ViTA: A Vision Transformer Inference Accelerator for Edge Applications

Shashank Nag *, Gourav Datta †, Souvik Kundu ‡, Nitin Chandrachoodan *, Peter A. Beerel †
* *Department of Electrical Engineering, Indian Institute of Technology Madras*
† *Ming Hsieh Department of Electrical and Computer Engineering, University of Southern California*
‡ *Intel Labs, USA*
{*shashank@smail , nitin@ee*}*.iitm.ac.in* , {*gdatta , pabeerel*}*@usc.edu , souvikk.kundu@intel.com*

***Abstract*—Vision Transformer models, such as ViT, Swin Transformer, and Transformer-in-Transformer, have recently gained significant traction in computer vision tasks due to their ability to capture the global relation between features which leads to superior performance. However, they are compute-heavy and difficult to deploy in resource-constrained edge devices. Existing hardware accelerators, including those for the closely-related BERT transformer models, do not target highly resource-constrained environments. In this paper, we address this gap and propose ViTA - a configurable hardware accelerator for inference of vision transformer models, targeting resource-constrained edge computing devices and avoiding repeated off-chip memory accesses. We employ a head-level pipeline and inter-layer MLP optimizations, and can support several commonly used vision transformer models with changes solely in our control logic. We achieve nearly 90% hardware utilization efficiency on most vision transformer models, report a power of 0.88W when synthesised with a clock of 150 MHz, and get reasonable frame rates - all of which makes ViTA suitable for edge applications.**

***Index Terms*—Vision Transformer, Swin Transformer, Hardware Accelerator, Computer Vision, Edge Computing, FPGA**

I. Introduction

The success of transformer models for NLP applications has led to self-attention-based models being applied to computer vision tasks. Although the initial works in this direction lacked scalability, subsequent works such as Vision Transformers (ViT) [1], Swin Transformers [2], TNT [3], and Data-efficient image Transformers (DeiT) [4] that adopted model architectures similar to the NLP-based Transformer [5], replacing word tokens with image patches, have yielded state-of-the-art (SOTA) results.

For some applications, such as autonomous driving and drone navigation, computer vision tasks have demanded real-time implementations on the edge. This has led to the development of energy-efficient hardware accelerators such as Eyeriss [6], [7], ShiDianNao [8], [9] for inferences of traditional CNN-based models. With vision transformer models outperforming the conventional CNN-based models, exploring energy-efficient hardware accelerators for ViTs, which could be implemented at the edge, is similarly important. Since the vision transformers are closely related to the BERT transformer architecture (by Vaswani et al. [5]), key insights can be derived from their acceleration. However, it has to be noted that most of the proposed accelerators for NLP tasks, including [10], [11] do not target edge devices, and their model parameters are quite different from that of the vision transformers.

This work was supported in part by IUSSTF and USC Viterbi under the IUSSTF Viterbi India Program 2022.

FPGAs for edge applications are useful because they provide flexibility in computation while still enabling very low latencies. For example, the Zynq Multiprocessor System-on-Chip (MPSoC) ZC7020 [12] and low-end Ultrascale MPSoC ZU3EG [13], [14] are some FPGAs that can be used in applications such as drones and similar low-energy vision oriented tasks. The defining characteristics of such platforms are limited parallelism (100s of DSP units rather than 1000s) and limited on-chip memory (100s of KB of Block RAM (BRAM) rather than MB). Naturally, this means any application targeting such platforms should focus on using the limited available parallelism and minimizing the amount of data transfer to and from off-chip memory.

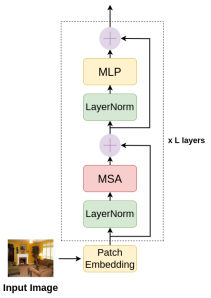
In this work, we propose ViTA - a hardware accelerator architecture and an efficient dataflow that supports several popular vision transformer models, targeting such resource-constrained FPGA devices. We evaluate the performance of ViTA for these models with commonly used configurations in terms of both hardware utilization efficiency and throughput.

The remaining of this paper is organized as follows. We provide a primer of the different vision transformer models, the computations involved therein, and existing related hardware accelerators in Section II. We propose ViTA - an architecture suitable for edge devices and an efficient dataflow within typical memory bandwidth constraints in Section III. We evaluate our design for different model architectures and configurations in Section IV. Section V concludes the paper.

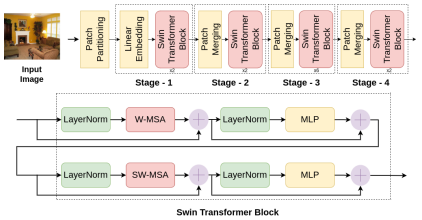
II. Preliminaries and Related Work

The Vision Transformer (ViT) [1] is one of the first proposed transformer-based models for vision tasks, and is similar to the encoder stack of the BERT transformer [5]. In ViT, the input image is split into patches of 16×16 pixels, which constitute a linear sequence of tokens, similar to words in the case of BERT. Its success motivated the development of other transformer-based models, such as the Swin Transformer [2], Data-efficient image Transformers [4], and Transformer-in-Transformer [3]. The key operation of all these models is the Multi-head Self Attention (MSA), applied on the sequence of image patches, followed by fully connected layers. The variation among these models is related to how these attention blocks are applied to the input patches, and the model parameters such as latent space dimensions, patch sizes, and the number of heads, for each of their variants. Figure 1 illustrates some of these model architectures.

There have been a few prior works on hardware accelerators for vision transformers [15] [16] [17]. Wang et al. [15] target



(a) ViT / DeiT



(b) Swin Transformer - two successive Swin Transformer blocks have a windowed and shifted windowed MSA

Fig. 1: Vision Transformers under consideration. The Swin Transformer has the same set of blocks as ViT, but the MSA is applied on disjoint windows on the image, and the patch merging blocks scales the image dimensions down in successive stages.

an application-specific integrated circuit (ASIC), but do not consider any off-chip data movement optimizations. Li et al. [16] and Sun et al. [17] target a large FPGA device, which enables storing all the weights and activations on-chip, thereby allowing a larger design space for exploration. However, edge computing devices, such as the Zynq ZC7020 MPSoC, pose more severe design constraints, owing to the lower on-chip computational resources, lower BRAM memory, and low bandwidth for off-chip access. In particular, off-chip memory accesses typically involve high energy and should be minimized.

As could be seen from the model architectures, the vision transformer models are quite similar to the BERT transformer [5]. There have been prior works on hardware accelerators targeting different stages of the BERT transformer. Liu et al. [18] proposed an accelerator for a quantized BERT model, and Lu et al. [10] proposed an accelerator design for the MSA and feed-forward blocks, with dedicated computation units for Softmax and LayerNorm. However, since NLP applications are typically not deployed at the edge, these target higher-end FPGA devices, and do not consider data movement from off-chip storage devices. ReTransformer [19] proposed a Re-RAM based in-memory computation engine with granular pipeline to accelerate the self-attention stage. In-memory computations typically pose additional limitations in terms of correctly aligning the results into the in-memory compute unit, which can be avoided in FPGA-based digital designs. However, the granular pipeline aspect of this work could potentially reduce the intermediate memory requirements while ensuring high efficiency, and we could explore this aspect in our design.

III. Proposed Architecture

A. Model Architecture

We base our design and analysis on the ViT-B/16 model, and in Section IV show how it can be extended to other vision transformers. This model considers a patch size $(P \times P)$ where P=16, having L=12 layers, a latent vector dimension (D) of 768, k=12 heads, a head level latent vector dimension $(D_h = D/k)$ of 64, and an MLP hidden layer dimension (M) of 3072. We consider an image dimension $(H \times W)$ of 256×256, thus giving the sequence length (N) as $(H.W)/P^2$ = 256.

We follow a post-training quantization approach, with all the weights and activations quantized to int8 representations for inference. We observe that when evaluated on ViT, this results in almost no degradation ($<0.04\%$) in the top-1 test accuracy on the ImageNet [20] dataset. We target the Zynq ZC7020 MPSoC for our design.

B. Hardware Architecture and Dataflow

As noted previously, targeting edge computing devices poses multiple constraints. Table I shows the memory requirements for the ViT-B model, which is beyond the typical on-chip memory capacity of edge FPGA devices, indicated in Table II. This necessitates most of the data to be stored in off-chip DRAMs, and brought into the on-chip BRAM cells when required, while minimizing back-and-forth data movement.

Input/Weights	Input	W^Q	W^K	W^V	MSA Weights
Memory (in KB)	192	576	576	576	576

TABLE I: Memory requirements for the input activations and weights for the ViT-B/16 model. The memory requirements for intermediate results are significantly higher.

	Zynq ZC7020 MPSoC	Zynq ZCU102
LUT6	53200	274080
DSP slices	220	2520
BRAM	630 KB	4 MB

TABLE II: Resources available on the Zynq ZC7020 MPSoC (an embedded platform) vs. the Zynq ZCU102 MPSoC (a high-end platform)

Most of the computations involved in the inference of vision transformers are matrix multiplication operations. Different schemes such as input-stationary, weight-stationary, or block multiplication can be adopted to meet our on-chip memory constraints. We note that in all the vision transformer model architectures in Fig. 1, multiple layers of self attention and MLP blocks are stacked one after the other. While the weights are different for each of these layers, the input activations for a layer are passed from the previous layer. This suggests that an input stationary scheme, with the weights being loaded onto the chip would be ideal. Once the input activations are loaded, the results of each layer are stored in the same on-chip location, and only after all the layers are computed, the final results are written back to the off-chip memory. Given this arrangement, it would be ideal to hide the off-chip memory access latency involved in fetching the weights, by scheduling computations and weight fetching appropriately. Our proposed approach is to implement this schedule at a column level of the weight matrix. For each weight matrix, BRAM cells that can store two columns of weights are allocated. The access latency can be hidden by ensuring that while one column of the weights is being operated upon, the other column is fetched from off-chip memory.

1) Architecture for MLP Layer: As noted in Table III, the major portion of MAC computations is in the MLP layer, accounting for nearly 60% of the total MAC operations across the different model variants. Thus, it is pertinent to accelerate this computation efficiently while designing the accelerator. The design for a simple feed-forward layer could be trivial, with the same weights and biases being applied on multiple

Model	Image Dim.	MSA	MLP	Patch Merging
ViT-B/16 [1]	$256 \times 256 \times 3$	36.8%	63.2%	-
ViT-B/16 [1]	$224 \times 224 \times 3$	36.1%	63.9%	-
DeiT-S [4]	$224 \times 224 \times 3$	38.6%	61.4%	-
DeiT-T [4]	$224 \times 224 \times 3$	43.1%	56.9%	-
Swin-T [2]	$224 \times 224 \times 3$	31.9%	63.8%	4.3%

TABLE III: Number of MAC operations in the MSA, MLP, and Patch Merging layers as a fraction of the total number of MAC operations involved for the considered models. We ignore the Softmax, LayerNorm, and Residual Connection layers.

input activations concurrently. However, we observe that the memory required to store the hidden layer, with dimensions of 256×3072 for the ViT-B model, is beyond the total on-chip memory capacity of the target device. To avoid repeated DRAM accesses, we employ the inter-layer optimization technique proposed in [21], using two sets of MAC units. As elaborated in Fig. 3, the hidden layer values computed in the first set of MAC units are broadcast to the second set of MAC units through the non-linear activation, to compute the partial products corresponding to the output layer.

This leads to one of the key ideas that inspire our design. In order to process this in a pipelined fashion, we allocate resources such that the hidden layer value computations and the output layer partial product computations take approximately equal times. This ensures they can be pipelined with minimal stalls. This implies that we should dedicate an equal number of MAC units to compute the hidden and output layers. While the MACs dedicated to the hidden layer accumulate the results over multiple cycles, those dedicated to the output layer compute different partial products in each of those cycles.

2) Architecture for MSA Layer: For all the model architectures illustrated in Table III, the self attention operation also accounts for about 30-40% of the total computations. Consequently, we try to optimize the hardware accelerator design to suit these computations, while adhering to the above scheme for the MLP layer. The MSA layer involves the following computations, where z refers to the input to the layer, i to the head number, and k to the total number of heads :

$$[Q|K|V]_i = z.[W^Q|W^K|W^V]_i \quad , W^{Q/K/V} \in R^{D \text{x} D_h}$$

$$SA_i(z) = \text{Softmax}(Q_i K_i^T / \sqrt{D_h}).V_i$$

$$MSA(z) = [SA_1(z), ..., SA_k(z)].W^{msa} \quad , W^{msa} \in R^{D \text{x} D}$$

As illustrated in the above equations, the MSA block involves computing self attention on each head, and finally concatenating the results. Although the operations on each head can be parallelized, the on-chip resource constraints dominate the design choice. Processing multiple heads implies that the computed values (say Q, K, and V) for all the heads will have to be staged until the next set of values ($Q.K^T$, $Softmax$, and $Softmax.V$) are computed. Given the on-chip memory constraints, storing these intermediate matrices for all the heads is not feasible. Hence, to avoid unnecessary data movement to and from off-chip, we instead perform head-wise computations. Even in the case of head-wise computation, we have a few dataflow design choices for the weights and the product computations. As detailed in Sec II, there have been several works focused on accelerating the MSA block, such as the ReTransformer [19], which proposes a row-level fine-grained pipeline for efficiently computing the self attention block using two PE engines. However, since the input matrix has been held stationary, the remaining available memory is not sufficient to hold the required weights for computing the Q, K, and V for a head. To avoid repeated off-chip accesses, we, in contrast, store the weights one column at a time and proceed to the next column once all rows of input activations are multiplied with this column. As a consequence of performing the multiplications column-wise, we can only compute the Q, K, and V matrices in a column-wise fashion, and hence the QK^T and further computations for the head cannot occur until both Q and K matrices are completely computed. To optimize the routing congestion and enable a streamlined dataflow for the weights from the off-chip memory, we propose a head-level coarse-grained pipeline with two dedicated sets of processing units with near-equal latency.

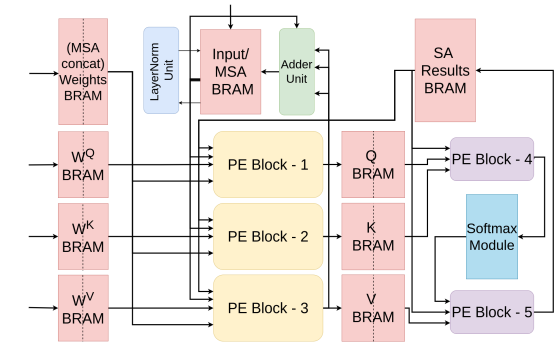
3) Overall Architecture and Optimal Configuration: Drawing conclusions from the observations and implications made on accelerating the MLP and self-attention layer, we propose a configurable processing element array-based hardware accelerator and the corresponding dataflow, illustrated in Figs. 2 and 4. The PE blocks 1, 2, and 3 together make up the first compute engine that performs the computations for generating Q, K, and V, while the PE blocks 4 and 5 form the second compute engine that performs the QK^T and $Softmax.V$ operation, respectively. We reuse the same PE blocks for computing the MSA concatenation and the MLP block. The condition required for efficient acceleration of the MLP layer is to have equal MAC units dedicated to computing the hidden layer and output layer. This is ensured by using half the rows in the PE blocks for computing the hidden layer and the other half for the output layer. Dedicated units for SoftMax, LayerNorm, skip connections and non-linear activations have been included, with the former adapted from [10].

As shown in Fig. 2, $k_1 \times k_2$ refers to the configuration of the PE block type I, while $k_3 \times k_4$ refers to the configuration of the PE block type II. As illustrated in Fig. 4, in order to time match the computations in the two compute engines and enable a head wise pipeline, the optimal values of k_1, k_2, k_3, and k_4 should satisfy $\frac{D}{k_1.k_2} = \frac{N}{k_3.k_4}$, subject to the on-chip resource constraints. Consequently, while PE Blocks 1, 2, and 3 operate on head h, PE Blocks 4 and 5 operate on head $h-1$. Similarly, PE Block 4, Softmax Module and PE Block 5 operate at a row granularity within a head. The optimal values chosen for the ViT-B/16 256×256 configuration are k_1=16, k_2=6, k_3=8, and k_4=4.

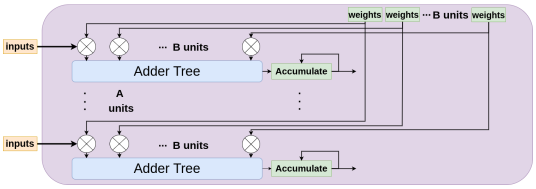
In the next section, we benchmark the performance of this architecture the considered vision transformer model architectures and compare it against other works on accelerating vision transformers.

IV. Analysis & Experiments

We evaluate the performance of ViTA across different model architectures in terms of the hardware utilization efficiency (HUE) [22] of the utilized resources on the FPGA. We choose a design configuration that works best for the ViT-B/16 model, while also restricting ourselves to the resources available on the Zynq ZC7020 MPSoC. This corresponds to the PE blocks



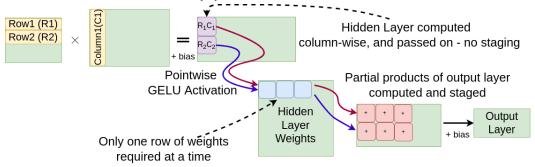
(a) Overall Hardware Accelerator Design



(b) PE unit architecture - the rows share weights. For PE Blocks - 1, 2 & 3, $A{=}k_1$ & $B{=}k_2$, and for PE Blocks 4 & 5, $A{=}k_3$ & $B{=}k_4$

Fig. 2: ViTA : Proposed design for the hardware accelerator. The weight BRAMs are split into two halves as indicated, with one of them acting as a buffer where a column is being loaded from off-chip, while the column in the other is being used for computations.

(a) MLP Block



(b) Scheduling MLP operations in two stages

Fig. 3: Inter-layer optimization for MLP

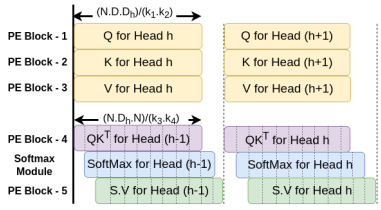


Fig. 4: Scheduling the MSA computations. PE Block 4, Softmax Module, and PE Block 5 process in a row-granular pipeline fashion.

1, 2, and 3 having a $k_1 \times k_2$ (16×6) configuration, while the PE blocks 4 and 5 have a $k_3 \times k_4$ (8×4) configuration. We rely on the LUTs, rather than DSP slices, for compute capability, and ensure that all the memory can be held and accessed as intended from the BRAM cells. Although the DSP slices are power efficient, they are limited in number for the considered FPGA. Moreover, these are optimized for 18bits $\times$ 27bits operations, and performing 8bits $\times$ 8bits operations on these would lead to significant under-utilization.

Although the design parameters of the PE array can be configured at run time, we demonstrate how having a single fixed configuration also allows for a run-time selection of the desired model and input image size, without a significant drop in efficiency. Since the other vision transformer models are typically made up of the MSA and MLP blocks, we can infer any of these models on ViTA by changing the control logic appropriately. For the case of Swin transformer, the W-MSA is performed on windows of $M \times M$, where $M{=}7$ by default. This would just correspond to the regular MSA being performed on $N{=}49$ repeatedly over these windows.

When synthesized on the Zynq ZC7020 MPSoC, the design is run at a clock of 150 MHz, and consumes a power of 0.88W for the given configuration. The obtained values of HUE and energy for processing an image are summarized in Table IV. We ensure that the latency in accessing the off-chip memory is hidden in all intermediate cases, with the DRAM access bandwidth well under 1 word/cycle - a reasonable number for the considered FPGA.

Model	Image Dim.	HUE	fps	Energy (J)
ViT-B/16	$256 \times 256 \times 3$	93.2%	2.17	0.406
ViT-B/16 or DeiT-B	$224 \times 224 \times 3$	92.8%	2.75	0.320
DeiT-S	$224 \times 224 \times 3$	87.2%	9.36	0.094
DeiT-T	$224 \times 224 \times 3$	66.2%	19.01	0.046
Swin-T	$224 \times 224 \times 3$	81%	8.71	0.101

TABLE IV: Overall hardware utilization efficiency (HUE), Frame Rate (fps) and energy for processing one image for different model architectures - for the optimal architecture configuration chosen for ViT-B/16 with a 256×256×3 image dimensions

Table V compares the performance our design against other vision transformer accelerator designs. As illustrated, ViTA achieves a significant reduction in power consumption compared to the other works. Since ViTA is focused on highly resource-constrained environments, it is difficult to compare it against other designs in terms of the processing frame rate. However, for the lower compute capability and technology node of our target device, the performance scales comparably with other works. Furthermore, the frame rates achieved by ViTA for the smaller model variants, such as DeiT-S, DeiT-T, and Swin-T are reasonable for embedded applications, such as drone navigation. Moreover, an FPGA environment makes ViTA reconfigurable for different model configurations for improved performance. This coupled with the energy efficiency of ViTA makes it a good choice for deployment on the edge.

Accelerator Design	Target Device	Power (W)	fps	fps/W
Row-wise-acc. [15]	ASIC (40nm)	*	44.5	*
Auto-vit-acc. [16]	FPGA (16nm)	9.40	25.9	2.76
ViTA (ours)	FPGA (28nm)	**0.88**	2.75	3.12

* not reported

TABLE V: Performance comparison of vision transformer accelerators for DeiT-B on $224 \times 224 \times 3$ dimensioned images

V. Conclusions

We proposed ViTA - a configurable hardware accelerator and dataflow design that can be employed for inference of ViT models on resource-constrained edge devices. In particular, we introduced a head-level coarse-grained pipeline and performed inter-layer optimization on the MLP layer to avoid unnecessary intermediate result staging. Our design avoids repeated off-chip memory accesses, achieves a high resource utilization efficiency of about 90%, and reports a significantly low power of 0.88W while having a reasonable frame rate - making it well suited for various edge applications.

References

[1] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[2] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," 2021. [Online]. Available: https://arxiv.org/abs/2103.14030

[3] K. Han, A. Xiao, E. Wu, J. Guo, C. Xu, and Y. Wang, "Transformer in transformer," 2021. [Online]. Available: https://arxiv.org/abs/2103.00112

[4] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jegou, "Training data-efficient image transformers amp; distillation through attention," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 10 347–10 357. [Online]. Available: https://proceedings.mlr.press/v139/touvron21a.html

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. [Online]. Available: http://arxiv.org/abs/1706.03762

[6] Y.-H. Chen, J. Emer, and V. Sze, "Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, 2016, pp. 367–379.

[7] Y.-H. Chen, T.-J. Yang, J. Emer, and V. Sze, "Eyeriss v2: A flexible accelerator for emerging deep neural networks on mobile devices," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 2, pp. 292–308, 2019.

[8] Z. Du, R. Fasthuber, T. Chen, P. Ienne, L. Li, T. Luo, X. Feng, Y. Chen, and O. Temam, "Shidiannao: Shifting vision processing closer to the sensor," in *2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, 2015, pp. 92–104.

[9] W. Huang, H. Wu, Q. Chen, C. Luo, S. Zeng, T. Li, and Y. Huang, "Fpga-based high-throughput cnn hardware accelerator with high computing resource utilization ratio," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 8, pp. 4069–4083, 2022.

[10] S. Lu, M. Wang, S. Liang, J. Lin, and Z. Wang, "Hardware accelerator for multi-head attention and position-wise feed-forward in the transformer," in *2020 IEEE 33rd International System-on-Chip Conference (SOCC)*, 2020, pp. 84–89.

[11] B. Li, S. Pandey, H. Fang, Y. Lyv, J. Li, J. Chen, M. Xie, L. Wan, H. Liu, and C. Ding, "Ftrans: energy-efficient acceleration of transformers using fpga," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, 2020, pp. 175–180.

[12] P. Li and C. Che, "Mapping yolov4-tiny on fpga-based dnn accelerator by using dynamic fixed-point method," in *2021 12th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP)*, 2021, pp. 125–129.

[13] H.-s. Suh, J. Meng, T. Nguyen, S. K. Venkataramanaiah, V. Kumar, Y. Cao, and J.-s. Seo, "Algorithm-hardware co-optimization for energy-efficient drone detection on resource-constrained fpga," in *2021 International Conference on Field-Programmable Technology (ICFPT)*, 2021, pp. 1–9.

[14] Y. Yang, Q. Huang, B. Wu, T. Zhang, L. Ma, G. Gambardella, M. Blott, L. Lavagno, K. Vissers, J. Wawrzynek, and K. Keutzer, "Synetgy: Algorithm-hardware co-design for convnet accelerators on embedded fpgas," in *Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, ser. FPGA '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 23–32. [Online]. Available: https://doi.org/10.1145/3289602.3293902

[15] H.-Y. Wang and T.-S. Chang, "Row-wise accelerator for vision transformer," in *2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 2022, pp. 399–402.

[16] Z. Li, M. Sun, A. Lu, H. Ma, G. Yuan, Y. Xie, H. Tang, Y. Li, M. Leeser, Z. Wang, X. Lin, and Z. Fang, "Auto-vit-acc: An fpga-aware automatic acceleration framework for vision transformer with mixed-scheme quantization," 2022. [Online]. Available: https://arxiv.org/abs/2208.05163

[17] M. Sun, H. Ma, G. Kang, T. Chen, X. Ma, Z. Wang, and Y. Wang, "Vaqf: Fully automatic software-hardware co-design framework for low-bit vision transformer," 01 2022.

[18] Z. Liu, G. Li, and J. Cheng, "Hardware acceleration of fully quantized bert for efficient natural language processing," 2021. [Online]. Available: https://arxiv.org/abs/2103.02800

[19] X. Yang, B. Yan, H. Li, and Y. Chen, "Retransformer: Reram-based processing-in-memory architecture for transformer acceleration," in *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2020, pp. 1–9.

[20] J. Deng and Dothers, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR*, 2009.

[21] S. Chen and Z. Lu, "Hardware acceleration of multilayer perceptron based on inter-layer optimization," in *2019 IEEE 37th International Conference on Computer Design (ICCD)*, 2019, pp. 164–172.

[22] J. Jin and C.-Y. Tsui, "Improving the hardware utilization efficiency of partially parallel ldpc decoder with scheduling and sub-matrix decomposition," in *2009 IEEE International Symposium on Circuits and Systems*, 2009, pp. 2233–2236.