and SRAMs) for two DNN models with different weight compression on a two-tier Mono3D system [16]. They use block-level partition techniques and show up to 22.3% iso-performance power savings. Do *et al.* integrate a two-tier Mono3D scratchpad memory on a GPU and provide 46% performance improvement [17]. In contrast, our work proposes a methodology to perform a comprehensive architecture-level performance, power, and temperature analysis for various DNNs and underlying architecture parameters to determine the most energy-efficient accelerator while also satisfying the thermal and performance constraints.

**Key Innovation.** To the best of our knowledge, this is the first work that offers a temperature-aware analysis and optimization framework for DNN accelerators based on Mono3D technology. The proposed framework enables navigating performance versus temperature tradeoffs for systolic array-based Mono3D DNN accelerators aiming mobile systems.

## 3 DESIGN AUTOMATION METHODOLOGY FOR MONO3D DNN ACCELERATORS

This section describes our proposed design automation methodology for optimization of Mono3D DNN accelerators in mobile systems. As shown in Fig. 3, our methodology takes a DNN topology (such as MobileNet or VGG11 [18]) and design constraints as inputs to a Mono3D optimizer that determines design parameters for the accelerators for the subsequent iterations and finally outputs a near-optimal accelerator with safe chip temperatures. This optimization flow starts with performance evaluation using SCALE-Sim, a cycle-accurate simulator for systolic array-based DNN accelerators [19]. SCALE-Sim outputs, along with CACTI-6.5 [20] and Mono3D power models, are then used to generate power traces for the accelerator. CACTI calculates power, energy, access time, and area of SRAMs. We then use HotSpot v6.0 (which we configure to simulate Mono3D systems) to obtain on-chip temperatures at steady state [21]. In addition, inter-tier thermal coupling can affect the temperature-dependent static power, which further influences peak temperature and energy. Therefore, we implement a feedback loop that updates the power traces with the static power, after which HotSpot reruns to obtain updated chip temperatures. This loop continues until the temperature converges.

### 3.1 Mono3D DNN Accelerator Design

To simulate a realistic Mono3D stack, we have limited our design to two tiers (see Fig. 2 for a cross-sectional view) because existing Mono3D technologies can typically support only two tiers due to the low temperature requirements during the fabrication process of upper tiers [4]. The number of metal layers, dielectric/device layer thickness, and other material properties of the stack are taken from recent work [4, 12]. The systolic array has a higher power consumption than SRAMs and is placed on the tier closer to the heat spreader. The systolic array and SRAMs have a high degree of connectivity through the MIVs since there are many read/write accesses to the SRAMs throughout the computations in the systolic array. We assume a high logic density for the tier with the systolic array, with SRAMs of the appropriate size on the other tier. Any whitespace (as a result of area mismatch between the two tiers) always appears on the SRAM tier in our design. We place whitespaces along chip edges so that thermal analyses are not affected.

### 3.2 Mono3D Optimizer

We construct a multi-start simulated annealing (MSA) based optimizer to systematically sweep a sufficient portion of the design space of accelerators and select near-optimal energy-efficient Mono3D architectures for mobile systems. MSA is a probabilistic algorithm that accepts solutions that temporarily degrade the optimization goal to escape from local minima. MSA can launch multiple "start"s in parallel to increase the probability of finding the global minima. As shown in Fig. 3, our optimizer takes a DNN topology and the following design constraints as inputs: (i) chip footprint budget; (ii) bounds on chip aspect ratio; (iii) limits on systolic array size, (iv) maximum SRAM size, (v) maximum allowed whitespace (as a result of mismatch between the two tiers in the Mono3D chip), (vi) thermal budget (i.e., maximum allowed peak temperature, $T_{threshold}$), and (vii) maximum performance loss ($C_{loss,max}$) w.r.t. the fastest design that satisfies the design constraints (i)-(vi). The optimizer generates performance, power, and thermal traces for systematically selected Mono3D accelerators, and converges to a near-optimal design for the user-specified optimization goal (e.g., minimizing energy or latency) while satisfying performance and thermal constraints. For the systematic selection of new design candidates, the optimizer uses the operating frequency, chip's aspect ratio, and combinations of systolic array and SRAMs (that satisfy the whitespace constraint) as its control knobs.

Algorithm 1 details our optimizer, which is inherently parallelizable because all the "start"s run in parallel (line 1). Each start is assigned an operating frequency and an aspect ratio range (*AR*), within which the optimizer determines a near-optimal solution by minimizing the objective function, $Obj$. $Obj$ can be execution time (i.e., inference latency), chip power, energy or another energy efficiency metric. $T_{start}$, $T_{finish}$, and decay ($\delta$) are parameters of the optimizer that define the annealing temperatures and the rate

**Algorithm 1:** MSA-based Temperature-Aware Optimizer

```
Input      : DNN, AR range, footprint budget, systolic array range,
             $T_{threshold}$, $C_{loss,max}$, frequencies
Output     : $S_{best}$ with minimum $Obj$
Initialize : $T_{start}$, $T_{finish}$, $N$, decay rate $\delta$, $multi\_starts$, $T$, AR range for
             each start $i$ ($AR_i$), $T_{peak}$ for each start $i$ ($T_{peak,i}$)
1  while $multi\_starts > 0$ do
2      $T_{peak,i} \leftarrow T_{threshold} + 1$
3      while $T_{peak,i} > T_{threshold}$ do
4          randomly select an accelerator ($S_i$) with $AR_i$ and a frequency
5          if $S_i$ meets design constraints (i)-(v) in Sec. 3.2 then
6              generate performance traces, calculate execution time $C_i$
7              generate power traces, estimate peak temperature ($T_{peak,i}$)
8      set current solution: $S_{curr} \leftarrow S_i$, $C_{curr} \leftarrow C_i$, $T_{peak,curr} \leftarrow T_{peak,i}$
9      calculate $Obj_{curr}$
10     initialize best performance, $C_{best} \leftarrow C_{curr}$
11     while $T > T_{finish}$ do
12         while $N > 0$ do
13             randomly perturb the systolic array design, $S_p$, belonging to
                 the neighborhood of $S_{curr}$ with $AR_i$
14             $N$ -= 1
15             if $S_i$ meets design constraints (i)-(v) in Sec. 3.2 then
16                 calculate $C_p$ and loss in performance $C_{loss} = \frac{C_p - C_{curr}}{C_{curr}}$
17                 initialize status ← ‘Reject’
18                 if $C_{loss} \leq C_{loss,max}$ then
19                     status ← ‘Accept’
20                 if status = ‘Accept’ then
21                     status ← ‘Reject’
22                     generate power traces and calculate $T_{peak,\,p}$
23                     calculate $Obj_p$
24                     if $T_{peak,p} \leq T_{threshold}$ then
25                         $\Delta Obj_{avg} = abs(Obj_p - Obj_{curr})$
26                         if $Obj_p \leq Obj_{curr}$ then
27                             status ← ‘Accept’
28                         else if $Obj_p > Obj_{curr}$ then
29                             if $random(0,1) < exp(-\frac{\Delta Obj}{\Delta Obj_{avg} * T})$
                                 then
30                                 status ← ‘Accept’
31                     if status = ‘Accept’ then
32                         $S_{curr} \leftarrow S_p$ , $C_{curr} \leftarrow C_p$ , $Obj_{curr} \leftarrow Obj_p$
33                         update $\Delta Obj_{avg}$
34                         if $C_p < C_{curr}$ then
35                             $C_{best} \leftarrow C_{curr}$
36             Store $S_{curr}$, $C_{curr}$, $Obj_{curr}$, $T_{peak,curr}$ in a data structure
37         $T \leftarrow T * \delta$
38     $multi\_starts$ -= 1
39 return $S_{best}$ s.t. $T_{peak} \leq T_{threshold}$, & $C_{loss} \leq C_{loss,max}$
```

of cooling[1]. Each start begins by randomly choosing an initial accelerator ($S_i$) that satisfies the design constraints (i)-(vi) listed above (lines 3-7). We set $S_i$ as the current solution ($S_{curr}$) and initialize the execution time ($C_{curr}$), fastest execution time ($C_{best}$), peak temperature ($T_{peak,curr}$), and $Obj_{curr}$ with $S_i$'s parameters (lines 8-10). We then randomly perturb $S_{curr}$ to a nearby (neighboring) and feasible design ($S_p$) based on chip footprint (lines 11-15). If the execution time of the DNN on the perturbed design ($C_p$) is smaller or within a user-specified performance degradation ($C_{loss,max}$) from $C_{best}$, this design (i.e., $S_p$) is ‘accepted’ for the next step. Otherwise, it is ‘rejected’ (lines 16-19). The ‘accepted’ $S_p$ is then thermally simulated for steady-state analysis. If the peak chip temperature ($T_{peak,p}$) is greater than the thermal budget ($T_{threshold}$), $S_p$ is rejected. Otherwise, $S_p$ is checked for a lower $Obj_p$. If $Obj_p$ is smaller than $Obj_{curr}$, then $S_p$ is ‘accepted’. Otherwise, it is ‘accepted’ with a certain probability (lines 20-30). The term $\Delta Obj$ is the difference between $Obj_p$ and $Obj_{curr}$, while $\Delta Obj_{avg}$ refers to the running average of $\Delta Obj$ for the accepted designs. The ‘accepted’ $S_p$ is then set as $S_{curr}$ for the next iteration (lines 31-35).

The algorithm terminates when the termination conditions are satisfied (lines 1, 11). The accepted designs are stored in a data structure. Finally, the optimizer selects the best design among all the starts with the least $Obj$ while satisfying the performance and thermal constraints (line 38). Note that if the objective of the user is to design one accelerator that can run multiple DNNs efficiently, then some additional meta strategies could be integrated to the optimizer. For example, the optimizer can select the fastest/most efficient design out of several optimized solutions for all target DNNs on average, or pick the design that yields the best results for the most frequently run DNNs.

## 3.3 Performance Model

SCALE-Sim is an open-source, state-of-the-art cycle-accurate simulator for DNN accelerators that operate on 8-bit integer data. It takes the size of systolic array and scratchpad memories and DRAM bandwidth as inputs, simulates a stall-free DNN inference, and outputs compute cycles, non-overlapping DRAM cycles, array utilization, SRAM accesses, and SRAM bandwidth to support stall-free inference. Compute cycles include cycles spent in data transfer between SRAMs and systolic array, along with DRAM cycles that overlap with the computation. We divide the compute cycles and non-overlapping cycles by chip and DRAM frequencies, respectively, to calculate the latency. Among the several dataflows SCALE-Sim supports, we use output stationary as it has been shown to outperform the other dataflows [19].

## 3.4 Mono3D Power Models

We use SCALE-Sim outputs to obtain the average dynamic power of the systolic array ($P_{SA,Dynamic}$) using Eqs. (1) and (2):

$$U_{av} = (\sum_{i=1}^{N} U_i * C_i)/(\sum_{i=1}^{N} C_i), \tag{1}$$

$$P_{SA,Dynamic} = U_{av} * P_{MAC,Dynamic} \ , \tag{2}$$

where $N$ is the total number of convolutional layers in the DNN, $U_i$ and $C_i$ are the utilization and compute cycles, respectively, for the $i^{th}$ layer, and $P_{MAC,Dynamic}$ is the dynamic power for a MAC unit. We also integrate an exponential leakage model for MAC (see Sec. 4.1.1 for details on MAC's power model).

We use the minimum SRAM bandwidth (*bytes per cycle*) generated by SCALE-Sim to decide the number of banks in SRAM. We use CACTI to calculate the SRAM dynamic power and leakage. To estimate SRAM leakage at a finer granularity than the 10 degree default granularity of CACTI, we fit a linear model (a linear model can accurately estimate leakage across close temperatures [22]).

We deploy a generic interconnect power model, where the interconnects consume 15% of the total chip dynamic power because (i) DNNs require large amounts of memory for inputs, weights, and outputs, and (ii) there is frequent data movement between the systolic array and SRAMs [23]. We then reduce the interconnect

[1] Annealing temperature is a unitless parameter in MSA that allows it to escape a local minima by accepting a design with a higher $Obj$ value. Rate of cooling is the rate at which the annealing temperature decays to achieve convergence.

| | |
|---|---|
| Systolic array size | 16×16 to 256×256 |
| Each SRAM size | {32, 64, 128, 256, 512, 1024, 2048, 4096} *KB* |
| Aspect ratio of the chip | 0.7 to 1.3 |
| Frequencies | {735, 600, 500} *MHz* |

**Table 1: Design space for DNN accelerators.**

power by 10%, which is equal to Mono3D iso-performance power savings obtained from a recent work [24]. The interconnect power is then uniformly distributed across the metal layers.

We evaluate energy efficiency for the accelerators using the following metrics: system energy ($E_{sys}$, includes both the chip and DRAM energy), energy-delay-area-product (*EDAP*), energy-delay$^2$-product (*ED2P*), and energy-delay-product (*EDP*). While *EDP* and *ED2P* emphasize the execution time, *EDAP* offers a comparison across accelerators with different chip footprints.

## 3.5 Mono3D Thermal Model

We build a compact thermal model (CTM) in HotSpot for the chip-stack shown in Fig. 2. The CTM has 32 layers (including tiers, metal layers, etc.). We use HotSpot's default ambient setting, i.e. 45 $^{\circ}C$, and *grid mode* (grid length = MAC length) to conduct steady-state thermal simulations with the power traces generated using CACTI and Mono3D power models. To model a mobile system, we set the heat spreader thickness to 50 $\mu m$ and remove the heat sink by setting its thickness to a negligible value. After every HotSpot run, we read the SRAM and MAC temperatures and update the power traces with their temperature-dependent leakage, and rerun HotSpot. This feedback loop continues till the difference is $< 1^{\circ}C$ for both MAC and SRAM temperatures between consecutive HotSpot runs.

We validate our CTM with a model for the same design in COMSOL, a multiphysics simulator that uses finite element method to solve a second order heat diffusion equation [25]. We model various aspect ratios, hot spot locations, sizes, and power densities. Overall, we observe a maximum error in peak temperature of 3.89% w.r.t. COMSOL. We also report average, maximum and RMS errors of $1.53^{\circ}C$, 4°C (corresponds to 3.2% w.r.t. COMSOL), and $1.76^{\circ}C$, respectively. We also observe that power profiles resembling our Mono3D setup show a maximum error of 1° C (corresponds to 1.3%) for peak temperatures close to $80^{\circ}C$ ($T_{threshold}$ in our analysis).

# 4 EXPERIMENTAL RESULTS

In this section, we describe our experimental setup, evaluate our optimizer for correctness and speed, and present the results of our optimization flow. For our analyses, we have used eight DNN inference benchmarks, six from MLPerf [18], namely VGG19, VGG16, VGG11, ResNet50, MobileNet, and GoogLeNet, along with Faster R-CNN [26] and Tiny-YOLO [27]. We group MobileNet, GoogLeNet, ResNet50, and Tiny-YOLO as *lower-complexity (LoC) DNNs* because of their lower memory usage and fewer number of MAC operations and the rest as *higher-complexity (HiC) DNNs* because of their greater number of MAC operations and higher memory usage [28].

## 4.1 Experimental setup

*4.1.1 SRAM/Systolic Array MAC model.* We synthesize a 65 *nm* 8-bit MAC unit at 250 *MHz* using the Synopsys Design Compiler (DC) and scale it down to 22 *nm* technology node. The scaled down area, dynamic power, and the frequency are 121 $\mu m^2$ (length = 11 $\mu m$), 0.25 $mW$, and 735 $MHz$, respectively. We also fit a temperature-dependent exponential leakage model for a MAC unit using data points (temperature, leakage) from our synthesized MAC model. Furthermore, we model 22 *nm* SRAMs in CACTI-6.5 and the off-chip DRAM is based on 8 *Gb* LPDDR2-800 x32 chips at 400 *MHz*, with 8.5 *GBps* bandwidth and 200 $pJ/byte$ energy consumption [29].

*4.1.2 Constraints and Design Space.* We set our chip footprint budget to 8 $mm^2$, desired systolic array size between 16×16 [2] and 256×256 (similar to Google's Tensor Flow Processor), total allowed SRAM size to 24 *MB*, thermal budget to $80^{\circ}C$, and the maximum whitespace allowed to 1% of the chip footprint. In addition to the chip frequency of 735 *MHz*, we include 600 *MHz* and 500 *MHz* in our search space. We set a constraint on maximum performance loss of $\leq$ 10% w.r.t. the design with the lowest latency under the given constraints. Note that this is a user-defined parameter and can change as required. Each SRAM has 4 banks and provides a bandwidth of 256 *bytes per cycle* to match with the minimum SRAM bandwidth for the given systolic array bounds as output by SCALE-Sim. Table 1 shows the total design space for DNN accelerators, i.e., 24.6$k$ (3 frequencies × 8.2$k$ accelerators) design points.

## 4.2 Optimizer Evaluation

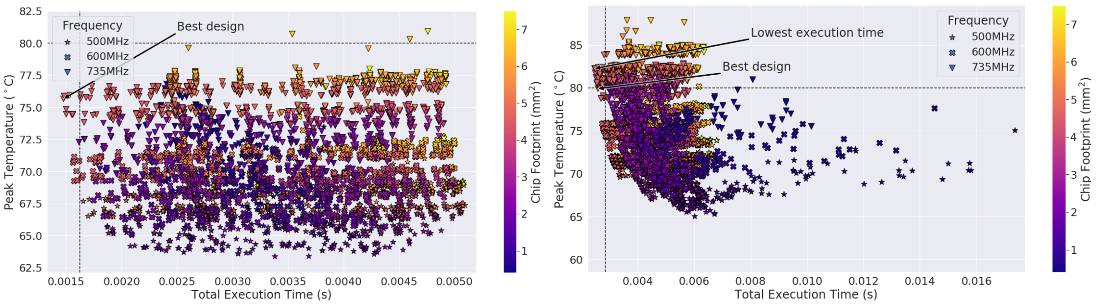
*4.2.1 Setup and Running Times.* We launch 6 starts for each frequency and each start is assigned an aspect ratio range. Each start has 6 annealing temperatures with 35 perturbations. We ensure convergence by observing that the optimizer does not accept worse designs as it approaches termination. We run the optimizer multiple times to tune its parameters, i.e., $T_{start}, T_{finish}$ and $\delta$, to achieve better solutions. Furthermore, our optimizer can work with a larger range of frequencies and still select a near-optimal point (this may require launching more starts in parallel).

SCALE-Sim and HotSpot take 10-60 and 5-45 mins, respectively, depending on the chip footprint and DNN. HiC DNNs have a higher number of MAC operations that lead to higher power densities and peak temperatures (more active PEs), which increase temperature-dependent leakage. Thus, these DNNs require more iterations (4-5) to converge in HotSpot. LoC DNNs require fewer iterations (2-3) due to fewer MAC operations [28] 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.

*4.2.2 Correctness of the Optimizer.* To demonstrate the correctness of our optimizer, we select a smaller design space with one frequency (735 *MHz*), 0.94 to 1 aspect ratio range (step size of 0.01), under the same constraints listed in Sec. 4.1.2. We evaluate the optimizer with 10%, 5%, and 3% performance constraints. In total, there are 1,196 valid accelerator configurations. We select 2 DNNs, Tiny-YOLO and VGG11, and compare the designs chosen by our optimizer to those determined by an exhaustive search in this smaller design space. The optimizer's parameters $\{T_{start}, T_{finish}, \delta\}$ for Tiny-YOLO and VGG11 are set to {1.446, 0.738611, 0.8} and {1.446, 0.885963, 0.85}, respectively. The 6 starts are assigned aspect ratio ranges: [0.94, 0.95], (0.95, 0.96], and so on till (0.99, 1]. Across all the objectives (performance, power, energy, EDP, ED2P, and EDAP), the near-optimal designs selected by the optimizer and the global optimal differ by $\leq$ 2% in $Obj$ values, showing close agreement. Exhaustive search for Tiny-YOLO and VGG11 requires 48.3 and 55 hours, respectively, with 6 parallel searches, while the optimizer requires 4.8 and 5.5 hours, respectively, with 6 parallel starts.

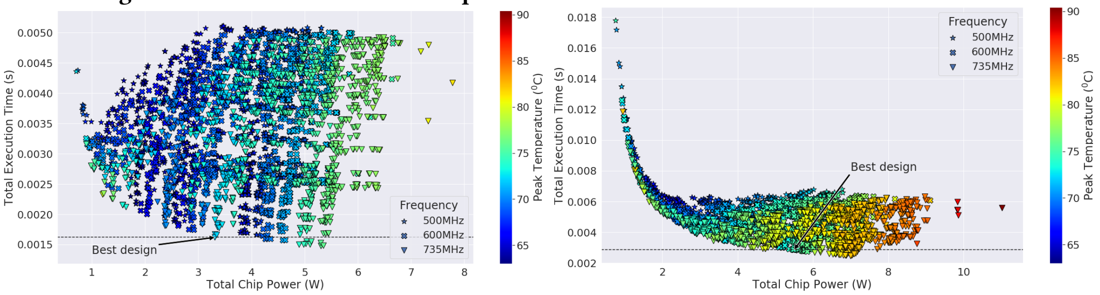
## 4.3 Optimization Results

We next discuss the temperature-aware optimization results for various objective functions. The 6 starts are assigned aspect ratio



(a) ResNet50 (b) VGG19

**Figure 4: Performance versus temperature tradeoffs in Mono3D DNN accelerators.**



(a) ResNet50 (b) VGG19

**Figure 5: Power-performance-temperature tradeoffs in Mono3D DNN accelerators.**

| *Optimization Goal* | *MobileNet* | *ResNet50* | *GoogLeNet* | *Tiny-YOLO* | *VGG11* | *VGG16* | *VGG19* | *Faster R-CNN* |
|---|---|---|---|---|---|---|---|---|
| Performance (Inference Latency) | 212×172 (735 *MHz*)<br>{32, 32, 4096} *KB* | 198×184 (735 *MHz*)<br>{4096, 32, 32} *KB* | 190×192 (735 *MHz*)<br>{32, 32, 4096} *KB* | 174×208 (735 *MHz*)<br>{4096, 32, 32} *KB* | 212×172 (735 *MHz*)<br>{4096, 32, 32} *KB* | 194×188 (735 *MHz*)<br>{4096, 32, 32} *KB* | 170×214 (735 *MHz*)<br>{4096, 32, 32} *KB* | 160×228 (735 *MHz*)<br>{4096, 32, 32} *KB* |
| Chip Power | 170×214 (735 *MHz*)<br>{32, 32, 4096} *KB* | 126×144 (735 *MHz*)<br>{2048, 32, 32} *KB* | 192×198 (735 *MHz*)<br>{32, 32, 4096} *KB* | 122×150 (735 *MHz*)<br>{2048, 32, 32} *KB* | 160×228 735 *MHz*)<br>{4096, 64, 32} *KB* | 210×174 (600 *MHz*)<br>{4096, 64, 32} *KB* | 180×202 (600 *MHz*)<br>{4096, 32, 32} *KB* | 152×120 (735 *MHz*)<br>{2048, 32, 32} *KB* |
| System Energy | 174×208 (735 *MHz*)<br>{32, 4028, 32} *KB* | 212×172 (735 MHz)<br>{4096, 64, 32} *KB* | 214×170 (735 *MHz*)<br>{32, 32, 4096} *KB* | 202×184 (735 *MHz*)<br>{4096, 128, 32} *KB* | 216×172 (735 *MHz*)<br>{4096, 128, 32} *KB* | 190×200 (735 *MHz*)<br>{4096, 256, 32} *KB* | 222×172 (600 *MHz*)<br>{4096, 256, 32} *KB* | 146×126 (735 *MHz*)<br>{2048, 64, 32} *KB* |
| System EDAP | 196×186 (735 *MHz*)<br>{32, 32, 4096} *KB* | 142×128 (735 *MHz*)<br>{2048, 32, 32} *KB* | 216×168 (735 *MHz*)<br>{32, 32, 4096} *KB* | 122×150 (735 *MHz*)<br>{2048, 32, 32} *KB* | 210×176 ( 735 *MHz*)<br>{4096, 128, 32} *KB* | 190×200 (735 *MHz*)<br>{4096, 256, 32} *KB* | 218×174 (600 *MHz*)<br>{4096, 256, 32} *KB* | 146x126 (735 *MHz*)<br>{2048, 64, 32} *KB* |

**Table 2: Designs selected by our optimizer: systolic array (operating frequency) and {IFMAP, Filter, OFMAP} SRAMs.**

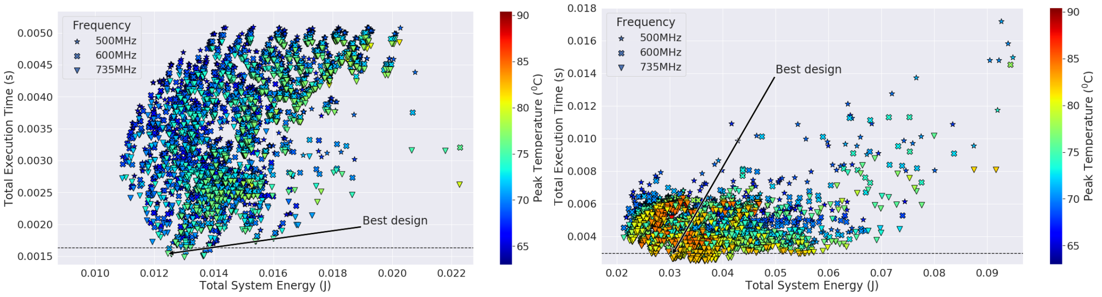
ranges: [0.7, 0.8], (0.8, 0.9], and so on till (1.2, 1.3]. Note that in the following results, the SRAM sizes are ordered as IFMAP, Filter, and OFMAP. If we mention one size, we refer to the total SRAM size.

*4.3.1 Performance.* Fig. 4 shows performance versus temperature results for all the designs that our optimizer evaluates before converging to near-optimal solutions for ResNet50 and VGG19 when minimizing latency. The dashed lines are the user-defined performance and thermal constraints. The optimizer selects a 198×184 systolic array with a 4160 *MB* SRAM at 735 *MHz* for ResNet50 (Fig. 4a). The figure also shows a few points with slightly worse performance but higher temperature within the performance constraint. Those points have a slightly larger footprint (1%) with more active PEs, which results in higher power and peak temperatures. LoC DNNs have adequate thermal headroom to run on big systolic arrays at 735 *MHz* without sacrificing performance (see Table 2).

In contrast, HiC DNNs have a higher array utilization (due to more MAC operations) and lead to more thermal violations (due to higher chip power) compared to the LoC DNNs (e.g., VGG19 in Fig. 4b). The optimizer selects 170×214 with 4160 *KB* SRAM for VGG19. Fig. 4b shows a 5% performance tradeoff w.r.t. the lowest execution time accelerator to obey the tight thermal budget for VGG19. The lowest execution time accelerator has higher utilization (with same SRAM size), which leads to better performance but higher dynamic power and temperature in the systolic array tier. The inter-tier thermal coupling in Mono3D further increases the static power by 4% (despite the same SRAM size), eventually leading to a 3°*C* higher peak temperature. On average, HiC DNNs tradeoff 2% performance to operate under safe temperatures (Table 2).

*4.3.2 Power.* Fig. 5 shows performance, power, and temperature tradeoffs for ResNet50 and VGG19. We see at low total chip power (< 1 W), peak temperatures can be high (80°C for ResNet50 and 82°C for VGG19). Here, the DNNs are running on smaller chip footprints (≈1 $mm^2$), i.e., with smaller systolic arrays and SRAMs , which leads to higher power density and peak temperatures. The optimizer selects 126×144 with 2112 *KB* SRAM at 735 *MHz* for ResNet50. 600 *MHz* designs present under the imposed constraints have a larger chip footprint with more PEs operating in parallel, which results in a net higher power (than the selected design). 500 *MHz* accelerators violate the performance constraint and thus, are not selected by the optimizer. Similarly, the optimizer selects 735 *MHz* designs for the other LoC DNNs (Table 2).

For VGG19, the optimizer selects a 180×202 systolic array with 2080 *KB* SRAM at 600 *MHz* (see Fig. 5b). At 735 *MHz*, the most power-efficient design under the user-specified constraints is almost of the same size as the selected design (≈0.99×) with a similar array utilization and same SRAM size. The higher dynamic power (due to faster PEs) causes higher temperature in the systolic array tier, which further increases the static power by 9% due to inter-tier thermal coupling (despite the same size of the SRAM), eventually resulting in a 7°*C* higher peak temperature. Similarly, a 600 *MHz*



**(a) ResNet50** **(b) VGG19**

**Figure 6: Energy distribution in Mono3D DNN accelerators under performance and temperature constraints.**

design is selected for VGG16. On the other hand, Faster-RCNN and VGG11 have lower power and lead to relatively fewer thermal violations. Hence the optimizer finds power-efficient accelerators at 735 $MHz$ under the thermal constraint (see Table 2).

*4.3.3 Energy Efficiency.* Fig. 6 shows system energy ($E_{sys}$) distribution for ResNet50 and VGG19. The optimizer selects 212×172 with 2112 $KB$ SRAM at 735 $MHz$ for ResNet50. While there exist few 600 $MHz$ accelerators with lower power under the performance constraint, the higher execution time negates the power savings. Similarly, 735 $MHz$ accelerators are selected for the other LoC DNNs. Even for HiC DNNs, the optimizer selects 735 $MHz$ accelerators for all but VGG19 (Table 2). VGG19, being the highest power DNN, benefits from both Mono3D iso-performance power savings and slower PEs, thus making up for the performance loss w.r.t. 735 $MHz$ designs. On average, our optimizer achieves 1.2× energy and 1.1× area savings, with a performance loss of 5.3% across all the DNNs. Finally, selections made by the optimizer for minimizing $EDAP$ achieve up to 2× chip footprint and 1.6× $E_{sys}$ savings, by sacrificing up to 9.7% latency (average: 1.2×, 1.4×, 5.5%, respectively).

## 5 CONCLUSION

We propose a design automation methodology that yields near-optimal energy efficient DNN accelerators based on Mono3D under user-specified thermal and performance constraints. Based on tradeoff analysis few conclusions can be drawn: (i) HiC DNNs with higher dynamic power result in higher temperature, which further increases leakage due to inter-tier thermal coupling, eventually resulting in thermal violations. As a result, HiC DNNs have to tradeoff performance to operate under safe temperatures. (ii) Although we can add more SRAM and PEs (i.e., larger systolic array) to utilize the two tiers in a given chip footprint, power efficiency can drop (even at lower frequencies) due to (a) higher dynamic power (more active PEs) and (b) higher SRAM static power, as a result of both SRAM size and inter-tier thermal coupling in Mono3D across all DNNs. (iii) HiC DNNs (e.g., VGG19) with more PEs running in parallel can benefit from running at lower frequency, along with Mono3D power savings, thereby achieving higher energy efficiency.

## REFERENCES

[1] C. Szegedy, A. Toshev, and D. Erhan. Deep neural networks for object detection. In *Adv. in neural information processing systems (NIPS)*, pages 2553–2561, 2013.

[2] 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 Design Automation Conference (DAC)*, pages 1–6, 2019.

[3] N. P. Jouppi et al. In-datacenter performance analysis of a tensor processing unit. In *Proc. of the Annual Int. Symp. on Computer Architecture*, pages 1–12, 2017.

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

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

[6] S. K. Samal et al. Fast and accurate thermal modeling and optimization for monolithic 3D ICs. In *ACM/IEEE DAC*, pages 1–6, 2014.

[7] S. Koppula et al. EDEN: Enabling energy-efficient, high-performance deep neural network inference using approximate DRAM. In *IEEE/ACM Proc. of International Symposium on Microarchitecture (MICRO)*, pages 166–181, 2019.

[8] S. Y. H. Mirmahaleh et al. Flow mapping and data distribution on mesh-based deep learning accelerator. In *IEEE/ACM Intl. Symp. on NOCS*, pages 1–8, 2019.

[9] B. Reagen et al. A case for efficient accelerator design space exploration via bayesian optimization. In *IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, pages 1–6, 2017.

[10] 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*, pages 751–764, 2017.

[11] 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*, pages 171–176, 2014.

[12] C. Yan and E. Salman. Mono3D: Open source cell library for monolithic 3-D integrated circuits. *IEEE Trans. on CAS I: Regular Papers*, 65(3):1075–1085, 2017.

[13] J. Shi et al. A 14nm FinFET transistor-level 3D partitioning design to enable high-performance and low-cost monolithic 3D IC. In *IEEE International Electron Devices Meeting (IEDM)*, pages 2–5, 2016.

[14] Y. Yu and N. K. Jha. SPRING: A sparsity-sware reduced-precision monolithic 3D CNN accelerator architecture for training and inference. *arXiv:1909.00557v2*, 2019.

[15] Y. Yu and N. K. Jha. A monolithic 3D hybrid architecture for energy-efficient computation. *IEEE Tran. on Multi-Scale Computing Systems*, 4(4):533–547, 2018.

[16] 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*, pages 1–6, 2017.

[17] C. T. Do, J. H. Choi, Y. S. Lee, C. H. Kim, and S. W. Chung. Enhancing matrix multiplication with a monolithic 3d based scratchpad memory. *IEEE Embedded Systems Letters*, 2020.

[18] V. J. Reddi et al. Mlperf inference benchmark. *arXiv:1911.02549*, 2019.

[19] A. Samajdar, Y. Zhu, P. Whatmough, M. Mattina, and T. Krishna. SCALE-Sim: Systolic CNN accelerator simulator. *arXiv preprint arXiv:1811.02883v2*, 2018.

[20] S. Thoziyoor, N. Muralimanohar, J. Ahn, and N. Jouppi. CACTI 6.5. *hpl.hp.com*, 2009.

[21] W. Huang et al. HotSpot: A compact thermal modeling methodology for early-stage VLSI design. *IEEE Trans. on VLSI systems*, 14(5):501–513, 2006.

[22] 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)*, pages 1–6, 2007.

[23] V. Adhinarayanan et al. Measuring and modeling on-chip interconnect power on real hardware. In *IEEE Int. Symp. on Workload Characterization*, pages 1–11, 2016.

[24] 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/EDAC/IEEE DAC*, pages 1–6, 2014.

[25] *"COMSOL Multiphysics Software." http://www.comsol.com.*

[26] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015.

[27] R. Huang, J. Pedoeem, and C. Chen. YOLO-LITE: a real-time object detection algorithm optimized for non-GPU computers. In *IEEE International Conference on Big Data*, pages 2503–2510. IEEE, 2018.

[28] S. Bianco, R. Cadene, L. Celona, and P. Napoletano. Benchmark analysis of representative deep neural network architectures. *IEEE Access*, 6:64270–64277, 2018.

[29] Micron Technology Inc. Embedded LPDDR2 SDRAM Features: https://www.micron.com/products/dram/lpdram/part-catalog/edb8132b4pb-8d-f. 2014.