An Architecture-Level Energy and Area Estimator for Processing-In-Memory Accelerator Designs

Yannan Nellie Wu* Vivienne Sze* Joel S. Emer*†
*Massachussets Institute of Technology †NVIDIA Corporation
{nelliewu, sze}@mit.edu, emer@csail.mit.edu

***Abstract*—Processing-in-memory (PIM) deep neural network (DNN) accelerators, which aim to improve energy/area efficiency of DNN processing by integrating computation into data storage, have gained popularity in recent years. Therefore, it is attractive to have a generally applicable framework that is able to quickly provide insights into the various trade-offs involved in PIM accelerator designs. We present an architecture-level design estimation framework for PIM accelerators that allows easy representations of the designs with provided architecture templates and component design templates, performs analytical runtime simulations, and produces technology-dependent area and energy estimations. We show that the framework can be easily used to evaluate state of the art PIM accelerator designs; it achieves 95% accurate total energy estimations and reproduces exact area breakdowns of the components in the design. Related open-source code is available at http://accelergy.mit.edu/.**

***Index Terms*—Architecture-Level Design Evaluations, Deep Neural Network Accelerators, Processing-in-Memory**

I. Introduction

In recent years, many DNN accelerators have been proposed [1]–[8] to improve the energy efficiency of DNN applications by exploiting the application-specific properties. A class of these proposed accelerators [5]–[8] are designed to reduce the amount of data movement by integrating the computation into SRAM, DRAM, emerging non-volatile memory, etc. As a result of the integration, these accelerators are able to perform the computations at the location where the weights are stored and only read/write the input data/computed partial sums from/to memories. We refer to them as PIM accelerators. Since building a physical PIM accelerator is time-consuming, it is attractive for designers to perform early-stage design estimations of a PIM architecture without developing the physical design layout of the memory bit-cells and the integrated mixed-signal circuits. We refer to these early design stage estimations as architecture-level estimations. Although some existing works report architecture-level estimations of their PIM accelerators [5], [6], each of them performs estimations with different evaluation frameworks, which are often not publicly available. This lack of generally applicable open-source PIM accelerator estimation framework makes it hard to quickly perform design space estimations on new designs, as well as to compare to existing designs. To address the above mentioned problems, we propose a generally applicable architecture-level energy and area estimation framework for PIM accelerators; the framework extends two existing works: Timeloop [9] and Accelergy [10].

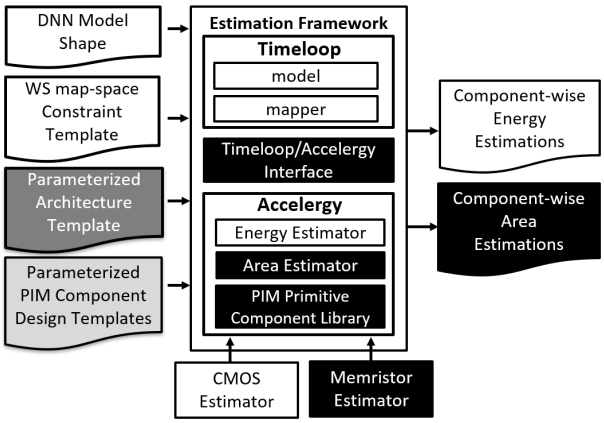

Fig. 1: The block diagram of the design estimation framework for PIM accelerators. The black-shaded blocks show the extensions added to the existing frameworks.

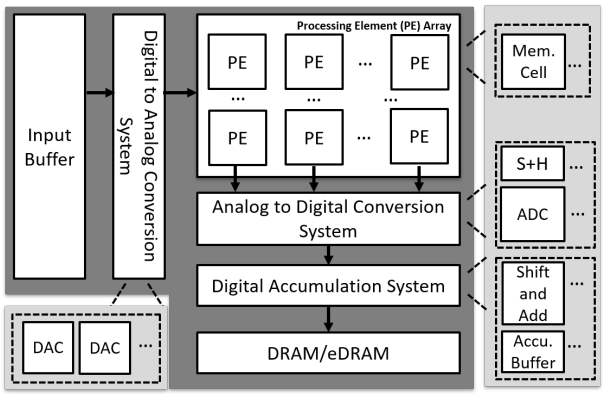

Fig. 2: The block diagram of the architecture template (shaded in dark gray) and the component design templates (shaded in light gray).

II. Estimation Framework

Figure 1 shows the high-level block diagram of our PIM accelerator estimation framework, which contains a Timeloop-based mapping space explorer and Accelergy-based energy/area estimator. The existing tools are extended and integrated to allow architecture-level energy/area estimations. There are four inputs to the framework: (1) a DNN model shape that specifies the dimensions of the workload; (2) map-space constraints file that describes the hardware resource

allocation limitations; (3) an architecture that describes the high-level components in the design; (4) a set of component designs that describes the hardware details of each component. As shown in Figure 1, we provide templates for (2), (3), and (4) to allow much easier PIM design representations. Figure 2 shows a block diagram of the architecture template and the set of component design templates. Each processing element (PE) in the architecture is responsible for performing multiply accumulate (MAC) operations. The analog-to-digital (A2D) and digital-to-analog (D2A) conversion systems are responsible for converting between digital and analog signals. An example design of the A2D conversion system, as shown in Figure 2, consists of multiple ADCs and sample-and-hold (S+H) units. Both the architecture and the component design templates are associated with parameters (*e.g.*, the number of PE rows/cols are parameters associated with the architecture template) that can be set by the designer to easily represent different designs. Finally, since many PIM accelerators implement weight stationary (WS) dataflow [11]. We provide an example map-space constraints file that describes the WS dataflow in terms of the the components described in the provided architecture template. The constraints file can be easily modified to represent other types of dataflows and its detailed format can be found in Timeloop [9]. With the available templates, designers only need to provide the necessary parameters associated with the templates to define a PIM accelerator. If the designs have unique features that are not captured by default, the templates are also easily customizable to represent the additional features. Additionally, we provide a primitive component library and a set of estimation plug-ins to allow easy incorporation of energy and area charaterizations of the unique devices used in a particular design.

III. Experimental Results

We first use the proposed framework to evaluate the energy consumption of the 65nm ADC-based architecture described in CASCADE [7], which consists of 80 tiles of memristor arrays. Each tile consists of 80 64-by-64 1-bit memristor arrays (represented as a 64-by-320 16-bit PE array), an A2D conversion system with 6-bit ADCs and S+Hs, a D2A conversion system with 1-bit DACs, and a digital accumulation system with 16-bit shift-and-add accumulators (SA). To fully define each IMA architecture, we only need the specify 2 architecture template parameters: # of PE rows/columns and input buffer size. We specified 11 more parameters for the component design templates. Other parameters are either set to default values or auto-derived. Figure 3 (a) shows the layer by layer absolute energy consumption of the convolutional layers in the VGG workload [12]. The total energy consumption is linearly related to the number of MACs in each layer. This relationship is expected as most of the energy is consumed by the A2D conversion system and the PE array, as shown in the figure, which are both linearly related to the number of analog MAC computations. Figure 3 (b) shows the total energy consumption and component-wise energy breakdown comparing to the results presented in CASCADE [7]. We are

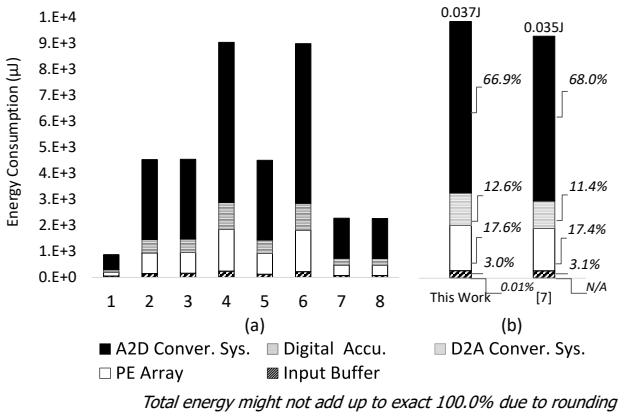

Fig. 3: (a) The energy breakdowns across VGG layers (b) Total energy reported by our framework and [7]. The D2A conversion system's negligible energy is not reported in [7] (N/A).

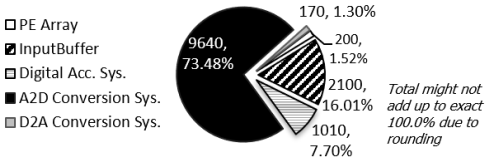

Fig. 4: The absolute area and the area breakdown of the components in an ISAAC IMA. Labeled in the format of *absolute area (μm^2), area breakdown*.

able to achieve approximately 95% match for total energy estimation, and closely capture the energy breakdown of the components in the design.

Since CASCADE deos not present exact area breakdowns, we then perform an area estimation validation on the 32nm ISAAC design's in-situ multiply-accumulate (IMA) units [6]. Each IMA is composed of eight 128x128 2-bit memory cells, eight 8-bit ADCs, 1024 S+Hs, 1024 1-bit DACs and 1024 16-bit SAs. The related parameters in the architecture template are updated to represent this new architecture. Figure 4 shows the area breakdown reported by the estimation framework for each type of components in an IMA. The reported architecture's area breakdown matches with the data presented in ISAAC, showing that our framework correctly interprets the user-specified parameters and performs accurate area estimations.

IV. Conclusion

This work proposes a generally applicable architecture-level energy and area estimation framework for PIM accelerator designs. To simplify the design representation, the proposed framework provides users a parameterized architecture template, a set of parameterizable component design templates, a library of primitive components, and a set of table-based estimation plug-ins. To simplify the dataflow representation, the proposed framework provides a constraint file that specifies a WS dataflow. We perform energy and area validations on PIM architecture to demonstrate the accuracy, simplicity, and flexibility of the proposed framework.

REFERENCES

[1] Y.-H. Chen, T. Krishna, J. Emer, and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," in *ISSCC*, 2016.

[2] Y.-H. Chen, T.-J. Yang, J. Emer, and V. Sze, "Eyeriss v2: A Flexible Accelerator for Emerging Deep Neural Networks on Mobile Devices," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2019.

[3] K. Hegde, J. Yu, R. Agrawal, M. Yan, M. Pellauer, and C. Fletcher, "UCNN: Exploiting Computational Reuse in Deep Neural Networks via Weight Repetition," in *ISCA*, 2018.

[4] V. Akhlaghi, A. Yazdanbakhsh, K. Samadi, R. K. Gupta, and H. Esmaeilzadeh, "SnaPEA: Predictive early activation for reducing computation in deep convolutional neural networks," in *ISCA*, 2018.

[5] P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, and Y. Xie, "PRIME: A novel processing-in-memory architecture for neural network computation in reram-based main memory," in *ISCA*, 2016.

[6] 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," in *ISCA*, 2016.

[7] T. Chou, W. Tang, J. Botimer, and Z. Zhang, "CASCADE: Connecting RRAMs to Extend Analog Dataflow In An End-To-End In-Memory Processing Paradigm," in *MICRO*, 2019.

[8] L. Song, X. Qian, H. Li, and Y. Chen, "PipeLayer: A pipelined reram-based accelerator for deep learning," in *HPCA*, 2017.

[9] A. Parashar, P. Raina, Y. S. Shao, Y.-H. Chen, V. A. Ying, A. Mukkara, R. Venkatesan, B. Khailany, S. W. Keckler, and J. Emer, "Timeloop: A Systematic Approach to DNN Accelerator Evaluation," in *ISPASS*, 2019.

[10] Y. N. Wu, J. S. Emer, and V. Sze, "Accelergy: An Architecture-Level Energy Estimation Methodology for Accelerator Designs," in *ICCAD*, 2019.

[11] Y.-H. Chen, J. Emer, and V. Sze, "Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks," in *ISCA*, 2016.

[12] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *ICLR*, 2015.