

An Embedded NAND Flash Based Compute-In-Memory Array Demonstrated in a Standard Logic Process

Minsu Kim, Muqing Liu, Luke Everson, and Chris H. Kim, *Fellow, IEEE*

Abstract— A neural network hardware inspired by the 3D NAND flash array structure was experimentally demonstrated in a standard 65nm CMOS process. Logic-compatible embedded flash memory cells were used for storing multi-level synaptic weights while a bit-serial architecture enables 8 bit x 8 bit multiply-and-accumulate operation. A novel back-pattern tolerant program-verify scheme reduces the cell current variation to less than 0.6 μ A. Positive and negative weights are stored in adjacent bitlines, generating a differential output signal. Our eNAND based neural network core achieves a 98.5% handwritten digit recognition accuracy which is within 0.5% of the software accuracy for the same weight precision. To our knowledge, this work represents the first physical demonstration of an embedded NAND Flash based compute-in-memory chip in a standard logic process.

Index Terms— 3D NAND, embedded flash, deep neural network, multiply-and-accumulate, compute-in-memory

I. INTRODUCTION

DEEP neural networks (DNNs) contain multiple computation layers each performing a massive number of multiply-and-accumulate (MAC) operations between the input data and trained weights. The number of layers and parameters vary significantly depending on the specific DNN architecture, with state-of-the-art designs exceeding 100 million parameters as shown Fig. 1 [1]. The performance and energy-efficiency of data-intensive DNN chips can be limited by the available memory bandwidth and the MAC engine throughput. An alternative approach that is gaining popularity is the compute-in-memory (CIM) approach where the computation occurs where the data is stored, with massively parallelized analog MAC engines [2-16]. For the analog MAC engines, the input data is typically loaded on to multiple memory wordlines generating parallel cell currents that are summed up in a single cycle and converted to a digital code. Ideally, memory cells used for CIM architectures should be non-volatile, to avoid the costly reloading of the weights after a power-down period. It is also highly desirable if the memory cell can support multi-level

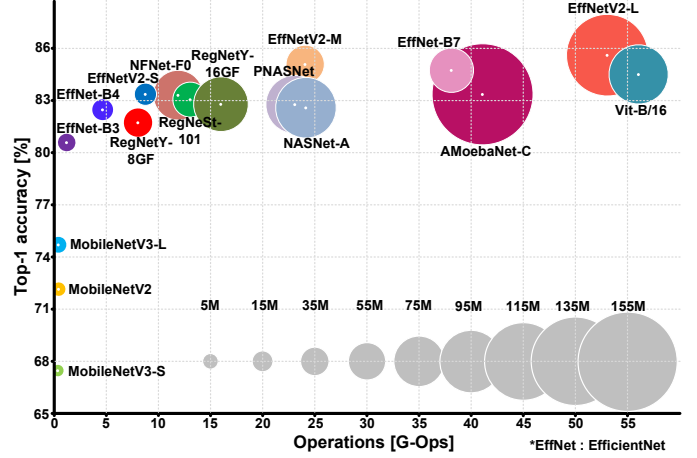


Fig. 1. Accuracy versus the number of operations required for a single forward pass of popular deep neural network architectures ([1], more recent DNNs added). The size of circle represents the number of network parameters which can exceed 100M in recent architectures.

storage as this can enhance the accuracy of inference tasks. One embodiment of this approach is a resistive RAM (ReRAM) crossbar array where weights are stored in the form of the transconductance of ReRAM cells while the MAC operation is performed in the analog domain [13-14]. Despite their tremendous potential, past CIM demonstrations based on emerging memory devices such as ReRAM and phase change RAM are mostly limited to small-scale networks with low precision operands (e.g. 1-2 bits) due to fabrication difficulties [2-16]. Most array level studies on emerging memory based CIM designs use compact models extracted from individual device measurements which cannot capture the true circuit and array level details. SRAM based CIMs have become a popular research topic as they can be fabricated reliably in a standard logic process [17-20]. However, SRAM cells are large compared to denser 1T or 1T1R memory technologies and hence cannot store the massive number of parameters required in today's DNN architectures. Furthermore, each bit can store only one bit of resolution and the cell current is prone to process

C. H. Kim is with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: chriskim@umn.edu).

M. Kim was with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 USA. He is now with SK Hynix, Icheon, South Korea.

M. Liu was with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 USA. She is now with Western Digital, Milpitas, CA.

L. Everson was with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 USA. He is now with Broadcom Inc., Bloomington, MN

variation, which is a critical disadvantage for analog computation.

To make CIM a practical reality for future DNNs with hundreds of millions of parameters, it is imperative that the in-memory neural network array be built in a non-volatile memory technology that can attain ultra high density, low cost, and highly manufacturability. 3D NAND Flash technology [21-24] is the leading candidate that meets all these requirements, however, no experimental data has been reported on 3D NAND flash-based CIM designs due to the proprietary nature of the technology and the difficulty in implementing the MAC function in standard flash chips. The feasibility of using a 3D NAND flash array for performing MAC functions was studied in [25-29], but the analyses were based on the cell current measured from individual cells rather than from a real MAC hardware.

In this paper, we investigate the array-level operation of a 3D NAND flash based Convolutional Neural Network (CNN) accelerator through a physical chip implementation. To get around the technology access problem, a demonstrator chip that mimics a 3D NAND flash array was fabricated in a 65nm foundry logic process. The proposed hardware features multi-level non-volatile weight storage using a 3T flash cell, single cycle current integration, bit-serial MAC operation, and multi-bit output sensing. One of the highlights of this work is the back-pattern tolerant program-verify sequence which reduces the cell current variation to less than $0.6\mu\text{A}$, allowing 28 individual cell currents to be summed up in a single cycle while delivering an MNIST classification accuracy of 98.5%.

Context and scope of this work: It should be noted that achieving an ultra-high density CIM array was not the main objective of this work as the logic-compatible 3T flash memory cell has a large footprint due to the thick oxide transistors required for the bit cell implementation. Instead, the goal was to use a logic compatible NVM cell that can be reliably manufactured in a standard logic technology to study practical aspects of a 3D NAND based CIM accelerator including, but not limited to, verifying MAC operation from a NAND array, developing an accurate and reliable erase and program scheme, testing charge loss behavior, designing a dedicated high voltage switch for controlling wordline signals, and demonstrating a multi-layer neural network architecture in the NAND array. It's worth noting that while the proposed CIM architecture was inspired by the 3D NAND flash array topology, significant discrepancies exist between the logic process used in work and a real 3D NAND flash process, that need to taken into account before the proposed concepts can be considered for real 3D NAND flash memory. These include the lower device mobility, higher bitline/wordline resistances, increased interference and disturbance issues, and the periphery circuit overhead of 3D NAND technology.

The remainder of this paper is organized as follows. Section II describes the design of a neuromorphic computing engine in a 3D NAND array. The 3T flash based NAND array mimicking a 3D NAND is described in section III. Implementation details

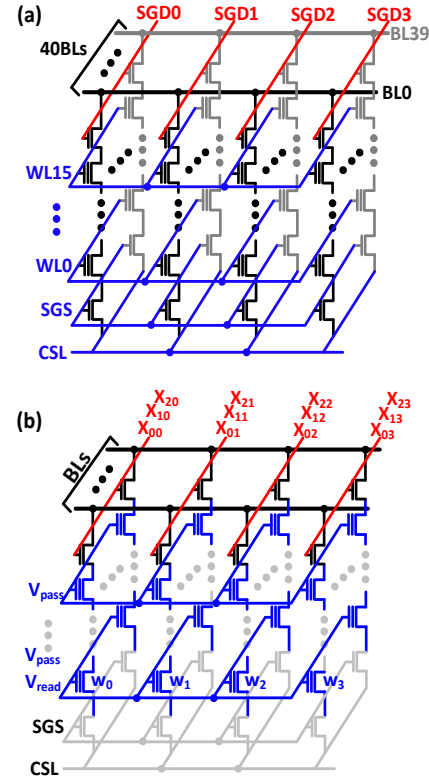


Fig. 2. (a) 3D NAND BiCS architecture [21] with 40 BLs x 4 SGDs x 16 WLs. (b) Analog MAC ($\sum X_i \times W_i$) implementation in a 3D NAND array.

of the proposed NAND flash based neural network core are given in section IV. Section V provides details of the HVS wordline driver and its measured results. Experimental results are discussed in section VI ranging from program and erase characteristics to inference results from a multi-layer neural network. Finally, conclusions are drawn in section VII. The conference version of this work was published in [30].

II. COMPUTE-IN-MEMORY INSPIRED BY 3D NAND FLASH ARRAY

A. Analog MAC Concept in 3D NAND Array

Fig. 2 shows the 3D NAND BiCS architecture [21] consisting of $8 \times 16\text{K}$ bitlines, 4 select lines, and 96 wordlines, along with the implementation of the analog MAC function. In the first cycle, the least significant bits of the input data $X_0=X_{04}X_{03}X_{02}X_{01}$ (4 bit example) is loaded on to the four individual SGD lines while the multi-bit weights are stored in the memory cell. Note that memory cells on the same layer of a 3D NAND array share the same wordline “plane” so the individual bits of the input data must be loaded onto the SGD lines rather than onto the shared wordline planes. Similar to a regular NAND flash read operation, a single wordline plane is enabled by applying a read voltage while the unselected wordline planes are biased at a pass voltage. To ensure good linearity between the input data and the cell current, we use binary SGD voltage levels (i.e. VDD or GND). For the weight storage, multi-bit weights (typically 2-3 bits) can be stored in each eflash cell in the same way as MLC (multi-level cell), TLC (ternary-level cell), or QLC (quad-level cell). The number of

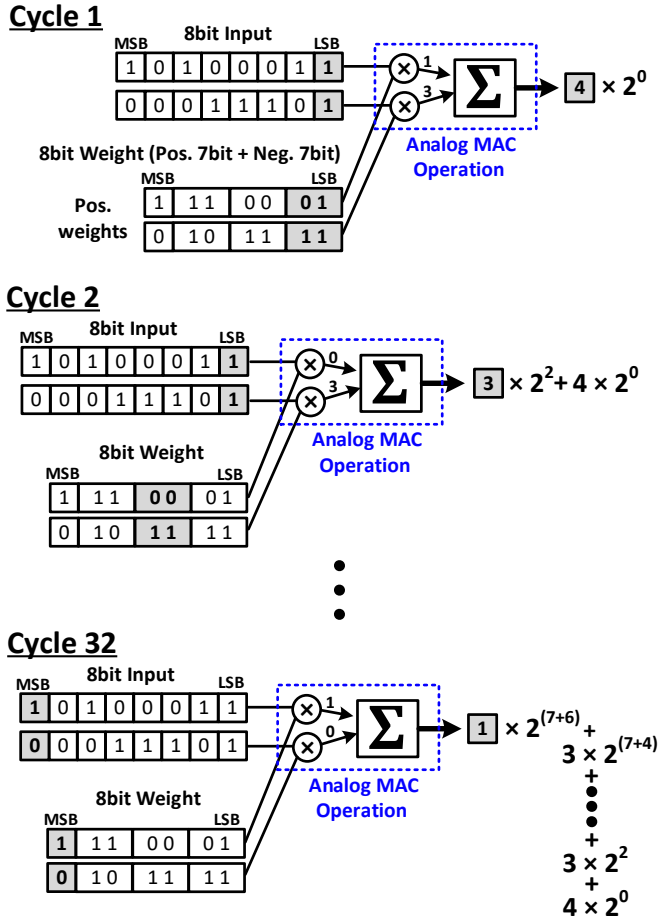


Fig. 3. Bit-serial operation realized in our test chip with 2 bits per cell weight storage and 8 bit weight and input parameters. The MAC operation is performed in the analog domain while the bit position shifting and summation was done in the digital domain.

bits stored in each cell is determined based on several factors including the maximum cell current, program disturbance, maximum sensing current, temperature effects, endurance limit, and retention characteristics. Each bitline is connected to 4 NAND strings within a single memory block. The bitlines are also connected to NAND strings in other blocks which allows a higher number of cell currents to be accumulated. A higher resolution (e.g. 8 bit) weight can be stored using multiple memory cells in the same stack, and a multi-bit MAC can be realized through bit-wise MAC operations where partial results are summed up while accounting for the bit position difference (details in section II-B). This step involves asserting the higher significant bits of the input data (e.g. $X_1 = X_{14}X_{13}X_{12}X_{11}$) to the NAND array as shown in Fig. 2.

B. Bit-Serial Operation for 8bit $\times$ 8bit MAC in 3D NAND

In this work, we incorporated the bit-serial technique for realizing an 8 bit input $\times$ 8 bit weight MAC in the proposed NAND array where each cell stores 2 bits of weight information. Fig. 3 shows the bit-serial operation for two 8 bit inputs and two 8 bit weights. Each flash cell can store 2 bits of information, necessitating 4 cells to store a single 8 bit weight. The example shown here is for positive weights but negative weights can be handled in a similar manner [32]. Our test chip

can incorporate both positive and negative weights using two separate bitlines. The bit-serial operation is briefly explained next. In cycle 1, LSBs of the two inputs and 2 LSBs of the weights are multiplied and accumulated as shown in the figure to produce the partial result $1_2 \cdot 01_2 + 1_2 \cdot 11_2 = 1 + 3 = 4$. In cycle 2, the same LSB bits of the two inputs are convolved with the next 2 bits of the two weights to obtain partial result $1_2 \cdot 00_2 + 1_2 \cdot 11_2 = 0 + 3 = 3$. When combining the results from cycle 1 and cycle 2, the bit position difference is accounted for by shifting the result from cycle 2 by 2 bits before adding it to the result from cycle 1. This produces an output of $3 \cdot 2^2 + 4 = 16$. This process is repeated until all the bits of the inputs and weights are accounted for. Finally, in cycle 32, the last MSB bits of the inputs and the last MSB bits of the weights are processed. The partial result of cycle 32 is multiplied by 2^{13} before the accumulation. For the 5x5 convolutional neural network implemented in our hardware, the actual number of inputs is 25.

III. EMBEDDED NAND FOR EVALUATING 3D NAND BASED COMPUTE-IN-MEMORY ARRAY

To experimentally verify the 3D NAND flash based CIM concept, ideally, we would have to build a hardware prototype in a real 3D NAND flash process. However, it is impossible for academic researchers to access a real 3D NAND technology due to its highly confidential nature. It is also impossible to test the idea in standard flash chips since chip vendors do not grant users control of the individual SGD and wordline signals. Due to these constraints, there has been no physical demonstration of a NAND flash based neuromorphic array to our knowledge. Only simulation and model based studies exist.

To demonstrate a real hardware prototype in spite of these constraints, we chose to use a 3T embedded flash memory cell [15, 30-31] that can be built in any foundry process to emulate the behavior of a 3D NAND neuromorphic array. Fig. 4 shows a comparison between a standard 1T flash cell and a 3T eflash cell vis-a-vis the erase, program, and read bias conditions. The 1T flash cell contains a floating gate layer surrounded by a thick dielectric layer for non-volatile charge storage. The 3T eflash cell on the other hand, consists of back-to-back connected PMOS transistors M1 and M2, and a separate read device M3. The floating gate node is formed by the gate poly shared by M1, M2, and M3 devices. The program and erase operations of both cell types are based on the Fowler Nordheim (FN) tunneling mechanism so their program/erase times and retention behaviors are equivalent. The standard 1T flash cell requires a high positive or negative voltage between the control gate and the substrate to inject or remove electrons from the floating gate. As for the 3T cell, the program or erase voltages are applied between the junction terminals of M1 and M2 (i.e. PWL and WWL signals). Fig. 5 shows 3D NAND strings with N stacked memory cells based on the 1T flash cell and the 3T embedded flash cell, respectively. Each wordline of the 3T NAND array is controlled by wordlines PWL and WWL. We can see that the current paths of the two NAND strings are basically the same, even though the program and erase of the

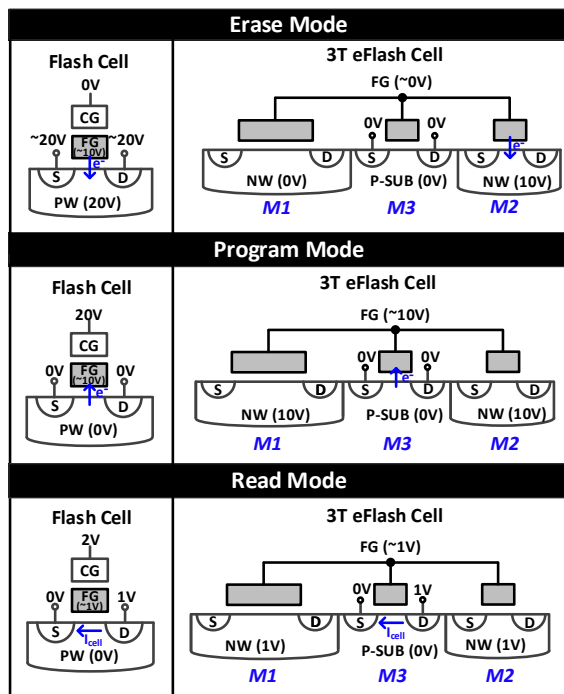


Fig. 4. Comparison between 1T flash cell (left) and 3T eflash cell (right). FG means floating gate and CG means control gate. For erase and program modes, electrons are either removed or injected to the GF node via FN tunneling mechanism [15, 25-26].

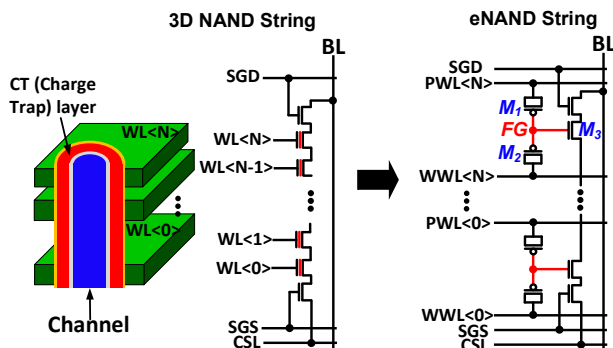


Fig. 5. 3D NAND string [18] (left and middle) and corresponding eNAND string based on 3T eflash cell (right). Each row of the eNAND array is controlled by PWL and WWL signals.

3T cell utilizes the two PMOS devices per cell. The equivalent between the two cell types allows us to experimentally study the MAC operation of a standard NAND flash array using a 3T flash based implementation.

IV. NEURAL NETWORK CORE CIRCUIT DESIGN

A. Overall eNAND Neural Network Core Architecture

Fig. 6 shows how the 3D NAND block in Fig. 2 composed of 40 BLs, 4 SGD, and 16 WLs, is flattened for implementation in a standard logic process. Unlike NOR-type designs where each memory cell requires a selection device, transistors in a NAND string share the source/drain nodes, and hence the selection devices are only required at the top and bottom of the NAND string. The logic-compatible 3T eFlash cell shown in Fig. 7 (left) consists of two asymmetrically sized PMOS

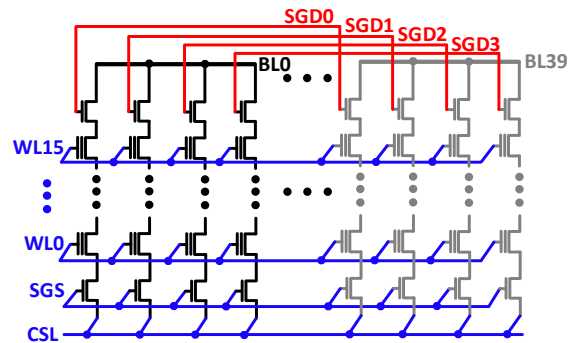


Fig. 6. Flattened 3D NAND architecture for demonstration in a standard logic process. The original 3D NAND block structure is shown in Fig. 2.

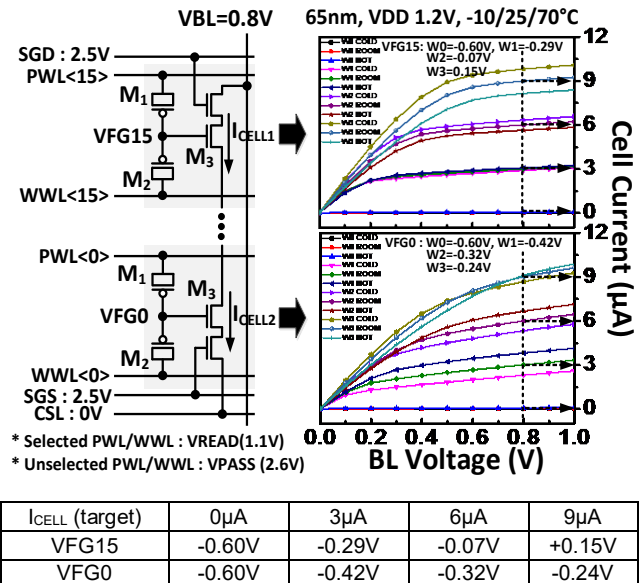


Fig. 7. Simulated I-V characteristics of top and bottom eFlash cells of a 16 stack NAND string at cold (-10°C), room (25°C), hot (70°C) temperatures. Series resistance varies depending on the location in the stack which can be compensated by incremental programming.

devices (M_1 and M_2) for efficient program and erase operation, and a separate NMOS read device (M_3) connected to the NAND string. Thick oxide IO devices were used to ensure practical retention times. The width/length of the 3 cell devices are $3.2\mu\text{m} / 280\text{nm}$ (M_1), $0.4\mu\text{m} / 280\text{nm}$ (M_2), and $0.4\mu\text{m} / 280\text{nm}$ (M_3), respectively. The M_1 width is 8 times larger than those of M_2 and M_3 to obtain a high coupling ratio which reduces the program and erase voltages. Due to the series resistance of the unselected devices, the I-V characteristics of the eNAND Flash cell may vary depending on its location in the stack. A flash memory cell can be programmed incrementally to mitigate systematic variation effects such as the aforementioned stack location dependence. The unique post-silicon tuning capability offers significant advantage over SRAM, DRAM, MRAM, or RRAM based neuromorphic approaches as the cell read current can be programmed to within $0.6\mu\text{A}$ of the target (Figs. 21 and 24). Further details on the incremental program method are given in section IV.

Fig. 7 (right) shows the I-V curves of the first and last 3T

flash cell in the 16 stack NAND string at -10, 25, and 70°C. We carefully adjusted the floating gate (FG) voltage to achieve uniform current levels of 0μA, 3μA, 6μA, and 9μA. A conservative 3μA step size was chosen to account for variation and charge loss effects, but further studies including baking tests at 150°C showed that a 1μA step size is also acceptable [33]. The initial FG voltages used in the simulations are listed in Fig. 7 (bottom) which represent the different programming levels of the flash cell. The bitline voltage was kept constant at 0.8V during program verify and read modes, which ensures that the cell current is accurately programmed. Simulation results in Fig. 7 (right) show that location-dependent variation can be canceled out by fine-tuning the FG voltage through incremental programming. The impact of temperature on the I-V curves was slightly different depending on the program level and the location of the cell in the stack, but was manageable as confirmed in our later measurements [33].

The overall chip architecture is shown in Fig. 8 (left) which comprises two high voltage switch (HVS) circuits per row for driving the PWL and WWL wordlines, a readout circuit, an eNAND cell array with 16 cells per stack, and scan chains. Each eNAND block of the chip architecture requires 32 HVSs for driving 16 PWL's and 16 WWL's. The HVS has to generate a high output voltage up to 10V during the program and erase operations. We included a repair block where three eNAND strings sharing the same PWL and WWL are reserved for redundancy and fixed bias weights. The input image data is simultaneously loaded on to 25 SGD lines, and the 3 additional SGD lines are used for enabling the bias. The circuit diagram and layout of the 16 stack eNAND string are shown in Fig. 8 (right). In this work, we adopted the bitline voltage regulator circuit from [15] to pin the bitline voltage to 0.8V while the bitline current is converted to an output voltage. An on-chip voltage-controlled oscillator (VCO) generates a frequency that corresponds to the output voltage. Other more compact and

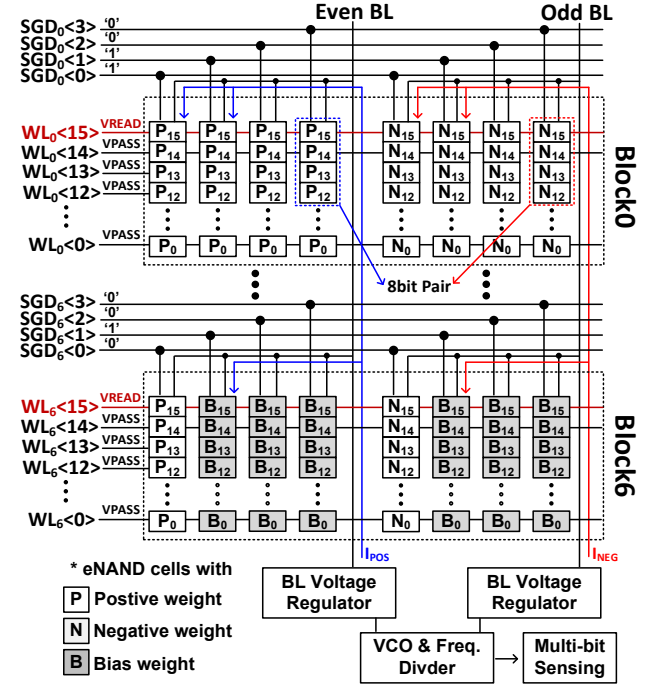


Fig. 9. Positive, negative and bias weight values are stored in two adjacent bitlines. 28 individual NAND string currents are summed up in a single cycle and converted to an output frequency for multi-bit sensing.

efficient analog-to-digital conversion methods [2-14] can be devised but given that this is the first ever attempt of building a NAND type neuromorphic array in a real chip, we intentionally kept the peripheral circuits simple so that we can focus on verifying the core functionality of the chip.

B. Storing positive and negative weights in eNAND Array

A pair of bitlines is used to store a single weight as illustrated in Fig. 9. If the weight is positive, the cell current of the left bitline is programmed accordingly while the cell current on the right bitline is programmed to <0.1μA, and vice versa for

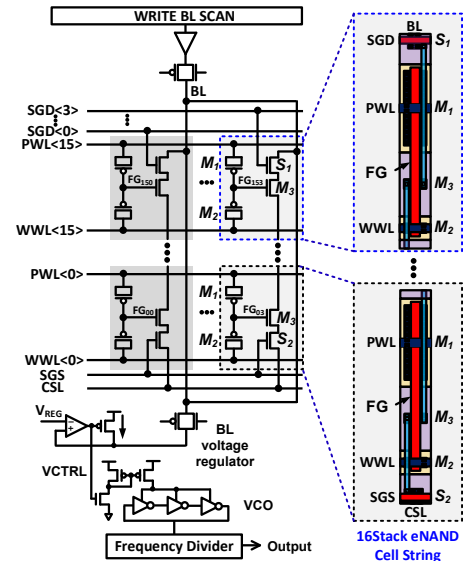
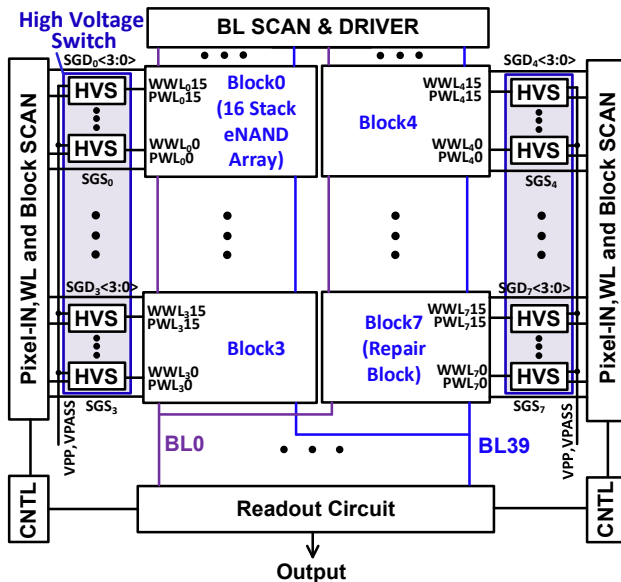


Fig. 8. (Left) Overall diagram of the CNN core with high voltage wordline drivers, a 16 stack eNAND Flash array, and readout circuit. (Right) Shared BL and readout circuit and layout of a 16 stack eNAND string.

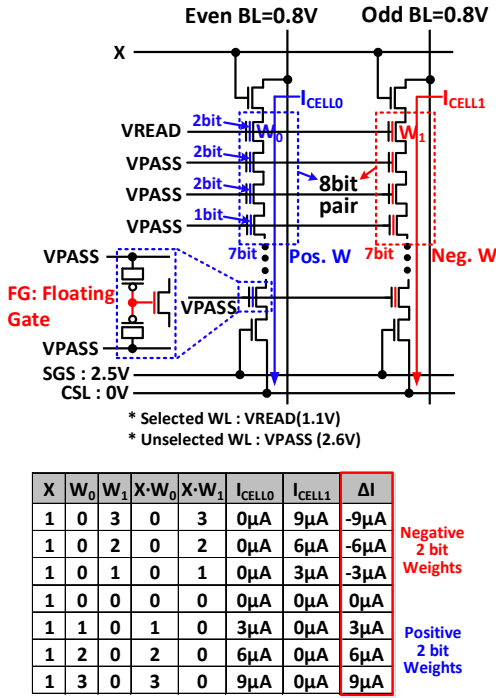


Fig. 10. Positive and negative weights for an 8 bit BL pair (upper). When input X is 1, I_{CELL0} and I_{CELL1} are decided by the positive 2 bit weight W_0 or negative 2 bit weight W_1 of the selected wordline, respectively (lower).

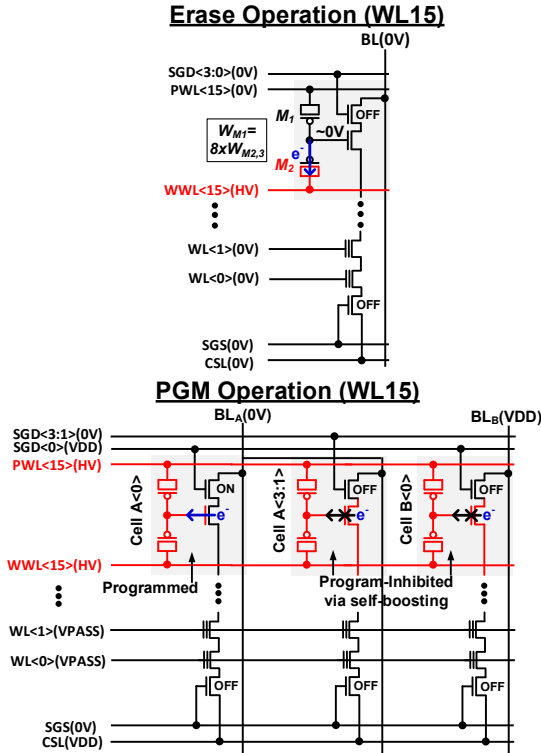


Fig. 11. Bias conditions of the 3T eNAND Flash cell for the erase and program modes. compensated by incremental programming.

negative weights. Input data is simultaneously loaded onto the SGD lines which activate multiple memory cell currents connected to the same bitline. During this operation, the selected and unselected wordlines are held to VREAD and

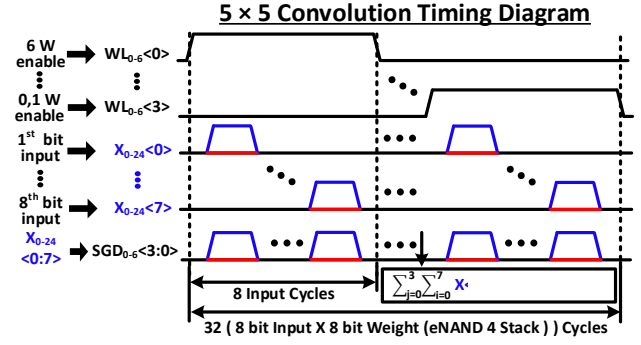


Fig. 12. Timing diagram of 5×5 convolution operation with 8 bit data and 8 bit weights. 4 cells are used to store a single 8 bit weight while each bit of the 8 bit data is fed to the correct SGD line. Bit serial operation produces a multi-bit inner product result.

VPASS, respectively. The sum of the individual cell currents flows through each bitline. The bitline pair generates two currents; i.e. positive weight and negative weight currents. Both currents are converted to the corresponding output voltages by the bitline regulator circuit (Fig. 9). Finally, the VCO circuit converts the output voltage to a frequency which is measured using off-chip equipment.

Fig. 10 shows an 8 bit weight composed of even and odd bitline pairs and the read operation of the first two cells in the NAND string. The 8 bit weight is represented as either a 7 bit positive weight or a 7 bit negative weight as shown in Fig. 10. A 7 bit positive or negative weight is stored across 4 stacked cells each storing 2 bit, 2 bit, 2 bit, and 1 bit, respectively. The read operation of the top 2 bit eNAND cell pair follows the bottom table of Fig. 10. As shown in the table, if the weight is negative, positive weight W_0 set to 0. If the weight is positive, it's reversed. If input X is 0, both I_{CELL0} and I_{CELL1} are 0μA, whereas if input X is 1, I_{CELL0} and I_{CELL1} are either 0μA or 3μA depending on the weight values. Based on this operation, we can generate the current difference $I_{CELL0} - I_{CELL1}$ that works correctly for both positive and negative weights.

C. Erase and Program Operation

To update the weights in the eNAND cells, we have to first erase and then program the cell data. Fig. 11 shows the bias condition for erasing and programming wordline 15 which is the upper most row of the NAND stack. The biasing scheme is the same for the other wordlines. During the erase operation, we applied 0V to the PWL and WWL signals of the unselected wordlines. As for the selected wordline 15, we applied a high voltage around 10V to the WWL 15 signal and 0V to the PWL 15 signal. Under these bias conditions, electrons move from the floating gate to the M2 PMOS channel through FN tunneling.

The program operation comprises two modes as shown in Fig. 11 (bottom). The first mode is the program mode while the second mode is the program inhibition mode. For the program mode, we applied a VPASS bias to the PWL and WWL signals of the unselected word lines. For the selected wordline 15, we applied a high voltage bias to the PWL 15 and WWL 15 signals, and 0V to the NMOS channel. Electrons move from the NMOS channel to the floating gate through FN tunneling in the selected

wordline. For program inhibition mode, the NMOS channel becomes floating and hence it is boosting by the floating gate. This limits the voltage across the tunneling oxide and hence the electrons do not escape the floating gate node. For testing, we used a program bias voltage ranging from 7.0V to 8.0V.

D. 5×5 CNN Operation in ENAND Array

The timing sequence of a 5×5 convolution operation with 8 bit inputs and 8 bit weights is shown in Fig. 12. Here, we applied the bit-serial operation described in section II where the operand is serialized and asserted one bit at a time to the array [32]. The bit-serial inner-product realizes the equation $\sum_{j=0}^3 \sum_{i=0}^7 X < i > \cdot 2^i \cdot W < j > \cdot 2^{(2-(3-i))}$.

V. HIGH VOLTAGE SWITCH CIRCUIT AND MEASUREMENTS

A. High Voltage Switch (HVS) WL Driver Circuit

Fig. 13 shows the HVS circuit for generating the proper wordline biases for erase, program, and read operations. The HVS circuit was inspired by our previous cascode driver circuit [30-31], but the following critical improvement was made in the new design. We added a level shifter circuit for generating the signal denoted as “F” in Fig 13, and introduced 2 additional PMOS transistors controlled by signals M and F to generate the VPASS bias during the program and read operations. The bias voltages required for each operating mode of the selected eNAND block are summarized next.

For the erase operation, we need the erase and default mode voltages shown in Fig.14 (left column). We apply a 10V erase voltage to the WWL signal of the selected wordline, and applied 0V to the PWL signal of the selected wordline as well as the WWL and PWL signals of the unselected wordlines for the default mode. To generate a stable 10V output while preventing voltage overstress, we set the VPP4, VPP3, VPP2, and VPP1 voltages to be 10V, 7.5V, 5.0V, and 2.5V, respectively, as shown in Fig. 13 and 14. Based on these biases and control signals, a 10V output pulse is generated by the HVS circuit.

For program operation, we need to realize the program and VPASS modes shown in Fig. 13 (middle column). The operation of the HVS circuit during program mode is almost the same as the erase mode. The main difference is the lower program voltage of 7 to 8V compared to the 10V erase voltage. WWL and PWL signals of the selected wordlines are charged to the program voltage while a lower VPASS bias is applied to the WWL and PWL signals of the unselected wordlines. A maximum VPASS of 3V was used to minimize program disturbance. When the voltage of signal M is kept at VPP1 (=0.25×VPP) and the voltages of F, G, and the Pulse signals are kept at 0V, the output signal is held at VPASS as shown in Fig. 13 (middle column).

For the read operation, we applied the read bias voltage (=VREAD) to WWL and PWL of the selected wordiness while

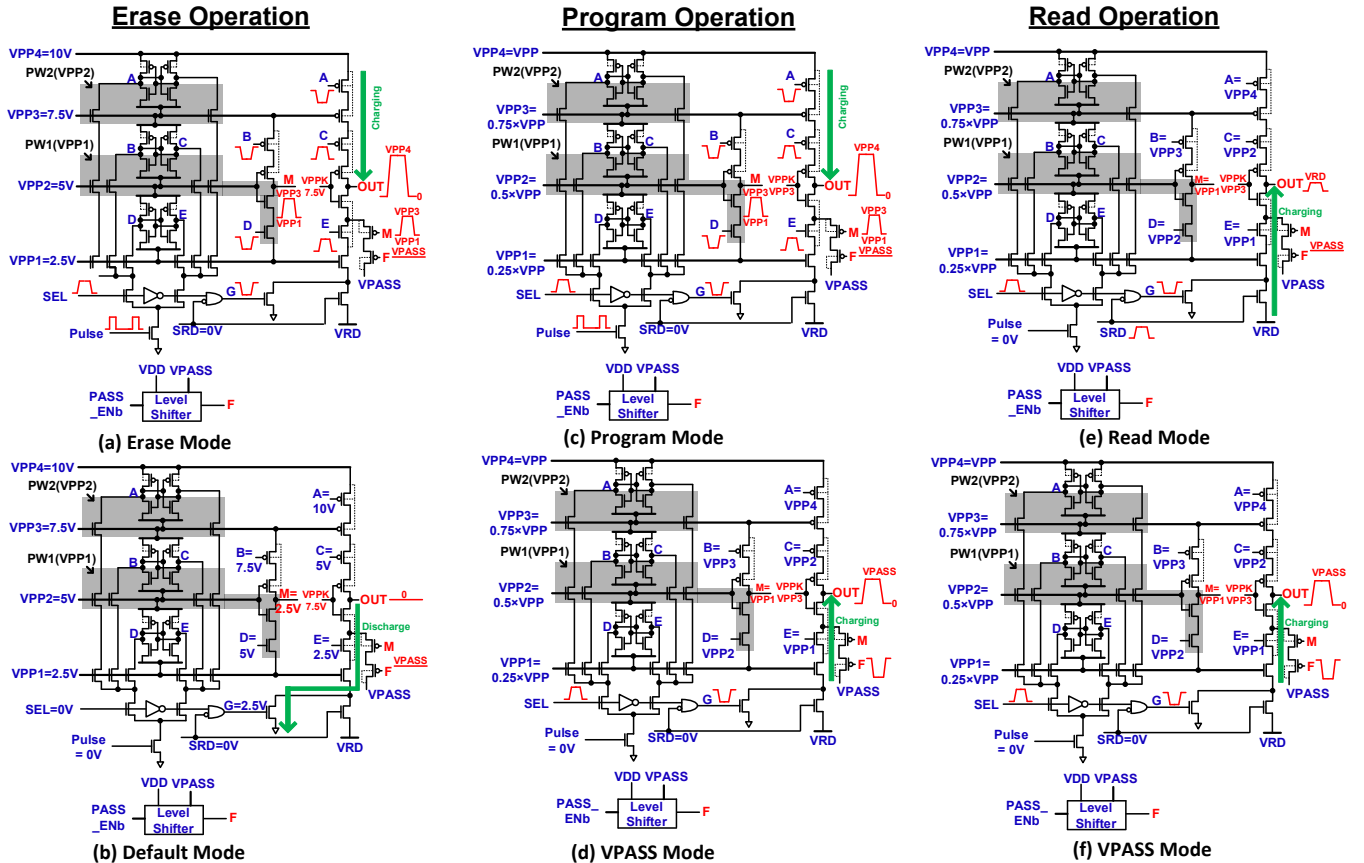


Fig. 13. Operation of the modified high voltage switch driver circuit to support (a) erase mode, (b) default mode and program operation (c) program mode, (d) VPASS mode and read operation (e) read mode, and (f) VPASS mode.

those of the unselected wordlines were biased at VPASS. The VPASS mode during read is the same as the VPASS mode during program operation. During read mode, the output signal of the HVS circuit is driven to VREAD by applying the following bias condition; VPASS for signal F, 2.5V IO VDD for signal SRD, and 0V for signal G as shown in Fig. 14 (right).

B. High Voltage Switch (HVS) WL Driver Measurements

Fig. 15 shows the measured results for PWL and WWL of unselected and selected wordlines in the eNAND array. First, for a 1ms erase period, we verified that the PWL and WWL signals of the selected and unselected wordlines are correct. The WWL of the selected wordline is raised to 10V while the PWL and WWL of unselected wordlines are kept at 0V during the erase operation. This is the desired result during erase and default modes of the HVS circuit. Next, we verified the

operation of the HVS circuit for program and VPASS modes. The PWL and WWL signals can support a VPP voltage from 7V to 8V, and a VPASS bias range from 2.6V to 4V. Fig. 15 (middle) shows the HVS output voltage for a 20 μ s program operation. PWL and SWL signals of the selected wordline are driven to VPP while PWL and SWL signals of the unselected wordlines are kept at a VPASS bias of 2.6 to 4V. Lastly, Fig. 15 (right) shows the PWL and WWL during read operation. We verified the voltage level of the selected wordline switches reliably to 1.1V while PWL and WWL signals of the unselected wordlines are kept at the VPASS bias ranging from 2.6 to 4V for the 2 μ s read duration. The measured waveforms follow the expected results of the program, read, and VPASS modes. The HVS circuit operates correctly without consuming any static

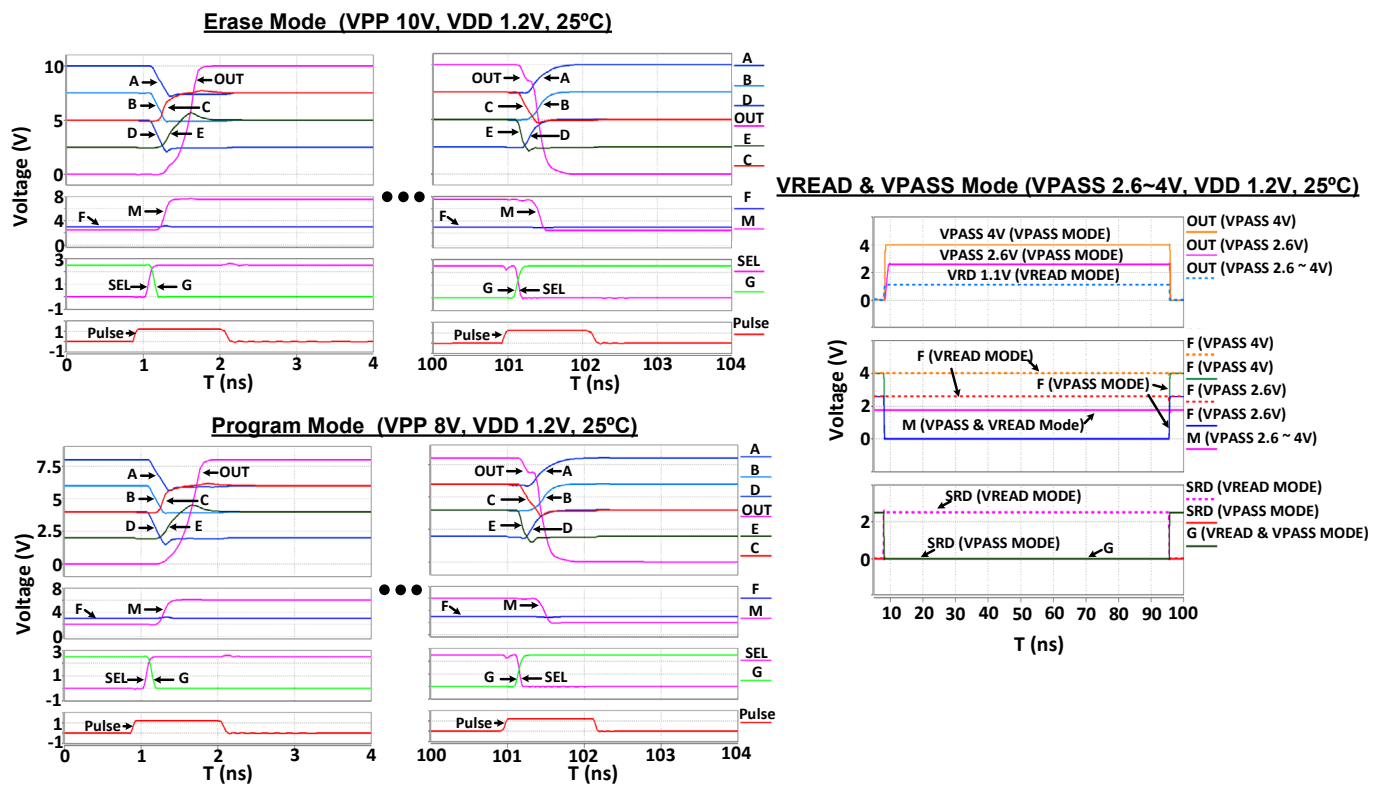


Fig. 14. Post-layout simulation waveforms of the HVS circuit for erase, program and VPASS modes

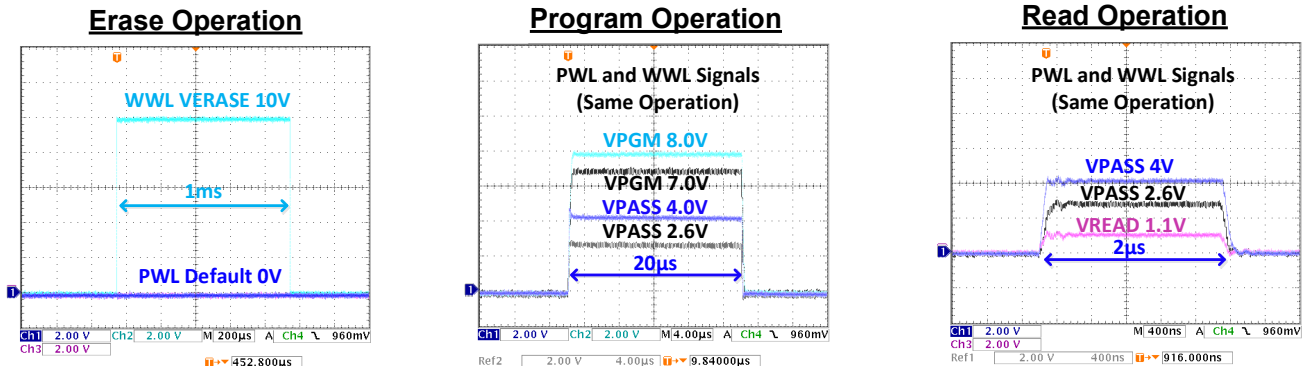


Fig. 15. Measured PWL and WWL signals of the selected and unselected wordlines for erase, program, and read operation.

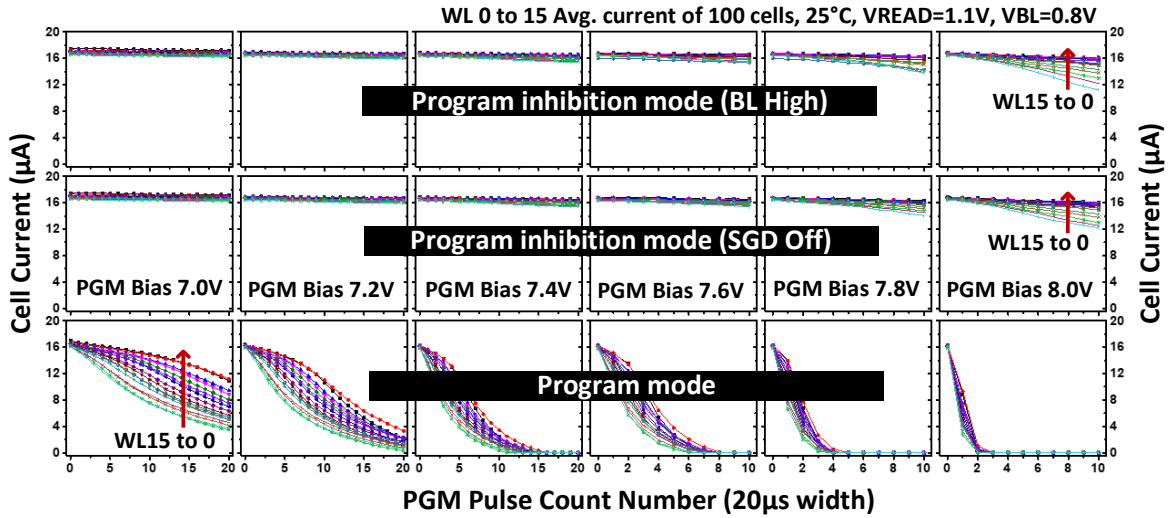


Fig. 16. Cell current versus the number of program pulses for two program inhibition modes (BL high and SGD off) and program mode (bottom row). The average current of 100 cells is shown for each wordline and 0.2V program bias increments from 7.0V to 8.0V for a constant pulse width of 20μs. Test chip data shows reliable programming with minimal program disturbance.

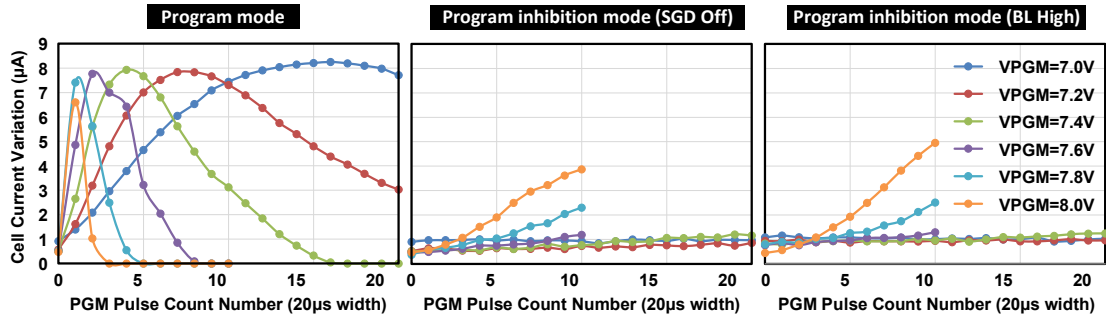


Fig. 17. Variation in the cell current versus the number of program pulses while programming a 16 layer NAND string. Results from program mode and two program inhibition modes are shown.

current making it suitable for the proposed eNAND based neural net hardware.

VI. EXPERIMENTAL RESULTS

Measured data in Fig. 16 confirms correct program and program inhibition characteristics for the eflash cells in the 16 stack NAND string. Here, the cell current was measured while increasing the program voltage from 7.0V to 8.0V with a 0.2V voltage step. As expected, a higher program voltage induces a larger threshold voltage shift but causes program disturbance. The data also shows how the location of the eflash cell in the NAND string affects the program operation. Cells connected to WL 15 (i.e. top of the string) show the largest cell current change for the same number of program pulses. This is due to the fact the same threshold voltage shift has a stronger impact on the overall NAND string current for the cells closer to the top of the string (further discussion can be found in Fig. 21). Fig. 17 shows the cell current variation as a function of the number of program pulses for the three program modes. Based on these results, we chose a program voltage of 7.0V for the final fine-tuning steps as the program disturbance is below 1.2μA. The VCO frequency versus the cell current was characterized (Fig. 18) which was used to convert the measured

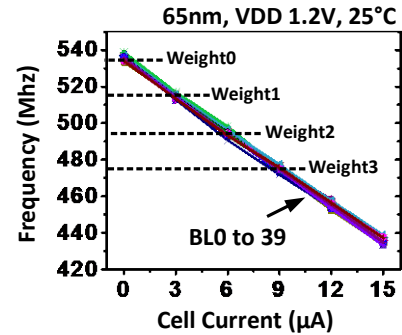


Fig. 18. VCO frequency versus cell current for BL0 to 39, and different weights

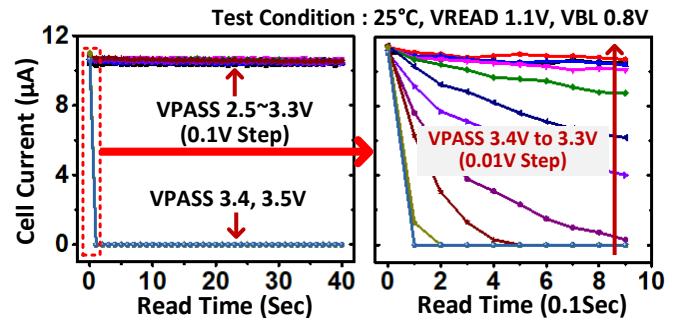


Fig. 19. VPASS disturbance characteristics of eNAND cells during read operation. Cell current remains constant for VPASS below 3.3V.

output frequency to the cell current values. VPASS applied to the unselected wordlines must be high enough to minimize the series resistance of the unselected cells in the NAND string. A high VPASS however, can lead to unwanted threshold voltage shift in the unselected cells (i.e. program disturbance) as shown in Fig. 19. Based on extensive testing, we chose a VPASS voltage of 2.6V which offered a good compromise between the two competing effects.

Another critical challenge we faced while programming the weights into the NAND string is the so-called back pattern dependency where the programmed cell current is affected by the program state of other cells in the NAND string. In our testing, we saw the cell current increase by up to $3\mu\text{A}$ depending on whether the rest of the array is in erase mode (i.e. all low V_t) or weight 0 mode (i.e. all high V_t). To overcome this issue, we devised a novel back-pattern tolerant program-verify scheme illustrated in Fig. 20 which ensures that the programmed cell current remains constant irrespective of the weight values stored in the rest of the array. The operating sequence is as follows. In the first phase, we programmed the weight 0 cells on a given wordline while inhibiting the weight 1, 2, and 3 cells. To ensure that the cell currents of all weight 0 cells are below the target current of $0.1\mu\text{A}$, we applied high voltage (8.0V) and long duration (20 μs) pulses until the target current is reached. Once this has been completed, we program the weight 1, 2, and 3 cell currents so that the cell current is 0-6 μA higher than the target depending on the weight value, using the additional program pulses as shown in Fig. 20. The first pulse is applied to weight 1, 2, 3 cells. The second pulse is applied to weight 1 and 2 cells, and the third pulse is applied to weight 1 cells. The specific program voltage of each pulse was chosen based on the program and program inhibition characteristics in Figs. 16 and 17. The same sequence was repeated for the rest of the wordlines until the entire array is programmed. In the second phase, using smaller and shorter pulses of 7.0V and 10 μs , we fine-tuned the weight 3, weight 2, and weight 1 cells to 9 μA , 6 μA , and 3 μA , respectively. A VPASS voltage of 2.6V was used throughout the program-verify sequence.

Fig. 21 shows how the cell current changes with the number of program pulses for weight 1, 2, and 3 cells. The number of program pulses required to converge to the desired current level depends on the location of the eflash cell in the 16 layer NAND stack. The general trend is that cells closer to the top of the stack (e.g. WL15) can be programmed with fewer pulses than those closer to the bottom of the stack (e.g. WL1). All eflash cells, regardless of their location in the stack, undergo the same bias condition during program mode so the difference in the number of program pulses can be attributed to the different sensitivity between the cell V_{th} and the stack current. That is, the same amount of V_{th} shift has a stronger impact on the stack current for cells closer to the top of the stack. After the program-verify operation is complete, the cell current variation is reduced from $3\mu\text{A}$ to $0.6\mu\text{A}$ which is significantly less than the intrinsic process variation. Our data suggests that systematic variation

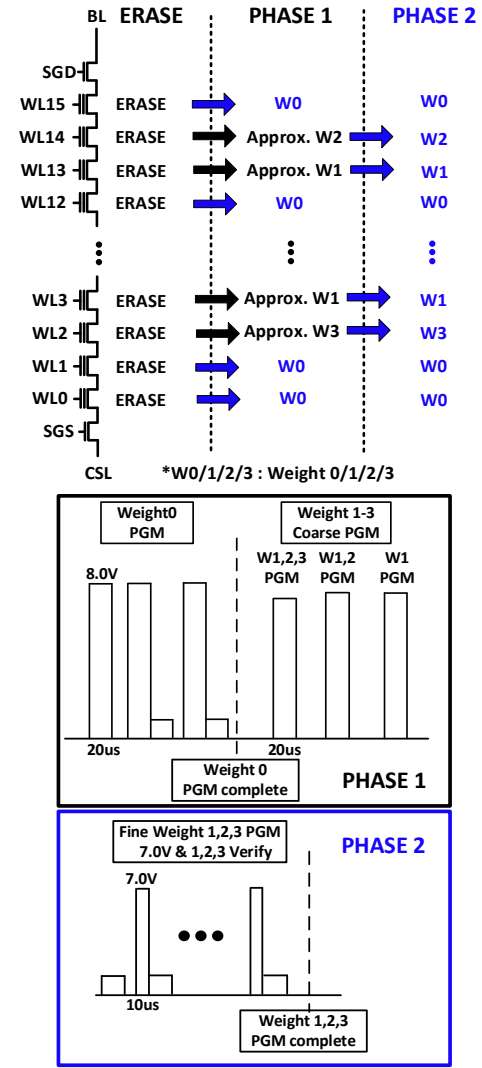


Fig. 20. Pulse sequence for programming weights 0, 1, 2 and 3 into the 16 stack eNAND array. 2 bit weight segments can be programmed into each cell using the proposed back-pattern tolerant program-verify scheme.

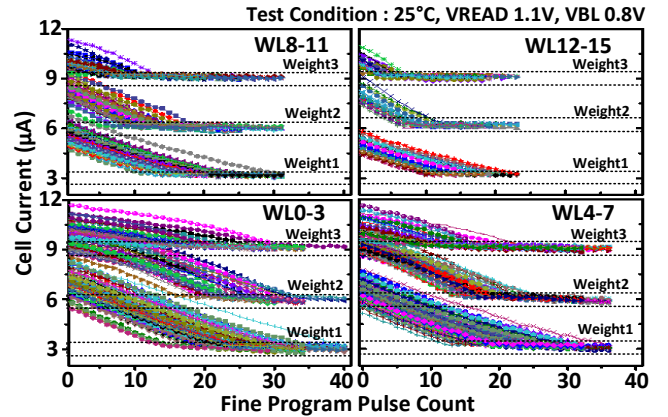


Fig. 21. Cell current versus program pulse count. Fewer program pulses are needed to reach the desired current level for the cells closer to the top of the NAND stack. The program-verify operation ensures a cell current variation less than $0.6\mu\text{A}$.

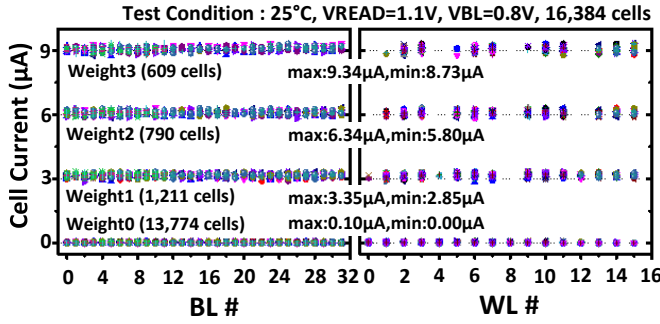


Fig. 22. Individual cell currents in bitline and wordline directions for MNIST trained weights. The number of cells for each weight level and their maximum and minimum programmed cell currents are denoted.

effects can be cancelled out in flash memory cells offering significant advantage over SRAM or MRAM based neuromorphic implementations which do not have any post-silicon tuning capabilities. Fig. 22 shows the measured cell current after programming the entire array using MNIST trained weights [34]. A total of 16,384 cells were programmed for this test, with the number of cells for each weight level being 609 (9µA), 790 (6µA), 1211 (3µA), and 13774 (0µA). Notice that 84.1% of the eflash array were programmed to zero because (1) the vast majority of the MNIST trained weights are zero and (2) the opposite polarity cells in Fig. 10 store a zero weight. The difference between the maximum and minimum cell currents was less than 0.61µA which is 20.3% of the current step size of 3.0µA. No systematic variation was observed in both bitline and wordline directions.

Fig. 23 shows the LeNet-5 CNN demonstration flow [35] for the MNIST handwritten digit recognition application. Weights were trained based on 60,000 handwritten digit images from the MNIST dataset and were preloaded to the test chip. During inference mode, the neural network core generates a frequency output based on an image with 28x28 pixels in 8 bit grayscale precision, and the preloaded 8 bit weights. Due to the long test time, this paper presents classification results of 1,000

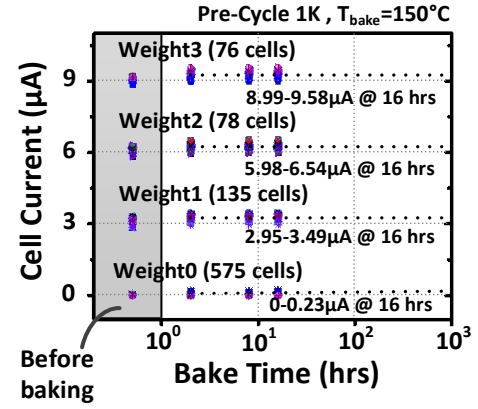


Fig. 24. Retention characteristics of weight 0, 1, 2 and 3 cell currents confirm that cell current variation is maintained below 0.59µA after baking the chip at 150°C for 16 hours.

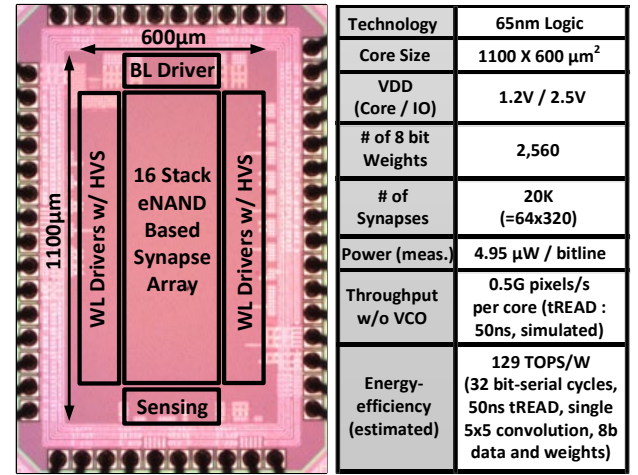


Fig. 25. Die microphotograph and test chip feature summary.

randomly chosen MNIST images. The classification accuracy measured from the test chip was 98.5% (Fig. 23) which is within 0.5% of the software accuracy for the same weight and data precision. The discrepancy can be attributed to noise and

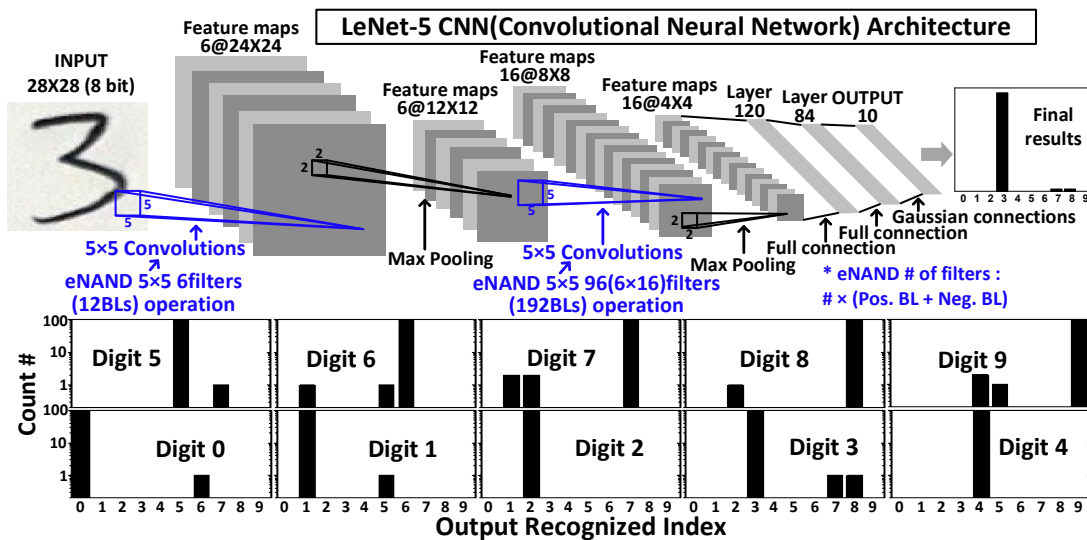


Fig. 23. (Upper) LeNet-5 Convolutional Neural Network (CNN) flow [26] using the proposed neural network core. (Low) Hand written digit recognition results measured from the test chip for 1,000 8 bit grayscale MNIST test images.

	This work	ISSCC'21 [37]	ISSCC'20 [36]	ISSCC'19 [14]	ISSCC'19 [11]	IEDM'18 [15]	ISSCC'18 [13]	IEDM'17 [16]
Technology	65nm	22nm	22nm	55nm	55nm	65nm	65nm	180nm
Voltage	1.2V	0.8V	0.8V	1.0V	1.0V	1.0V	1.0V	2.7V
Cell Type	NAND	NOR	NOR	NOR	NOR	NOR	NOR	NOR
Non volatile?	Yes (eFlash)	Yes (ReRAM)	Yes (ReRAM)	Yes (ReRAM)	No (SRAM)	Yes (eFlash)	Yes (ReRAM)	Yes (Eflash)
Pure CMOS?	Yes	No	No	No	Yes	Yes	No	No
Program-verify?	Yes	No	No	No	No	Yes	No	Yes
Weight Resolution	8 Bits	8 Bits	4 Bits	3 Bits	5 Bits	2.3 Bits	3 Bits	2 Bits
Input Resolution	8 Bits	8 Bits	4 Bits	2 Bits	2 Bits	1 Bit	3 Bits	1 Bit
# of Currents Summed up	28 Cells	4 Cells	4 Cells	8 Cells	32 Cells	68 Cells	14 Cells	4 Cells
# of Levels per Cell	4 Levels	2 Levels	2 Levels	2 Levels	2 Levels	3 Levels	2 Levels	4 Levels
Neural Net Architecture	CNN	CNN	CNN	CNN	CNN	MLP	CNN	MLP

Fig. 26. Comparison with prior works

variation effects in the real chip. The measure inference power was $4.95\mu\text{W}$ per bitline. This power does not include the peripheral circuit power which was too small to be measured due to the low speed testing setup. The estimated energy-efficiency assuming a 50nsec read time is 129 TOPS/W where an operation is defined as a single 5×5 convolution using 8 bit data and 8 bit weights which requires 32 bit-serial cycles (Fig. 4). Charge loss was minimal when the eflash cells were baked at 150°C for 16 hours as shown in Fig. 24. Our MNIST demonstration was based on a conservative 2 bit per cell weight storage, but the cell retention data shows that 3 bit storage is possible. A more detailed study on the cell retention time and variation characteristics can be found in [33].

The die photo and chip feature summary are given in Fig. 25. Comparison with previous NOR-type neuromorphic core designs based on various memory technologies is shown in Fig. 26. To our knowledge, this work represents the first physical demonstration of an embedded flash based network engine inspired by the 3D NAND flash architecture with 16 cells per string, 8 bit MAC operation, and summation of 28 cell currents.

VII. CONCLUSIONS

An embedded flash based $8\text{bit} \times 8\text{bit}$ convolutional neural network core inspired by the 3D NAND flash array architecture was demonstrated in a 65nm standard logic process. The eNAND array mimics the operation of a 3D NAND flash and consists of 16 stack NAND strings where each eFlash cell can storage a 2 bit weight. We adopted the bit-serial scheme to achieve the full $8\text{bit} \times 8\text{bit}$ MAC operation. A novel back-pattern tolerant program verify operation enables precise weight storage within $0.3\mu\text{A}$ of the target current with a program disturbance less than $1.1\mu\text{A}$ @ $\text{VPGM}=7.0\text{V}$. Experimental data from a 65nm test chip confirms a 98.5% digit recognition accuracy, which is within 0.5% of the software accuracy for an equivalent 2-layer LeNet5 CNN benchmark.

REFERENCES

[1] An Analysis of Deep Neural Network Models for Practical Applications," arXiv: 1605.07678[cs.CV], 2016.

[2] X. Si, W. Khwa, J. Chen, et al., "A dual-split 6T SRAM-based computing-in-memory unit- macro with fully parallel product-sum operation for binarized DNN edge processors," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 66, no. 11, pp. 4172–4185, Nov. 2019.

[3] K. Ando, K. Ueyoshi, K. Orimo, et al., "BRin memory: A single-chip binary/ternary reconfigurable in-memory deep neural network accelerator achieving 1.4 tops at 0.6 w," IEEE J. Solid State Circuits, vol. 53, no. 4, pp. 983–994, Apr. 2018.

[4] J. Zhang, Z. Wang, and N. Verma, "In-memory computation of a machine-learning classifier in a standard 6T SRAM array," IEEE J. Solid-State Circuits, vol. 52, no. 4, pp. 915–924, Apr. 2017.

[5] W. Khwa, J. Chen, J. Li, et al., "A 65nm 4Kb algorithm-dependent computing-in- memory SRAM unit-macro with 2.3ns and 55.8TOPS/W fully parallel product-sum operation for binary DNN edge processors," IEEE International Solid-State Circuits Conference (ISSCC), Feb. 2018.

[6] A. Biswas and A. P. Chandrakasan, "Conv-RAM: An energy-efficient SRAM with embedded convolution computation for low-power CNN-based machine learning applications," IEEE International Solid-State Circuits Conference (ISSCC), Feb. 2018.

[7] Z. Jiang, S. Yin, M. Seok, et al., "XNOR-SRAM: In-memory computing SRAM macro for binary/ternary deep neural networks," in Proc. IEEE Symp. VLSI Technology, Jun. 2018, pp. 173–174.

[8] Q. Dong, S. Jeloka, M. Saligane, et al., "A 0.3V VDDmin 4 2T SRAM for searching and in-memory computing using 55nm DDC technology," in Proc. Symp. VLSI Circuits, Jun. 2017, pp. 160–161.

[9] H. Valavi, P. J. Ramadge, E. Nestler, et al., "A mixed-signal binarized convolutional-neural-network accelerator integrating dense weight storage and multiplication for reduced data movement," in Proc. IEEE Symp. VLSI Circuits, Jun. 2018, pp. 141–142.

[10] D. Bankman, L. Yang, B. Moons, et al., "An always-on $3.8\mu\text{J}/86\%$ CIFAR-10 mixed-signal binary CNN processor with all memory on chip in 28-nm CMOS," IEEE International Solid-State Circuits Conference (ISSCC), Feb. 2018.

[11] X. Si, J. Chen, Y. Tu, et al., "A Twin-8T SRAM computation-in-memory macro for multiple-bit CNN-based machine learning," IEEE International Solid-State Circuits Conference (ISSCC), Feb. 2019.

[12] A. Biswas and A. P. Chandrakasan, "CONV-SRAM: An Energy-Efficient SRAM With In-Memory Dot-Product Computation for Low-Power Convolutional Neural Networks," IEEE Journal of Solid-State Circuits, vol. 54, no. 1, pp. 217–230, Jan. 2019.

[13] W. Chen, K. Li, W. Lin, et al., "A 65nm 1Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors," IEEE International Solid-State Circuits Conference (ISSCC), 2018, pp. 494–496.

[14] C. Xue, W. Chen, J. Liu, et al., "A 1Mb Multibit ReRAM Computing-In-Memory Macro with 14.6ns Parallel MAC Computing Time for CNN Based AI Edge Processors," IEEE International Solid-State Circuits Conference (ISSCC), 2019, pp. 388–390.

[15] M. Kim, J. Kim, G. Park, et al., "A 68 Parallel Row Access Neuromorphic Core with 22K Multi-Level Synapses Based on Logic-Compatible Embedded Flash Memory Technology," IEEE International Electron Devices Meeting (IEDM), 2018, pp. 15.4.1–15.4.4.

- [16] X. Guo, F. Bayat, M. Bavandpour, et al., "Fast, energy-efficient, robust, and reproducible mixed-signal neuromorphic classifier based on embedded NOR flash memory technology," IEEE International Electron Devices Meeting (IEDM), 2017, pp. 6.5.1-6.5.4.
- [17] X. Si, Y. Tu, W. Huang, et al., "A 28nm 64Kb 6T SRAM Computing-in-Memory Macro with 8b MAC Operation for AI Edge Chips", IEEE International Solid-State Circuits Conference (ISSCC), Feb. 2020
- [18] Q. Dong, M. Sinangil, B. Erbagci, et al., "A 351TOPS/W and 372.4GOPS Compute-in-Memory SRAM Macro in 7nm FinFET CMOS for Machine-Learning Applications", IEEE International Solid-State Circuits Conference (ISSCC), Feb. 2020
- [19] H. Kim, Q. Chen, and B. Kim, "A 16K SRAM-Based Mixed-Signal In-Memory Computing Macro Featuring Voltage-Mode Accumulator and Row-by-Row ADC," IEEE Asian Solid-State Circuits Conference (ASSCC), Nov. 2019
- [20] C. Yu, T. Yoo, T. Kim, K. Chai, and B. Kim, "A 16K Current-Based 8T SRAM Compute-In-Memory Macro with Decoupled Read/Write and 1-5bit Column ADC," IEEE Custom Integrated Circuits Conference (CICC), Mar. 2020
- [21] H. Maejima, K. Kanda, S. Fujimura, et al., "A 512Gb 3b/Cell 3D flash memory on a 96-word-line-layer technology," IEEE International Solid-State Circuits Conference (ISSCC), 2018, pp. 336-338.
- [22] C. Siau, K. Kim, S. Lee, et al., "A 512Gb 3-bit/Cell 3D Flash Memory on 128-Wordline-Layer with 132MB/s Write Performance Featuring Circuit-Under-Array Technology," IEEE International Solid-State Circuits Conference (ISSCC), 2019, pp. 218-220.
- [23] P. Du, H. Lue, Y. Shih et al., "Overview of 3D NAND Flash and progress of split-page 3D vertical gate (3DVG) NAND architecture," IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), pp. 1-4, 2014
- [24] S. Venkatesan and M. Aoulaiche, "Overview of 3D NAND Technologies and Outlook Invited Paper," Non-Volatile Memory Technology Symposium (NVMTS), Sendai, 2018, pp. 1-5.
- [25] P. Wang, F. Xu, B. Wang, et al., "Three-Dimensional nand Flash for Vector-Matrix Multiplication," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 27, no. 4, pp. 988-991, April 2019.
- [26] S. K. Gonugondla, M. Kang, Y. Kim, et al., "Energy-Efficient Deep In-memory Architecture for NAND Flash Memories," IEEE International Symposium on Circuits and Systems (ISCAS), 2018, pp. 1-5.
- [27] H. Lue, P. Hsu, M. Wei, et al., "Optimal Design Methods to Transform 3D NAND Flash into a High-Density, High-Bandwidth and Low-Power Nonvolatile Computing in Memory (nvCIM) Accelerator for Deep-Learning Neural Networks (DNN)", International Electron Devices Meeting (IEDM), 2019
- [28] S. Lee, H. Kim, J. Bae, et al., "High-Density and Highly-Reliable Binary Neural Networks Using NAND Flash Memory Cells as Synaptic Devices", International Electron Devices Meeting (IEDM), 2019
- [29] M. Kim, M. Liu, L. Everson, et al., "A 3D NAND Flash Ready 8-Bit Convolutional Neural Network Core Demonstrated in a Standard Logic Process," IEEE International Electron Devices Meeting (IEDM), 2019, pp. 38.3.1-38.3.4.
- [30] S. Song, K. Chun, and C. H. Kim, "A logic-compatible embedded flash memory for zero-standby power system-on-chips featuring a multi-story high voltage switch and a selective refresh scheme," IEEE J. Solid-State Circuits, vol. 48, no. 5, pp. 1302-1314, May 2013.
- [31] S. Song, K. Chun, and C. H. Kim, "A bit-by-bit re-writable eflash in a generic logic process for moderate-density embedded non-volatile memory applications," in Proc. IEEE Custom Integrated Circuits Conf. (CICC), 2013, pp. 1-4.
- [32] P. Judd, J. Albericio, T. Hetherington, et al., "Stripes: Bit-serial deep neural network computing," IEEE/ACM International Symposium on Microarchitecture (MICRO), 2016, pp. 1-12.
- [33] M. Kim, J. Song and C.H. Kim, "Reliability Characterization of Logic-Compatible NAND Flash Memory based Synapses with 3-bit per Cell Weights and 1uA Current Steps", International Reliability Physics Symposium (IRPS), Mar. 2020
- [34] MNIST dataset, <http://yann.lecun.com/exdb/mnist/index.html>
- [35] Y. Lecun, L. Bottou, Y. Bengio, et al., "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998.
- [36] C. Xue, T. Huang, J. Liu, et al., "A 22nm 2Mb ReRAM Compute-in-Memory Macro with 121-28TOPS/W for Multibit MAC Computing for Tiny AI Edge Devices," IEEE International Solid-State Circuits Conference (ISSCC), 2020, pp. 244-246
- [37] C. Xue, J. Hung, H. Kao, et al., "A 22nm 4Mb 8b-Precision ReRAM Computing-in-Memory Macro with 11.91 to 195.7TOPS/W for Tiny AI Edge Devices," 2021 IEEE International Solid-State Circuits Conference (ISSCC), 2021, pp. 245-247



Minsu Kim received the B.S degree in electronic and electrical and computer engineering from Hanyang University, Seoul, Korea, in 2007, and the Ph.D. degree in electrical engineering from the University of Minnesota, Minneapolis, Minnesota, USA, in 2020.

He joined the NAND Flash Division, SK Hynix, Icheon, Korea, in 2007. From 2007 to 2016, he involved in NAND Flash circuit design. After his Ph.D. study at the U of M, he rejoined SK Hynix in 2020, where he has worked for high density and speed NAND development. Dr. Kim is the recipient of a SK Hynix Ph.D. Scholarship for outstanding employees. His research interests include digital, mixed-signal, and memory circuit designs, computing in memory with non-volatile memory, and energy efficient circuits and systems.



Muqing Liu (Member, IEEE) received the B.S. degree in applied physics from Tongji University, Shanghai, China, in 2013, the M.S. degree in electrical engineering from Columbia University, New York, NY, USA, in 2015, and the Ph.D. degree from University of Minnesota, Minneapolis, MN, USA in 2019, also in electrical

engineering.

In 2015, she joined the VLSI Research Laboratory, University of Minnesota, with a focus on neuromorphic circuits design and hardware security, including physical unclonable functions and counterfeit electronics sensor design. In 2019, she joined IBM as a hardware developer, developing circuits to monitor server product reliability. She is now with Western Digital Corporation as a R&D engineer, focusing on applying emerging non-volatile memory/memristor technologies in neuromorphic computing and next generation platforms development. Ms. Liu was a recipient of the Best Paper Award in 2017 ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED) and Best Poster Award, 2nd Place, in Women in Hardware and System Security (WISE) workshop in 2019.



Luke R. Everson (Member, IEEE) received the B.S.E.E., M.S.E.E., and Ph.D. degrees from the University of Minnesota, Minneapolis, MN, USA, in 2015, 2016, and 2019, respectively.

He is currently a Memory Designer with the Central Engineering Group, Broadcom Inc., San Jose, CA, USA. He has coauthored over 20 articles in a diverse range of circuit topics ranging from machine learning, neural recording, radiation effects, and architecture.



Chris H. Kim (Fellow, IEEE) is currently the Louis John Schnell professor in Electrical and Computer Engineering at the University of Minnesota. His group has expertise in digital, mixed-signal, and memory IC design, with emphasis on circuit reliability, hardware security,

memory circuits, radiation effects, time-based circuits, machine learning, and quantum-inspired hardware design.

Prof. Kim was a recipient of the University of Minnesota's Taylor Award for Distinguished Research, the SRC Technical Excellence Award for his "Silicon Odometer" work, NSF CAREER Award, the McKnight Foundation Land-Grant Professorship, the DAC/ISSCC Student Design Contest Awards, the IBM Faculty Partnership Awards, the ISLPED Low Power Design Contest Awards, and the ISLPED Best Paper Awards.