Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

SAPER-AI accelerator: a systolic array-based power-efficient reconfigurable AI accelerator

Fahad Bin MUSLIM[†‡§1], Kashif INAYAT[§2,3], Muhammad Zain SIDDIQI[1], Safiullah KHAN[4], Tayyeb MAHMOOD[5], Ihtesham ul ISLAM[6]

[1]*Faculty of Computer Science and Engineering, GIK Institute, Topi 23460, Pakistan*

[2]*Barcelona Supercomputing Center, Barcelona 1-3 08034, Spain*

[3]*Department of Electronics Engineering, Incheon National University, Incheon 22006, Republic of Korea*

[4]*Department of Computing and Mathematics, Manchester Metropolitan University, Manchester M15 6BX, UK*

[5]*Nextwave Inc., Daejeon 34134, Republic of Korea*

[6]*Department of Computer Software Engineering, National University of Sciences and Technology, Islamabad H-12, Pakistan*

[†]E-mail: fahad.muslim@giki.edu.pk

Received Sept. 21, 2024; Revision accepted Jan. 24, 2025; Crosschecked Aug. 22, 2025; Published online Sept. 16, 2025

Abstract: Deep learning (DL) accelerators are critical for handling the growing computational demands of modern neural networks. Systolic array (SA)-based accelerators consist of a 2D mesh of processing elements (PEs) working cooperatively to accelerate matrix multiplication. The power efficiency of such accelerators is of primary importance, especially considering the edge AI regime. This work presents the SAPER-AI accelerator, an SA accelerator with power intent specified via a unified power format representation in a simplified manner with negligible microarchitectural optimization effort. Our proposed accelerator switches off rows and columns of PEs in a coarse-grained manner, thus leading to SA microarchitecture complying with the varying computational requirements of modern DL workloads. Our analysis demonstrates enhanced power efficiency ranging between 10% and 25% for the best case 32×32 and 64×64 SA designs, respectively. Additionally, the power delay product (PDP) exhibits a progressive improvement of around 6% for larger SA sizes. Moreover, a performance comparison between the MobileNet and ResNet50 models indicates generally better SA performance for the ResNet50 workload. This is due to the more regular convolutions portrayed by ResNet50 that are more favored by SAs, with the performance gap widening as the SA size increases.

Key words: Artificial intelligence (AI) accelerators; Application-specific integrated circuit (ASIC) design; Systolic arrays; Low-power designs

https://doi.org/10.1631/FITEE.2400867 **CLC number:** TP39

1 Introduction

The meteoric rise in deep learning (DL)-based solutions has revolutionized a variety of domains such as image processing, pattern recognition, and transportation. To make full use of the benefits that such DL models can accord, huge operational costs need to be incurred due to the computational complexities accompanying such models (Yüzügüler et al., 2023). Therefore, specialized DL accelerators are necessary to offer enhanced computational prowess while still being energy efficient. Several such accelerators have been proposed on both sides of the DL continuum, i.e., the cloud (Bobda et al., 2022; Li et al., 2023) and the edge (Seshadri et al., 2022; Loh et al., 2025). Deep neural networks (DNNs)

‡ Corresponding author
§ These two authors contributed equally to this work
ORCID: Fahad Bin MUSLIM, https://orcid.org/0000-0002-4153-360X

incorporate matrix multiplication as the primary primitive, which in turn, fortunately offers a lot of parallelism, and its acceleration is imperative to achieve tremendous processing demands of the DL workloads (Muslim et al., 2024).

Several general matrix multiplication (GEMM) accelerators found in the literature are based on systolic arrays (SAs) (Jouppi et al., 2017; Song et al., 2019; Lai and Zhang, 2024). SA is a two-dimensional mesh of processing elements (PEs) with each PE being fed inputs from the left and top sides. Each PE performs a multiplication and accumulation (MAC) operation every clock cycle, and the partial result and the input are fed to the neighboring PE via pipeline registers. In this way, the PEs collaborate to offer enhanced parallelism in processing data efficiently and accelerate the DNN computation. A typical SA architecture is depicted in Fig. 1. Multiplier $\boldsymbol{X}$ is indicated on the left while the multiplicand $\boldsymbol{Y}$ and the bias term $\boldsymbol{W}$ are on the top side of SA in Fig. 1.

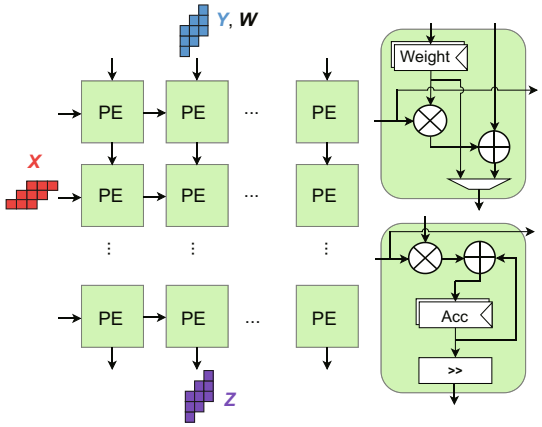

Fig. 1 A typical SA architecture with output and weight stationary data flows

One of the main factors necessitating the usage of customized architectures for DL acceleration is the resulting improvement in the energy efficiency offered by the accelerators. Such customized hardware must comply with the tight power budget, which is important both at the cloud and the edge. However, the power efficiency of such designs at the edge due to the constrained power availability is of even greater importance (Kim et al., 2020). Thus, a lot of research has been done in recent years to improve the power efficiency of such hardware accelerators. The research found in the literature mainly targets complex microarchitectural optimizations, e.g., by reducing expensive memory access optimizations or by quantization of the networks leading to approximate computing, thus leading to enhanced energy savings (Chen YJ et al., 2016; Moons et al., 2016).

Moreover, the modern DNNs are accompanied by increased sparsity, much like their biological counterparts and can generalize well when compared to their denser versions (Hoefler et al., 2021; Guo et al., 2024). Owing to their reduced memory footprints, enhanced power efficiency and lightweight implementation suited to the inference regime, such networks are primarily suited for edge artificial intelligence (AI) devices (Hoefler et al., 2021). The sparsity, however, is counterproductive when DNN implementation via SA-based accelerators is concerned. This is due to the fact that the SAs are well-suited for dense, regular computation patterns accompanied by predictable dataflow in the array (Xu et al., 2021a). The irregularity in data accesses as well as in computation patterns causes several PEs to remain idle, consuming power while not doing any useful computation, thus adversely affecting the power efficiency (Chen YH et al., 2016; Xu et al., 2023). The impact of various types of convolutions on the utilization of SAs is depicted graphically in Fig. 2. Fig. 2a shows the typical GEMM, while the matrix–vector multiplication (mimicking depthwise convolution) is depicted in Fig. 2b. Finally, the impact on SA utilization when the SA size is large, considering depthwise convolution, is shown in Fig. 2c. The SA utilization is indicated by the number of shaded PEs to the total number of PEs in Fig. 2. The interdependence between the SA size and the type of convolution operation with respect to SA utilization can clearly be seen in the figure as well.

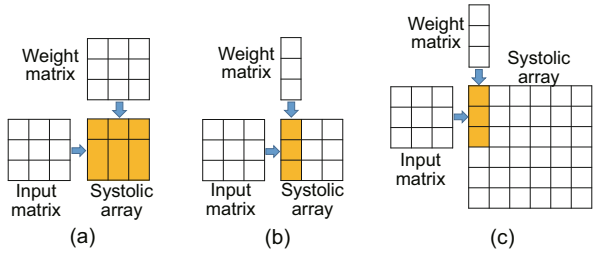

Fig. 2 Typical GMEM (a), matrix–vector multiplication (b), and matrix–vector multiplication with a large SA size (c)

The aforementioned discussion leads us to the primary motivation behind this research effort. This includes the ability to reconfigure SAs which are designed for dense computations in accordance with the required computational complexity of various network layers and to bulk power-gate idle PEs (in a coarse-grained manner). This shall ensure enhanced power efficiency at the cost of additional logic caused by the introduction of power gating logic. The architectural details of two neural network architectures, i.e., MobileNet and ResNet50 (both trained on the ImageNet dataset), considered for design evaluation in this work are given in Table 1. We observe a reduction in the output feature map (OFMap) size as the number of network layers increases (Xu et al., 2021a). The depthwise and pointwise convolutions in the MobileNet layers can also be observed. The frequent transition between these two convolutions has a profound impact on the SA performance evaluation considering MobileNet, as will be presented in Section 4.

Power gating is achieved by writing the power intent for the SA accelerator in an IEEE standard unified power format (UPF). The power management logic is inserted at a coarse-grain level and adjusted in accordance with the layerwise computational demands of the considered DNN models. Power intent description via UPF allows power logic to be handled separately from the logic intent description (Chadha and Bhasker, 2012). This, in turn, leads to simplified power inference with trivial design changes required to be made at the register transfer level (RTL), mainly to provide power control signals to control the insertion of low-power logic after logic synthesis. In comparison to our previous work (Inayat et al., 2023) which targets fine-grained power gating by shutting down the idle multipliers of the MAC units inside the PEs, this work targets a coarse-grained approach. This approach better fits the sparse nature of modern DNN workloads. Moreover, this effort includes microarchitectural optimizations to divide the multipliers in the MAC units into smaller submultipliers to increase the power gating instances (at the cost of increased area). In comparison, this work purely exploits the network architecture without having to rely on any expensive microarchitectural modifications, apart from the requirement to write the power management module at the RTL and instantiate it inside the top SA module.

The major contributions targeted in this work are as follows:

1. Application-specific integrated circuit(ASIC) implementation of 32×32 and 64×64 reconfigurable SA-based AI accelerators, termed as SAPER-AI accelerator, to cater to the varying workloads of the considered DNN model layers. This ensures the full utilization of the active PEs of the underlying SA designs at any given time.

2. Coarse-grained power gating of row and column PEs of the SA based on the varying computational requirements of various network layers of the DNN workload. The main aim of the power optimization strategy is to achieve power savings without expensive microarchitectural modifications.

Table 1 Structure of MobileNet and ResNet50 models

Architecture	Layer name	Kernel size	OFMap size
MobileNet	Conv1	3×3	112×112
	Conv2_dw/pw	3×3/1×1	112×112
	Conv3_dw/pw	3×3/1×1	56×56
	Conv4_dw/pw	3×3/1×1	56×56
	Conv5_dw/pw	3×3/1×1	28×28
	Conv6_dw/pw	3×3/1×1	28×28
	Conv7_dw/pw	3×3/1×1	14×14
	Conv8_dw/pw	3×3/1×1	14×14
	Conv9_dw/pw	3×3/1×1	7×7
ResNet50	Conv1	7×7	112×112
	Conv2_3	1×1, 3×3, and 1×1	56×56
	Conv3_4	1×1, 3×3, and 1×1	28×28
	Conv4_6	1×1, 3×3, and 1×1	14×14
	Conv5_3	1×1, 3×3, and 1×1	7×7

3×3/1×1 means the kernel size of the convolutional layer Conv2_dw/pw respectively, with similar meanings in other cases as well

3. Realistic power analysis of the proposed design cases by using the actual workloads of some DNN architectures, i.e., MobileNet and ResNet50.

4. A detailed comparison of the proposed SA design based on power, performance, and area (PPA), and power delay product (PDP) parameters.

2 Related works

This section presents a mix of some recent works targeting power-efficient SA-based accelerators, while specifically dealing with reconfigurable accelerators that can adjust their features to cater to evolving DNN workloads.

Moghaddasi and Nam (2024) presented serial/parallel and octet serial/parallel SA (SPSA and OSPSA) approaches. The designs presented improve the DNN computational efficiency by activating serial processing in an adjustable manner with respect to the runtime and layerwise precision. This is done in accordance with the computation required by the DNN model. While both the activations and weights fed to the PEs are done in a parallel manner in traditional SAs, their proposed approach introduces support for serializing the activation matrices while keeping the weight matrices input in a parallel manner. The OSPSA design further improves the throughput by processing a column of eight simultaneous bits of different activations in the same bit position. Energy consumption is reduced via the simpler serial/parallel PE architecture and the replacement of complex multipliers with simpler and cost-effective serial circuits. Other bit-serial approaches were also proposed (Lee et al., 2018; Ryu et al., 2019), which obviously offer advantages in terms of interconnections, area, and power. However, this gain traditionally comes with a sacrifice in throughput of the underlying SA-based designs.

Inayat et al. (2023) proposed a power intent systolic array (PI-SA), which relies on UPF-based power intent, similar to our present work. However, power gating is done in a more fine-grained manner by targeting multipliers for power shutoff in case of inactivity. This is different from what we are doing, i.e., using coarse-grained power gating. Power efficiency is further improved by splitting the multipliers into smaller parallel submultipliers, thus increasing the power shutoff instances. Moreover, the power analysis includes introducing zero entries randomly and not necessarily indicating data traffic of the actual DNN workloads considered in this work.

Xu et al. (2021b) presented the concept of a heterogeneous systolic array architecture (HeSA) to resolve the issue of suboptimal SA performance when processing compact convolutional neural network (CNN) models, such as depthwise convolution. This work introduces heterogeneous PEs supporting multiple dataflows to cater to changing DNN workloads. Since the structural changes are happening to the PEs within the SAs, the traditional SA structure remains intact. While dealing with standard convolution, the HeSA acts as a traditional SA, but for the depthwise convolution, the SA adjusts the dataflow by using the heterogeneous PEs, thus resulting in significant energy savings when compared to the naive SA architecture. While Xu et al. (2021b) proposed a way around the depthwise convolutions which are also considered in this work, their approach is considerably different and arguably more complicated than our approach.

Another approach to achieve energy efficiency in AI accelerators is via approximate computing based on the quantization of weights and inputs at each layer (Moons et al., 2016). Other approximate computing approaches based on dynamic voltage and frequency scaling (DVFS) to vary the threshold voltage leading to supply voltage variation for differing precision requirements were also proposed (Moons et al., 2017a, 2017b). These works achieve energy efficiency at the cost of loss in precision, which may be crucial in some DNN applications, such as medical imaging and diagnostics, and military and defense.

Thus, the presented literature can be summarized by inferring that energy efficiency in DL accelerators is of prime importance. This is usually achieved by complicated microarchitectural modifications and other approximation approaches, which, in turn, lead to a loss of accuracy. Based on the discussion above, the presented literature can be broadly summarized in terms of optimization strategies, accuracy loss, and optimization effort. Such a comparison of the presented literature with the proposed work in these aspects is given in Table 2. This work intends to improve the accelerator energy efficiency by specifying power intent via UPF with trivial changes in the SA structure. The optimizations can be adjusted according to the actual DNN workload, and significant energy savings can be achieved as demonstrated by

performance analysis considering a couple of widely used DNN models.

3 System model

This section details the system architecture, indicating primarily how the power intent description process will vary the SA architecture. We then present the 32×32 and 64×64 SA accelerator designs considered in this work. Each design has adjustable PE configurations to cater to the varying DNN workloads. Finally, we present the details of our power analysis framework, considering MobileNet and ResNet50 DNN workload models.

3.1 Bird's eye illustration of the system

A high-level abstraction of the architecture of the 64×64 SA top module with various module instantiations is shown in Fig. 3. The power management unit (PMU) is an RTL module basically responsible for providing the control signals to shut off the power domains. These power domains are labeled as PD_X, where X ranges from 2 to 5, indicating the switchable power domains. Power domains are basically a collection of PEs that operate at the same operating condition; i.e., all the power domains collectively constitute a power-aware SA design. Another variable of interest is the mesh_select signal, which (as the name implies) decides which power domains (or a collection of PEs) are being put to sleep at any instant. When the mesh_select variable is set to 0, all the switchable domains are put to rest, thus allowing only the 7×7 SA size to compute actively, i.e., the so-called default power domain (PD_default). The term "default" indicates that this power domain will never be switched off under any

Table 2 Comparative analysis with relevant literature

Design	Optimization strategy	Accuracy loss	Effort
Lee et al. (2018); Ryu et al. (2019); Xu et al. (2021b); Moghaddasi and Nam (2024)	Microarchitectural optimizations	No	High
Inayat et al. (2023)	PG (fine) & microarchitectural optimizations	No	Medium
Moons et al. (2016)	Approximate computing (quantization)	Yes	Medium
Moons et al. (2017a, 2017b)	Approximate computing (DVFS)	Yes	Low
This work	Coarse-grained power gating	No	Low

PG: power gating; DVFS: dynamic voltage and frequency scaling

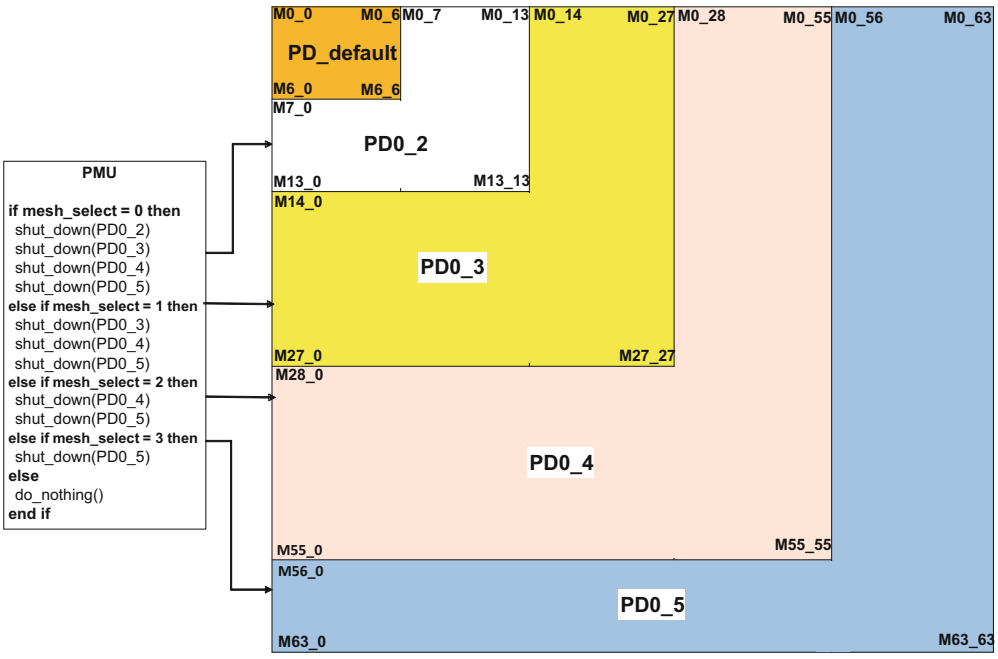

Fig. 3 Abstracted illustration of the power-aware 64×64 SA design (PD: power domain)

circumstances. The PMU module is also part of the default power domain, as it is responsible for providing the necessary power control signals and must be active all the time. Similarly, with mesh_select equaling 1, the 14×14-sized SA computes actively while the rest of the PEs are put to sleep. The rest of the active SA sizes selected via the mesh_select variable are 28×28, 56×56, and the fully loaded 64×64 for mesh_select values of 2, 3, and beyond, respectively. We do not depict the 32×32 SA since it has a similar architecture. The mesh_select values of 0, 1, and 2 correspond to 7×7, 14×14, and 28×28 active SA sizes, respectively, with the full-blown active SA size being 32×32. The PMU is meant to provide the power control signals in an appropriate manner. We do not discuss its detailed logic but inquisitive readers are encouraged to explore the same in other reported works (Qamar et al., 2017; Inayat et al., 2023).

At the gate level, committing power intent results in the insertion of low-power logic in the design (Qamar et al., 2017). This logic includes low-power cells, such as isolation cells, to isolate the default power domain from the floating output values of the power-gated domain. These cells are usually placed at the intersection of the two domains (at the output of the switchable domain). Another class of low-power cells is the retention cells. These are used to save the state of some sequential logic in the power-gated domain before it is put to sleep. These cells are inserted if the logic synthesis tool senses the need for them. Furthermore, the power switch header/footer cells are used to cut off the supply to the switchable domain and are usually placed in the design after physical implementation (Chadha and Bhasker, 2012). A generalized view of a typical power-aware hardware with a single default always-on domain and a power shut-off (PSO) domain similar to the one created in this work is shown in Fig. 4. The PMU module indicating the power control signals, iso (isolation) and ret (state retention), and the resulting low-power cells, i.e., ISO and RET, can be clearly seen as well. Both the always-on and PSO domains contain a group of PEs in our case.

3.2 Detailed microarchitecture of the designs

This subsection presents detailed microarchitectural representation of the power-aware 32×32 and 64×64 SAs. The 32×32 design is depicted in Fig. 5,

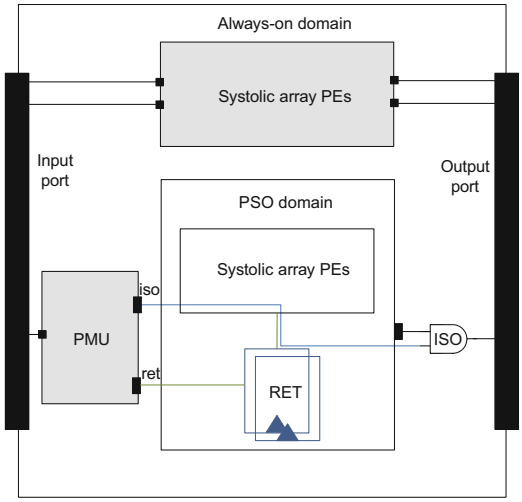

Fig. 4 Generalized power-aware hardware design

while its 64×64 counterpart is shown in Fig. 6. A multiplexer at the input of the SA in both cases is used to provide the inputs to the active PEs. The SA inputs as well as the PMU signals in both figures are color-coded in accordance with the active power domains at any specific time. The low-power logic depicted earlier in Fig. 4 with respect to various power domains is inserted by the logic synthesis tool. This logic ensures that the appropriate power domains are active at any given time as dictated by the DNN computation workload. Sequential logic is also depicted at the output of the SA in Figs. 5 and 6. It is necessary to add the appropriate delay in each of the low-power SA operations to ensure that the SA output is always fed to any succeeding components in an amicable fashion. The only exception, as far as the registered SA outputs are considered, is the full-blown SA computation. This computation would always be active if the power logic had not been used in the design. Moreover, it should be noted that even though a single register is shown to store the various active PE outputs, it is only to simplify the depiction. In reality, the number of registers differs depending on the active SA size. As an instance, for the 32×32 case depicted in Fig. 5, with 7×7 active PEs, we use 50 registers, while in the case of 64×64 SA depicted in Fig. 6 and considering 7×7 active PEs, we use 114 registers. To ensure appropriate SA outputs all the time, the general number of registers required to store the active SA outputs, considering the active SA size of $n \times n$ and the full-blown SA size of $N \times N$, is $2(N-n)$. The register count required at the SA output for different cases is given in Table 3.

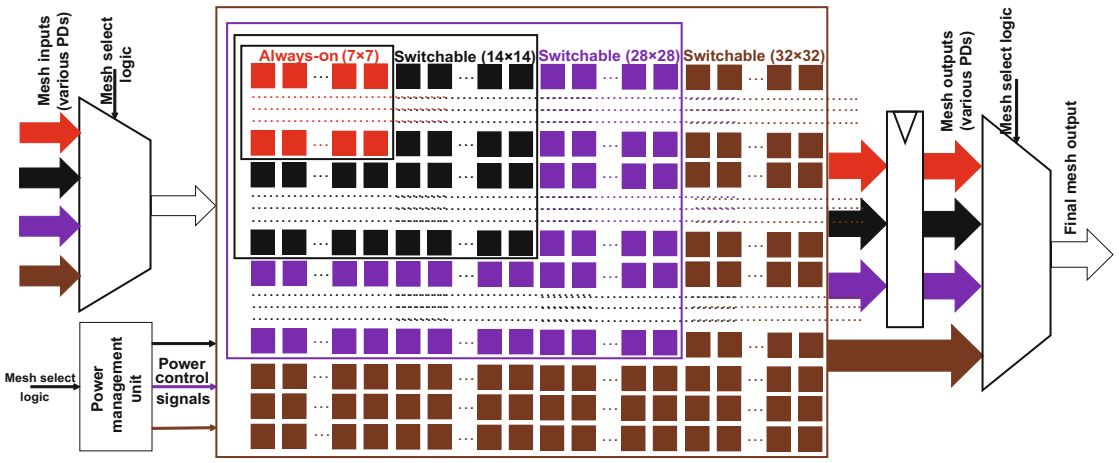

Fig. 5 Detailed view of the 32×32 SA with power control signals. References to color refer to the online version of this figure

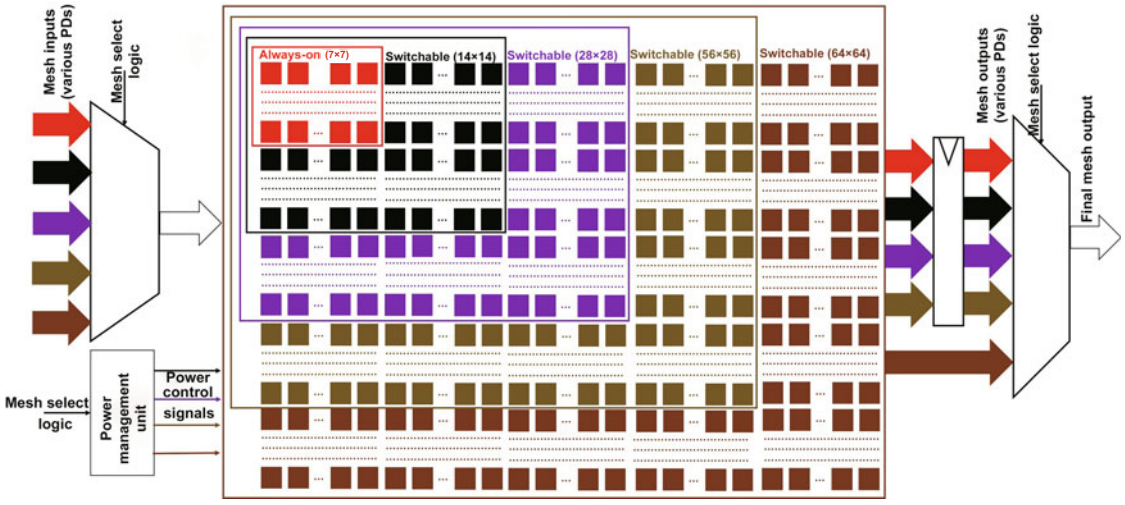

Fig. 6 Detailed view of the 64×64 SA with power control signals. References to color refer to the online version of this figure

Table 3 Number of registers required at the SA output for various sizes

Total SA size $N \times N$	Active SA size $n \times n$	Number of O/P registers $2(N-n)$
32×32	7×7	50
	14×14	36
	28×28	8
64×64	7×7	114
	14×14	100
	28×28	72
	56×56	16

3.3 Power analysis framework

This subsection details the power evaluation framework used in this work. The power-aware SA designs proposed in this work are based on the baseline design generated by the Gemmini SA generator (Genc et al., 2021). The generated SAs are evaluated by running DNNs such as MobileNet and ResNet50 according to the baseline design evaluation framework (Genc et al., 2021). We perform initial simulations in the bare metal environment using our recently developed in-house field-programmable gate array (FPGA)-accelerated simulation environment called "Quickloop" (Inayat et al., 2024). These simulations are meant to extract the number of clock cycles taken by each layer of the DNN workloads, i.e., to do activity profiling of the DNN workloads. This number is then used to estimate the index numbers (with respect to the whole DNN activity index)

to perform power analysis while complying with the actual workloads of the DNN models considered in this work.

Furthermore, the gate-level netlist produced after logic synthesis is simulated to generate the switching activity information of various signals. This information is captured using the switching activity interchange format (SAIF) file that is then annotated to the gate-level netlist to perform power analysis. The activity profile for various computation sizes of both the considered DNN workloads as obtained from the FPGA-accelerated simulations is provided in Table 4. The total number of clock cycles taken by the FPGA-assisted SA simulations of both networks is also indicated in the table. The MobileNet simulation seems to be taking more clock cycles to complete. This may be attributed to the frequent switching between depthwise and pointwise convolutions, which in turn causes dataflow patterns to change significantly, thereby leading to underutilization of the SA. Moreover, the delay caused by excessive memory accesses and synchronization overheads due to frequent switching of convolution patterns in the MobileNet model also adversely impacts the simulation cycle count (Lym and Erez, 2020; Xu et al., 2021a).

4 Results

This section discusses the details regarding the evaluation environment used in this work. This is followed by a detailed performance analysis of the developed 32×32 and 64×64 SA designs considering the DL workloads.

4.1 Evaluation environment

All the designs have been synthesized using the SAED 32-nm technology library supporting low-power designs. We have employed various Synopsys tools, i.e., Design Compiler for synthesis, VCS for simulation, and Primetime for power analysis. Functional simulations have been performed both at the RTL and the gate-level design representations. The power measurement has been done at 3-ns clock period. All the analysis has been done on a Linux machine with 128 GB of memory. This memory constraint imposes a limit on the largest SA size being deployed, especially considering the power-gated designs. This is the reason behind the maximum SA size of 64×64 considered in this work.

Table 4 Activity profile of various feature sizes of the DNN workload

Model	Output size	Activity (%)
MobileNet (1278M cycles)	56×56	15.63
	28×28	19.50
	14×14	20.78
	7×7	11.16
ResNet50 (109M cycles)	56×56	14.56
	28×28	25.13
	14×14	21.02
	7×7	13.12

M means million

4.2 32×32 SA performance analysis

The performance analysis for the 32×32 design in terms of delay and area, with and without power intent, is given in Table 5. The cases with the best delay and area are emboldened in Table 5. The insertion of low-power cells has an adverse effect on the delay and area of the design. The main culprits, as far as the area increase of the power-gated designs is concerned, are the sequential area and the interconnect area. This is because the power gating results in excessive sequential logic in the form of output registers, in addition to an increase in resulting interconnections. These contribute significantly to the delay of the power-aware designs as well as the delay caused by the additional low-power cells and the interconnects that the insertion of these cells shall result in.

As far as the power analysis is concerned, the 32×32 SA design shows power saving of around 10% in the case of MobileNet and around 12% in the case of ResNet50 workload, as depicted in Table 5. The combined impact of both the power and delay can be captured in terms of PDP. In terms of the PDP, our power-gated design worsens by around 9% in the case of MobileNet workload, and by around 8% in the case of the ResNet50 workload. Thus, besides improving the power consumption by around 10%–12% for realistic DNN workloads, our developed power-aware 32×32 SA design does not perform all that well in the other performance parameters. This prompts us to explore bigger-sized SAs, as 64×64 and 128×128 SAs are not that uncommon, especially for processing large-scale matrix multiplications that are fundamental to DNNs (Chen YJ et al., 2014, 2016; Gao et al., 2017; Jouppi et al., 2017; Chen YH et al., 2019). Due to memory limitations of our

evaluation setup, we could not synthesize a 128×128 SA design. Therefore, we adopt the 64×64 SA design presented in the next subsection.

4.3 64×64 SA performance analysis

The performance evaluation for the power-aware 64×64 SA accelerator is presented in Table 6. The best-case performance parameters in each case are emboldened, similar to Table 5. The trend is the same as that of the 32×32 case when the non-power-aware and power-aware cases are compared; i.e., the delay and area increase with a decrease in the power consumption. The delay cost in the case of power-gated 64×64 SA is around 26%, while the associated area cost is around 11%. The power-gated 64×64 design shows higher delay compared to the power-gated 32×32 design. The delay values of the 32×32 and 64×64 SA designs, considering the same power awareness regimes, are almost the same, however. This is because of the critical path in both the 32×32 and 64×64 cases being constructed within a single PE, which is technically the same. This work does not change the PE microarchitecture but instead makes slight microarchitectural changes to the SA itself. There is a 3× increase in the area of the 64×64 design compared to the 32×32 design, due to a massive increase in the computational and interconnect logic in the case of the 64×64 design.

The delay in the case of 64×64 SA worsens by around 26%, while the area by around 11% as a result of the lower power logic. The power, however, improves by 22% for the MobileNet case and by about 25% for the ResNet50 case. More significantly, the PDP shows an improvement of around 2% for the MobileNet case and around 6% for the ResNet50 case when considering the power-aware 64×64 SA design. This positively cements our hypothesis that the proposed optimizations are more useful when considering larger SA designs. Such larger arrays are beneficial for different applications such as high-performance cloud-based AI accelerators and inference of large models, e.g., BERT and GPT.

4.4 Comparative performance of DNN models

This subsection performs a comparative assessment (across both the power-aware and non-power-aware) of the best-case performance of the 32×32 and 64×64 SA designs. The best case performance for the 32×32 SA design is depicted graphically in Fig. 7. The area and delay performances are dependent on the design itself and do not vary with the DNN workloads, and hence they are the same for both MobileNet and ResNet50 workloads. They are still, however, included in the analysis for the sake of completeness. The best-case power results are obtained for the 32×32 case by the power-aware design, while the non-power-aware case achieves the best PDP performance. The two DNN models cannot be separated much in this case, with better performance achieved by ResNet50 as far as the design power consumption is concerned.

The best-case performance results for the DNN models in the case of 64×64 SA design are depicted in Fig. 8. Just like the 32×32 SA design, the 64×64 array also shows better performance in terms of power

Table 5 Performance analysis of the 32×32 SA across DNN workloads with (w/) and without (w/o) UPF

Workload	Delay (ns)		Area (mm^2)		Power (W)		PDP (W·ns)	
	w/o UPF	w/ UPF	w/o UPF	w/ UPF	w/o UPF	w/ UPF	w/o UPF	w/ UPF
MobileNet	**1.56**	1.91	**7.74**	8.41	1.094	**0.980**	**1.71**	1.87
ResNet50	**1.56**	1.91	**7.74**	8.41	1.077	**0.950**	**1.68**	1.81

Better results are in bold

Table 6 Performance analysis of the 64×64 SA across DNN workloads with (w/) and without (w/o) UPF

Workload	Delay (ns)		Area (mm^2)		Power (W)		PDP (W·ns)	
	w/o UPF	w/ UPF	w/o UPF	w/ UPF	w/o UPF	w/ UPF	w/o UPF	w/ UPF
MobileNet	**1.56**	1.96	**30.8**	34.2	3.72	**2.90**	5.80	**5.68**
ResNet50	**1.56**	1.96	**30.8**	34.2	3.60	**2.70**	5.62	**5.29**

Better results are in bold

and PDP for the ResNet50 model. The model-wise difference in performance, however, is much more prominent as compared to the 32×32 array. This is due to the consistent switching between pointwise and depthwise convolutions portrayed by MobileNet that the SA architecture is not very good at handling (Park et al., 2024). This structural difference between the DNN models seems to cause a larger variation in the performance as the SA size goes up. Moreover, the best-case power and PDP performances in the case of 64×64 SA, considering the DNN models for the power-aware case, are much better compared to the non-power-aware case. This indicates that the proposed design subsequently performs much better in terms of DL acceleration considering larger SA sizes.

4.5 State-of-the-art comparison

A conceptual comparison of the proposed work with similar state-of-the-art endeavors is presented in Table 2. Those works are still considerably different in approaches, and thus, a fair numerical comparison with all the works is not possible. We have thus chosen a subset of those works based on the same baseline naive SA design (Genc et al., 2021) for a numerical comparison with our proposed work. Furthermore, since our proposed work does not use approximation of any sort, we include only the literature exhibiting no loss of accuracy. The analysis presented here must be considered in combination with other factors, such as optimization effort and real-time DL workload consideration to assess where the proposed work would fit in the current design landscape. The comparison is tabulated in Table 7, which shows that the maximum clock frequency, the maximum SA size, and technology size considered in this work are all in line with the parameters considered in similar studies.

Table 7 primarily indicates the change in area (Δ_{Area}) and the change in energy (Δ_{energy}), among the optimizations in the cited works and the naive SA representation. While Inayat et al. (2023) portrayed better performance as far as the area and the energy consumption are concerned, it uses random zero inputs to cause fine-grained powering off of the PEs. The values thus do not represent the real workload handled by modern DL frameworks. In comparison, Xu et al. (2021b) considered real DL workloads, with energy savings almost equal to those offered by our

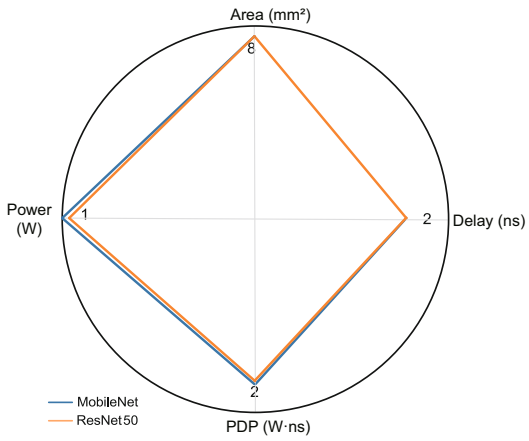

Fig. 7 Best-case performance analysis of the 32×32 SA design

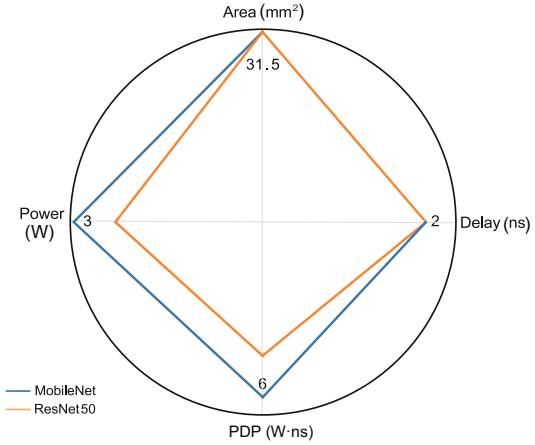

Fig. 8 Best-case performance analysis of the 64×64 SA design

Table 7 Numerical comparison with state-of-the-art works

Design	Δ_{Area} (%)	Δ_{Energy} (%)	Maximum clock frequency (MHz)	Maximum SA size	DNN model	Technology size (nm)
Xu et al. (2021b)	N/A	10	500	32×32	MobileNet, MixNet	45
Inayat et al. (2023)	32	56	250	64×64	N/A	32
This work	10	6	333	64×64	ResNet50, MobileNet	32

N/A: not available

proposed design. Our work, however, causes a slight degradation of area due to the additional low-power logic and the sequential logic at the design output. The numbers presented in Table 7, when considered in combination with the optimization effort requirement of each approach, emphasize the significance of our proposed work compared to the relevant state-of-the-art research.

4.6 Limitations

While we have analyzed the performance of our designed accelerator on MobileNet and ResNet50 models, we realize the enhanced dominance of Transformer-based models such as GPT, in the prevailing AI workloads. Transformer models rely heavily on matrix multiplications in self-attention and feedforward networks, but the matrices involved are considerably large and not as structured as in convolutions (Chang and Kim, 2024). Thus, scaling the proposed SA design to Transformer-based models would pose significant challenges in dealing with memory access bandwidth and dynamic dataflow requirements of these models (Amirshahi et al., 2023a, 2023b).

Besides, we have a constraint on the maximum size of the SA, i.e., 64×64, in this work. The large Transformer models may involve splitting the computations into multiple blocks (with the maximum SA size limitation), leading to idle PEs due to partial computations. The power (especially dynamic power) may therefore increase because of the resulting frequent switching. This mismatch in workload granularity may require fine-grained power gating at the PE level to achieve better energy efficiency but at the cost of higher power logic. Additionally, the performance evaluation in this work, as well as other similar works presented earlier, is based on the Gemmini framework (Genc et al., 2021). This framework uses MobileNet and ResNet50 models for evaluating performance, and thus, we use the same workloads for a fair comparison to other state-of-the-art models.

To summarize, extending the proposed work to incorporate the Transformer models would require several low-level interventions, such as on-chip caching and tiling strategies to reduce memory access overhead as well as compression mechanisms to reduce attention weights and activations. This would in turn add additional optimization effort, which is not the main scope of this activity, i.e., to introduce simplified power intent in the SA-based accelerator and analyze its performance on DL workloads. We thus intend to address these significant challenges in our future endeavors to enhance the scalability of our proposed work towards Transformer models for targeting applications such as natural language processing and computer vision.

5 Conclusions

In this research endeavor, we propose a power-aware SA-based AI accelerator termed as the SAPER-AI accelerator. This accelerator is meant to exploit the structural changes in modern DNN workloads and reconfigurably select active PEs to cater to the changing workloads. Unlike similar efforts found in the literature that rely on complex micro-architectural optimizations or fine-grained power optimizations, the proposed accelerator relies on much simpler UPF-based power intent descriptions to put unutilized PEs to sleep in a coarse-grained manner. The accelerator involves modifying the naive chisel-generated SA trivially by introducing a power management module that is responsible for providing the appropriate power control signals. The low-power cells introduced in the design after logic synthesis ensure the power intent integrity of the design. The same is validated by the post-synthesis power-aware simulations as well.

Evaluating the performance of the proposed designs with scaling demonstrates that the optimizations generally cause area and delay penalties due to the additional low-power logic. Power and PDP evaluation, considering MobileNet and ResNet50 DL workloads, indicate that both the 32×32 and 64×64 SA designs are more power-efficient comparatively in the power-aware case. The efficiency, however, improves further as the SA sizes increase. Moreover, the PDP of the power-aware 64×64 SA is considerably better when compared with the non-power-aware case, thus demonstrating the effectiveness of the proposed optimizations, especially for larger SAs. Finally, the proposed accelerator provides better performance on the ResNet50 workload compared to the MobileNet case for larger SAs owing to greater comparative uniformity in convolutions of ResNet50 that are more favored by the underlying SA architecture.

6 Future work

As the future scope of work, further research is required to validate the enhancement in the performance of the proposed optimizations with increasing SA sizes. This could not be obtained in this work due to memory capacity limitations in our evaluation setup. Moreover, other more complex power optimization techniques based on UPF, such as dynamic voltage scaling, can be used to optimize the design power even more. We can explore some simpler microarchitectural optimizations to keep area and delay values in check while optimizing power. Additionally, we intend to scale the proposed work towards a power optimized SA-based accelerator design targeting Transformer-based models. This will require the necessary microarchitectural interventions to circumvent the challenges posed by the complicated matrix operations constituting the Transformer models, as well as by the considerably larger number of parameters of such models.

Contributors

Fahad Bin MUSLIM conceptualized the research. Fahad Bin MUSLIM and Kashif INAYAT implemented the idea and performed the analysis. Fahad Bin MUSLIM and Muhammad Zain SIDDIQI drafted the paper. Safiullah KHAN and Tayyeb MAHMOOD helped organize the paper. Ihtesham ul ISLAM revised and finalized the paper.

Conflict of interest

All the authors declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

Amirshahi A, Ansaloni G, Atienza D, 2023a. Accelerator-driven data arrangement to minimize transformers runtime on multi-core architectures. 15th Workshop on Parallel Programming and Run-Time Management Techniques for Many-Core Architectures and 13th Workshop on Design Tools and Architectures for Multicore Embedded Computing Platforms, p.2:1-2:13. https://doi.org/10.4230/OASIcs.parma-ditam.2024.2

Amirshahi A, Klein JAH, Ansaloni G, et al., 2023b. TiC-SAT: tightly-coupled systolic accelerator for transformers. Proc 28th Asia and South Pacific Design Automation Conf, p.657-663. https://doi.org/10.1145/3566097.3567867

Bobda C, Mbongue JM, Chow P, et al., 2022. The future of FPGA acceleration in datacenters and the cloud. *ACM Trans Reconfig Technol Syst*, 15(3):34. https://doi.org/10.1145/3506713

Chadha R, Bhasker J, 2012. An ASIC Low Power Primer: Analysis, Techniques and Specification. Springer Science & Business Media, New York, NY, USA. https://doi.org/10.1007/978-1-4614-4271-4

Chang SW, Kim DS, 2024. Scalable Transformer accelerator with variable systolic array for multiple models in voice assistant applications. *Electronics*, 13(23):4683. https://doi.org/10.3390/electronics13234683

Chen YH, Emer J, Sze V, 2016. Eyeriss: a spatial architecture for energy-efficient dataflow for convolutional neural networks. *ACM SIGARCH Comput Archit News*, 44(3):367-379. https://doi.org/10.1145/3007787.3001177

Chen YH, Yang TJ, Emer J, et al., 2019. Eyeriss v2: a flexible accelerator for emerging deep neural networks on mobile devices. *IEEE J Emerg Sel Top Circ Syst*, 9(2):292-308. https://doi.org/10.1109/JETCAS.2019.2910232

Chen YJ, Luo T, Liu SL, et al., 2014. DaDianNao: a machine-learning supercomputer. 47th Annual IEEE/ACM Int Symp on Microarchitecture, p.609-622. https://doi.org/10.1109/MICRO.2014.58

Chen YJ, Chen TS, Xu ZW, et al., 2016. DianNao family: energy-efficient hardware accelerators for machine learning. *Commun ACM*, 59(11):105-112. https://doi.org/10.1145/2996864

Gao MY, Pu J, Yang X, et al., 2017. TETRIS: scalable and efficient neural network acceleration with 3D memory. *ACM SIGARCH Comput Archit News*, 45(1):751-764. https://doi.org/10.1145/3093337.303770

Genc H, Kim S, Amid A, et al., 2021. Gemmini: enabling systematic deep-learning architecture evaluation via full-stack integration. Proc 58th Annual Design Automation Conf, p.769-774. https://doi.org/10.1109/DAC18074.2021.9586216

Guo C, Xue FC, Leng JW, et al., 2024. Accelerating sparse DNNs based on tiled GEMM. *IEEE Trans Comput*, 73(5):1275-1289. https://doi.org/10.1109/TC.2024.3365942

Hoefler T, Alistarh D, Ben-Nun T, et al., 2021. Sparsity in deep learning: pruning and growth for efficient inference and training in neural networks. *J Mach Learn Res*, 22(1):124.

Inayat K, Muslim FB, Iqbal J, et al., 2023. Power-intent systolic array using modified parallel multiplier for machine learning acceleration. *Sensors*, 23(9):4297. https://doi.org/10.3390/s23094297

Inayat K, Muslim FB, Mahmood T, et al., 2024. FPGA-assisted design space exploration of parameterized AI accelerators: a Quickloop approach. *J Syst Archit*, 155:103260. https://doi.org/10.1016/j.sysarc.2024.103260

Jouppi NP, Young C, Patil N, et al., 2017. In-datacenter performance analysis of a tensor processing unit. Proc 44th Annual Int Symp on Computer Architecture, p.1-12. https://doi.org/10.1145/3079856.3080246

Kim B, Lee S, Trivedi AR, et al., 2020. Energy-efficient acceleration of deep neural networks on realtime-constrained

embedded edge devices. *IEEE Access*, 8:216259-216270. https://doi.org/10.1109/ACCESS.2020.3038908

Lai CT, Zhang W, 2024. gem5-NVDLA: a simulation framework for compiling, scheduling and architecture evaluation on AI system-on-chips. *ACM Trans Des Autom Electr Systt*, 29(5):84. https://doi.org/10.1145/3661997

Lee J, Kim C, Kang S, et al., 2018. An Energy-Efficient Unified Deep Neural Network Accelerator with Fully-Variable Weight Precision for Mobile Deep Learning Applications. https://old.hotchips.org/hc30/3posters/An_EnergyEfficient_Unified_Deep_Neural_Network_Accelerator.pdf [Accessed on Sept. 20, 2024].

Li WQ, Liu TY, Xiao ZY, et al., 2023. TCADer: a tightly coupled accelerator design framework for heterogeneous system with hardware/software co-design. *J Syst Archit*, 136:102822. https://doi.org/10.1016/j.sysarc.2023.102822

Loh J, Dudchenko L, Viga J, et al., 2025. Towards hardware supported domain generalization in DNN-based edge computing devices for health monitoring. *IEEE Trans Biomed Circ Syst*, 19(1):5-15. https://doi.org/10.1109/TBCAS.2024.3418085

Lym S, Erez M, 2020. FlexSA: flexible systolic array architecture for efficient pruned DNN model training. https://doi.org/10.48550/arXiv.2004.13027

Moghaddasi I, Nam BG, 2024. Enhancing computation-efficiency of deep neural network processing on edge devices through serial/parallel systolic computing. *Mach Learn Knowl Extr*, 6(3):1484-1493. https://doi.org/10.3390/make6030070

Moons B, De Brabandere B, Van Gool L, et al., 2016. Energy-efficient ConvNets through approximate computing. IEEE Winter Conf on Applications of Computer Vision, p.1-8. https://doi.org/10.1109/WACV.2016.7477614

Moons B, Uytterhoeven R, Dehaene W, et al., 2017a. 14.5 envision: a 0.26-to-10TOPS/W subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28nm FDSOIF. IEEE Int Solid-State Circuits Conf, p.246-247. https://doi.org/10.1109/ISSCC.2017.7870353

Moons B, Uytterhoeven R, Dehaene W, et al., 2017b. DVAFS: trading computational accuracy for energy through dynamic-voltage-accuracy-frequency-scaling. Design, Automation & Test in Europe Conf & Exhibition, p.488-493. https://doi.org/10.23919/DATE.2017.7927038

Muslim FB, Inayat K, Khan S, 2024. LPCHISEL: automatic power intent generation for a chisel-based ASIC design. *Comput Electr Eng*, 115:109115. https://doi.org/10.1016/j.compeleceng.2024.109115

Park M, Hwang S, Cho H, 2024. BIRD: bi-directional input reuse dataflow for enhancing depthwise convolution performance on systolic arrays. *IEEE Trans Comput*, 73(12):2708-2721. https://doi.org/10.1109/TC.2024.3449103

Qamar A, Muslim FB, Iqbal J, et al., 2017. LP-HLS: automatic power-intent generation for high-level synthesis based hardware implementation flow. *Microproc Microsyst*, 50:26-38. https://doi.org/10.1016/j.micpro.2017.02.002

Ryu S, Kim H, Yi W, et al., 2019. BitBlade: area and energy-efficient precision-scalable neural network accelerator with bitwise summation. Proc 56th ACM/IEEE Design Automation Conf, p.1-6. https://doi.org/10.1145/3316781.3317784

Seshadri K, Akin B, Laudon J, et al., 2022. An evaluation of edge TPU accelerators for convolutional neural networks. IEEE Int Symp on Workload Characterization, p.79-91. https://doi.org/10.1109/IISWC55918.2022.00017

Song J, Cho Y, Park JS, et al., 2019. 7.1 an 11.5 TOPS/W 1024-MAC butterfly structure dual-core sparsity-aware neural processing unit in 8nm flagship mobile SoC. IEEE Int Solid-State Circuits Conf, p.130-132. https://doi.org/10.1109/ISSCC.2019.8662476

Xu R, Ma S, Wang YH, et al., 2021a. Configurable multi-directional systolic array architecture for convolutional neural networks. *ACM Trans Archit Code Optim*, 18(4):42. https://doi.org/10.1145/3460776

Xu R, Ma S, Wang YH, et al., 2021b. Heterogeneous systolic array architecture for compact CNNs hardware accelerators. *IEEE Trans Parallel Distrib Syst*, 33(11):2860-2871. https://doi.org/10.1109/TPDS.2021.3129647

Xu R, Ma S, Guo Y, et al., 2023. A survey of design and optimization for systolic array-based DNN accelerators. *ACM Comput Surv*, 56(1):20. https://doi.org/10.1145/3604802

Yüzügüler AC, Sönmez C, Drumond M, et al., 2023. Scale-out systolic arrays. *ACM Trans Archit Code Optim*, 20(2):27. https://doi.org/10.1145/3572917