An Energy Efficient Time-Multiplexing Computing-in-Memory Architecture for Edge Intelligence

Rui Xiao, *Student Member, IEEE,* Wenyu Jiang, *Senior Member, IEEE,* Piew Yoong Chee

***Abstract*—The growing data volume and complexity of Deep Neural Networks (DNN) require new architectures to surpass the limitation of the von-Neumann bottleneck, with Computing-in-memory (CIM) as a promising direction for implementing energy-efficient neural networks. However, CIM's peripheral sensing circuits are usually power and area hungry components. We propose a time-multiplexing Computing-in-Memory architecture (TM-CIM) based on memristive analog computing to share the peripheral circuits and process one column at a time. The memristor array is arranged in a column-wise manner that avoids wasting power/energy on unselected columns. In addition, DAC (digital-to-analog converter) power and energy efficiency, which turns out to be an even greater overhead than ADC (analog-to-digital converter), can be fine-tuned in TM-CIM for significant improvement. For a 256*256 crossbar array with a typical setting, TM-CIM saves 18.4× in energy with 0.136 pJ/MAC efficiency, and 19.9× area for 1T1R case and 15.9× for 2T2R case. Performance estimation on VGG-16 indicates that TM-CIM can save over 16× area. A trade-off between the chip area, peak power, and latency is also presented, with a proposed scheme to further reduce the latency on VGG-16, without significantly increasing chip area and peak power.**

***Index Terms*—Time Multiplexing, Column-wise array, Computing-in-Memory, Neuromorphic Computing, Memristive Analog Computing, Edge Intelligence.**

I. Introduction

DEEP Neural Networks (DNNs) have been widely implemented in various fields with unprecedented success, such as autopilot, aerospace, wearables, security, etc [1]. With the ever-increasing complexity of DNNs, the modern computing systems have to cope with massive parameters and operations. Due to the physical separation between the processing units and memory units, conventional Von-Neumann architectures suffer from the limited on-chip memory size and memory bandwidth, resulting in the "Von-Neumann Bottleneck" [2]. What's more, conventional processors for DNNs such as GPUs require ultra-high power consumption, which is not suitable for some applications such as edge intelligence.

Computing-in-Memory (CIM) is considered as a promising candidate to surpass the "Von-Neumman bottleneck" with much lower power consumption and much higher energy efficiency. CIM performs in-situ computing within the memory, significantly reducing data movement and thus facilitating high energy efficiency. Emerging Non-Volatile memristors such as Phase Change Memory (PCM), Spin-Torque-Transfer memory (STT-MRAM), and resistive random-access memory (RRAM) [3]–[5] have been widely explored as fundamental building blocks of CIM schemes. Fig.1 shows the diagram of conventional CIM schemes, the input circuit for each row is usually composed of a digital-to-analog converter (DAC) with an operational amplifier (OP-AMP) based voltage follower output stage [6]. The memristors are usually arranged as a crossbar array. The conductance of the memristors behaves like a synaptic weight, and according to Kirchhoff's law, the combined bit-line current of each column corresponds linearly to the weighted sum of the respective neuron. This arrangement of the array corresponds to the matrix in a layer of a neural network, and implements what is referred to as a cross-bar in neural network hardware implementation.

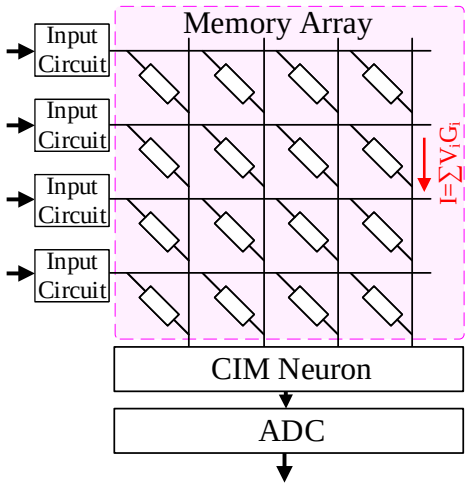


Fig. 1. The Diagram of Computing-in-Memory Scheme.

In conventional CIM schemes, each input circuit is required to drive multiple devices (e.g. 256), which means the DAC will become area and power hungry [6]. Some research works use digital input signals to avoid such overhead [7]–[9]. However, it requires multiple cycles to compute a high-precision activation, which will increase its total energy consumption. What's more, the shift & add operation for different input bits will accumulate the quantization errors if each cycle requires an ADC conversion, leading to a reduction in the robustness of this architecture. Moreover, in above two architectures, each

R. Xiao is with the College of Information Science & Electronic Engineering, Zhejiang University, 38 Zheda Road, Hangzhou, China, 310027, and also with the Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore, 138632, email: xiaor@zju.edu.cn.

W. Jiang and C. Yoong are with the Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore, 138632, email: wjiang@i2r.a-star.edu.sg, cheepy@i2r.a-star.edu.sg

column has its dedicated peripheral circuits, including CIM neuron and ADC. However, both the CIM neuron and ADC are power-hungry components and have significant chip area. In [8], the trans-impedance amplifier (TIA) neuron consumes more than 95% power in the scheme. In [9], the analog-to-digital converter (ADC) consumes more than 92% energy and 75% area in the system. Since edge intelligence prefers compact and energy-efficient DNN chips over those with high computing throughput, one way to reduce the area overhead of peripheral circuits is to share them via Time-Multiplexing (TM). However, for conventional memristor array, forcing simple TM would lead to significant leakage currents on unselected columns, which will be discussed in detail in Section II.

In light of above limitations, we proposed a novel Time-Multiplexing Computing-in-Memory (TM-CIM) architecture to save area without incurring additional power consumption. In addition, since TM-CIM computes one column at a time, the input circuit only needs to drive one device (at a time). Therefore, the area and power consumption of input circuits can be significantly reduced. The major contributions of this paper are summarized as follows:

1) The Time-Multiplexing architecture shares the TIA and ADC with multiple columns to reduce the area overhead of the peripheral circuits. In a typical setting, the area can be saved 19.9× for a 256*256 1T1R array and 16.9× for a 256*256 2T2R array at a latency of 5140 ns, vs 210 ns without time-multiplexing.
2) The memristor array is arranged in column-wise form rather than row-wise form. The cells on the same column are controlled with a column-wise signal. In this way, unselected columns will be completely turned off, thus avoiding leakage currents.
3) TM-CIM is flexible and efficient at implementing complex DNNs. Compared with conventional architectures with analog input, the area saving is 16×, and energy saving is 30× on VGG-16 [10]. Compared with conventional architecture with digital input, the area and energy saving is 14.2× and 5.9×, respectively.
4) A trade-off analysis between chip area, peak power, and system latency gives the best time-multiplexing strategy for different DNNs. Under similar setting as in previous points, TM-CIM can implement VGG-16 with an area of 118.09 mm^2, peak power of 0.797 W, and energy consumption of 1.968 pJ/image at a latency of 16.056 ms.

The rest of the paper is organized as follows: Section II introduces the background and related works. Section III discusses the detailed design of the proposed TM-CIM architecture. Section IV provides performance evaluation of the proposed architecture. Finally the conclusion is drawn in Section V.

II. Background and Related Works

In a CIM crossbar array, typically each memristor is connected with a select transistor to form a 1T1R cell. As shown in Fig. 2 (a), each Word-Line controls the gate of each 1T1R cell on a row, generally the Bit-Line of each 1T1R cell on a row is fed by the same corresponding voltage (Act_i) which corresponds to the input activation x_i in a DNN, and all columns are computed and sensed in parallel. Based on Kirchhoff's law, Each column's current is therefore the weighted sum of products of input activations and weights on the corresponding neuron $\sum_i x_i w_{ij}$. The conventional 1T1R array is widely used in recent research [7], [9], [11], with some researchers using Source-Line to represent the activations and Bit-Line current to represent the weighted sum [12], [13].

As shown in Fig. 2 (b), 2T2R cells have been proposed to represent signed weights [14], [15]. In a 2T2R cell, the first memristor represents the positive portion of a weight and the second represents the negative portion. So if the weight is positive, the second memristor will generally be programmed with high(est) resistance state (HRS) possible so that it represents a weight of zero. Similar converse arrangement is made if the weight is negative. If the input activation x_i is positive, then it will be represented by the voltage $Act_{i,p}$, and $Act_{i,n}$ will be zero (with respect to ground seen at SL_i, which could either true ground or virtual ground depending on the implementation). Conversely, if x_i is negative, then voltage $Act_{i,n}$ will be negative, and $Act_{i,p}$ be zero (again w.r.t. ground seen at SL_i). With this arrangement, the difference of the pair of memristors' currents would represent $x_i w_{ij}$. These architectures all have per-column peripheral circuits including CIM neurons and ADCs, which will lead to high power consumption and area overhead.

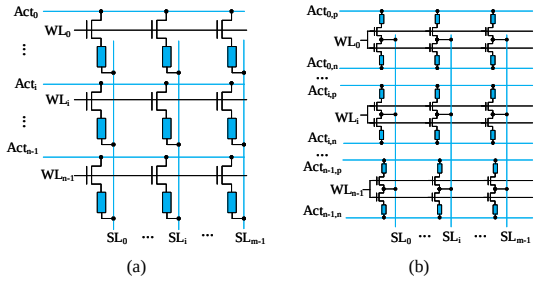


Fig. 2. Diagram of conventional (a) 1T1R array and (b) 2T2R array. The 2T2R array is used to represent signed weights.

One promising way to reduce the power consumption and area overhead of the peripheral circuits is to share them via time multiplexing. [9] shares each ADC with 4 columns to reduce the area overhead. However, it still requires per-column sample & hold (S&H) circuits which also implies per-column TIAs, and the ADCs still consume most of the energy and area. [12] uses 32-to-1 multiplexers to share a 4-bit ADC with 32 columns. Nevertheless, the ADC precision is so low that it requires additional shift & add circuits to achieve high precision. However, the effective number of bits (ENOB) of the weighted sum is limited by the ADC resolution, rendering the shift & add unuseful in practical sense. In addition, its binary input mode requires multiple cycles to implement n-bit activations. In this paper, a novel time-multiplexing architecture is proposed with analog input and high precision ADC output, which will be discussed in Section III.

III. Time-Multiplexing CIM Architecture

TM-CIM is designed to reduce the peripheral circuit area overhead of CIM architectures as well as to avoid additional power/energy overhead. Fig. 3 shows the top view of proposed TM-CIM. For simplicity of illustration, the column-wise array is composed of 1T1R cells, which can be replaced by 2T2R cells to represent signed weights. The neuron is shared with multiple columns to reduce the area overhead. The details of each block will be introduced in the rest of this section.

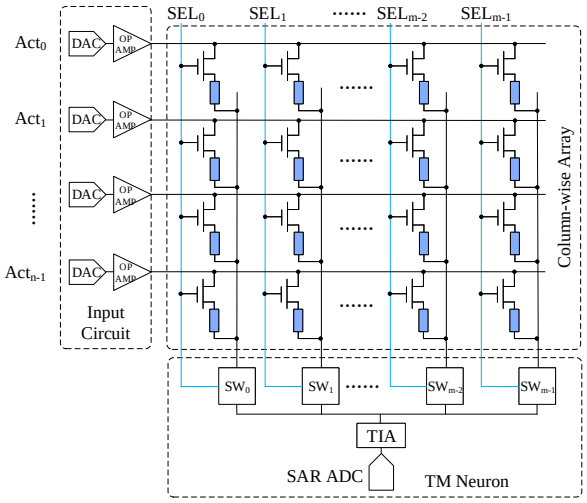


Fig. 3. The top view of proposed TM-CIM architecture based on 1T1R array.

A. Column-wise Array

Memristor array is the key component for CIM architecture. On a conventional CIM memristor array (e.g. 1T1R or 2T2R), because during inference Word Line i would turn on all transistors in row i, if we force time-multiplexing when one column is selected for processing, the unselected columns will continue to draw current and hence waste power and energy. In TM-CIM, not only the array is selected and computed on a column by column basis, but the array is also designed to be column-wise to avoid wasting energy on unselected columns. As shown in Fig. 4, the activation voltages (Act_i) are sent into the array by rows, the cells on the same column are controlled with a column-wise signal (SEL_j), and the weighted sum is computed by the Multiply-and-Accumulate (MAC) operations, which can be represented as

$$I_{SL,j} = \sum_i Act_i G_{i,j} \tag{1}$$

where V_i is corresponding voltage of the input activations, $G_{i,j}$ is the corresponding conductance of the cell representing the i_{th} weight of neuron j.

To program the cells, SEL_j is set to an on-voltage to turn on the gates of the 1T1R cells of the selected column j, and for the selected row i, for SET operations, the input (Act_i) is set to a high voltage (such as V_{prog}) and the Source Line (SL_j) is set to a low voltage (such as 0V); and for RESET operations,

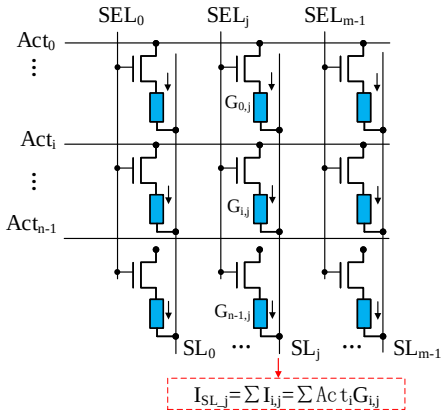


Fig. 4. Proposed 1T1R column-wise array that avoids wasting energy on unselected columns.

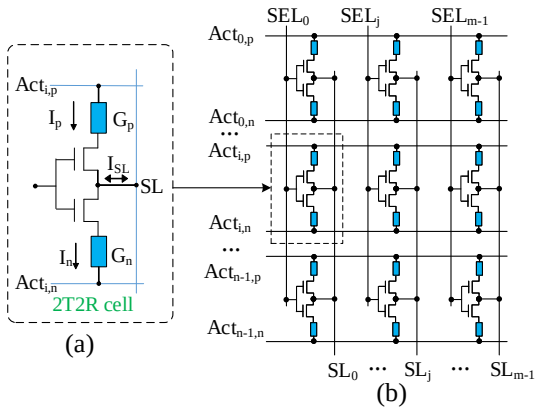


Fig. 5. (a) Proposed 2T2R column-wise array and (b) 2T2R cell.

SL_j is set to V_{prog} and Act_i is set to 0V. V_{prog} may take on different values depending on the target state (i.e. target conductance value). If there is need to program multiple cells on the same column, each row can be a corresponding selected row with its corresponding suitable Act_i. For the unselected rows, the input activation lines are set to floating so as not to alter states of these unselected cells. For the unselected columns, SLs can be set to any voltage that does not alter states of these unselected cells. During inference, all input activation lines are fed with voltages that correspond to input activations from a DNN.

The scheme of the 2T2R cell is proposed by [14], which is shown in Fig. 5 (a). For positive weights, the weight value is stored as G_p, with G_n = conductance of HRS. For negative weights, the weight value is stored in G_n, with G_p = conductance of HRS. Therefore, the weight can be represented as the difference of G_p and G_n. The gates of two transistors are connected with the selected signal (SEL_j). V_p and V_n are connected to the input activations. Then the output current can be expressed as

$$I_{SL} = (V_{Act_{i,p}} - V_{SL}) * G_p - (V_{SL} - V_{Act_{i,n}})G_n \tag{2}$$

For positive activations, $V_{SL} - V_{Act_{i,n}} = 0$, and for negative

activations, $V_{Act_{i,p}} - V_{SL} = 0$. Therefore, the current on j_{th} SL can be represented as

$$I_{SL,j} = \sum_i V_i(Gp_{i,j} - Gn_{i,j}) \tag{3}$$

where $G_{i,j}$ is the corresponding conductance of the signed weights, and

$$V_i = \begin{cases} V_{Act_{i,p}} - V_{SL}, & activation_i \geq 0 \\ V_{SL} - V_{Act_{i,n}}, & activation_i < 0 \end{cases} \tag{4}$$

Fig. 5(b) shows the proposed column-wise 2T2R array which is used to represent signed weights. Programming cells in the proposed column-wise 2T2R array is similar to that of the proposed column-wise 1T1R array, with specialization for 2T2R. When programming a positive weight for SET, $Act_{i,p}$ should be V_{prog} while $Act_{i,n}$ should be either equal to SL_j (which is preferably 0V) or floating, so as to not SET the negative portion of the weight. Conversely, when programming a negative weight for SET, $Act_{i,n}$ should be V_{prog} while $Act_{i,p}$ should be either equal to SL_j (which is preferably 0V) or floating. Similarly, for RESET, the activation line of the selected polarity should be 0V while that of the unselected polarity should be either V_{prog} or floating, and selected column's SL_j should V_{prog}.

Note that in the proposed column-wise array structure, whether for 1T1R or 2T2R, if there is a verification procedure after programming (often referred to as write-verify), since all cells in a column will be turned on, to avoid unselected rows from this column to contribute unwanted read current, their activation lines should be set to floating, to guard against the case where supposed 0V activations on unselected rows have systematic offset voltages.

B. Energy-Efficient Time-Multiplexing Neuron

The proposed time-multiplexing neuron consists of a series of switches, a TIA, and an ADC such as high-precision Successive-approximation (SAR) type [16]. As shown in Fig. 3, the source lines are connected to the TIA via a series of switches (SW) acting as a MUX. The switch on the j^{th} column (SW_i) is controled by the signal SEL_j, which is also the select signal of the array. TIA converts the source line current to voltage and sends it into the ADC. Finally, the ADC latches the TIA's output and converts it into a digital signal.

Fig. 6 shows the work flow of the proposed time-multiplexing neuron. In the first phase (P_0), SEL_0 is turned on, and the first (0^{th}) column generates the results as a current signal, which is passed by SW_0 to the TIA for converting the current to voltage. At end of P_0, the ADC latches TIA's output voltage to start A/D conversion. In the second phase (P_1), the ADC converts the output of 0^{th} column to digital signal, while SEL_1 gets turned on and the TIA converts SL_1's current to voltage. At the end of P_1, the ADC completes the conversion and then latches the TIA's output voltage to start A/D conversion, and so forth. We are assuming that memristor column current stabilization and TIA output voltage stabilization are happening within the same phase. Therefore, if an array shares a ADC with m columns, the latency of each phase is t_p, the latency of this array would be $(m+1) \times t_p$.

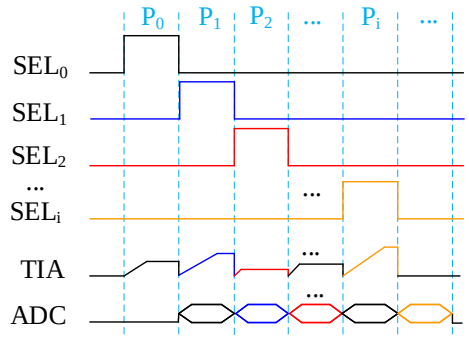


Fig. 6. Work flow of the proposed time-multiplexing neuron.

C. Network Implementation

When implementing a DNN, such as a Convolutional Neural Network (CNN) with TM-CIM, typically each column of the array stores the synaptic weights of a Conv neuron, and each row corresponds to an input activation of current convolution window. As illustrated in Fig. 7 (a), each neuron has $N \times k \times k$ synapses, where N is the input feature maps, and there are M neurons for M output feature maps. For a practical CNN, the first convolution layer is usually small and is able to fit on a single array. For the later layers, the number of inputs can be much bigger than the number of rows in a array. Therefore, as shown in Fig. 7 (b), multiple arrays are required to map a layer, and the partial weighted sums of each array can be summed together in digital domain and in synchronized time-multiplexing across these cores/arrays to obtain the final weighted sum with negligible extra latency.

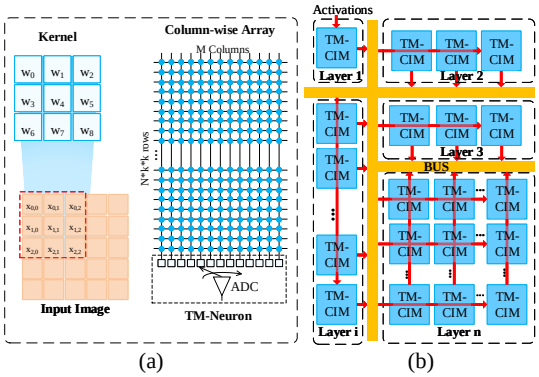


Fig. 7. The Diagram of implementing a DNN with the proposed TM-CIM. (a) Convolution within an array, (b) mapping the DNN with multiple array.

IV. Performance Evaluation

In this section, the energy and area of the proposed TM-CIM are evaluated both on a 256×256 core and on the VGG-16 CNN. The evaluation is based on known parameters in 65nm technology. What's more, a trade-off strategy between the area overhead, peak power, and system latency is also illustrated.

A. Core-level Evaluation

Here we assume 4-bit input is sufficient for quantization-aware trained DNNs like VGG-16, hence core level evaluation

is based on 4-bit input and 256*256 crossbar 1T1R/2T2R cells. The area, peak power, and latency of each 1T1R cell is 0.169 μm^2, 1 μW, and 10 ns, respectively [17][1]. By definition, the area of the 2T2R cell is twice that of the 1T1R, but the power and latency remain the same, because only one out of the two RRAM devices in a 2T2R cell is turned on at a time. 6-bit DAC is used to represent 4-bit inputs. The 6-bit DAC speed is 100MS/s, driving 256 devices, consumes 390.6 μm^2 in area and 60 mW in power by scaling and deriving from Fig. 7 of [6]. In contrast, the input circuit driving 1 device is assumed to only consume 50 μm^2 and 0.001 mW for DAC, and 10 μm^2 and 0.005 mW for OP-AMP [6].To reduce the area overhead of ADC, [17] uses Single-Slope (S/S) ADC to complete the data conversion, whose area and power consumption are 3000 μm^2 and $0.2mW$. However, its latency is as long as 200 ns. In the proposed TM-CIM, a 9-bit 100MS/s SAR ADC whose area and power consumption are 13000 μm^2 and 1.2 mW (on 65nm) is assumed per [16]. A TIA with 2000 μm^2 and 0.5 mW is assumed to be used to provide input to the ADC, and the latency of the TIA is 10 ns. We further assume that both the DAC's and the TIA's output stabilization times can be overlapped and merged as 10 ns.

TABLE I
ENERGY&AREA ESTIMATION COMPARISON BETWEEN CONVENTIONAL CIM AND PROPOSED TM-CIM ON A 256×256 ARRAY WITH 4-BIT INPUT

	Component			Quantity	Area (mm^2)	Peak Power (mW)	Latency (ns)	Energy/MAC (pJ)
Conventional	Array		1T1R	256*256	0.011	65.536	10	0.010
			2T2R	256*256	0.022	65.536	10	0.010
	DAC+OP-AMP*			256	0.100	15360	10	2.343
	S/S ADC			256	0.768	51.200	200	0.156
	Total	**Analog Input**	**1T1R**	-	**0.879**	**15476.736**	**210**	**2.509**
			2T2R	-	**0.890**	**15476.736**	**210**	**2.509**
		Digital Input	**1T1R**	-	**0.779**	**116.736**	**840****	**0.665****
			2T2R	-	**0.790**	**116.736**	**840****	**0.665****
Time-Multiplexing	Array		1T1R	256*256	0.011	0.256	2560	0.010
			2T2R	256*256	0.022	0.256	2560	0.010
	DAC*			256	0.013	0.256	10	0.010
	OP-AMP*			256	0.003	1.280	10	0.050
	SW			1	0.003	-	-	-
	TIA			1	0.002	0.500	2560	0.019
	SAR ADC			1	0.013	1.200	2560	0.047
	Total	**Digital Input**	**1T1R**	-	**0.029**	**1.956**	**10280****	**0.308****
			2T2R	-	**0.040**	**1.956**	**10280****	**0.308****
		Proposed	**1T1R**	-	**0.045**	**3.492**	**5140**	**0.136**
			2T2R	-	**0.056**	**3.492**	**5140**	**0.136**

* The DAC and OP-AMP are only used in analog-input architecture.
**The latency and energy are increased due to multiple computing cycles in digital-input architecture.

Table. I gives the energy and area comparison between conventional CIM, digital-input CIM, and the proposed TM-CIM architecture. Since conventional architecture computes all columns simultaneously, the analog input circuits would consume huge power (15360 mW) and energy (2.343 pJ per MAC), which is extremely unfriendly for edge intelligence implementations. In contrast, the circuit of proposed TM-CIM array only consumes 3.492 mW and 0.136 pJ (per MAC). Digital-input architecture removes the DACs to avoid the substantial power consumption of the input circuits, but multiple cycles are required to implement high-precision inputs, incurring additional energy consumption and latency. Furthermore, digital input's latency is much higher than TM-CIM's. Therefore, the proposed TM-CIM shows best energy efficiency in contrast with other architectures.

[1] [17] assumes $0.9\mu W$ on 65nm RRAM process, whereas we assume 1 μW for ease of illustration.

The 256*256 1T1R array consumes 0.011 mm^2, while the 2T2R array would consume 0.022 mm^2. For conventional-analog architecture, the input circuits would consume 0.100 mm^2, and ADC would consume 768 mm^2. There are more than 98.7% and 97.5% area consumed by the peripheral circuits, which will lead to overall a huge chip yet with incompetitive capacity. The digital-input architecture removes the input circuits to reduce area overhead. However, the TIAs and ADCs for each column still consume considerable area. In time-multiplexing architecture, the area consumed by TIAs and ADCs can be saved significantly by sharing them with multiple columns. Nevertheless, the digital-input architecture incurs extra energy consumption and latency increase as the input resolution increases. In the proposed TM-CIM, the DAC and OP-AMP can be designed with low power and small area since only one column is computed at a time. Moreover, if every 256 columns share one ADC, the area consumed by peripheral circuits can be saved significantly even though a single SAR ADC will consume more area. In TM-CIM, 1T1R architecture has a total area of only 0.045 mm^2, saving around 19.53 times compared to conventional analog-input scheme. The total area of 2T2R architecture is 0.056 mm^2, which is 15.89 times less than that of conventional analog-input scheme.

TABLE II
CORE-LEVEL COMPARISON BETWEEN CONVENTIONAL ARCHITECTURES AND PROPOSED TM-CIM WITH 1T1R CELLS

		Throughput (GMACs)	Peak Power (mW)	Area (mm^2)	Efficiency (TMACs/W)	Density (GMACs/mm^2)
Conventional	Analog-In	312.076	15476.736	8.879	0.399	35.148
	Digital-In	78.019	116.736	0.779	1.504	100.153
Time-Multiplexing	Digital-in	6.375	1.956	0.029	3.246	219.828
	Proposed	12.750	3.492	0.045	**7.352**	**283.333**

*The thoughput estimation is based on 4-bit inputs.
**At the network level, TM-CIM can achieve a throughput comparable to conventional CIM by simply increasing the number of ADCs in the early layers.

Because the DAC and op-amp in TM-CIM are tuned for driving 1 device only at 100MS/s (i.e. 10ns), and yet during row initialization (before column time-multiplexing starts) they will see parasitic capacitance of a whole row of (i.e. 256) transistors, we further conservatively assume that the initialization will take as long as multiplexing all 256 columns, which is 2570ns. Hence, the latency of TM-CIM is 2570*2 = 5140 ns, which is reasonable for edge intelligence since the latency is mostly determined by the slowest layer in the network level Table. II gives the core-level comparison between conventional CIM and the proposed TM-CIM. The proposed TM-CIM shows the best energy efficiency and density, even though with a lower throughput. Moreover, at the network level, TM-CIM can achieve a throughput comparable to conventional CIM by simply increasing the number of ADCs in the early layers, and the trade-off strategy will be illustrated in Section IV-C.

Table. III gives the comparison between the proposed TM-CIM and other ADC-shared architectures. [9] shares each ADC with 4 columns to reduce the area overhead. [12] uses 32-to-1 multiplexers to share a 4-bit ADC with 32 columns. However, these architectures still require per-column S&H circuits which also implies per-column TIAs, and the ADCs

TABLE III
PERFORMANCE COMPARISON BETWEEN TM-CIM AND OTHER ADC-SHARED ARCHITECTURES

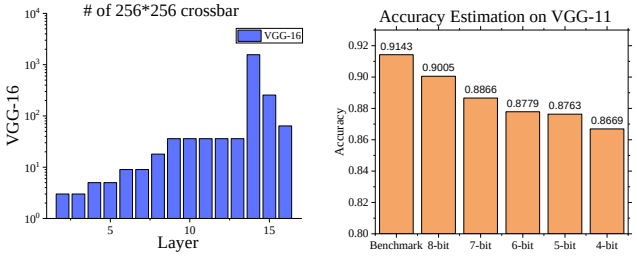
	[9]	[12]	**TM-CIM**
Technology (nm)	130	40	65
Size	128*128	256*256	256*256
MUX	4-to-1	32-to-1	256-to-1
Latency (ns)	400	-	5140
Area (mm^2)	0.0704	0.437	0.056
Energy/MAC (pJ)	371.89	1158.49	0.136

still consume most of the energy and area. Therefore, the proposed TM-CIM still consumes the lowest area and energy.

B. Network-level Evaluation

The network-level energy estimation is based on VGG-16 using ImageNet dataset, and the accuracy estimation is based on VGG-11 using CIFAR-10 dataset. Multiple 256*256 2T2R crossbars are required to implement signed weights in each layer of the network. Fig. 8(a) shows the required number of crossbars for each layer of VGG-16, and the total number of required crossbars is 2121. The accuracy simulation is performed in PyTorch platform using the DoReFa quantization-aware training framework [18] and core-size limitation. As shown in Fig. 8(b), the accuracy drops slightly when activations and weights are quantized to 8-bit, and is suitable for implementation on the proposed TM-CIM.

Note that in this evaluation we focus on the aggregation of power/energy consumption of CIM cores, and the aggregate latency seen by the application. Hence in this study we do not include energy overhead for data traffic in the chip, nor for input control blocks that manage and fetch the input data for each convolutional layer, as their estimations would vary significantly depending on the choice of implementation of Network-on-Chip routers and input control blocks.



(a) # of Crossbar for each layer. (b) Accuracy estimation.

Fig. 8. (a)Required Number of Crossbars for each layer of VGG-16 and ImageNet dataset, and (b)accuracy estimation of the proposed TM-CIM with different precision of activations and weights on VGG-11 and Cifar-10 dataset.

Table. IV gives the performance estimation of implementing VGG-16 with 256*256 2T2R arrays and 8-bit ADC, and Table. V gives the performance comparison between the proposed TM-CIM and other architectures. It is assumed that the arrays used to implement the same convolution layers can be computed in parallel. For conventional analog-input architecture, the total area is 1887.997 mm^2, while the array area is only 46.983 mm^2. The chip area is huge and most of which is consumed by the peripheral circuits. In fact, due to size constraints of photomasks, a chip even if occupying the entire mask is usually limited to around 800 mm^2 (exemplified by some of the biggest GPU dies). For conventional digital-input architecture, the total area is reduced to 1675.911 mm^2, most of which is consumed by per-column TIAs and ADCs. The time-multiplexing digital-input architecture has the minimum area of 85.161 mm^2. However, its total energy consumption and latency is still higher than the proposed TM-CIM, after considering the number of computing cycles. In TM-CIM, if every 256 columns share one ADC, the area consumed by peripheral circuits can be reduced significantly to 70.757 mm^2 even though a single faster ADC will consume more area, the 2T2R array consumes 46.983 mm^2, and the total area is 117.739 mm^2. Compared with the conventional analog-input architecture, the area saving is more than 16 times.

TABLE IV
PERFORMANCE ESTIMATION FOR IMPLEMENTING VGG-16 WITH PROPOSED ARCHITECTURE

Component	Quantity	Area (mm^2)	Peak Power (W)	Latency (ms)	Energy/Img (mJ)
2T2R Array	2121	46.983	0.042	32.113	0.154
DAC	542976	27.149	0.042	32.113	0.155
OP-AMP	542976	5.430	0.208	32.113	0.774
MUX	2121	6.363	-	-	-
TIA	2121	4.242	0.133	32.113	0.068
SAR-ADC	2121	27.573	0.318	32.113	0.817
Total	-	**117.739**	**0.742**	**64.225**	**1.968**

TABLE V
PERFORMANCE ESTIMATION COMPARISON ON VGG-16 BETWEEN THE PROPOSED ARCHITECTURE AND THE OTHERS

Architecture		Area (mm^2)	Peak Power (W)	Latency (ms)	Energy/Img (mJ)
Conventional	Analog Input	1887.997	2527.996	10.537	59.256
	Digital Input	1675.911	30.376	20.070	11.517
Time-multiplexing	Digital Input	85.161	0.492	128.451	4.159
	Proposed	**117.739**	**0.742**	**64.225**	**1.968**

In conventional architecture, up to 256 columns in each array will compute at a time. Therefore, the peak power will be particularly high, especially for the analog-input circuits. This is evidenced in Table. V, where total peak power is as high as 2527.996 W which is in fact impractical. Note that this estimation is already significantly reduced by considering that the Fully Connected (FC) layers can be calculated gradually instead of all at once. The key reason is that the DACs here are too power-hungry despite scaling down their resolution following the benchmark in [6]. The digital-input architecture reduces the peak power significantly, but it requires more computing cycles which also increases energy consumption, and for our evaluation on VGG-16 the proposed TM-CIM is still the most energy efficient.

In TM-CIM, only one column will be turned on in each array, thus the peak power can be efficiently reduced. Since the unselected columns can be totally turned off, TM-CIM would not consume extra energy. Contrasted with conventional anlog-input architecture, the energy consumption per image can be saved by 7.67 times . The latency in proposed TM-CIM is acceptable since in this illustrative example we adopt an SAR ADC with higher speed by sacrificing area moderately.

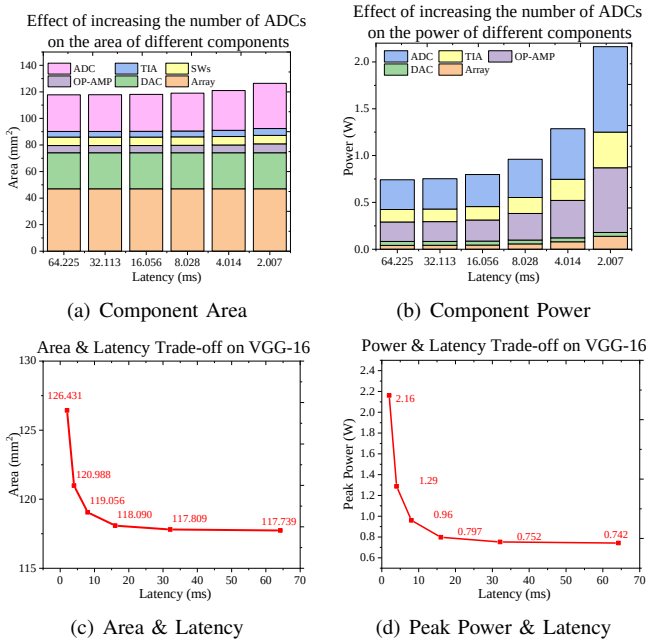
TABLE VI
AREA, PEAK POWER AND LATENCY FOR MINIMIZING THE LATENCY OF VGG-16 WITH MULTIPLE ADCS IN EARLIER CONVOLUTIONAL LAYERS.

Layer	# of operations	Effective columns in array	# of array	# of ADC in each array/layer	Area (mm^2)	Peak Power (W)	Latency (ms)
Conv3-64	224*224	64	1	32/32	0.600	0.060	2.007
Conv3-64			3	32/96	1.800	0.274	2.007
Conv3-128	112*112	128	3	16/48	0.957	0.137	2.007
Conv3-128			5	16/80	1.595	0.248	2.007
Conv3-256	56*56	256	5	8/40	0.892	0.124	2.007
Conv3-256			9	8/72	1.606	0.235	2.007
Conv3-256			9	8/72	1.606	0.235	2.007
Conv3-512	28*28	256	18	2/36	1.315	0.091	2.007
Conv3-512			36	2/72	2.631	0.182	2.007
Conv3-512			36	2/72	2.631	0.182	2.007
Conv3-512	14*14	256	36	1/36	1.998	0.093	1.004
Conv3-512			36	1/36	1.998	0.093	1.004
Conv3-512			36	1/36	1.998	0.093	1.004
FC-4096	(512*7*7)*4096	256	1568	1/1568	87.042	0.112	0.253
FC-4096	4096*4096	256	256	1/256	14.211	0.112	0.253
FC-1000	4096*1000	256	64	1/64	3.553	0.112	0.253
Total	-	-	**2121**	-/**2616**	**126.431**	**2.162**	**2.007**

C. Area, Power, and Latency trade-off

The latency can be further reduced by increasing the number of ADC in some arrays. In VGG-16, the latency is mainly determined by the first two convolutional layers. Therefore, two ADCs can be adopted in the four arrays which implement the first two layers. In this way, these arrays will compute two columns simultaneously, and the latency will be reduced in half with minimal area increase.

As shown in Table. VI, to minimize the latency of VGG-16, 32 ADCs are adopted in the first two convolution layers, and in the later layers, the number of TIAs and ADCs can be scaled down. The minimum latency is 2.007 ms with the area consumption of 1276.431 mm^2.



(a) Component Area (b) Component Power

(c) Area & Latency (d) Peak Power & Latency

Fig. 9. The effect of increasing the number of ADCs on the (a) area and (b) power of different component, and the trade-off between the latency and (c) area overhead, (d) peak power.

However, it is not efficient to push to the minimal latency, because increasing the number of ADCs in each array requires the input circuits to drive more devices, which results in a significant increase in peak power and area. Fig. 9(a), 9(b) shows the effect of increasing the number of ADCs on the area and power of different components. The number of TIAs and ADCs has increased, resulting in more area and power consumption. The number of OP-AMPs remains the same, but the area and power consumption have been increased to drive more columns at the same time. Therefore, there is a trade-off between the area, peak power, and latency. As shown in Fig. 9(c), the latency can be reduced to 16.056 ms (1/4 of not using TIA&ADC overhead) by only increasing the area of 0.297 mm^2 (0.25%) and peak power of 0.055 W (7.41%) on VGG-16. In contrast, the area and peak power will increase considerably if much smaller latency is sought.

V. CONCLUSION

In this paper, an energy-efficient time-multiplexing memristive analog computing architecture is proposed. A column-wise memory array is designed to reduce the peak power and area consumption as well as avoiding wasting energy on unselected columns. The time-multiplexing neuron is designed to take full advantage of the ADC performance. The core-level evaluation on 256*256 crossbar has shown that the proposed TM-CIM has a small energy consumption of 0.136 pJ/MAC, and the area is only 0.044 mm^2 for 1T1R array and 0.055 mm^2 for 2T2R array. When implementing complex DNNs such as VGG-16, TM-CIM can save area and energy consumption significantly. The trade-off strategy between the area, power and latency is used to find the best way to implement a DNN like VGG-16. The proposed TM-CIM has low energy consumption, small area overhead, and acceptable latency, which is well-suited to edge intelligence applications.

Because the DAC and op-amp only need to drive 1 device in TM-CIM, their power and energy is much improved compared to the case where they need to drive an entire row (of 256) devices. However, more optimizations should be possible for the input circuit, which we expect to investigate as future work.

ACKNOWLEDGMENT

This research is supported by Programmatic grant no. A1687b0033 from the Singapore government's Research, Innovation and Enterprise 2020 plan (Advanced Manufacturing and Engineering domain) and administered by the Agency for Science, Technology and Research.

REFERENCES

[1] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," Computer Science Review, vol. 40, p. 100379, 2021.

[2] G. W. Burr, R. M. Shelby, A. Sebastian, S. Kim, S. Kim, S. Sidler, K. Virwani, M. Ishii, P. Narayanan, A. Fumarola et al., "Neuromorphic computing using non-volatile memory," Advances in Physics: X, vol. 2, no. 1, pp. 89–124, 2017.

[3] K. Huang, Y. Ha, R. Zhao, A. Kumar, and Y. Lian, "A low active leakage and high reliability phase change memory (pcm) based non-volatile fpga storage element," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 61, no. 9, pp. 2605–2613, 2014.

[4] S. Yin, Y. Kim, X. Han, H. Barnaby, S. Yu, Y. Luo, W. He, X. Sun, J.-J. Kim, and J.-s. Seo, "Monolithically integrated rram- and cmos-based in-memory computing optimizations for efficient deep learning," IEEE Micro, vol. 39, no. 6, pp. 54–63, 2019.

[5] Z. Wang, C. Li, P. Lin, M. Rao, Y. Nie, W. Song, Q. Qiu, Y. Li, P. Yan, J. P. Strachan et al., "In situ training of feed-forward and recurrent convolutional memristor networks," Nature Machine Intelligence, vol. 1, no. 9, pp. 434–442, 2019.

[6] A. Amirsoleimani, F. Alibart, V. Yon, J. Xu, M. R. Pazhouhandeh, S. Ecoffey, Y. Beilliard, R. Genov, and D. Drouin, "In-memory vector-matrix multiplication in monolithic complementary metal–oxide–semiconductor-memristor integrated circuits: Design choices, challenges, and perspectives," Advanced Intelligent Systems, vol. 2, no. 11, p. 2000115, 2020.

[7] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," ACM SIGARCH Computer Architecture News, vol. 44, no. 3, pp. 14–26, 2016.

[8] S. Zhang, K. Huang, and H. Shen, "A robust 8-bit non-volatile computing-in-memory core for low-power parallel mac operations," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 67, no. 6, pp. 1867–1880, 2020.

[9] P. Yao, H. Wu, B. Gao, J. Tang, Q. Zhang, W. Zhang, J. J. Yang, and H. Qian, "Fully hardware-implemented memristor convolutional neural network," Nature, vol. 577, no. 7792, pp. 641–646, 2020.

[10] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[11] W. Wan, R. Kubendran, S. B. Eryilmaz, W. Zhang, Y. Liao, D. Wu, S. Deiss, B. Gao, P. Raina, S. Joshi et al., "33.1 a 74 tmacs/w cmos-rram neurosynaptic core with dynamically reconfigurable dataflow and in-situ transposable weights for probabilistic graphical models," in 2020 IEEE International Solid-State Circuits Conference-(ISSCC). IEEE, 2020, pp. 498–500.

[12] J.-H. Yoon, M. Chang, W.-S. Khwa, Y.-D. Chih, M.-F. Chang, and A. Raychowdhury, "29.1 a 40nm 64kb 56.67 tops/w read-disturb-tolerant compute-in-memory/digital rram macro with active-feedback-based read and in-situ write verification," in 2021 IEEE International Solid-State Circuits Conference (ISSCC), vol. 64. IEEE, 2021, pp. 404–406.

[13] W.-H. Chen, W.-J. Lin, L.-Y. Lai, S. Li, C.-H. Hsu, H.-T. Lin, H.-Y. Lee, J.-W. Su, Y. Xie, S.-S. Sheu, and M.-F. Chang, "A 16mb dual-mode reram macro with sub-14ns computing-in-memory and memory functions enabled by self-write termination scheme," in 2017 IEEE International Electron Devices Meeting (IEDM), 2017, pp. 28.2.1–28.2.4.

[14] Q. Liu, B. Gao, P. Yao, D. Wu, J. Chen, Y. Pang, W. Zhang, Y. Liao, C.-X. Xue, W.-H. Chen et al., "33.2 a fully integrated analog reram based 78.4 tops/w compute-in-memory chip with fully parallel mac computing," in 2020 IEEE International Solid-State Circuits Conference-(ISSCC). IEEE, 2020, pp. 500–502.

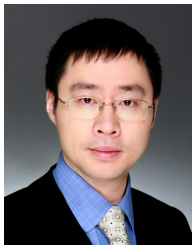
[15] F. Tan, Y. Wang, Y. Yang, L. Li, T. Wang, F. Zhang, X. Wang, J. Gao, and Y. Liu, "A reram-based computing-in-memory convolutional-macro with customized 2t2r bit-cell for aiot chip ip applications," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 67, no. 9, pp. 1534–1538, 2020.

[16] X. Zhu, Y. Chen, S. Tsukamoto, and T. Kuroda, "A 9-bit 100ms/s tri-level charge redistribution sar adc with asymmetric cdac array," in Proceedings of Technical Program of 2012 VLSI Design, Automation and Test. IEEE, 2012, pp. 1–4.

[17] Q. Wang, X. Wang, S. H. Lee, F.-H. Meng, and W. D. Lu, "A deep neural network accelerator based on tiled rram architecture," in 2019 IEEE international electron devices meeting (IEDM). IEEE, 2019, pp. 14–4.

[18] S. Zhou, Z. Ni, X. Zhou, H. Wen, Y. Wu, and Y. Zou, "DoReFa-Net: Training Low Bitwidth Convolutional Neural Networks with Low Bitwidth Gradients," CoRR, vol. abs/1606.06160, 2016. [Online]. Available: http://arxiv.org/abs/1606.06160

Rui Xiao (Student Member, IEEE) received the Bechalor degree from the College of Information Science Electronic Engineering, Zhejiang University in 2019. Currently she is pursuing the Ph.D degree in the School of Information Science and Electronic Engineering, Zhejiang University. She is also a joint Ph.D student with the A*STAR Research Attachment Programme in Agency for Science, Technology, and Research under the supervision of Dr. Jiang. Her research interests include in-memory computing circuits and systems design using emerging resistive non-volatile memories, deep learning accelerators, and embedded system design.



Wenyu Jiang (Senior Member, IEEE) received the Ph.D. degree in computer science from Columbia University, New York City, in 2003. From 2003 to 2011, he was with Dolby Laboratories, as a Staff Engineer, and researched on quality of service for streaming media over wireless, P2P-based digital rights management (DRM) friendly content distribution, and multimedia fingerprinting and retrieval. Since 2011, he has been with the Institute for Infocomm Research, A*STAR. He is currently a senior scientist. His research areas span from memory computing-based multimedia retrieval and fuzzy pattern search, signal processing and applications for fiber Bragg gratings (FBG) sensors to neuromorphic computing, including spiking neural network algorithms and in-memory computing. He is the Vice Chair of IEEE SSCS Singapore Chapter.

Piew Yoong Chee received the B.E. degree in electrical engineering from the University of Malaya. Malaya, Malaysia, and the M. Eng. degree from Nanyang Technological University, Singapore. Prior to joining Institute for Infocomm Research; he worked at Institue of Microelectronics, Philips Semiconductor, Cadence Design Centre in Singapore for several years. He is currently a Senior Manager with the Robotics and Autonomous Systems Dept. His research interests and activities include analog and RF integrated circuit (RFIC) designs; analog mixed signal peripheral circuits for neuromorphic computing system.