23.4 Nebula: A 28nm 109.8TOPS/W 3D PNN Accelerator Featuring Adaptive Partition, Multi-Skipping, and Block-Wise Aggregation

Changchun Zhou[1], Tianling Huang[1], Yanzhe Ma[1], Yuzhe Fu[1], Xiangjie Song[1], Siyuan Qiu[1], Jiacong Sun[1], Min Liu[1], Ge Li[1], Yifan He[2], Yuchao Yang[1,3], Hailong Jiao[1]

[1]Peking University, Shenzhen, China
[2]Reconova Technologies, Xiamen, China
[3]Peking University, Beijing, China

Three-dimensional (3D) point clouds are increasingly deployed across various emerging fields, such as autonomous driving, robots, drones, and virtual reality (VR) [1]-[6]. Point-based point-cloud neural networks (PNNs) [3]-[6] have demonstrated superior performance in point-cloud analysis, compared to both sparse 3D convolution-based networks [7], [8] and graph-based convolutional neural networks [9], [10]. Due to the high computational complexity, low parallelism, and frequent irregular external memory accesses, deploying PNNs in hardware is a great challenge. PNN hardware accelerators have been developed [11]-[20]. However, three key challenges remain unsolved in these accelerators, as illustrated in Fig. 23.4.1. 1) The inherent farthest point sampling (FPS) features serial computation and suffers from quadratic growth in inference latency with rising point counts. The existing uniform block-wise FPS techniques [13], [21] fail to achieve a well-balanced block segmentation, due to a typically non-uniform point distribution. 2) A large amount of redundant operations exist for both discarded points (DPs) and retained points (RPs) in FPS. These operations exist in the sampling operations of RPs (① in Fig. 23.4.1), as well as grouping (②), convolution (③), and aggregation (④) for DPs, introducing unnecessary energy and latency costs. 3) The irregular memory accesses in the aggregation operation cause significant latency penalties. Channel-wise aggregation in [11] relieves irregularity, yet is unsuitable for large-scale point clouds, as the external memory access of features and the neighbor index table (NIT) is quadratically increased due to the iterative loading of features or the NIT.

To address the challenges above, an energy-efficient neural network accelerator, Nebula, is developed in this work for 3D point-cloud analysis. As depicted in Fig. 23.4.2, the overall architecture mainly includes seven modules: a partition unit, a sampling unit, a grouping unit, a convolution unit with 2 cores (each consists of a 16×16 PE array), an aggregation unit, a system controller, and a 64KB shared memory. Nebula has three key features: 1) A tree-based adaptive partitioning sampling (APS) technique hierarchically and adaptively partitions point clouds into blocks based on the 3D spatial density of points to reduce and balance the number of points in each block, thereby significantly reducing the latency. 2) A sampling-based multi-skipping strategy (SMS) skips redundant operations in two parts for RPs and DPs in PNNs. Firstly, for the sampling skipping of RPs (SSR), by freezing the initial point across all sampling layers, the calculation path of FPS is deterministic, allowing SMS to reversely skip sampling operations of already computed RPs (① in Fig. 23.4.2). Secondly, for sparse skipping of DPs (SSD), FPS-based sparse grouping (FSG) is presented in this work by replacing the random sampling of sparse grouping [22] with the inherent FPS of PNNs. This thereby enables the skipping of grouping (②), convolution (③), and aggregation (④) of DPs. 3) A pipelined block-wise delayed-aggregation (BDA) is introduced, which partitions delayed features and the corresponding NIT into blocks and aggregates the point features within each block. BDA reads features and NIT only once, eliminating the irregular loop-read (i.e. iterative loading) of exponentially growing features, and the loop-read of linearly increased NIT. BDA allows for pipelined execution of grouping, convolution, and aggregation in each block, thereby reducing latency. To evaluate the effectiveness of the presented techniques, PointNeXt [6], a state-of-the-art PNN based on the classical PointNet [3]/PointNet++ [4], classifying the ModelNet40 dataset [23] (PointNeXt-Cls@ModelNet40) and segmenting the Stanford 3D Indoor Scene (S3DIS) dataset [24] (PointNeXt-Seg@S3DIS) in 8b data width, are chosen as two evaluation benchmarks. The ModelNet40 dataset is a classical classification dataset with 40 object classes and 12311 samples, each of which contains 1k points. The S3DIS dataset contains 6 large-scale indoor areas with 271 rooms, a covered area of over 6,000m^2, and over 215 million points, captured from various indoor environments.

The APS technique performs partitioning and accelerates sampling, as shown in Fig. 23.4.3. The workflow of APS for sampling acceleration is divided into two steps: prediction and sampling. Sampling, guided by the average predicted number of points from multi-stream sparse blocks, is conducted on each block. The key of APS is partitioning, which determines the global structure of point clouds. At the beginning of the tree-based adaptive partitioning, the expected average number of points per block (*AP*) and the threshold number of points per block (*Pth*) are calculated based on the targeted number of blocks. At each level, if the number of points in a block exceeds *Pth*, the block is further partitioned at the next level, until the numbers of points in all blocks fall below *Pth*. *Pth* is always set larger than *AP* to prevent blocks with insufficient points from being further partitioned at the next level. In the hardware implementation of adaptive partitioning, the corresponding block ID of a point is computed by comparing the coordinates with block boundaries. If the number of points exceeds *Pth*, the address of the block is recorded, and the corresponding points are further partitioned at the next level. In the hardware implementation of sampling, 16 cores handle 16 blocks, respectively. A mask (see Fig. 23.4.3), in which “0” indicates RP, is used to skip redundant distance calculations between the already-generated RPs, thereby reducing the number of cycles in sampling. When evaluated on PointNeXt-Cls@ModelNet40, with the same number of blocks (e.g. 16 blocks), APS achieves a more balanced partition than [13], reducing the mean square error by 83.6%, improving the accuracy by 1.5%, and reducing the latency by 76.8%.

The sampling-based multi-skipping strategy (SMS) is applied to skip redundant operations and storage in PNNs, including sampling, grouping, aggregation, and convolution, as shown in Fig. 23.4.4. For sampling skipping, the RPs for all sampling layers are computed in reverse and output during the first sampling layer, allowing all subsequent sampling layers to be directly skipped. Suppose the point number in the input point cloud is N, the sampling rate is $1/R$ ($1<R\leq N$), and the number of sampling layers is M. The sampling unit sequentially outputs RPs from the 1st point to the last ($(N/R)^{th}$) point. Among these N/R points, the first N/R^L points are the RPs of the L^{th} sampling layer ($1<L\leq M$). For grouping skipping, after enabling FSG, operation skipping is performed in two types of DPs: neighbor point search for DPs (from the current sampling layer (L^{th} layer)) as neighboring points (NPs), and NIT computation for DPs (from the next sampling layer ($(L+1)^{th}$ layer)) as centroid points (CPs). The skipping process for aggregation follows a similar pattern, also in two types of DPs: neighbor search for DPs (discarded in the current sampling layer), and the aggregation centered on DPs (discarded in the next sampling layer) because of grouping skipping. In the next layer ($(L+1)^{th}$ layer), the feature extractions performed by the convolution unit for DPs of the $(L+1)^{th}$ sampling layer are skipped with a ratio of $(1-1/R)$. With these four skipping mechanisms, total computations are reduced by 60.1% and 58.2% for PointNeXt-Cls@ModelNet40 and PointNeXt-Seg@S3DIS, respectively. SMS requires only simple control logic for skipping, occupying 0.5% of the core area.

The pipelined BDA is intended to address the memory access bottleneck, including irregular memory access and the loop-read of feature maps and the NIT, as shown in Fig. 23.4.5. In the computation flow of BDA, the grouping-guided aggregation is delayed after point convolution. Meanwhile, the point cloud is divided into multiple blocks, with each block undergoing a series of convolution layers followed by an aggregation. In the hardware implementation of BDA, external point features and the NIT are read into on-chip memory through consecutive regular block-level reads. Each core fetches the NIT of a block, and then fetches the corresponding features of neighbor points within the block point by point. The core compares and identifies the channel-wise maximum feature across all neighbor points (NPs), and outputs the subtraction of the centroid point (CP) feature and channel-wise maximum feature of neighbor points. By reducing the search space of aggregation from the global level to block level, BDA eliminates extensive irregular external memory access, especially in delayed aggregation where point features grow exponentially after multiple convolution layers. Furthermore, BDA reduces latency by allowing grouping, convolution, and aggregation of multiple blocks to be pipelined. Meanwhile, BDA reduces external memory access by enabling the aggregation unit to directly reuse both point features generated by the convolution unit and the NIT produced by the grouping unit on-chip. As a result, BDA reduces the external memory accesses by 99% and 95%, for PointNeXt-Cls@ModelNet40 and PointNeXt-Seg@S3DIS, respectively. BDA reduces latency by 51.3% and 55% for PointNeXt-Cls@ModelNet40 and PointNeXt-Seg@S3DIS, respectively.

Nebula is fabricated in 28-nm HPC+ technology. The measurement results are shown in Fig. 23.4.6. This chip can work at 20-200MHz with 0.57-0.94V supply voltage, while the peak energy efficiency is 109.8TOPS/W for PointNeXt-Cls@ModelNet40 and 59.4TOPS/W for PointNeXt-Seg@S3DIS at 20MHz/0.57V. Operating at 200MHz/0.94V, the frame rate, energy efficiency, and energy consumption per frame are 5263.2fps, 48.1TOPS/W, and 0.033mJ, respectively, for PointNeXt classifying the ModelNet40 dataset, and 275.4fps, 26TOPS/W, and 0.823mJ for PointNeXt segmenting the large-scale indoor S3DIS dataset. The die photo and chip summary are shown in Fig. 23.4.7, where the die area and core area are 3mm^2 and 1.5mm^2, respectively.

Acknowledgement:
This work was supported by the National Key R&D Program of China (2023YFB4502200 and 2023YFB4502802), Guangdong Provincial Key Laboratory of In-Memory Computing Chips (Shenzhen Graduate School, Peking University), and National Natural Science Foundation of China (61925401, 92064004, and 62474010). The corresponding authors are Hailong Jiao and Yuchao Yang (jiaohailong@pku.edu.cn; yuchaoyang@pku.edu.cn).

2025 IEEE International Solid-State Circuits Conference (ISSCC) | 979-8-3315-4101-9/25/$31.00 ©2025 IEEE | DOI: 10.1109/ISSCC49661.2025.10904703

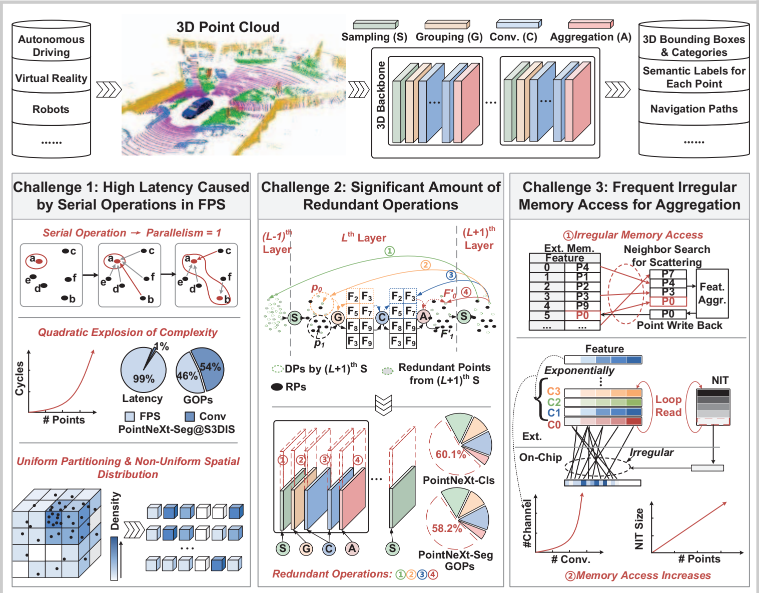

3D Point Cloud
Autonomous Driving
Virtual Reality
Robots
3D Backbone
Sampling (S)
Grouping (G)
Conv. (C)
Aggregation (A)
3D Bounding Boxes & Categories
Semantic Labels for Each Point
Navigation Paths
Challenge 1: High Latency Caused by Serial Operations in FPS
Challenge 2: Significant Amount of Redundant Operations
Challenge 3: Frequent Irregular Memory Access for Aggregation
Serial Operation → Parallelism = 1
Quadratic Explosion of Complexity
Uniform Partitioning & Non-Uniform Spatial Distribution
Redundant Operations
Irregular Memory Access
Memory Access Increases
PointNeXt-Seg@S3DIS
PointNeXt-Cls
PointNeXt-Seg

Figure 23.4.1: Application and challenges of 3D point-cloud neural network (PNN) accelerators.

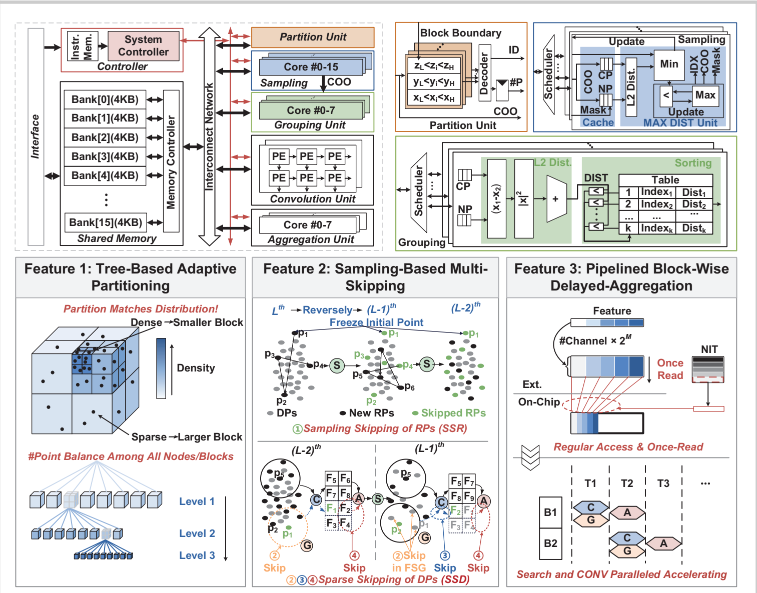

Controller
System Controller
Interface
Memory Controller
Interconnect Network
Shared Memory
Partition Unit
Sampling
Grouping Unit
Convolution Unit
Aggregation Unit
Block Boundary
Decoder
Scheduler
MAX DIST Unit
Feature 1: Tree-Based Adaptive Partitioning
Partition Matches Distribution!
#Point Balance Among All Nodes/Blocks
Feature 2: Sampling-Based Multi-Skipping
Sampling Skipping of RPs (SSR)
Sparse Skipping of DPs (SSD)
Feature 3: Pipelined Block-Wise Delayed-Aggregation
Regular Access & Once-Read
Search and CONV Paralleled Accelerating

Figure 23.4.2: Overall architecture and three features of the energy-efficient PNN accelerator, Nebula.

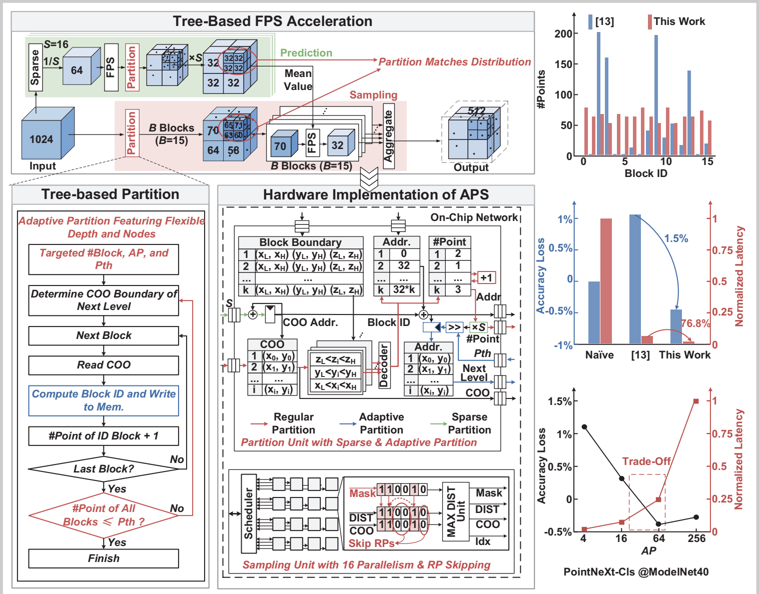

Tree-Based FPS Acceleration
Partition Matches Distribution
Tree-based Partition
Adaptive Partition Featuring Flexible Depth and Nodes
Targeted #Block, AP, and Pth
Determine COO Boundary of Next Level
Next Block
Read COO
Compute Block ID and Write to Mem.
#Point of ID Block + 1
Last Block?
#Point of All Blocks ≤ Pth ?
Finish
Hardware Implementation of APS
Partition Unit with Sparse & Adaptive Partition
Sampling Unit with 16 Parallelism & RP Skipping
[13]
This Work
Block ID
Trade-Off
PointNeXt-Cls @ModelNet40

Figure 23.4.3: Tree-based adaptive partitioning sampling (APS) for balanced partitioning and acceleration.

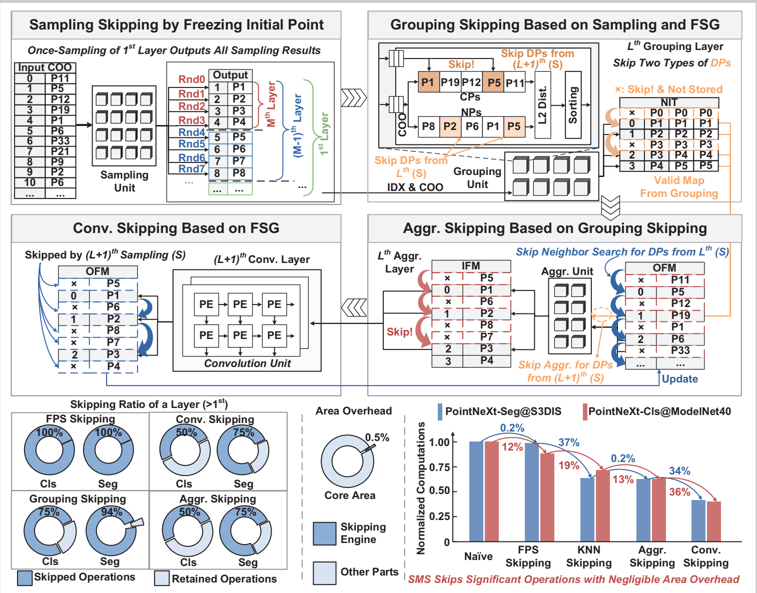

Sampling Skipping by Freezing Initial Point
Once-Sampling of 1st Layer Outputs All Sampling Results
Grouping Skipping Based on Sampling and FSG
Conv. Skipping Based on FSG
Aggr. Skipping Based on Grouping Skipping
Skipping Ratio of a Layer (>1st)
Area Overhead
PointNeXt-Seg@S3DIS
PointNeXt-Cls@ModelNet40
SMS Skips Significant Operations with Negligible Area Overhead

Figure 23.4.4: Sampling-based multi-skipping (SMS) strategy for operation skipping and storage savings.

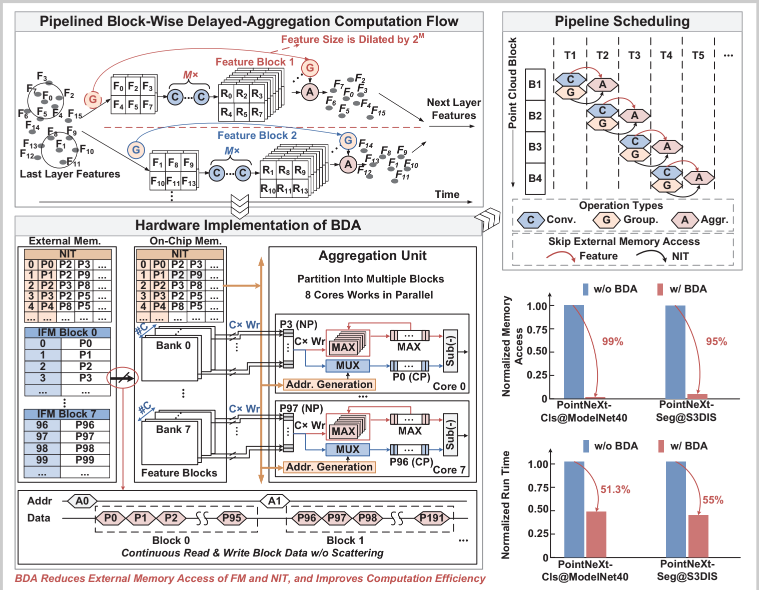
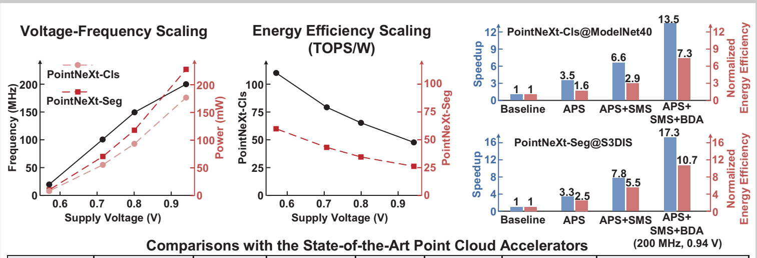

Pipelined Block-Wise Delayed-Aggregation Computation Flow
Feature Size is Dilated by 2M
Pipeline Scheduling
Hardware Implementation of BDA
Aggregation Unit
Partition Into Multiple Blocks
8 Cores Works in Parallel
Continuous Read & Write Block Data w/o Scattering
BDA Reduces External Memory Access of FM and NIT, and Improves Computation Efficiency
w/o BDA
w/ BDA
PointNeXt-Cls@ModelNet40
PointNeXt-Seg@S3DIS

Figure 23.4.5: Pipelined block-wise delayed-aggregation (BDA) with regular memory access and lower access times.

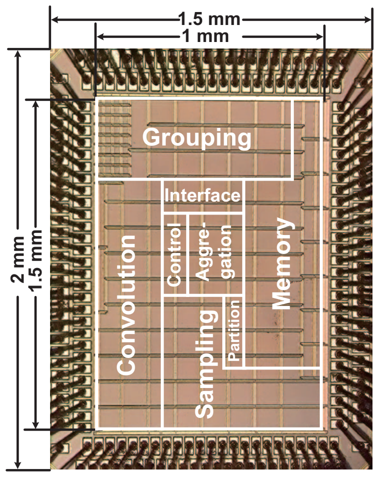


Comparisons with the State-of-the-Art Point Cloud Accelerators

	PointAcc [12] MICRO 2021	PNNPU [13] VLSI 2021	[14] VLSI 2022	DSPU [16] ISSCC 2022	[17] ISSCC 2023	Hawkeye [20] CICC 2024	This Work	
Technology	40 nm (Simulation)	65 nm	65 nm	28 nm	28 nm	28 nm	28 nm	
Area [mm^2]	3.9 (Core)	16 (Die)	4.1 (Die)	12.96 (Die)	2.69 (Core)	12.96 (Die)	1.5 (Core)	
SRAM [KB]	274	364	108.5	806	176	695.5	64	
Frequency [MHz]	1024	50-200	50-300	250	40-450	40-320	20-200	
Supply Voltage [V]	N/A	0.78-1.1	0.5-1.2	0.72-1.1	0.48-0.92	0.74-1.1	0.57-0.94	
Data Type	A: 16b, W: 8b	8b	8b	4b, 8b, 12b, 16b	8b	5b, 9b, 13b,16b, 17b, 32b	8b	
MAC	256	4608	100	12288	256	2160	512	
Model & Dataset	PointNet-Cls@ModelNet40, PointNet++-Seg@S3DIS	VoteNet-OD @SUN RGB-D[1)]	MinkowskiNet-Seg @ScanNet & Synthia	VoteNet-OD @SUN RGB-D	SECOND-OD @KITTI	PointPillars-OD @KITTI	PointNeXt-Cls @ModelNet40[2)]	PointNeXt-Seg @S3DIS[3)]
Accuracy	N/A	57.34% mAP @0.25 IoU	49% mIoU @ScanNet 61% mIoU @Synthia	57% mAP	78.49% mAP @Car Moderate 0.7	76.57% mAP @Car Moderate 0.7	91.78%	53.8% mIoU
Power Consumption [mW]	1057.1 @ModelNet40, 464.5 @S3DIS[4)]	266.8 @200 MHz	80 @300 MHz, 6.9 @50 MHz	350.6 @VoteNet	125 @400 MHz, 6.6 @60 MHz	855.5 @320 MHz, 66.1 @40 MHz	175.1 @200 MHz, 7.67 @20 MHz	226.7 @200 MHz, 9.93 @20 MHz
Frame Rate [Frame/s]	687.2 @ModelNet40, 618.4 @S3DIS[4)]	84.4 @200 MHz, 21.1 @50 MHz	7.2 @300 MHz, 1.2 @50 MHz	129.9 @VoteNet	16.9 @400 MHz, 3.3 @60 MHz	44.2 @320 MHz	5263.2 @200 MHz, 526.3 @20 MHz	275.4 @200 MHz, 27.54 @20 MHz
Performance [GOPS]	1168.2 @ModelNet40, 4455.6 @S3DIS[4)]	1760.9 @200 MHz	567.2 @300 MHz, 93.8 @50 MHz[5)]	11600 @CNN, 2500 @k-NN	193.8 @400 MHz[6)], 27.3 @60 MHz[7)]	1480 @320 MHz[8)]	8421.1 @200 MHz, 842.1 @20 MHz	5893 @200 MHz, 589.3 @20 MHz
Area Efficiency [GOPS/mm^2]	299.5 @ModelNet40, 1142.5 @S3DIS[4)]	110.1 @200 MHz	138.3 @300 MHz, 22.9 @50 MHz[5)]	N/A	72 @400 MHz, 10.1 @60 MHz	114.2 @320 MHz[8)]	5614.1 @200 MHz, 561.4 @20 MHz	3928.7 @200 MHz, 392.9 @20 MHz
Energy Efficiency [TOPS/W]	1.1 @ModelNet40, 9.6 @S3DIS[4)]	6.6 @200 MHz, 11.9 @50 MHz	7.1 @300 MHz, 13.6 @50 MHz[5)]	19.0 @CNN, 99.6 @k-NN	1.55 @400 MHz, 4.1 @60 MHz	11.1 @320 MHz, 38.5 @40 MHz[8)]	48.1 @200 MHz, 109.8 @20 MHz	26 @200 MHz, 59.4 @20 MHz
Energy Consumption [mJ/Frame]	1.5 @ModelNet40, 0.75 @S3DIS[4)]	3.2 @200 MHz, 1.47 @50 MHz	11.1 @300 MHz, 5.77 @50 MHz	2.7 @VoteNet	5.0 @400 MHz, 1.9 @60 MHz	19.4 @320 MHz	0.033 @200 MHz, 0.0155 @20 MHz	0.823 @200 MHz, 0.36 @20 MHz

[1)]OD: Object Detection. [2)]29.1% activation sparsity (AS), and 89% weight sparsity (WS). [3)]22.8% AS, and 59% WS. [4)]Derived from [11]. [5)]Conv. Layer. [6)]24% AS. [7)]40% AS. [8)]94.6% AS and 34.5% WS.

Figure 23.4.6: Measurement results and comparison table.



Technology	28-nm HPC+
Chip Area	2 mm × 1.5 mm
Core Area	1.5 mm × 1 mm
On-Chip SRAM	64 KB
Precision	INT8
Voltage	0.57-0.94 V
Frequency	20-200 MHz
Power	7.67-226.7 mW
Frame Rate [Frame/s] @0.94 V, 200 MHz	5263.2 (PointNeXt-Cls), 275.4 (PointNeXt-Seg)
Area Efficiency [TOPS/mm^2] @0.94 V, 200 MHz	5.6 (PointNeXt-Cls), 3.9 (PointNeXt-Seg)
Energy Efficiency [TOPS/W] @0.57 V, 20 MHz	109.8 (PointNeXt-Cls), 59.4 (PointNeXt-Seg)
Energy Efficiency [TOPS/W] @0.94 V, 200 MHz	48.1 (PointNeXt-Cls), 26 (PointNeXt-Seg)

Figure 23.4.7: Die photo and chip summary.

References:
[1] R. Abbasi, A. K. Bashir, H. J. Alyamani, F. Amin, J. Doh, and J. Chen, "Lidar Point Cloud Compression, Processing and Learning for Autonomous Driving," *IEEE Trans. on Intelligent Transportation Systems*, vol. 24, no. 1, pp. 962-979, 2023.
[2] L. Kastner, V. C. Frasineanu, and J. Lambrecht, "A 3D-Deep-Learning-Based Augmented Reality Calibration Method for Robotic Environments Using Depth Sensor Data," *IEEE ICRA*, pp. 1135-1141, 2020.
[3] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep Learning On Point Sets for 3D Classification and Segmentation," *CVPR*, pp. 77-85, 2017.
[4] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep Hierarchical Feature Learning in Point Sets in a Metric Space," *NeuroIPS*, pp. 5105-5114, 2017.
[5] X. Ma, C. Qin, H. You, H. Ran, and Y. Fu, "Rethinking Network Design and Local Geometry in Point Cloud: A Simple Residual MLP framework," *ICLR*, 2022.
[6] G. Qian, Y. Li, H. Peng, J. Mai, H. A. A. K. Hammoud, M. Elhoseiny, and B. Ghanem, "PointNeXt: Revisiting PointNet++ with Improved Training and Scaling Strategies," *NeuroIPS*, pp. 23192-23204, 2022.
[7] B. Graham, M. Engelcke, and L. V. D. Maaten, "3D Semantic Segmentation with Submanifold Sparse Convolutional Networks," *CVPR*, pp. 9224-9232, 2018.
[8] C. Choy, J. Y. Gwak, and S. Savarese, "4D Spatio-Temporal Convnets: Minkowski Convolutional Neural Networks," *CVPR*, pp. 3070-3079, 2019.
[9] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic Graph CNN for Learning on Point Clouds," *ACM Trans. on Graphics*, vol. 38, no. 5, pp. 1-12, 2019.
[10] Q. Xu, X. Sun, C.-Y. Wu, P. Wang, and U. Neumann, "Grid-GCN for Fast and Scalable Point Cloud Learning," *CVPR*, pp. 5661-5670, 2020.
[11] Y. Feng, B. Tian, T. Xu, P. Whatmough, and Y. Zhu, "Mesorasi: Architecture Support for Point Cloud Analytics via Delayed-Aggregation," *IEEE/ACM MICRO*, pp. 1037-1050, 2020.
[12] Y. Lin, Z. Zhang, H. Tang, H. Wang, and S. Han, "PointAcc: Efficient Point Cloud Accelerator," *IEEE/ACM MICRO*, Oct. 2021, pp. 449-461, 2021.
[13] S. Kim, J. Lee, D. Im and H.-J. Yoo, "PNNPU: A 11.9 TOPS/W High-Speed 3D Point Cloud-Based Neural Network Processor with Block-Based Point Processing For Regular DRAM Access," *IEEE Symp. VLSI Circuits*, 2021.
[14] Q. Cao and J. Gu, "A Sparse Convolution Neural Network Accelerator for 3D/4D Point-Cloud Image Recognition on Low Power Mobile Device With Hopping-Index Rule Book for Efficient Coordinate Management," *IEEE Symp. VLSI Circuits*, 2022.
[15] Y. Feng, G. Hammonds, Y. Gan, and Y. Zhu, "Crescent: Taming Memory Irregularities for Accelerating Deep Point Cloud Analytics," *IEEE/ACM ISCA*, pp. 962-977, 2022.
[16] D. Im, G. Park, Z. Li, J. Ryu, S. Kang, D. Han, J. Lee, and H.-J. Yoo, "DSPU: A 281.6mW Real-Time Depth Signal Processing Unit for Deep Learning-Based Dense RGB-D Data Acquisition With Depth Fusion and 3D Bounding Box Extraction in Mobile Platforms," *ISSCC*, pp. 510-512, 2023.
[17] W. Sun, X. Feng, C. Tang, S. Fan, Y. Yang, J. Yue, H. Yang, and Y. Liu, "A 28nm 2D/3D Unified Sparse Convolution Accelerator with Block-Wise Neighbor Searcher for Large-Scaled Voxel-Based Point Cloud Network," *ISSCC*, pp. 328-330, 2023.
[18] X. Yang, T. Fu, G. Dai, S. Zeng, K. Zhong, K. Hong, and Y. Wang, "An Efficient Accelerator for Point-Based And Voxel-Based Point Cloud Neural Networks," *IEEE/ACM DAC*, 2023.
[19] C. Zhou, Y. Fu, M. Liu, S. Qiu, G. Li, Y. He, and H. Jiao, "An Energy-Efficient 3D Point Cloud Neural Network Accelerator with Efficient Filter Pruning, MLP Fusion, and Dual-Stream Sampling," *IEEE/ACM ICCAD*, 2023.
[20] S. Lim, J. Heo, J. Yang, and J.-Y. Kim, "A 38.5TOPS/W Point Cloud Neural Network Processor with Virtual Pillar and Quadtree-Based Workload Management for Real-Time Outdoor BEV Detection," *IEEE CICC*, 2024.
[21] C. Zhou, Y. Fu, Y. Ma, E. Han, Y. He, and H. Jiao, "Adjustable Multi-Stream Block-Wise Farthest Point Sampling Acceleration in Point Cloud Analysis," *IEEE TCAS-II*, vol. 71, no. 7, pp. 3523-3527, 2024.
[22] S. Kim, S. Kim, J. Lee, and H.-J. Yoo, "A 54.7 fps 3D Point Cloud Semantic Segmentation Processor with Sparse Grouping Based Dilated Graph Convolutional Network for Mobile Devices," *IEEE ISCAS*, 2020.
[23] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D Shapenets: A Deep Representation for Volumetric Shapes," *CVPR*, pp. 1912-1920, 2015.
[24] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3D Semantic Parsing of Large-Scale Indoor Spaces," *CVPR*, pp. 1534-1543, 2016.