5

Emerging In-memory Computing for Neural Networks

Nellie Laleni[1], Taha Soliman[2], Alptekin Vardar[1], and Thomas Kämpfe[1]

[1]Fraunhofer IPMS
[2]Robert Bosch GmbH

Abstract

Recently, deep neural networks (DNNs) have proved their success in performing various tasks at high accuracy. However, these networks come at a high cost of computational and memory requirements and with the continuously growing neural networks sizes, conventional von Neumann accelerators hit the memory wall. Processing-in-memory (PIM) acceleration is heavily investigated to deliver the aforementioned requirements with a great potential to further accelerate these application and meet the possible future needs. In this chapter, we explore the state-of-the-art, challenges and future possibilities of the PIM based DNN accelerators. First, we explore various volatile and non-volatile memory cells that are commonly used for PIM architectures. Second, we discuss the possible approaches to design a PIM accelerator (digital, analog, mixed-signal processing). Third, we investigate the operational accuracy these architectures are offering, the requirements these architectures enforce when it comes to the inferred network quantization. Finally, we conduct an extensive comparison between these architectures.

5.1 Memory Technologies

5.2 In-Memory Architectures

5.2.1 Volatile Memories

For many years, the main memory cells to provide storage space for any computational process were volatile memories. Volatile Memory is called the memory which retains the data only as long as there is power supplied. The most common volatile memories are the dynamic random access memory (DRAM) and the static random-access memory (SRAM).

An SRAM cell is constructed from two transistors and four more transistor forming two cross-coupled inverters storing one single bit (Figure 5.1). The SRAM cell is preferred among the volatile memories due to its low access time and high performance comparing with the DRAM of which the threshold voltage of the access transistor is very high [1]. However, SRAM is considered as en expensive memory and dominates a high amount of area in a digital chip and also the total chip leakage current [2]. Although, in advanced technologies, the decreased VDD can lower the leakage current, the storage capacitance of a bitcell SRAM is reduced and soft error rate (SER) is introduced [3]. Moreover, with respect to the NVM, it lacks of high power efficiency and exposes higher read delay, for higher temperature [4]. SRAM cell is often used as the main memory cell where the MAC operations are performed [5], [6].

At the moment, DRAM is the most popular type of memory when designing an AI accelerator and needing memory storage. Its simple design consists of a transistor and a storage capacitance (Figure 5.1). The need for very large and dense quantities of memory, led to the usage of DRAM to be the main off-chip memory [7]. The cost of the DRAM cell is less than the SRAM, but the DRAM memory needs a circuit to periodically refresh the memory since the

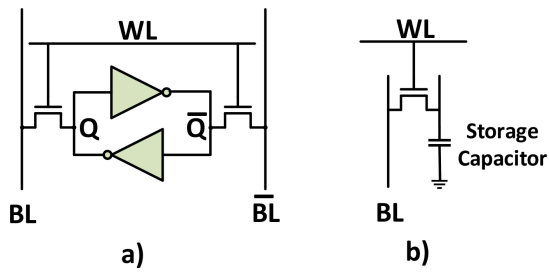

Figure 5.1 Conventional volatile memory cells a) 6T SRAM cell and b) DRAM cell.

capacitance needs to be discharge also, DRAM's capacitance leaks current and the data has to be transferred at the main chip, so higher latency and power needed [7]. Recent studies has designed new techniques to compensate those effects like the time minimization of the DRAM access [8] or the latency in sense of energy per access [9]. Nevertheless, the biggest concern of the DRAM memory seems to be the scaling limit with the newer technologies and the smaller sizes of the transistor [10].

5.2.2 Non Volatile Memories

The aforementioned disadvantages of the volatile memories lead the researchers to investigate another type of memories, the non volatile memories (NVM). These memories have the ability to retain the data even if the power supply is disconnected. They present high power efficiency with respect to the volatile memories and low latency since the network's operations happen inside the memory. Although the low cost and high density, they may present some reliability issues like data retention and finite endurance, resulting in high bit-error-rates (BER) in the stored weights [11]. Recent studies have showed some solutions for the BER problem like error correction code (ECC) [12, 13], but these techniques demand high power during the read operation which can not be compatible with the new edge technologies. Some of the most popular NVM are the flash memory, the resistive random-access memory (ReRAM or RRAM) and the ferroelectric RAM (FeRAM, F-RAM or FRAM).

A Flash memory cell is simply a MOSFET cell, except that a polysilicon floating gate (or a silicon nitride charge trap layer) is sandwiched between a tunnel oxide and an interpolyoxide to form a charge storage layer [10]. The floating gate is used to store the data and it provides programming and erasing process. However, the Flash memory lacks of scalability since a conventional Flash type of memory needs a tunnel oxide layer thickness of 8nm to avoid charge loss and maintain the data (data retention) for 10 years [14]. As a result, a reduction of device dimension could cause threshold voltage shift, retention, endurance and dielectric leakage [10, 15].

The ReRAM cell consists of one Memristor and one transistor. Memristor is a device which acts as a programmable resistance, so the voltage level of the transistor can be determined. This voltage level represents the state/value of the weights in a neural network. However, concerning to the power consumption, the ReRAM presents gate leakage and relatively high power consumption for low latency and vise versa [4]. Moreover, a significant effect

which should be taken into consideration when designing a NN accelerator with ReRAM, is that for any small variation of the V_{th}, the write delay is increased exponentially [4].

In order to avoid the ReRAMs high write power and long read latency (RC delay), studies focused on FeRAM as one popular upcoming technologies for NVM [16]. It is firstly introduced in [17] where ferroelectricity in silicon doped hafnium oxide ($Si : HFO_2$) is presented as a high scalable and complementary metal-oxide semiconductor (CMOS) compatible technology (ferroelectric field-effect transistor - FeFET). It consists of a transistor and a capacitor structure which gives the transistor the ability to be programmed and erased in different levels with respect to its V_{th} It has already been integrated into various CMOS new edge technologies [18, 19] and it presents low device-to-device variation. One disadvantage of the FeFET is that during the read operation, a leakage current can be detected which is involved to a small writing pulse to each cell [10]. It is a fast memory (higher read speed than Flash and SRAM memory) with high endurance and low hold power making FeFET a competitive technology of NVM.

In this section, we review the different design trends in the field of in-memory computing architectures based on different memory technologies and targeting various neural networks. The different architectures are explored according to their computational domain, flexibility and programmability, used technology, target networks and their representation and finally the reliability and accuracy of the computations.

5.2.3 Computational Domain

As In-memory architectures main idea is to perform the target operation in memory by leveraging the memory cell properties in the analog domain or in more digital approach. However, pure analog domain usually targets neuromorphic computations which is not the main scope of this survey. In this section, we will be exploring two main trends in In-memory architecture; the mixed signal based architectures and digital based architectures. For each, we will investigate the possible advantages and disadvantages each is offering as well as the potential each hold for future applications.

5.2.3.1 Mixed signal approach

In this approach, the main computation is realized by using the analog properties of the memory cell within the memory crossbar or sub-array. As shown in Figure 5.2, the min idea here depends on storing the weight value

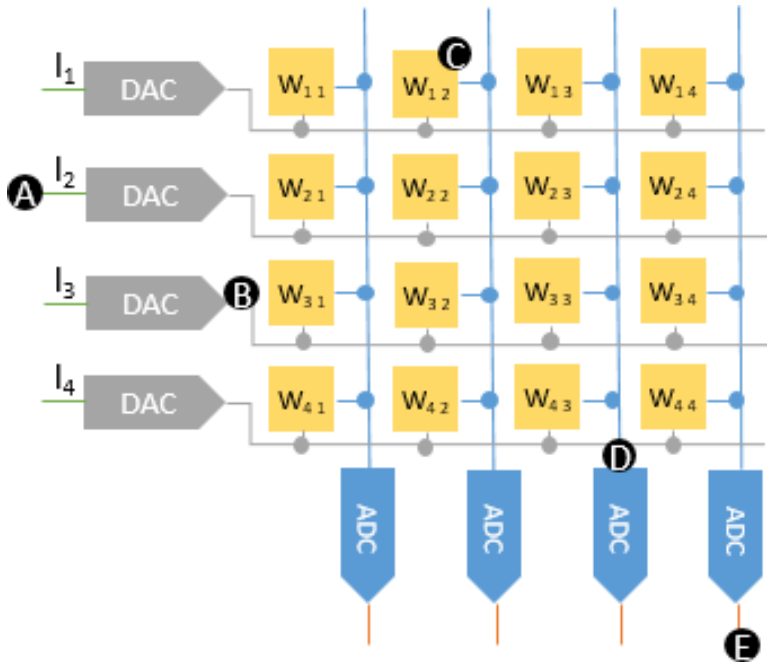

Figure 5.2 General concept of mixed-signal in-memory crossbar (A) The digital activation of the computed layer. (B) DACs convert the digital input into an analog signal to be applied to the memory cell. (C) Memory cell storing the kernel value of the currently computed layer. (D) The summation line which accumulates the result signal out of the memory cell representing the operation results. (E) ADCs convert back the result into the digital domain for any further processing.

within memory cell and using a digital to analog converter (DAC) to represent the input feature value as an applied voltage. The result of such multiplication operation between the input and weight values are represented by the output signal of the memory cell as explained previously. Based on kirchhoff law, the result of different multiplications along the BL is accumulated and finally forwarded to the analog to digital converter (ADC) unit that yields the final result of the performed MAC operations.

Several architectures [20], [21] adopt this crossbar organization as their main processing element. This structure became a very popular crossbar structure because of its very high throughput as well as matching the dominant MAC operation. However, this approach holds couple of drawbacks as it requires a number of ADCs and DACs which reflects on the chip area and the power consumption of the overall processing element. These components

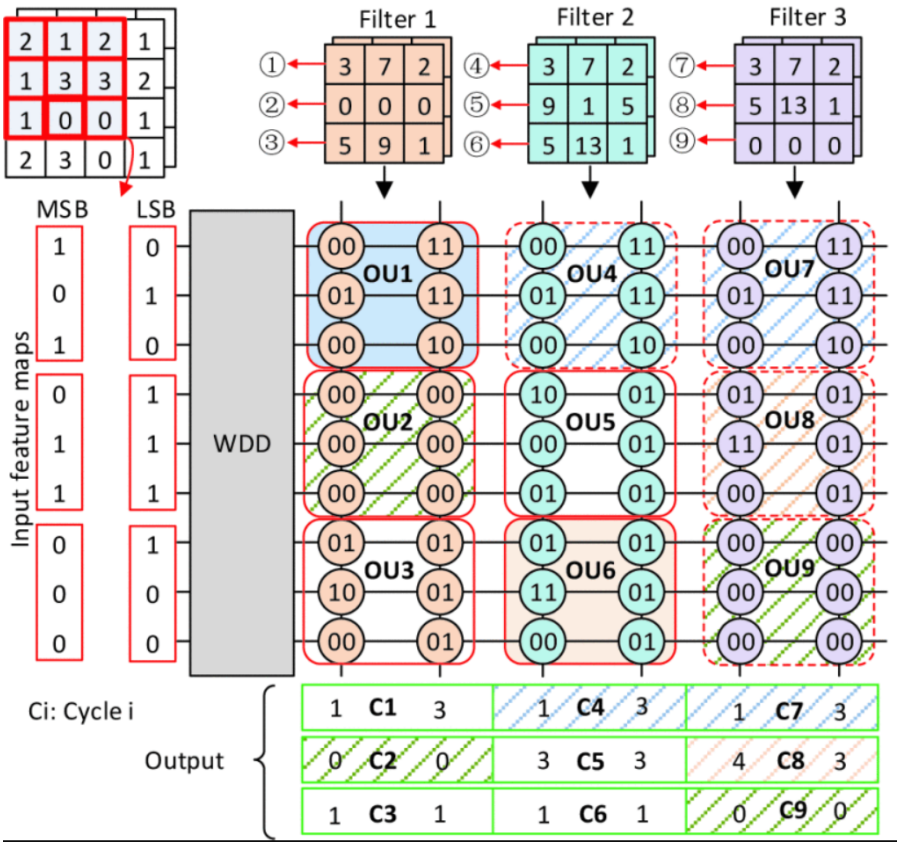

Figure 5.3 Eliminated the DACs and instead serialize the activation by applying only a single bit at each cycle [24].

can amount up to 23% and 61% of the system area and power respectively as shown in [22] or in extreme cases up to 99% and 85% as in [23]. To limit these drawbacks, several architectures [24], [25] eliminated the DACs and instead serialize the activation by applying only a single bit at each cycle as shown in Figure 5.4. This approach also reduce the ADCs size as the accumulated analog value is also smaller. However, this approach requires more cycles to perform single operation (usually number of cycles equivalent to the activation precision.),

Another approach to limit such drawback was to adapt a bit decomposition approach by either decomposing the activation as the previously

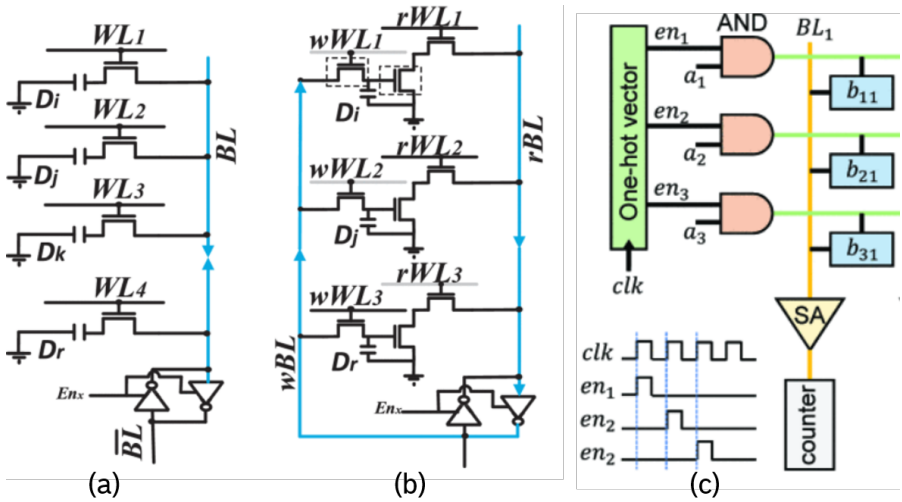

Figure 5.4 (a) Several row activation approach such as Ambit's TRA [28]. (b) Changing subarray unit cell structure whether with extra transistors or operation mode as in DRISA 3T1C [29]. (c) Activating only one row at a time and use the row activation as an operand as in FlexPim [28].

mentioned approach or by decomposing the weight stored as well [26], [27]. In this approach, a compromise between the number of cycles needed and the size of ADCs and DACs is investigated to balance the throughput, area and power of the architecture.

5.2.3.2 Digital approach

Another way for in-memory computation is adapting a completely digital approach. Such structure depends on either decomposing both the weights and activations completely or quantizing the parameters to binarized representation. This in return converts the MAC operation into bulk logical bitwise operations that need to be followed by additions, shiftings and comparisons. The memory cells are usually used to perform the bitwise operations and the rest of operations are done by supporting computational blocks. Several architectures realize the bulk operations as shown in Figure 5.4 through parallel sub-array activations representing the operands [28] or through single row activation based on one of the operands [29], [30]. Another realization is possible through modifying the cell to perform the target logical operation as in [31], [32].

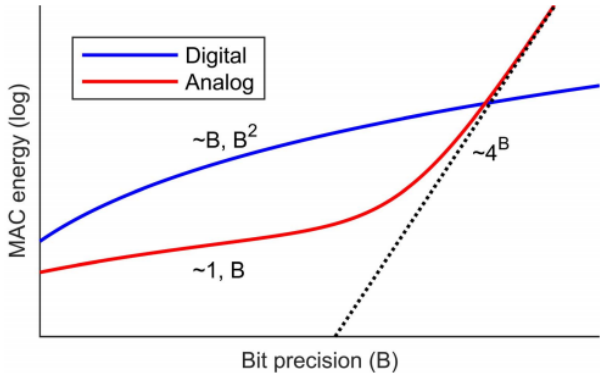

Figure 5.5 The relation between the energy cost for digital and analog MAC operations versus bit precision. [33].

Compared to the mixed signal approach, this approach allows for high speed due to the eliminations of the analog blocks as well as high power efficiency. However, the decomposition of the MAC operations reflects on the operation latency. Also, low bit quantization limits the architecture usage to neural networks models that can tolerate such quantization.

5.2.4 Target Network Quantization

In this section, we investigate the possible targeted neural network weights' quantization and representation. Ranging from floating point representation to binary representation, wide range of presented architectures has been offered with each targeting a specific representation or in some cases try to be flexible and target several possible weights' quantization. As highlighted in earlier sections and shown in Figure 5.5, such network properties affect directly the architecture choices such as the computational domain, selected technology, etc but also it is reflected on the network accuracy and possibility of using the architecture for training as well as inference.

5.2.4.1 Floating point architectures

Floating point representation is considered as the most accurate form of the targeted network since it is usually the representation used during the training and design phase. Architectures targeting such representation are usually used for training mainly [34], [35]. The main advantage such architectures are offering is the elimination of accuracy loss from network representation.

Also, these architectures targets the largest range of networks as the only limitation in that case is the support of network layers type or not.

However, these architectures are usually suffer from a trade off between high power consumption or lower throughput. Depending on the design choices, generally expensive power blocks are used which maintain high throughput on the expense of high power consumption. This makes them way efficient when compared to general purposed computing devices such as graphic processing units (GPUs) but losing the edge compared to low power ASIC designs.

5.2.4.2 Fixed-point architectures

Architectures targeting fixed-point weight representation are the most popular among in-memory architectures. Due to advanced neural network optimization techniques [36], [22], weights can be represented using fixed-point precision as low as 4-bit in large complex networks. With such low representation, these architectures store the weight in the memory cell which boost the system throughput and performance as shown previously. However, such architectures suffer from several drawbacks related to accuracy losses that can occur due to sever compression. Also, such representation limit the usage of these architectures in training related tasks and confine them more to inference based tasks.

5.2.4.3 Binarized architectures

Motivated by the extremely reduced memory/computational requirements with marginal degradation in accuracy for some networks [37], [38], several architectures are built to target binary/ternary operations. In these architectures [39], [5], the main operation performed by the memory cell is usually very simple logical operation. Also, these architectures reduce the memory cell irregularities to the minimum as each cell stores a single bit.

However, the binarization limits the architecture usage to a limited number of networks that can currently adopt such representation. The main argument these architecture is dependant on is the consistent advance in the binarization techniques that can allow for more networks to be using such architecture.

5.2.4.4 Flexible precision architectures

A recent popular growing trend is to target various representations simultaneously where these representations can range from binarized to full precision floating point as in [26], [30]. In these architectures, the weights and inputs

precision can be traded for either higher throughput or reduced power consumption.

Most of these architectures depend on weight bit decomposition or input serialization explained earlier. Such flexibility allows for the use of these architectures for a wider range of networks and for both inference and training purposes. However, such flexibility comes with a cost compared to fixed representation architectures. For example in [29], to achieve such flexibility, a hierarchical network on chip is required which adds extra hardware either to the chip busses or the complexity of the chip control.

References

[1] Y. Liang, K. Gopalakrishnan, P. B. Griffin, and J. D. Plummer, "From dram to sram with a novel sige-based negative differential resistance (ndr) device," in *IEEE InternationalElectron Devices Meeting, 2005. IEDM Technical Digest.*, 2005, pp. 959–962.

[2] N. S. Kim, K. Flautner, D. Blaauw, and T. Mudge, "Circuit and microarchitectural techniques for reducing cache leakage power," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, no. 2, pp. 167–184, 2004.

[3] B. H. Calhoun and A. P. Chandrakasan, "A 256-kb 65-nm sub-threshold sram design for ultra-low-voltage operation," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 3, pp. 680–688, 2007.

[4] A. I. Soumitra Pal, Subhankar Bose, "Design of memristor based low power and highly reliable reram cell," in *Springer*, 2019. [Online]. Available: https://link.springer.com/article/10.1007/s00542-019-04582-1

[5] K. Ando, K. Ueyoshi, K. Orimo, H. Yonekawa, S. Sato, H. Nakahara, M. Ikebe, T. Asai, S. Takamaeda-Yamazaki, T. Kuroda, and M. Motomura, "Brein memory: A 13-layer 4.2 k neuron/0.8 m synapse binary/ternary reconfigurable in-memory deep neural network accelerator in 65 nm cmos," in *2017 Symposium on VLSI Circuits*, 2017, pp. C24–C25.

[6] B. Moons, R. Uytterhoeven, W. Dehaene, and M. Verhelst, "Envision: A 0.26-to-10tops/w subword-parallel dynamic-voltage-accuracy-frequency-scalable convolutional neural network processor in 28nm fdsoi," in *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, 2017, pp. 246–247.

[7] Y. Chen, T. Luo, S. Liu, S. Zhang, L. He, J. Wang, L. Li, T. Chen, Z. Xu, N. Sun, and O. Temam, "Dadiannao: A machine-learning supercomputer," in *2014 47th Annual IEEE/ACM International Symposium on Microarchitecture*, 2014, pp. 609–622.

[8] J. Li, G. Yan, W. Lu, S. Jiang, S. Gong, J. Wu, and X. Li, "Smartshuttle: Optimizing off-chip memory accesses for deep learning accelerators," in *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2018, pp. 343–348.

[9] R. V. Wicaksana Putra, M. Abdullah Hanif, and M. Shafique, "Drmap: A generic dram data mapping policy for energy-efficient processing of convolutional neural networks," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[10] J. Meena, S. Sze, U. Chand, and T.-Y. Tseng, "Overview of emerging non-volatile memory technologies," *Nanoscale Research Letters*, vol. 9, pp. 1–33, 09 2014.

[11] M. Hasan and B. Ray, "Tolerance of deep neural network against the bit error rate of nand flash memory," in *2019 IEEE International Reliability Physics Symposium (IRPS)*, 2019, pp. 1–4.

[12] Y. Cai, S. Ghose, E. F. Haratsch, Y. Luo, and O. Mutlu, "Error characterization, mitigation, and recovery in flash-memory-based solid-state drives," *Proceedings of the IEEE*, vol. 105, no. 9, pp. 1666–1704, 2017.

[13] N. Mielke, T. Marquart, Ning Wu, J. Kessenich, H. Belgal, E. Schares, F. Trivedi, E. Goodness, and L. R. Nevill, "Bit error rate in nand flash memories," in *2008 IEEE International Reliability Physics Symposium*, 2008, pp. 9–19.

[14] C.-W. Hu, K.-M. Chang, C.-H. Tu, C.-N. Chiang, C.-C. Lin, S. Sze, and T.-Y. Tseng, "Nisige nanocrystals for nonvolatile memory devices," *Applied Physics Letters*, vol. 94, pp. 062 102–062 102, 02 2009.

[15] D. Ielmini, "Reliability issues and modeling of flash and post-flash memory," *Microelectronic Engineering - MICROELECTRON ENG*, vol. 86, pp. 1870–1875, 07 2009.

[16] Y. Long, D. Kim, E. Lee, P. Saha, B. A. Mudassar, X. She, A. I. Khan, and S. Mukhopadhyay, "A ferroelectric fet-based processing-in-memory architecture for dnn acceleration," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 5, no. 2, pp. 113–122, 2019.

[17] D. B. U. S. T. S. Böscke1, J. Müller and U. Böttger, "Ferroelectricity in hafnium oxide thin films," *Applied Physics Letters*, 08 2011.

[18] M. Trentzsch, S. Flachowsky, R. Richter, J. Paul, B. Reimer, D. Utess, S. Jansen, H. Mulaosmanovic, S. Müller, S. Slesazeck, J. Ocker, M. Noack, J. Müller, P. Polakowski, J. Schreiter, S. Beyer, T. Mikolajick, and B. Rice, "A 28nm hkmg super low power embedded nvm technology based on ferroelectric fets," in *2016 IEEE International Electron Devices Meeting (IEDM)*, 2016, pp. 11.5.1–11.5.4.

[19] S. Dünkel, M. Trentzsch, R. Richter, P. Moll, C. Fuchs, O. Gehring, M. Majer, S. Wittek, B. Müller, T. Melde, H. Mulaosmanovic, S. Slesazeck, S. Müller, J. Ocker, M. Noack, D. . Löhr, P. Polakowski, J. Müller, T. Mikolajick, J. Höntschel, B. Rice, J. Pellerin, and S. Beyer, "A fefet based super-low-power ultra-fast embedded nvm technology for 22nm fdsoi and beyond," in *2017 IEEE International Electron Devices Meeting (IEDM)*, 2017, pp. 19.7.1–19.7.4.

[20] A. Biswas and A. P. Chandrakasan, "Conv-ram: An energy-efficient sram with embedded convolution computation for low-power cnn-based machine learning applications," in *2018 IEEE International Solid - State Circuits Conference - (ISSCC)*, 2018, pp. 488–490.

[21] Q. Dong, M. E. Sinangil, B. Erbagci, D. Sun, W. Khwa, H. Liao, Y. Wang, and J. Chang, "15.3 a 351tops/w and 372.4gops compute-in-memory sram macro in 7nm finfet cmos for machine-learning applications," in *2020 IEEE International Solid- State Circuits Conference - (ISSCC)*, 2020, pp. 242–244.

[22] 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 *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, 2016, pp. 14–26.

[23] L. Ni, Y. Wang, H. Yu, W. Yang, C. Weng, and J. Zhao, "An energy-efficient matrix multiplication accelerator by distributed in-memory computing on binary rram crossbar," in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2016, pp. 280–285.

[24] Y. Zhang, Z. Jia, Y. Pan, H. Du, Z. Shen, M. Zhao, and Z. Shao, "Pattpim: A practical reram-based dnn accelerator by reusing weight pattern repetitions," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[25] G. Murali, X. Sun, S. Yu, and S. K. Lim, "Heterogeneous mixed-signal monolithic 3-d in-memory computing using resistive ram," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 2, pp. 386–396, 2021.

[26] T. Soliman, R. Olivio, T. Kirchner, M. Lederer, T. Ka'mpfe, A. Guntoro, and N. Wehn, "A ferroelectric fet based in-memory architecture for multi-precision neural networks," in *2020 33rd IEEE International System-on-Chip Conference (SOCC)*, 2020.

[27] Y. Cai, T. Tang, L. Xia, B. Li, Y. Wang, and H. Yang, "Low bit-width convolutional neural network on rram," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 7, pp. 1414–1427, 2020.

[28] V. Seshadri, D. Lee, T. Mullins, H. Hassan, A. Boroumand, J. Kim, M. A. Kozuch, O. Mutlu, P. B. Gibbons, and T. C. Mowry, "Ambit: In-memory accelerator for bulk bitwise operations using commodity dram technology," in *2017 50th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2017, pp. 273–287.

[29] Y. Long, E. Lee, D. Kim, and S. Mukhopadhyay, "Flex-pim: A ferroelectric fet based vector matrix multiplication engine with dynamical bitwidth and floating point precision," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–8.

[30] Y. Long, D. Kim, E. Lee, P. Saha, B. A. Mudassar, X. She, A. I. Khan, and S. Mukhopadhyay, "A ferroelectric fet-based processing-in-memory architecture for dnn acceleration," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 5, no. 2, pp. 113–122, 2019.

[31] S. Liu, H. Zhu, C. Chen, L. Zhang, and C. . Richard Shi, "Xnoram: An efficient computing-in-memory architecture for binary convolutional neural networks with flexible dataflow mapping," in *2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 2020, pp. 21–25.

[32] S. Li, D. Niu, K. T. Malladi, H. Zheng, B. Brennan, and Y. Xie, "Drisa: A dram-based reconfigurable in-situ accelerator," in *2017 50th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2017, pp. 288–301.

[33] B. Murmann, "Mixed-signal computing for deep neural network inference," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 1, pp. 3–13, 2021.

[34] S. Gupta, M. Imani, H. Kaur, and T. S. Rosing, "Nnpim: A processing in-memory architecture for neural network acceleration," *IEEE Transactions on Computers*, vol. 68, no. 9, pp. 1325–1337, 2019.

[35] S. R. Nandakumar, I. Boybat, V. Joshi, C. Piveteau, M. Le Gallo, B. Rajendran, A. Sebastian, and E. Eleftheriou, "Phase-change memory

models for deep learning training and inference," in *2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, 2019, pp. 727–730.

[36] 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 *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, 2016, pp. 27–39.

[37] M. Courbariaux, Y. Bengio, and J. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," *CoRR*, vol. abs/1511.00363, 2015. [Online]. Available: http://arxiv.org/abs/1511.00363

[38] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," *CoRR*, vol. abs/1603.05279, 2016. [Online]. Available: http://arxiv.org/abs/1603.05279

[39] 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 Journal of Solid-State Circuits*, vol. 54, no. 6, pp. 1789–1799, 2019.