2025 5th International Conference on Artificial Intelligence and Industrial Technology Applications (AIITA)

A Reconfigurable Hardware Accelerator based on SRAM In-Memory Computing

Chengsi Wu
South China University of Technology
Guangzhou, China
1045601490@qq.com

Xiaoyong He*
South China University of Technology
Guangzhou, China
eemorrishe@scut.edu.cn
*Corresponding author

***Abstract*—Traditional artificial intelligence (AI) accelerator suffers from the "memory wall" which brings the bottleneck of energy efficiency improvement. In this paper, a reconfigurable in-memory computing accelerator (RIA) is proposed to further improve the energy efficiency of AI accelerator. The proposed RIA integrates a sram in-memory computing macro and multiple operators to support different multi-layer convolutional neural networks. It can also be configured into 1/2/4/8-bit computational modes. The proposed accelerator exhibits superior energy efficiency with 309.5TOPS/W at 1-bit mode, 2.08TOPS/W at 8-bit mode in the ResNet-18 testbench.**

Keywords—Artificial intelligence, In-memory computing, Reconfigurable accelerator

I. INTRODUCTION

The development of the neural network algorithms deployed in edge devices brings higher energy efficiency requirements for the hardware architecture. However, traditional Von Neumann architectures, which rely on separate memory and processing units, are increasingly struggling to meet the computational and bandwidth requirements of AI applications. For example, the energy consumption of a 32-bit addition operation is only 0.1pJ, while the memory access operations will require hundreds of pJ in 40nm technology [1]. Therefore, a new computing architecture is required to minimize excessive data transfer.

As an alternative, in-memory computing (IMC) technologies have emerged as promising solutions to address these challenges. Different from Von Neumann architectures, CIM integrates memory and processing elements into a single unit, enabling data to be processed directly within the memory itself. This approach eliminates the traditional bottleneck associated with the data transfer between the processor and the memory, thus significantly improving both performance and energy efficiency. The need for IMC has become even more pronounced as AI models grow larger and more complex, demanding massive amounts of data to be moved between different levels of memory and processing elements. Most works focus on custom IMC macro designs [2],[3],[4],[5],[6] to improve the energy efficiency greatly for a specific application or operator, such as a convolution operator. To improve the flexibility of different AI models, IMC macros are also extended to IMC-based accelerators [7], [8], [9], [10], [11], [12], [13], [14], [15].

A majority of IMC - grounded accelerators make use of analog-mixed-signal (AMS) IMC macros. These AMS IMC macros employ resistance and capacitance for computational purposes and analog-to-digital converters(ADCs) for converting signal. The utilization of AMS IMC enables the attainment of high energy efficiency as well as area efficiency. Nevertheless, analog computing hardware has the potential to yield inaccurate matrix multiplication outcomes due to process, voltage, and temperature (PVT) fluctuations. As a result, the accuracy of inferences is diminished [11]. In contrast, the digital IMC macro utilizes digital arithmetic circuits like accumulators and adder tree which composed of full adders and compressors. It can perform MAC operations stably even in the face of PVT variations. Nevertheless, an IMC - based accelerator usually comprises multiple IMC macros. Taking up a large amount of chip area is the main difficulty faced by current digital IMC accelerators, because digital IMC requires more computing logic and has lower storage density.

In this paper, we present a new accelerator based on digital sram IMC, which combines a reconfigurable IMC macro, multiple operators and scheduling modules. We facilitate the developed scheduling modules to maximize the utilization of IMC macro for the neural networks of different scales. The operators that can be enabled based on requirements will help us to complete the deployment of a full neural network. And the reconfigurable IMC macro allows us to implement the application of quantitative neural networks with different precision. Through the three components above, we will achieve a balance between the area and throughput.

II. BACKGROUND

There are various IMC-based accelerators which can categorize them as domain-specific accelerators [12], [13] and general programmable accelerators [9], [14], [15].

The domain-specific accelerators aim to reduce the power consumption of data access and improve the data bandwidth greatly for a specific set of neural networks. For instance, [12] introduced a dual-stationary dataflow architecture for RNNs, significantly reducing memory access. Additionally, predictive early BN (batch normalization) and Binary quantization units were integrated to reduce the computational complexity. Similarly, the algorithm and hardware co-designed by Dbouk et al. [13] for keyword detection, utilizing IMC SRAM for convolution layers and digital hardware for layers requiring higher precision.

979-8-3315-0976-7/25/$31.00 ©2025 IEEE

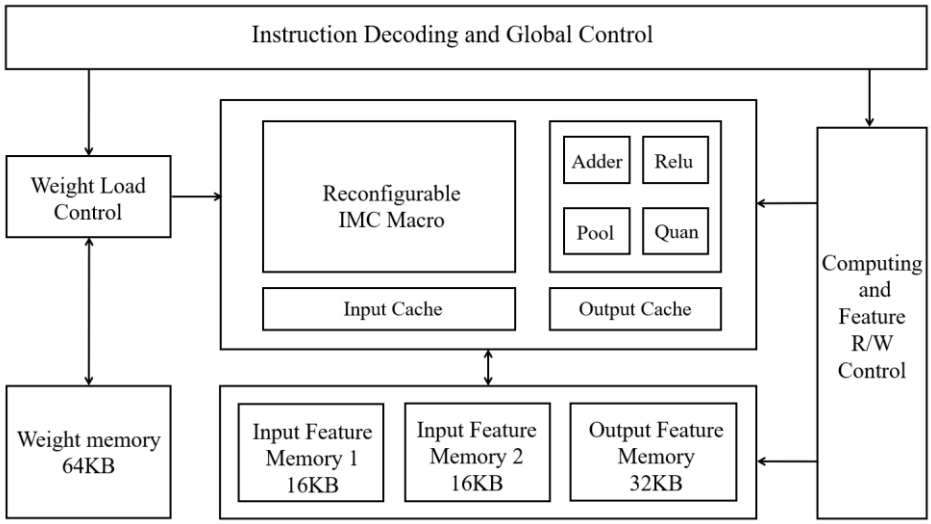


Fig.1 RIA Architecture

Another approach involves programmable accelerators that offer flexibility for various NN models. These accelerators typically integrate numerous IMC macros to accommodate large-weight matrices. A notable example is [14], the accelerator in this research contains a 590-kb compute-in-memory-unit and near-memory digital operators, which can realize complex multiplications of matrix and nonlinear functions. A RISC-V CPU core combined with on-chip SRAM is embedded for scheduling matrix-vector operations. By employing bit-serial and bit-parallel schemes for multi-bit inputs and weights, it supports flexible bit precision for convolution and FC layers.

These specialized and programmable accelerators demonstrate impressive energy efficiency and bandwidth utilization. However, specialized accelerators may lack flexibility for diverse NN models, while programmable accelerators may introduce higher complexity and overhead. Future research could focus on striking a balance between specialization and programmability to achieve optimal energy efficiency and bandwidth utilization for a broader range of NN applications. However, a huge silicon area will be occupied. And it is also difficult to fully utilize all IMC macros across different neural networks.

III. RIA Architecture

Figure 1 shows the overall structure of the RIA, which is mainly composed of IMC macro, other operator modules, input and output cache, decoding and global control module, weight load control module, calculation and feature read and write control module and on-chip memory. The weight memory is used to store the weights of the model and is 64KB in size. Two input feature memories are used to store the input features of each layer, each with a size of 16KB, and the output feature memories are used to temporarily store the output results of the macro, with a size of 32KB. In order to support common neural networks, such as ResNet and VGGNet, the accelerator is also equipped with a larger capacity of off-chip memory to transfer data to on-chip memory during operation. For two input feature memory, when one is being read by the IMC macro, the other one receives the input feature from the off-chip memory for the next calculation, such a structure can effectively improve the throughput.

A. Accelerator scheduler

The status scheduler of the RIA is shown in Figure 2. In the idle state of the system, the accelerator listens or reads the register information in the global control module. The global control module has a total of five 32-bit global registers, three of which are used to record the current IRA's configuration and data address, and the others are used to record the operating status and error status. The current configuration information contains the precision of a certain layer of the neural network, the type of convolutional kernel, and the size of the input/output features of that layer. Table I shows a detailed description of the parameters related to the network model in the global register. The accelerator supports binary neural networks (BNN) and 2/4/8-bit convolutional neural networks (CNNS) by configuring WDP as 0/1/2/3, respectively. By configuring WIFW, WIFD, and WIFC, the maximum size of the input/output features supported is 64×64×512. By configuring WCT, two classical convolution kernel types, 1×1 and 3×3, are also supported. In the idle state, the accelerator will detect whether the weight refresh signal in the operation status register is enabled. When the weight refresh signal is detected, the platform reads the configuration and address information and enters the system refresh state.

Table I Configure instruction description

Name	Description	Width
WDP	Data precision	2
WCT	Convolution Type	1
WIFW	Input feature width	7
WIFH	Input feature height	7
WIFC	Input feature channel	10
WOFC	Output feature channel	10

In the system refresh state, the weight loading control module reads the weights from the weight memory and writes them to the IMC macro. The IMC macro is used to store the weights, and multiple convolution calculations can be carried out in the macro through inputting feature data, without transferring the weights multiple times. The amount of weight data depends on the WIFW, WIFD, and WIFC parameters of the layer. If the amount of weights written is greater than the capacity of the macro circuit, the system retains the configuration parameters and records the memory address, so that the unused weights can continue to be loaded after the calculation is completed. When the weight writing is complete and the computation start signal is valid, the accelerator will enter the system working state from the system refresh state.

In the operating state of the system, the calculation and feature loading control module will control the feature input, macro calculation and feature output at the same time according to WIFW, WIFD, WIFC and WOFC parameters and address information, while managing the operation of other operator modules in the process. The adder module is used to add up the results of multiple calculations of the macro or other models such as the jump operation in ResNet [16]. Relu and pooling operator module is mainly composed of FIFO and comparator to realize activation and pooling. The quantization operator module completes the quantization and dequantization operation of the result, which is mainly composed of shifters and interceptors, and is responsible for quantizing the calculation result of each layer to the input precision of the next layer. When the calculation corresponding to the current loaded weights is completed, if there are still unloaded weights in this layer, the accelerator re-enters the system refresh state. If all calculations of the layer network are completed, the accelerator enters the system idle state and waits for the next layer network calculation to start.

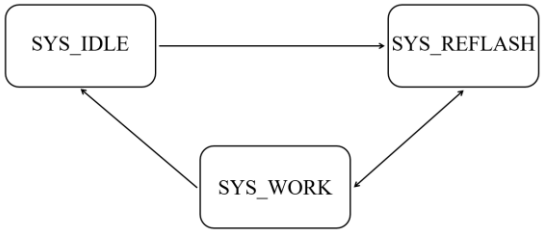


Fig.2 RIA Scheduler

B. Reconfigurable IMC Macro

Figure 3 shows the architecture of the used reconfigurable IMC macro in our accelerator, which mainly consists of the IMC array and peripheral logic. The IMC array is composed of 8 sub-arrays, each of which is the key circuit for memory computing. Since the in-memory computing architecture of the macro is based on SRAM memory, the peripheral logic of the general SRAM memory is required to achieve the weight data writing. It is worth noting that in order to save resource consumption, the macro does not have the read-out function of the general SRAM memory. At the same time, the peripheral logic controls the generation of the gated clock, and the gated clock is input into the IMC array and other peripheral logic to reduce the power consumption at runtime.

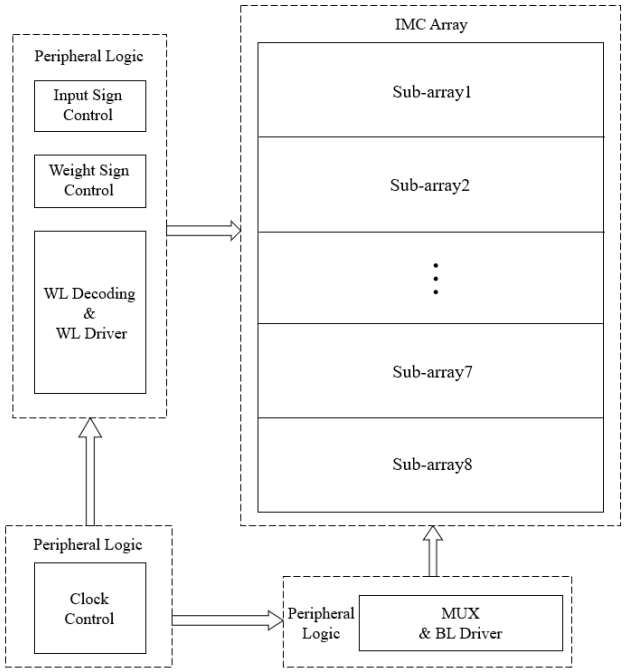
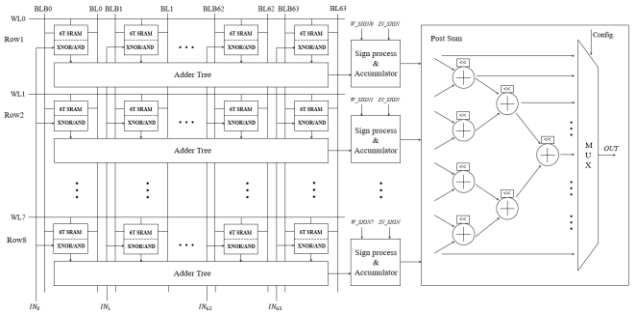
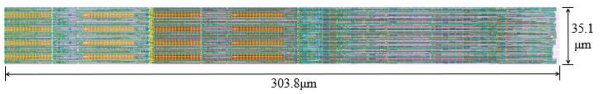


Fig.3 IMC Macro Architecture

The schematic of sub-array is shown in Figure 4. The sub-array is composed of 8x64 IMC cells. Each cell is a 6T SRAM combined with a multiplier that can be configured as the XNOR or AND function. The way of writing weights to the IMC cell array is the same as writing data to SRAM. Each column of the sub-array is used to store different bits of the same weight, if the weight is configured to be 8 bits, then one column can store one weight, if the weight is configured to be 1 bit, then one column can store eight weights, and the same is true of 2/3 bits. Each row (with weights of 1 bit) or multiple rows (with weights of 8 bits) of the sub-array is used to store different weights on the same output channel, so the sub-array can store up to 64 weights on one output channel.



(a) The schematic of sub-array



(b) The layout of sub-array

Fig. 4 The schematic(a) and layout(b) of sub-array

An adder tree is shared by all IMC cells in each row in the array to sum the bitwise multiplication results. The adder tree is composed of a mixture of full adders/half adders and

compressors to achieve a balance between performance, area, and power consumption.

In a row sub-structure of the IMC macro, the dedicated adder tree for each row adds the multiplication results of 64 cells in each row within one clock cycle, thus obtaining the MAC result for [1bit input/1bit weight]. When the input/weight width is set to 8 bits, the accumulator continuously accumulates these [1bit/1bit] results within 8 clock cycles, and then the MAC result for [8bit input/1bit weight] can be obtained. Finally, the post-sum circuit adds the results of the eight accumulators, and the MAC result for [8bit input/8bit weight] can be obtained. In other input/weight width configurations, it is only necessary to select different after and circuit outputs. Therefore, the macro can realize the configuration of input data and weight data from 1bit to 8bit, and multiple sub-arrays can implement matrix multiplication.

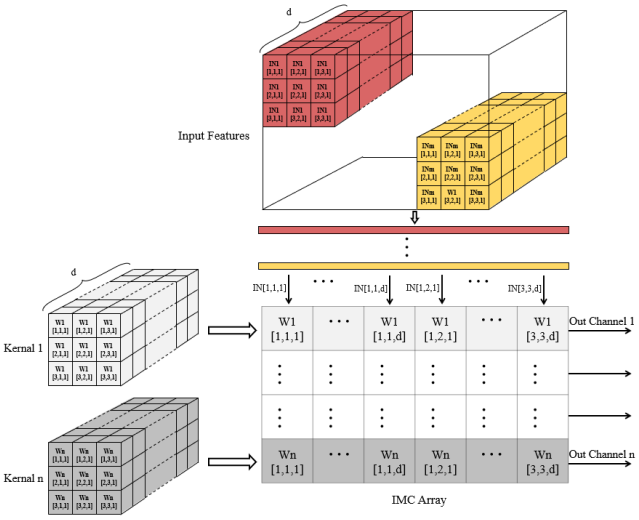


Fig. 5 Convolution operation mapping process

IV. Experimental Result

The RIA is implemented in 55nm CMOS SMIC Technology. Simulations are performed to evaluate the energy efficiency of RIA in different neural networks. The parasitic parameters are extracted by StarRC. HSIM is used to carry out analog-digital mixed analysis because our design contains the custom IMC macro and digital circuits design. The whole system works at 0.8V, 100Mhz.

A. Energy Efficiency Analysis

Figure 5 shows the convolution operation in RIA. It uses the way of bit-series to transfer the input feature from the feature memory, which needs 8 cycles to compute an single 8-bit data. The whole IMC array can support 8×64 8-bit data computation simultaneously. When the layer size exceeds 8×64, the remaining weight and feature data can be updated by our weight load controller and computing controller.

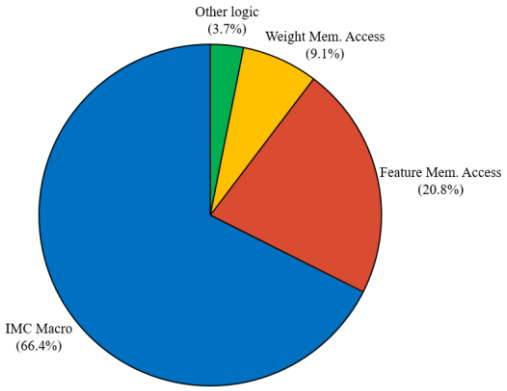


Fig.6 The power consumption breakdown of RIA in the ResNet-18 testbench

Fig. 6 shows the power consumption breakdown at 0.8V 100Mhz 8-bit mode in the ResNet-18 testbench. We simulate the transfer process between the off-chip memory and the memory in the accelerator in testbench, so the energy consumption generated by the off-chip memory access is not taken into account when measuring the energy consumption. The total power is 5.76mW. The IMC macro accounts for 69.4% of the total power dissipation which is 3.82mW. The routing memory and weight memory access power account for 20.8% and 9.1%, respectively. The other control logical which accounts for 3.7% which is 0.21mW. Since the number of times to carry the weights to the computing circuit is greatly reduced, the access energy for the weight memory is significantly reduced, which is the greatest advantage of the RIA.

An IMC cell can perform one-bit multiplication and addition, which contains two one-bit operates. Hence the maximum throughput of RIA at 1-b mode/100Mhz and 8-b mode/100Mhz are 819.2 GOPS ($64*64*2*10^8$), 12.8 GOPS ($64*64/8*2*10^8/8$), respectively. We also evaluate the average energy efficiency of RIA in three kinds of neural networks: ResNet-18, VGG-6, LeNet-5. Table II shows the results for the three kind neural networks. For example, RIA has the average energy efficiency of 8.96 TOPS/W, 9.43 TOPS/W and 9.88 TOPS/W at TT corner, respectively. Note that we ignore the cost of initial feature and weight storing. Table II also shows the energy efficiency in 1-b mode which is 8.4~11.7x larger than the one in 8-b mode because no extra cycles and accumulator are needed to perform the shift-and-add operation.

Table II The energy efficiency (TOPS/W) of IRA in different neural networks

	SS	TT	FF
ResNet/8b	7.36	8.96	11.20
VGG/8b	7.85	9.43	11.68
LeNet/8b	8.73	9.88	12.15
ResNet/1b	1099	1312	1478

Table III shows the comparison with the recent IMC-based accelerator works. As compared to prior works, RIA have better programmability and configurability than the work [9] and work [10] which can support the three types of neural networks: CNN, FC and ResNet. And compared with the work [8], RIA shows better energy efficiency with 2.28× improvement.

Table III The comparison of other AI accelerators

	ISSCC20 [9]	JSSC23 [8]	ISSCC22 [10]	RIA (Proposed)
Technology	65nm	28nm	22nm	55nm
Supply voltage (V)	0.9-1.05	1.0	0.55-0.9	0.8V
AI model	RNN/FC	CNN/FC/ResNet	ResNet	CNN/FC/ResNet
IMC size	4kb	3456kb	2560kb	4kb
Clock frequency (Mhz)	50-100	42	100	100
Bit precision(input/weight)	2-8b	1b/2b	7b	1b/2b/4b/8b
Peak Throughput (TOPS)	2/0.17	49	128	0.82/0.012
Area (mm^2)	5.66	20.9	10.21	1.562
Area efficiency ($GOPS/mm^2$)	63.5	2345	6560	525(1b)
Energy eff.(TOPS/W)	113.9 (2b)	136(1b)/32(2b)	N/A	309.5(1b)/2.08(8b) (TTG sim)

IV. CONCLUSION

In this paper, a reconfigurable accelerator is proposed, named RIA. It integrates a reconfigurable digital IMC macro, multiple operators and parallel scheduling system, which supports various kinds of neural networks and different computational precision. The simulation results with 55nm CMOS technology shows that RIA has the area efficiency of 525 $GOPS/mm^2$ at 1-bit mode and the energy efficiency of 2.08TOPS/W in 8-bit mode for ResNet-18.

REFERENCES

[1] W. Shan et al., "AAD-KWS: A Sub-μ W Keyword Spotting Chip With an Acoustic Activity Detector Embedded in MFCC and a Tunable Detection Window in 28-nm CMOS," in IEEE J. Solid-State Circuits, vol. 58, no. 3, pp. 867-876, March 2023.

[2] X. Si et al., "A twin-8T SRAM computation-in-memory macro for multiple-bit CNN-based machine learning," in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2019, pp. 396–398.

[3] Y. -C. Chiu et al., "A 4-Kb 1-to-8-bit Configurable 6T SRAM-Based Computation-in-Memory Unit-Macro for CNN-Based AI Edge Processors," in IEEE J. Solid-State Circuits, vol. 55, no. 10, pp. 2790-2801, Oct. 2020.

[4] E. Choi et al., "A 133.6 TOPS/W compute-in-memory SRAM macro with fully parallel one-step multi-bit computation," in Proc. IEEE Custom Integr. Circuits Conf. (CICC), Apr. 2022, pp. 1–2.

[5] J. Song et al., "A 28 nm 16 Kb Bit-Scalable Charge-Domain Transpose 6T SRAM In-Memory Computing Macro," in IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 70, no. 5, pp. 1835-1845, May 2023.

[6] R. Sehgal et al., "A Bit-Serial, Compute-in-SRAM Design Featuring Hybrid-Integrating ADCs and Input Dependent Binary Scaled Precharge Eliminating DACs for Energy-Efficient DNN Inference," in IEEE J. Solid-State Circuits, vol. 58, no. 7, pp. 2109-2124, July 2023.

[7] H. Valavi, P. J. Ramadge, E. Nestler, and N. Verma, "A 64-tile 2.4-Mb in-memory-computing CNN accelerator employing charge-domain compute," IEEE J. Solid-State Circuits, vol. 54, no. 6, pp. 1789–1799, Jun. 2019, doi: 10.1109/JSSC.2019.2899730.

[8] Zhang, Bo, et al. "PIMCA: A programmable in-memory computing accelerator for energy-efficient DNN inference." IEEE Journal of Solid-State Circuits 58.5 (2022): 1436-1449.

[9] Dbouk, Hassan, et al. "KeyRAM: A 0.34 uJ/decision 18 k decisions/s recurrent attention in-memory processor for keyword spotting." 2020 IEEE Custom Integrated Circuits Conference (CICC). IEEE, 2020.

[10] Ueyoshi, Kodai, et al. "DIANA: An end-to-end energy-efficient digital and ANAlog hybrid neural network SoC." 2022 IEEE International Solid-State Circuits Conference (ISSCC). Vol. 65. IEEE, 2022.

[11] Wang, Hechen, et al. "A PVT Robust 8-Bit Signed Analog Compute-In-Memory Accelerator with Integrated Activation Functions for AI Applications." 2024 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits). IEEE, 2024.

[12] Guo, Ruiqi, et al. "A 5.1 pJ/neuron 127.3 us/inference RNN-based speech recognition processor using 16 computing-in-memory SRAM macros in 65nm CMOS." 2019 Symposium on VLSI Circuits. IEEE, 2019.

[13] Jia, Hongyang, et al. "A programmable heterogeneous microprocessor based on bit-scalable in-memory computing." IEEE Journal of Solid-State Circuits 55.9 (2020): 2609-2621.

[14] H. Jia, H. Valavi, Y. Tang, J. Zhang and N. Verma, "A Programmable Heterogeneous Microprocessor Based on Bit-Scalable In-Memory Computing," in IEEE Journal of Solid-State Circuits, vol. 55, no. 9, pp. 2609-2621.

[15] Tu, Fengbin, et al. "A 28nm 29.2 TFLOPS/W BF16 and 36.5 TOPS/W INT8 reconfigurable digital CIM processor with unified FP/INT pipeline and bitwise in-memory booth multiplication for cloud deep learning acceleration." 2022 IEEE International Solid-State Circuits Conference (ISSCC). Vol. 65. IEEE, 2022.

[16] Wu, Zifeng, Chunhua Shen, and Anton Van Den Hengel. "Wider or deeper: Revisiting the resnet model for visual recognition." Pattern recognition 90 (2019): 119-133.