Are Conventional SNNs Really Efficient? A Perspective from Network Quantization

Supplementary Material

A. Experimental Details

A.1. Static Datasets

In order to verify the efficiency of QSNN, we performed a series of comprehensive evaluations on several datasets. We describe these datasets in more detail below.

ImageNet [5] With over a million categorized images spread across $1,000$ diverse categories, ImageNet stands as a colossal and pivotal dataset. The images in ImageNet, which cover a wide spectrum from tangible items to abstract notions, demonstrate considerable variations in attributes like scale, pose, and illumination. Consequently, it provides a rigorous testing ground for diverse computer vision tasks and serves as an industry-standard benchmark.

CIFAR [14] The CIFAR datasets, particularly CIFAR10 and CIFAR100, are quintessential datasets in the field of machine learning, commonly employed for assessing image classification algorithms. CIFAR10 comprises $60,000$ 32×32 color images, evenly distributed across 10 distinct categories, with each category containing $6,000$ images. In contrast, CIFAR100 maintains the same overall number of images and resolution but is divided into 100 categories, each containing 600 images, offering a finer granularity of classification.

A.2. Neuromorphic Datasets

Neuromorphic datasets provide a benchmark for assessing algorithms tailored to neuromorphic hardware, which often deal with spiking neural networks (SNNs) and event-driven data. Such datasets capture real-world, dynamic visual information in a format suitable for SNNs, emphasizing the importance of time and asynchronous events in information processing.

CIFAR10-DVS [16] The CIFAR10-DVS dataset is a neuromorphic version of the popular CIFAR10 dataset. Instead of static images, CIFAR10-DVS provides sequences of events generated by a Dynamic Vision Sensor (DVS), a type of neuromorphic camera that only captures pixel-level brightness changes, making it more power-efficient and better suited for real-time applications.

DVS-Gesture [1] The DVS-Gesture dataset is a benchmark collection in the field of neuromorphic vision, designed for the task of gesture recognition. The DVS-Gesture dataset includes a variety of hand gestures from multiple subjects, performed under different lighting conditions, and from various angles, challenging the robustness of spiking neural network (SNN) models in recognizing and classifying dynamic patterns.

N-Caltech101 [27] The N-Caltech101 dataset is a neuromorphic version of the well-known Caltech101 dataset, which has been converted using a DVS camera to create event-based vision data. Unlike the original dataset with static images, N-Caltech101 provides a sequence of events as each image is presented to the sensor, capturing the temporal aspect of visual perception. This dataset contains the same categories as the original Caltech101, encompassing a wide range of objects such as animals, vehicles, and everyday items, thus providing a comprehensive challenge for testing the effectiveness of SNNs in processing and classifying neuromorphic vision data.

A.3. Experimental Settings

Following the framework defined by Zhou et al. [43] for consistency and comparability, we set the input dimensions for the ImageNet dataset to 224×224. For other static image datasets, their native image dimensions were retained. The neural morphology datasets underwent a resizing process, adjusting the event streams to 48×48 to streamline computational demands. We used a batch size of 128 and selected the AdamW optimizer [23]. Training involved 310 epochs for the ImageNet dataset, whereas for other datasets, 400 epochs were deemed sufficient. We initialized our models with a learning rate of 0.0005.

Our experiments spanned various neural network architectures, prominently SEW-ResNet [7] and SpikFormer [43]. Although the architectural blueprints of our models mirrored the original specifications, we took the liberty of introducing modifications to accentuate their event-driven nature. This involved imposing bit width restrictions on the output of SEW-ResNet's blocks and SpikFormer's attention matrices. Our thorough evaluations then centered on the performances of S-ACE and NS-ACE for these models across our selected datasets.

A.4. Additional Evaluation

In Tab. 5, various methods and configurations showcase their performance on both CIFAR10 and CIFAR100. Burst[33], although demonstrating impressive performance across different configurations, retains a relatively large number of parameters due to its limited consideration of pulse patterns and its lack of quantization for model weights. Building upon Spikiformer [43], our further experiments reveal that models with multiple pulse patterns can better utilize the bit budget. When combined with

weight quantization, we managed to achieve an accuracy of 96.84% on CIFAR10 and 80.13% on CIFAR100 with just 1/4 of the original model parameters and 1/5 of the S-ACE. These results indicate a performance improvement of 1.33% and 1.92% respectively over the non-optimized methods.

From the neuromorphic results in Tab. 3, it's evident that as we modify the allocation between the spike patterns and simulation steps (S and T), there are significant performance trade-offs. The most notable decline in performance is observed for the DVSC10 dataset when adjusting the S/T ratio from 1/16 to 8/2, indicating a substantial sensitivity to the bit allocation strategy. This is further accentuated in the 1/16/1 configuration, leading to a staggering decrease of 44% in accuracy. However, the impact of weight bit-width on model performance is not particularly pronounced. This might be attributed to the relatively small dataset size of the neural morphology dataset, where weight bit-width may not be the predominant factor influencing model performance.

Building upon our previous discussions and the observed performance on the CIFAR dataset, these results further underscore the intricacy of employing spiking neural networks on the neural morphology dataset. The judicious selection and allocation of bits are of paramount importance when dealing with neural morphology data characterized by temporal dynamics.

B. Hardware Implementation

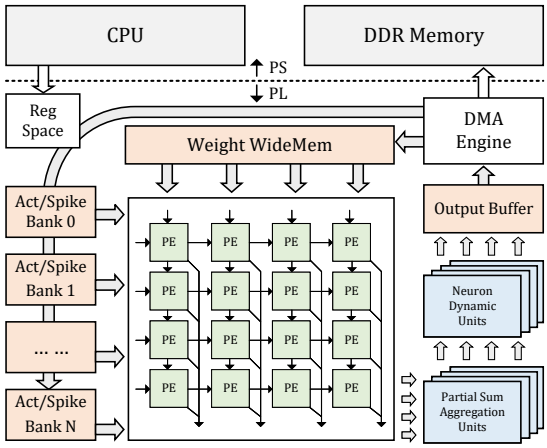

Figure 7. System block design of our hardware accelerator.

To assess the hardware efficacy of various bit allocation strategies, we developed a flexible FPGA accelerator prototype that has the flexibility to support different time steps and activation bit widths. Drawing inspiration from established advancements in bit-serial accelerator, the accelerator first breaks down multi-bit spikes across multiple time steps into several iterations of general matrix multiplication between binary inputs and multi-bit weights. These iterations' resulting partial sum matrices are then aggregated and transmitted to the neurodynamic units to generate the final output. In this way, the hardware efficiency of models with distinct bit allocation strategies can be assessed using the same hardware backend to ensure a fair and accurate comparison. Notably, bit-serial accelerators are prevalent in deep learning accelerator research due to their ability to provide a finer computation granularity, offering a more expansive optimization space. This includes the implementation of mixed bit-width quantization across different layers, enabling more extensive optimization opportunities.

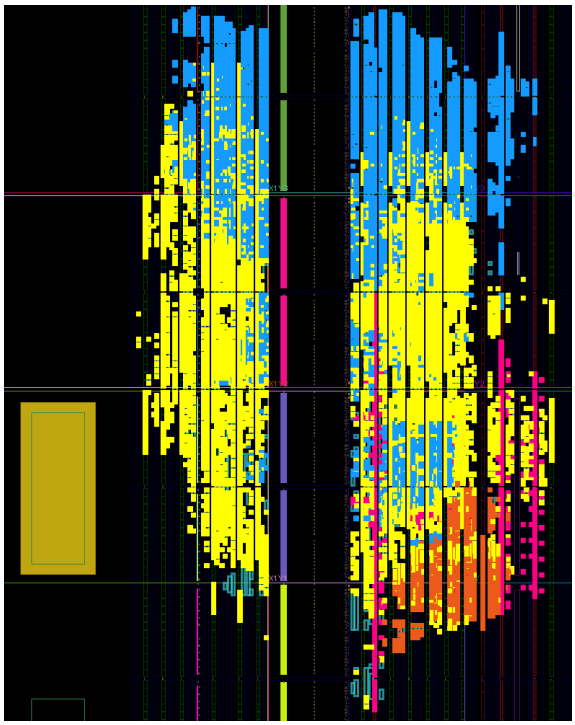

Figure 8. Place and route of the accelerator on xck26 FPGA device.

The system architecture of the accelerator is shown in Fig. 7. This accelerator is implemented on the Xilinx Zynq Ultrascale series FPGA platform, which features embedded host CPUs on the Processing System (PS) side. The PS manages task scheduling, while the Programmable Logic (PL) side handles heavy computational workloads within the FPGA fabric. The system diagram is depicted below. The DMA engine is responsible for input/output activation and weight data transfer between the PL logic and external DDR memory. To optimize data reuse and alleviate DDR bandwidth constraints, on-chip weight and input activation buffers are instantiated. Input buffers are split into several banks for a flexible data layout arrangement. A systolic array, consisting of Processing Elements (PEs) arranged in a

Table 5. Additional evaluation on CIFAR10/100. † represent results reproduced with the same experimental setup.

Methods	Architecture	Assignment b_w / b_s / T	Params (M)	S-ACE (G)	Accuracy CIFAR10	Accuracy CIFAR100
PLIF [8]	CIFARNet-Fang	16 / 1 / 8	0.58	3.57	93.50	74.36†
Diet-SNN [29]	ResNet-20	16 / 1 / 5	0.27	3.32	92.54	64.07†
tdBN [41]	ResNet-19	16 / 1 / 4	12.63	139	92.92	70.86
TET [6]	ResNet-19	16 / 1 / 4	12.63	139	94.44	74.47
Burst [33]	ResNet-19	16 / 2 / 1	12.63	69.6	95.94	77.86
	ResNet-19	16 / 2 / 2	12.63	139	96.01	78.04
	ResNet-19	16 / 2 / 4	12.63	278	96.21	78.12
	ResNet-19	16 / 2 / 6	12.63	417	96.32	78.31
Spikformer [43]	Spikformer-4-256	16 / 1 / 4	4.15	26.28	93.94	75.96
	Spikformer-2-384	16 / 1 / 4	4.15	44.50	94.80	76.95
	Spikformer-4-384	16 / 1 / 4	4.15	59.10	95.51	78.21
Quantized Spikformer	Spikformer-4-384	1 / 1 / 1	0.26	0.77	90.48	70.11
	Spikformer-4-384	1 / 4 / 1	0.26	3.08	95.00	76.90
	Spikformer-4-384	1 / 2 / 2	0.52	3.08	94.43	75.91
	Spikformer-4-384	1 / 1 / 4	0.26	3.69	93.91	74.13
	Spikformer-4-384	2 / 2 / 1	0.52	3.08	95.41	76.67
	Spikformer-4-384	2 / 1 / 2	0.52	3.09	93.56	75.91
	Spikformer-4-384	4 / 1 / 1	1.04	3.09	94.51	74.61
	Spikformer-4-384	4 / 4 / 1	1.04	12.32	96.84	80.13
	Spikformer-4-384	4 / 2 / 2	1.04	12.33	96.50	80.71
	Spikformer-4-384	4 / 1 / 4	1.04	14.77	95.94	78.77
	Spikformer-4-384	8 / 1 / 2	2.08	12.35	95.55	77.72
	Spikformer-4-384	8 / 2 / 1	2.08	12.33	96.29	80.00

two-dimensional grid, executes arithmetic-intensive matrix multiplications using binary inputs and multi-bit weights. Each PE comprises a multiplexer and an accumulator; the binary input controls the multiplexer to determine whether the accumulator adds the current weight. A central controller is designed to fetch the necessary inputs and weight data for the systolic array's operations. Following the matrix multiplications, a post-processing unit aggregates partial sums from each binary slice of inputs and generates output spikes through the neurodynamic units. Subsequently, the output data is organized in the output buffer and transferred back to the external DDR memory to serve as the input for the next layer.

The accelerator decomposes the multi-bit activation (or spike) tensor into binary slices across multiple time steps. The bit-width of the activations (or spikes) and the quantity of time steps determine the number of binary slices. Hence, it is unsurprising that SNNs and QANNs sharing the same bit budget will result in an identical computational workload using such a hardware backend.

The out-of-context Place and Route implementation results are depicted in Fig. 8. In the illustration, the highlighted bright yellow area signifies the systolic array. In contrast, the highlighted blue area encompasses the post-processing units responsible for the partial sum aggregation process and the neurodynamic process. The pink line demarcates the utilized block RAM for the weight buffer, whereas the orange area indicates the split input buffer banks. The dark yellow rectangle on the left represents the CPUs on the PS side of the Zynq Ultrascale device. The comprehensive resource utilization is detailed in Tab. 6.

Table 6. Utilization of different resources on the xck26 FPGA device.

Resource	Used	Total
Look Up Table	23298	117120
Flip Flops	44084	234240
DSP48E2	512	1248
Block Ram	89.5	144
Ultra Ram	8	64