Received 9 September 2024; revised 29 October 2024; accepted 7 November 2024.
Date of publication 15 November 2024; date of current version 23 December 2024.

Digital Object Identifier 10.1109/JXCDC.2024.3498837

SpecPCM: A Low-Power PCM-Based In-Memory Computing Accelerator for Full-Stack Mass Spectrometry Analysis

KEMING FAN[1], ASHKAN MORADIFIROUZABADI[1] (Graduate Student Member, IEEE), XIANGJIN WU[2] (Graduate Student Member, IEEE), ZHEYU LI[3], FLAVIO PONZINA[3], ANTON PERSSON[2], ERIC POP[2,4] (Fellow, IEEE), TAJANA ROSING[3] (Fellow, IEEE), and MINGU KANG[1]

[1]Department of Electrical and Computer Engineering, University of California at San Diego, San Diego, CA 92093 USA
[2]Department of Electrical Engineering, Stanford University, Palo Alto, CA 94305 USA
[3]Department of Computer Science and Engineering, University of California at San Diego, San Diego, CA 92093 USA
[4]Department of Materials Science and Engineering, Stanford University, Palo Alto, CA 94305 USA

CORRESPONDING AUTHOR: M. KANG (mingu@ucsd.edu)

This work was supported by the Center for Processing with Intelligent Storage and Memory (PRISM) Center within JUMP 2.0, a Semiconductor Research Corporation (SRC) Program with the Defense Advanced Research Projects Agency (DARPA).

This article has supplementary downloadable material available at https://doi.org/10.1109/JXCDC.2024.3498837, provided by the authors.

ABSTRACT Mass spectrometry (MS) is essential for proteomics and metabolomics but faces impending challenges in efficiently processing the vast volumes of data. This article introduces SpecPCM, an in-memory computing (IMC) accelerator designed to achieve substantial improvements in energy and delay efficiency for both MS spectral clustering and database (DB) search. SpecPCM employs analog processing with low-voltage swing and utilizes recently introduced phase change memory (PCM) devices based on superlattice materials, optimized for low-voltage and low-power programming. Our approach integrates contributions across multiple levels: application, algorithm, circuit, device, and instruction sets. We leverage a robust hyperdimensional computing (HD) algorithm with a novel dimension-packing method and develop specialized hardware for the end-to-end MS pipeline to overcome the nonideal behavior of PCM devices. We further optimize multilevel PCM devices for different tasks by using different materials. We also perform a comprehensive design exploration to improve energy and delay efficiency while maintaining accuracy, exploring various combinations of hardware and software parameters controlled by the instruction set architecture (ISA). SpecPCM, with up to three bits per cell, achieves speedups of up to 82× and 143× for MS clustering and DB search tasks, respectively, along with a four-orders-of-magnitude improvement in energy efficiency compared with state-of-the-art (SoA) CPU/GPU tools.

INDEX TERMS Hardware-software co-design, hyperdimensional computing (HD), in-memory computing (IMC), instruction set architecture (ISA), mass spectrometry (MS), phase change memory (PCM).

I. INTRODUCTION

MASS spectrometry (MS) is a key analytical tool used by proteomics and metabolomics, aiding in drug discovery and chemical analysis by identifying and quantifying molecules based on their mass-to-charge ratios [1]. Its high sensitivity and precision have made it one of the most widely used techniques for detecting even the smallest molecular variations. However, one of the key challenges in MS processing is handling the vast and continually growing data volumes. For example, the MassIVE database (DB), a publicly accessible repository for proteomics MS data [2], now exceeds 600 TB (as of September 2024) and continues to grow at an accelerating pace, with hundreds of terabytes of new spectral data added annually. The MS analysis process involves comparing spectra generated from MS experiments against an extensive reference library to identify proteins, a procedure known as DB search [3]. To accelerate the search process, spectral clustering is done by grouping similar reference spectra together. During search, the query is first compared to the cluster centroids, quickly focusing the search on an appropriate cluster and thereby speeding up the overall process [4]. Ideally, such a DB should be clustered on a daily basis as new samples are continually added, but this is currently done only once per year due to the excessive time

required, resulting in lower accuracy. Traditional systems with separate memory and processor units are hindered by limited data movement bandwidth and computing efficiency in processing such a sheer amount of data. Modern MS analysis tools [5], [6], [7], whether based on conventional CPU or GPU architectures, often spend more than 60% of their time on large matrix operations with significant memory footprint. These tools struggle to manage large datasets efficiently, mainly due to the high energy consumption and latency involved in data transfer.

To overcome these challenges, in-memory computing (IMC) has emerged as an alternative paradigm that processes data directly within the memory where it is stored, substantially reducing the latency and energy overhead caused by data movement. To capitalize on this opportunity, various memory topologies have been explored, including DRAM, SRAM, RRAM, PCM, NAND Flash, and other emerging memory technologies [8]. While RRAM has been widely adopted [9], [10], [11] for its well-recognized high-density and efficient read operations, particularly due to its support for multilevel cells (MLCs), it faces limitations, such as high energy consumption and high voltage requirements during write operations. This drawback will significantly degrade the energy efficiency of the clustering process, where frequent data updates are required to adapt to the newly collected MS data. While NAND flash memories provide superior memory density and fabrication maturity, they suffer from relatively high latency, e.g., several microsecond [12], due to the high resistance in the read-path, which is caused by the nature of reading data from serially connected cells.

This work adopts a recently developed multilevel phase change memory (PCM) [13] based on superlattice materials, which features lower error rates, reduced voltage requirements, faster, and more energy-efficient programming. In particular, we aim to explore the analog IMC, which offers dramatic efficiency improvements, achieving more than two orders of magnitude benefit [8] compared with conventional digital counterparts by utilizing low-voltage swing analog operations. Additionally, the analog IMC performs both reading and computation across the entire bitcell array simultaneously, enabling high parallelism and significant compute density improvement. Consequently, the analog IMC on PCM facilitates efficient processing for classification while also enabling the effective updating of stored weights to adapt to newly collected MS data.

From an algorithmic perspective, we employ hyperdimensional computing (HD), a brain-inspired computing paradigm that leverages lightweight and highly parallel operations by encoding input features into high-dimensional (long) binary vectors. HD replaces costly and complex floating-point arithmetic with simpler binary or integer operations, which can be executed in parallel, leading to dramatic throughput improvements as demonstrated in [14], [15], and [16]. Furthermore, HD's data representation in hyperspace offers significant error resilience, with data points being well separated by large geometric distances. This property has been demonstrated in previous work [10], where HD tolerated up to a 10% bit error rate for MS DB search tasks. This resilience creates a strong synergy with analog IMC on emerging devices, helping to overcome their computing and storage nonidealities while achieving greater storage density and computing efficiency.

The use of MLC PCM involves complex trade-offs between efficiency and accuracy, which require careful optimization across both hardware and algorithms. These interrelated challenges cannot be addressed at a single abstraction level. Therefore, we introduce SpecPCM, an IMC accelerator designed for the efficient processing of MS workloads. This framework integrates design efforts across the entire vertical stack, spanning application, algorithm, circuit, device, and instruction set levels, to enhance performance throughout the end-to-end MS pipeline. The detailed contributions of this work are summarized as follows.

1) We propose an analog IMC system with architecture and circuits specifically tailored for MS algorithms. While prior works have applied IMC to MS DB search tasks in the HD domain, our work is the first to apply IMC for both clustering and DB search.
2) At the algorithm level, we introduce a new HD encoding method, called dimension packing, to maximize storage density by leveraging multilevel PCM devices while maintaining the simplicity of the binary representation of HD vectors.
3) At the device level, we propose customized PCM devices to meet the distinct requirements of clustering and MS DB search by optimizing the materials differently for each task, based on measured characterization results from the fabricated devices.
4) We conduct hardware–software co-design through a comprehensive analysis to balance trade-offs between latency, energy efficiency, and accuracy, taking into account various parameters such as bits per cell, write-verify cycles, analog-to-digital converter (ADC) precision, and HD dimensions, all controlled by the instruction set.

The results indicate that the proposed SpecPCM demonstrates speedup of up to 82× for clustering and 143× for DB search over state-of-the-art (SoA) solutions, with a four-orders-of-magnitude in energy efficiency improvement, while maintaining on-par accuracy across datasets of different scales.

II. BACKGROUND AND MOTIVATION

This section provides background on HD, the MS algorithm, and the IMC for the MS analysis to motivate the proposed PCM-based analog IMC architecture.

A. HYPERDIMENSIONAL COMPUTING

HD is a brain-inspired computing paradigm that leverages lightweight and highly parallel operations, by encoding input features into high-dimensional binary vectors called hypervectors (HVs), typically with a dimension of 1k–10k [14].

1) HD ENCODING

Encoding is the process of mapping raw data to an HD vector. Despite its diversity, we introduce the ID-level encoding [10] approach. Initially, two sets of HVs are created, ID HVs and level HVs. Both sets consist of D-dimensional HVs, where each element is either -1 or 1. The encoding scheme assigns a unique ID HV to each feature position. These ID HVs are randomly generated to ensure orthogonality among all features. Additionally, a set of level HVs is generated to represent the value of each feature. To create these level HVs, we identify the minimum and maximum feature values across all data points, denoted as $l_{\min}$ and $l_{\max}$. The range $[l_{\min}, l_{\max}]$ is then quantized into m levels. Each level is associated with a corresponding level HV, e.g., $\mathrm{LV}_1, \ldots, \mathrm{LV}_m$ for the m different levels. To encode a feature vector, the encoder performs an element-wise multiplication and accumulation (MAC) between the position ID HV (ID_i) and the corresponding level HV (hv_i) for the ith feature position. The resulting HV (HV_i) is then binarized to complete the encoding process, i.e., $\mathrm{HV}_i = \mathrm{sign}(\mathrm{ID}_i \cdot hv_i)$. The following equation demonstrates how a D-dim feature vector is mapped into the HD space:

$$\mathrm{HV}_i = \mathrm{sign}(hv_{i1} * \mathrm{ID}_{i1} + \cdots + hv_{iD} * \mathrm{ID}_{iD}) \qquad (1)$$

where $hv_i \in \{\mathrm{LV}_1, \ldots, \mathrm{LV}_m\}$, $\mathrm{LV}_k \in \{-1, 1\}^D$, $\mathrm{ID}_i \in \{-1, 1\}^D$, and sign outputs 1 when the input is positive and -1 otherwise. The ID_{ij} and hv_{ij} are the jth element of ID_i and hv_i, respectively.

2) CLASSIFICATION AND INFERENCE

The HD model classifies input samples using class HVs, each representing a specific class. Classification involves calculating the similarity between the encoded query and class HVs, with the predicted class being the one whose HV is most similar to the input. Similarity is computed using Hamming distance, which equals the dot product of two bipolar vectors, and the class with the highest score is chosen $y = \mathrm{argmax}_j(\sigma(\vec{\mathcal{Q}}, \vec{\mathcal{C}}_j))$, where σ represents the similarity function, $\vec{\mathcal{Q}}$ is the encoded query vector, $\vec{\mathcal{C}}$ is the class HV, and y is the predicted label.

B. MS AND ITS ACCELERATIONS

Proteomics is essential for understanding biological processes and drug discovery, as it identifies key proteins, biomarkers, and therapeutic targets involved in health and disease. MS is a key tool in this field, enabling detailed analysis of protein compositions. MS measures the mass-to-charge ratio (m/z) of ionized proteins, with the results presented as a spectrum—a plot of intensity as a function of the m/z ratio. Modern MS experiments generate millions of spectra, requiring the efficient processing of hundreds of terabytes (TB) of data [2]. MS analysis involves two main tasks: clustering and DB search, both of which can be implemented using HD. In the clustering process (Fig. 1), spectra are first divided into several buckets based on biofeatures. Inside each

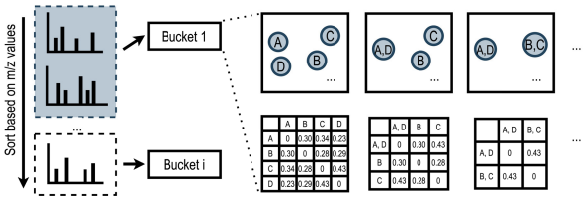

FIGURE 1. Overview of spectral clustering for MS analysis.

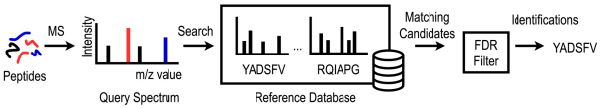

FIGURE 2. Overview of MS DB search.

bucket, spectra are encoded into HVs, and pairwise distances between data points are calculated using dot products to form a distance matrix. The algorithm begins with each data point in its own cluster and iteratively merges the closest clusters until a distance threshold is reached. This process generates the representative reference DB with a condensed volume of data. For the DB search (Fig. 2), a Hamming similarity search compares query HVs to reference HVs in the DB, identifying the closest reference based on the dot product score. Finally, matching candidates are filtered with a false discovery rate (FDR), a common technique using decoy spectra to evaluate accuracy [17].

In both clustering and DB search, each spectrum from an MS experiment is extensively compared against a collection of spectra, a process that is particularly time-consuming due to the sheer volume of data involved. Previous works, such as [18] and [19], and ANN-SoLo [5], have attempted to accelerate MS analysis using techniques including hashing, approximate nearest neighbor search, and efficient dot products. However, these tools have limited efficiency due to their reliance on complex, high-precision floating-point arithmetic. In contrast, HD-powered tools, such as HyperSpec [6] for clustering and HyperOMS [7] for DB search, demonstrate the fastest operations by utilizing simple Boolean operations and enabling significantly higher hardware parallelism.

Despite the latency improvements provided by HD, latency profiling results (Fig. 3) reveal a new scalability challenge when handling large datasets. Even with an NVIDIA 4090 GPU equipped with 24 GB VRAM, distance calculation remains the primary bottleneck in clustering, while Hamming similarity search is the main bottleneck in DB search. Both stages involve large-scale matrix computations, leading to significant data movement, particularly when the dataset exceeds the GPU's onboard memory capacity. This data movement between the GPU and main memory hampers processing efficiency and results in substantial performance overheads. These observations underscore the motivation for accelerating key operations, such as distance calculation and Hamming similarity search, using analog IMC. By processing directly within memory in combination with HD, this approach aims to minimize the costly data movement and enhance the overall efficiency of MS analysis.

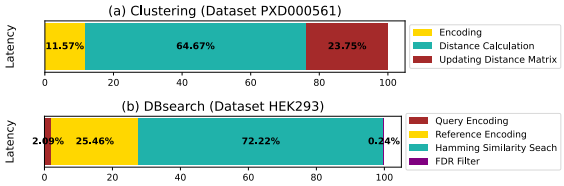

FIGURE 3. Latency breakdown for GPU tools. (a) Clustering on dataset PXD000561 using HyperSpec. (b) DB search on dataset HEK293 using HyperOMS.

TABLE 1. SpecPCM Hardware Configurations.

Component	Configurations
PCM Array	128×128 2T2R cells, 3 bits per cell
Flash ADC	6-bit precision, 16 units each shared between eight rows of cells
DAC	3-bit precision, 128 units each for one col of cells
SL Gen / Drive	64 units each shared between four cols of cells
Read Gen	Two units for each row of cells, activated for the target row
WL Decode / Drive	8-bit decoder, 256 driver units, two per one row of cells
Sense Amp	3-bit precision, 32 units each shared between four cols of cells

C. IMC FOR MS ANALYSIS AND APPLICATION OF PCM

The IMC has been exploited to accelerate MS analysis using various memory technologies. While DRAM [20] and NAND flash [12] have been employed in near-memory computing architectures for MS analysis, they are rarely used for analog IMC. This is due to DRAM's volatility and NAND flash's serially connected structures, which prevent the analog IMC operations from all bitcells. In contrast, RRAM has been actively explored for analog IMC in MS analysis in conjunction with HD, due to its fast read access, simple fabrication process, and suitability for MLC operations, primarily focusing on classification for DB search [10]. Despite these advantages, RRAM suffers from orders of magnitude higher write latency compared to read latency and the requirement for significantly high programming voltages (e.g., more than 2 V). These drawbacks are less apparent during classification tasks, where the weights remain static after the model is trained. However, in tasks such as training and spectral clustering, frequent data updates are required, necessitating repeated write operations, which exacerbate the impact of these limitations.

Unlike previous studies that focus either on clustering or DB search, this work aims to provide an end-to-end solution for both. To achieve this, we adopt a recently introduced PCM device with a superlattice structure based on nanocomposites of $Ge_4Sb_6Te_7$ [13] to leverage the following unique advantages: 1) a very low programming voltage (less than 1.0 V), which ensures great compatibility with a logic die without requiring specialized peripheral circuitry to support high voltage; 2) low switching energy (approximately pJ) due to the low forming voltage; 3) reduced resistance drift, which significantly supports stable MLCs and increases the effective storage density; and 4) PCM's well-established technology and process maturity.

Despite the above benefits, utilizing MLC on PCM remains challenging due to increased susceptibility to noise and variability, with error rates often exceeding 10% even after meticulous write-verify operations [10]. These challenges impede real-world deployment, underscoring the need for error-tolerant algorithms. This work leverages HD computing for its superior error resilience, as demonstrated in [10].

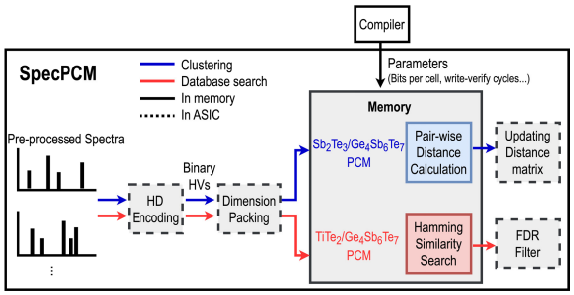

FIGURE 4. Overview of the SpecPCM accelerator.

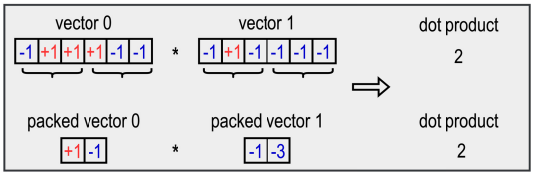

FIGURE 5. Dimension packing for MLC with 3 bits per cell.

III. PROPOSED SpecPCM USING ANALOG IMC BASED ON PCM

This section outlines the hardware and software implementations of SpecPCM, covering algorithmic, architectural, and device perspectives.

A. SpecPCM OVERVIEW

Fig. 4 illustrates the system overview of the SpecPCM accelerator, which supports both clustering and DB search. The memory-intensive distance calculations in clustering and Hamming distance calculation in DB search, identified as the main bottlenecks, are offloaded to PCM-based memory for the IMC processing. The remaining steps are handled by near-memory ASIC blocks. Encoded HVs are packed and then passed to memory. Each memory bank array is sized as 128 × 128 cells and 2T2R per cell, with multiple arrays to enable parallel processing. For clustering, distance calculations are performed directly in $Sb_2Te_3/Ge_4Sb_6Te_7$ PCM, which supports lower programming power whereas hamming similarity searches are executed in $TiTe_2/Ge_4Sb_6Te_7$ PCM, which offers longer retention time and lower error rates [13]. A detailed discussion of the selection of these PCM types is provided in Section III-E. Various parameters for controlling IMC operations, including the number of bits per cell, write-verify cycles, and ADC precision, are managed by the software through a custom instruction set architecture (ISA), which will be discussed in Section III-F.

B. DIMENSION PACKING IN HD ALGORITHM

Previous work has shown that ID-level encoding performs best for MS analysis [6], [7] as it captures essential spectral information with minimal loss. Each spectrum is encoded as a HV in binary format, as shown in (1) and later operations (e.g., dot product) on those binary vectors require bit-wise

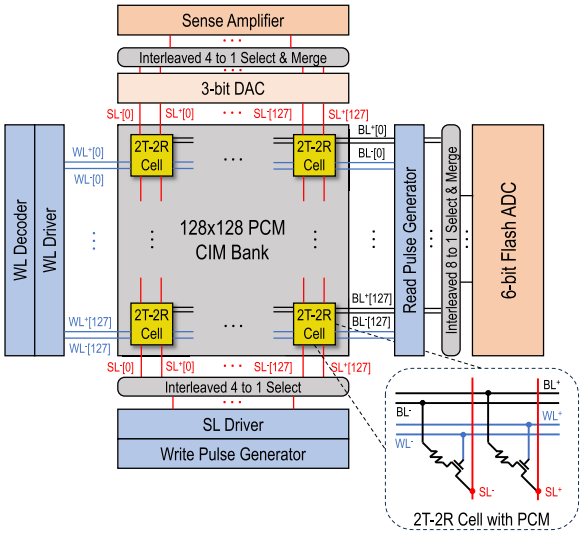

FIGURE 6. IMC array with 2T2R PCM cell structure.

operations. In contrast, MLC hardware is designed for nonbinary, integer-based computations. Therefore, directly storing and processing binary vectors on MLC devices is suboptimal in terms of storage and computational efficiency. To address this, we propose a dimension packing method (see Fig. 5), which is processed in ASIC. This involves converting the binary vector of length D into a compressed vector of length D/n by summing n adjacent bits, where n is the number of bits per cell. This compression aligns the data with the MLC devices' optimal format, improving the storage and compute density by n times while maintaining accuracy with only a negligible drop compared to the original binary version, as will be shown in Section IV.

C. HARDWARE OPERATIONS

The clustering begins with the encoding and dimension packing of spectral data. These packed HVs are then programmed into the PCM memory arrays, each configured as a 128 × 128 matrix using a 2T2R cell structure for each element. As shown in Fig. 6, each element is represented by a pair of PCM cells to store signed numbers, with the value expressed as the difference in conductance between the two cells, as introduced in [9] and [11]. Each row in the array contains the data for a single HV, with multiple HVs stored across different rows in the array. Due to the high dimensionality of the HVs (e.g., 1024 dimensions after packing), a single row in the array cannot store an entire HV. Instead, each row in an array stores a different segment of HV, with parts of the same HV distributed across multiple arrays at the same row. Multiple arrays can operate in parallel for higher throughput.

1) PROGRAMMING

To program the HVs into the array, pulses are generated by the generator module and transmitted to the array through source-line (SL) drivers. The voltage level of each SL is modulated based on the target resistance value of each cell inside the write pulse generator. The bit-lines (BLs) are connected to the ground and the target row is activated by the word-line (WL) decoder.

TABLE 2. Clustering speedup versus prior works.

Tools	Falcon	msCRUSH	HyperSpec	SpecHD	SpecPCM
Hardware	CPU	CPU	GPU	FPGA	TSMC 40nm
Dataset			PXD001468		
Latency	573s	358s	38s	13.17s	5.46s
Speedup	1×	1.6×	15.1×	43.5×	104.94×
Dataset			PXD000561		
Latency	134 min	42 min	17 min	179s	98.4s
Speedup	1×	3.2×	7.9×	44.9×	81.7×

2) NORMAL READ OPERATION

To perform a normal read operation, a target row is activated by the WL decoder. The pulse generator module creates pulses smaller than 0.4 V to prevent read disturbances in neighboring cells. The read pulses are applied through BLs of the target row based on the activated WL and the values of a whole row are read at a time through SLs using sense amplifiers.

3) IMC FOR CLUSTERING

During clustering, the distances between all HVs need to be calculated. The retrieved HV from the array through the normal read operation serve as an input for the pair-wise distance calculation with all the other HVs in the memory through the IMC operation. During the IMC, the signed input is sent to the array through the SLs using a 3-bit DAC. To enable parallel computing of the whole array, all the WLs are enabled at the same time to deliver the inputs to multiple rows. The distances between the input HV and the stored HVs in all the rows are computed at a time via the dot-product operations and generated on the positive and negative BLs (BL+ and BL−) as analog outputs. These BL voltages are converted to digital values by 6-bit Flash ADCs with 63 dynamic comparators, with each ADC shared across every eight rows. Each ADC operation takes one cycle to complete, and the entire IMC process requires ten cycles, including input generation overhead via the DACs.

Then, the generated distance matrix is stored in a separate block of PCM memory array. This distance matrix is dynamically updated by the near-memory ASIC logic, which manages the merging of data points into a group based on the computed distances. The ASIC employs the complete linkage method, where the maximum distance between one element from each of two clusters determines the distance between the clusters. This process iteratively merges the closest clusters and updates the distance matrix accordingly. At each iteration of such operations, the newly generated clusters are written to the memory array through programming operations. Such a programming overhead is mitigated via the device optimization in Section III-E and controlling the write-verify iterations.

4) IMC FOR DB SEARCH

For DB search, an input query HV needs to be compared with all the stored reference HVs simultaneously. The query HV

is converted via the proposed dimension packing, and applied through the SLs as an input of IMC operation. The PCM array then performs a dot product of the input query vector with all the stored reference HVs simultaneously, same as in the clustering. The resulting partial sums from these operations are transmitted to the peripheral ASIC logic. The ASIC then processes these sums and identifies the highest score as the matching candidate, effectively determining the best match between the query and reference HVs.

D. EFFICIENCY VERSUS ACCURACY TRADE-OFFS

We utilize multiple control knobs to manage the efficiency versus accuracy trade-offs, depending on the target application and processing stages.

1) HD VECTOR DIMENSION

The dimension of the HV is an important parameter for controlling the amount of information. A higher HD dimension can be achieved by simply utilizing more memory space, with additional arrays for the distributed storage.

2) RECONFIGURABLE ADC BITS

While the 6-bit ADC is employed with 63 comparators, the effective bit precision can be modulated to be 1–6 bits by partially enabling the ADCs to reduce the energy overhead without modifying the hardware.

3) WRITE-VERIFY CYCLES

For each write-verify cycle, the resistance of the cell is read after each write and compared to the target resistance level. An additional pulse is applied if needed, e.g., if the resistance is lower than the target, a pulse with higher amplitude or iterative pulse is applied, and vice versa. Such a write-verify operation increases the programming overhead by the amount of cycle numbers. Fig. 7 shows bit error rate versus write-verify cycles measured from 100 fabricated devices, averaged over 100 rounds of measurements. The bit error rate decreases as the number of write-verify cycles (and thus write latency) increases.

E. PCM DEVICE OPTIMIZATION

In addition to reconfigurable parameters in Section III-D, we leverage two different PCM device technologies, based on superlattices of $Sb_2Te_3/Ge_4Sb_6Te_7$ and $TiTe_2/Ge_4Sb_6Te_7$, to maximize the efficiency for different use cases. Due to the differences in fundamental material properties, these two technologies provide unique trade-offs between programming energy, retention, and resistance on/off ratio (Supplementary Table S1). $Sb_2Te_3/Ge_4Sb_6Te_7$ and $TiTe_2/Ge_4Sb_6Te_7$ require programming voltages of 0.65–0.8 V and 0.85–1 V, respectively, with higher voltages needed to program higher resistance levels. The clustering stage requires iterative programming to update the newly generated clusters periodically, as discussed in Section III-C. Given that both types of PCM devices have an endurance of over 10^8 cycles and clustering typically involves fewer than 100 iterations, resulting in under 100 writes per cell, the system theoretically supports over 10^6 clustering processes. Furthermore, many iterations involve minor adjustments, particularly in later stages, where the algorithm fine-tunes existing clusters, leading to relatively mild write operations. For this reason, we expect the proposed system to practically support well over 10^6 clustering processes in realistic scenarios. The write-intensive nature of clustering makes energy efficiency in programming crucial, while the retention period of the device can be significantly relaxed. The $Sb_2Te_3/Ge_4Sb_6Te_7$ provides higher ON-resistance (with a similar ON/OFF-resistance ratio of $\approx 100\times$) and lower programming current, reducing energy consumption in peripheral circuits during the programming operation. On the other hand, the DB search requires storing the reference HVs for a long enough retention time to be compared with the input HVs. $TiTe_2/Ge_4Sb_6Te_7$ offers longer retention time and lower error rate at operation temperature (105 °C) at the cost of 2.6× higher programming energy. As a result, the MS system for the clustering is implemented using $Sb_2Te_3/Ge_4Sb_6Te_7$ PCM while DB search, which requires intensive read operations and longer retention time, is implemented using $TiTe_2/Ge_4Sb_6Te_7$ PCM.

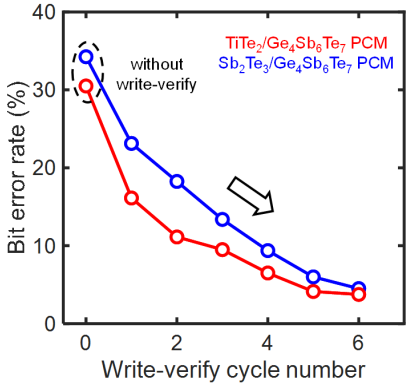

FIGURE 7. Experimentally measured bit error rate as a function of write-verify cycles for 3-bits per cell.

TABLE 3. DB search speedup versus prior works.

Tools	ANN-SoLo	HyperOMS	RRAM	3D NAND	SpecPCM
Hardware	CPU-GPU	GPU	130nm	ASAP 7nm	TSMC 40nm
Dataset	iPRG2012				
Latency	6.45s	2.08s	1.22s	0.145s	0.049s
Speedup	1×	3.1×	5.3×	44.2×	131.63×
Dataset	HEK293				
Latency	45.14s	10.4s	-	-	0.316s
Speedup	1×	4.34×	-	-	142.84×

F. INSTRUCTION SET

Given various parameters discussed in Section III-D, we developed an ISA to effectively control them from the software. This ISA, detailed in Table S2 manages memory operations in the proposed system including read, program, write-verify, and in-memory dot product used in clustering and DB search. The instruction set also configures parameters such as write_cycles, MLC_bits, ADC_bits, and HD_dimensions given the user's requirements.

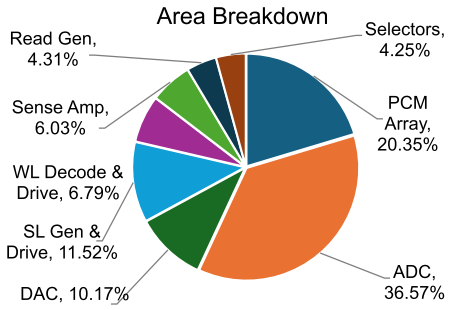

FIGURE 8. Area breakdown of SpecPCM system.

IV. EVALUATION

This section presents the experimental results of the proposed MS system, highlighting the energy and delay benefits achieved through the proposed optimizations, along with their impact on accuracy for various benchmarks.

A. EXPERIMENTAL SETUP

1) DATASETS

We evaluate the design using two real-world datasets for clustering and two for DB search, representing small and large scales. For clustering, the following two datasets are employed: 1) a small-scale dataset, PXD001468 [21] and 2) a large dataset, PXD000561 [22]. For DB search, we use: 1) the small-scale iPRG2012 [23]; and 2) the larger scale HEK293. The details are described in Supplementary Section S.A.

2) QUALITY METRICS

For clustering, we evaluate the quality using the cluster spectra ratio, which is the number of clustered spectra divided by the total number of spectra. This metric assesses the clustering capability of each tool while keeping the incorrect clustering ratio fixed. For DB search, we compare the number of total identified peptides given the fixed FDR rate against those identified by other tools.

3) HARDWARE CONFIGURATIONS

By default, three write-verify cycles, 3-bit MLC, HD dimension of 8192, and a 6-bit ADC are employed for the DB search. On the other hand, the HD dimension is set to 2048 for the clustering. While the 3-bit MLC and a 6-bit ADC are employed for the clustering similar to the DB search, no write-verify is used for the default setup due to the strong error tolerance of clustering process [see Supplementary Fig. S3(a)]. We built an in-house simulator to model our system based on the methodology in Supplementary Section S.B. The configuration of each hardware component is detailed in Table 1. The hardware system is built with the CMOS 40 nm process at a target clock frequency of 500 MHz. The area breakdown is shown in Fig. 8 indicating high overhead from the ADC. Therefore, an ADC unit is shared across eight rows of cells to minimize the area overhead. The detailed power breakdown is shown in Supplementary Table S3.

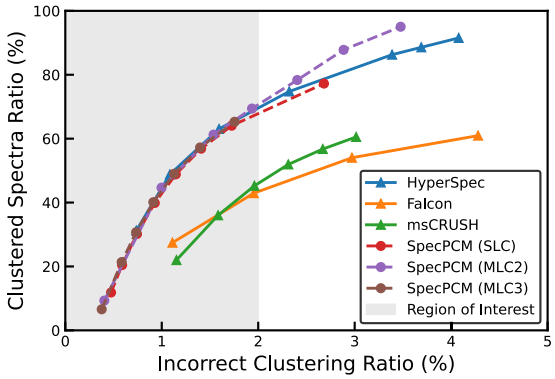

FIGURE 9. Clustering quality for PXD000561 dataset. Given incorrect clustering ratio, higher clustered spectra ratios indicate superior clustering performance. The region of interest is an incorrect clustering ratio of less than 2%.

4) BASELINE DESIGNS

For clustering, we evaluate our approach against four SoA tools, including Falcon [18], msCRUSH [19], HyperSpec [6], and SpecHD [24]. For DB search, we compare our system against ANN-SoLo [5] on GPU and HyperOMS [7] on GPU, and IMC accelerators including RRAM-based [10] and 3-D NAND-based [12]. All DB search results are evaluated at a fixed 1% FDR threshold, the same as existing works. The baseline systems are tested on NVIDIA GeForce RTX 4090 GPU with 24 GB VRAM and Intel Core i7-11700K CPU with 64 GB of RAM.

B. EXPERIMENT RESULTS

1) SEARCH QUALITY

Fig. 9 shows that the SpecPCM outperforms existing tools such as Falcon [18] and msCRUSH [19], while delivering performance comparable to HyperSpec [6] across all configurations, including single-level-cell (SLC), 2-bit MLC (MLC2), and 3-bit MLC (MLC3). With an incorrect clustering ratio of up to 1.5%, SpecPCM achieves around 60% clustered spectra ratio. Compared to SLC, the performance reduction in both MLC2 and MLC3 is minimal, indicating that the dimension-packing method has a negligible impact on accuracy.

In DB search, we validate the functionality of SpecPCM by comparing it against existing tools. Fig. 10 illustrates the total number of peptides identified using the large-scale HEK293 dataset, and Fig. S1 visualizes the identified data points for the specific query b1931 in a Venn diagram. SpecPCM demonstrates higher search quality compared to SpectraST [25] and comparable performance to HyperOMS [7]. Although ANN-SoLo identifies the highest number of peptides, it comes at the cost of significantly higher power consumption and latency. While the proposed SpecPCM provides balanced accuracy, parameters such as HD dimensions and others can be further adjusted to enhance search quality at the cost of increased energy and latency (see Supplementary Fig. S3(a) and (b)].

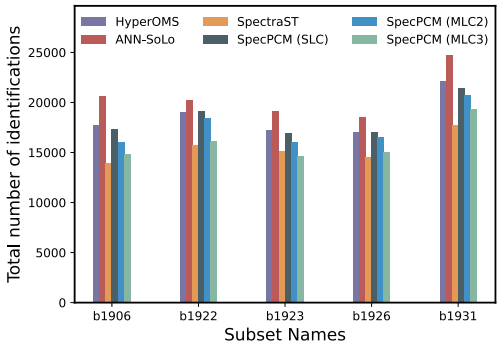

FIGURE 10. DB search quality for the HEK293 dataset, showing the number of identified peptides for each subset.

2) SPEEDUP AND ENERGY EFFICIENCY

SpecPCM achieves up to 82× speedup in clustering and up to 143× speedup in DB search compared to traditional CPU and GPU tools on large datasets as shown in Tables 2 and 3, respectively. This performance gain is attributed to the reduced data movement in SpecPCM, which is the dominant overhead in traditional CPU/GPU architectures. Additionally, SpecPCM fully leverages MLC technology due to the dimension packing. This packing enables SpecPCM to leverage the benefits of 3-bit MLC, boosting computation throughput by 3× compared to the SLC implementation. The energy consumption of SpecPCM is 3.27 J on the PXD00561 dataset for the entire clustering process and 0.149 J for the DB search to process a subset of data, which requires 46 665 query processing on average for the HEK293 dataset. Given that GPU-based tools typically operate at an average power of 450 W with longer delays, SpecPCM is expected to provide an energy efficiency improvement of four orders of magnitude.

When compared to prior MS systems based on 3-D NAND [12] and RRAM [10], SpecPCM achieves a speedup of 2.96× to 24.9× on the DB search task as shown in Table 3. While RRAM shows strong potential, it suffers from high voltage requirements for programming and increased noise, leading to larger peripheral circuits and a limited number of activated rows in the array at a time. In contrast, NAND flash has slower read latency compared to PCM due to its serial array structure.

3) ACCURACY AND EFFICIENCY TRADE-OFFS

Our PCM-based IMC accelerator allows the ISA-based control of following parameters that impact the performance. In this section, we explore various combinations of these parameters to analyze their impacts on the quality of MS analysis.

1) *Bits Per Cell:* Increasing the number of bits per cell enhances memory and compute density. However, this leads to higher error rates, degrading the MS performance as illustrated in Figs. 9 and 10. In clustering, the accuracy decreases from 60.57% with SLC to 59.80% and 59.54% with 2-bit and 3-bit MLC, respectively, at an incorrect clustering ratio of 1.5%. In contrast, the DB search shows a more pronounced drop in quality, highlighting its greater sensitivity to noise.
2) *HD Dimension:* Higher HD dimensions generally enhance performance at the cost of linearly increased storage and processing delay and energy (Supplementary Figs. S4 and S5).
3) *Write-Verify Cycles:* DB search operations are predominantly read-oriented, allowing the cost of write-verify operations on reference HVs to be effectively amortized. This means that a higher number of write-verify cycles can be used to ensure accurate storage. In contrast, spectral clustering involves frequent write operations to update the distance matrix. Thus, increasing the number of write-verify linearly increases the latency and energy consumption. Fortunately, the performance in terms of clustered spectra ratio remains largely unaffected by the number of write-verify cycles, as shown in Supplementary Fig. S3(a). As a result, no write-verify cycles are used for clustering in this work.
4) *ADC Resolution:* Since HD vectors have positive and negative values with an equal probability, partial sums are likely to approach near-zero values. In addition, HD computing has a high inherent error resiliency. For this reason, the performance gracefully degrades as ADC bit precision reduces. Therefore, a 4-bit flash ADC can achieve roughly 4× less area and energy compared to a 6-bit flash ADC at the marginally degraded accuracy [Supplementary Fig. S3(b)].

By tuning the above parameters using the provided instruction sets, we can optimize the performance and efficiency of the proposed SpecPCM, highlighting its flexibility and adaptability given the target specification.

V. CONCLUSION

This work presents a full-stack effort for a PCM-based analog IMC platform that significantly enhances energy and delay efficiency for MS analysis, addressing both clustering and DB search. In consideration of clustering, which requires frequent memory updates, we utilize superlattice-material-based PCM for low-voltage programming and employ different PCM materials to optimize the distinct requirements of clustering and DB search. At the algorithm level, we introduce dimension packing to compress binary data into MLCs, maximizing storage and IMC processing efficiency. Additionally, we leverage a HD platform for its strong error tolerance and parallel processing capabilities with simple operations. The proposed ISA enables software-driven trade-offs between accuracy and efficiency by controlling various hardware and algorithmic parameters. Experimental results based on a 40-nm CMOS process demonstrate that SpecPCM achieves speedups of up to 82× for clustering and 143× for DB search, alongside a four-orders-of-magnitude improvement in energy efficiency while maintaining accuracy. These full-stack efforts underscore the potential for greater efficiency through automated compiler-based optimizations and pave the way for future work in many data-intensive tasks, overcoming the nonidealities of various emerging technologies.

REFERENCES

[1] M. Wilhelm et al., "Mass-spectrometry-based draft of the human proteome," *Nature*, vol. 509, no. 7502, pp. 582–587, May 2014.

[2] *Massive: Mass Spectrometry Interactive Virtual Environment*, UCSD, San Diego, CA, USA, 2024.

[3] H. Lam, "Building and searching tandem mass spectral libraries for peptide identification," *Mol. Cellular Proteomics*, vol. 10, no. 12, Dec. 2011, Art. no. R111.008565.

[4] J. Griss et al., "Recognizing millions of consistently unidentified spectra across hundreds of shotgun proteomics datasets," *Nature Methods*, vol. 13, no. 8, pp. 651–656, Aug. 2016.

[5] I. Arab, W. E. Fondrie, K. Laukens, and W. Bittremieux, "Semisupervised machine learning for sensitive open modification spectral library searching," *J. Proteome Res.*, vol. 22, no. 2, pp. 585–593, Feb. 2023.

[6] W. Xu, J. Kang, W. Bittremieux, N. Moshiri, and T. Rosing, "HyperSpec: Ultrafast mass spectra clustering in hyperdimensional space," *J. Proteome Res.*, vol. 22, no. 6, pp. 1639–1648, Jun. 2023.

[7] J. Kang, W. Xu, W. Bittremieux, N. Moshiri, and T. Rosing, "Accelerating open modification spectral library searching on tensor core in high-dimensional space," *Bioinformatics*, vol. 39, no. 7, p. 404, Jul. 2023.

[8] N. R. Shanbhag and S. K. Roy, "Comprehending in-memory computing trends via proper benchmarking," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Apr. 2022, pp. 1–7.

[9] W. Wan et al., "A compute-in-memory chip based on resistive random-access memory," *Nature*, vol. 608, no. 7923, pp. 504–512, Aug. 2022.

[10] K. Fan, W.-C. Chen, S. Pinge, H.-S.-P. Wong, and T. Rosing, "Efficient open modification spectral library searching in high-dimensional space with multi-level-cell memory," in *Proc. 61st ACM/IEEE Design Autom. Conf.*, Jun. 2024, pp. 1–6.

[11] W. Wan et al., "A voltage-mode sensing scheme with differential-row weight mapping for energy-efficient RRAM-based in-memory computing," in *Proc. IEEE Symp. VLSI Technol.*, Jun. 2020, pp. 1–2.

[12] P.-K. Hsu, W. Xu, T. Rosing, and S. Yu, "An in-storage processing architecture with 3D NAND heterogeneous integration for spectra open modification search," in *Proc. Int. Symp. Memory Syst.*, Oct. 2023, pp. 1–7.

[13] X. Wu et al., "Novel nanocomposite-superlattices for low energy and high stability nanoscale phase-change memory," *Nature Commun.*, vol. 15, no. 1, p. 13, Jan. 2024.

[14] G. Karunaratne, M. Le Gallo, G. Cherubini, L. Benini, A. Rahimi, and A. Sebastian, "In-memory hyperdimensional computing," *Nature Electron.*, vol. 3, no. 6, pp. 327–337, Jun. 2020.

[15] J. Kang, B. Khaleghi, T. Rosing, and Y. Kim, "OpenHD: A GPU-powered framework for hyperdimensional computing," *IEEE Trans. Comput.*, vol. 71, no. 11, pp. 2753–2765, Nov. 2022.

[16] W. Xu, J. Kang, and T. Rosing, "FSL-HD: Accelerating few-shot learning on ReRAM using hyperdimensional computing," in *Proc. Design, Autom. Test Eur. Conf. Exhibit. (DATE)*, Apr. 2023, pp. 1–6.

[17] J. E. Elias and S. P. Gygi, "Target-decoy search strategy for increased confidence in large-scale protein identifications by mass spectrometry," *Nature Methods*, vol. 4, no. 3, pp. 207–214, Mar. 2007.

[18] W. Bittremieux, K. Laukens, W. S. Noble, and P. C. Dorrestein, "Large-scale tandem mass spectrum clustering using fast nearest neighbor searching," *Rapid Commun. Mass Spectrometry*, p. e9153, Feb. 2021.

[19] L. Wang, S. Li, and H. Tang, "MsCRUSH: Fast tandem mass spectral clustering using locality sensitive hashing," *J. Proteome Res.*, vol. 18, no. 1, pp. 147–158, Dec. 2018.

[20] J. Kang, W. Xu, W. Bittremieux, N. Moshiri, and T. Š. Rosing, "DRAM-based acceleration of open modification search in hyperdimensional space," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 43, no. 9, pp. 2592–2605, Sep. 2024.

[21] J. M. Chick et al., "A mass-tolerant database search identifies a large proportion of unassigned spectra in shotgun proteomics as modified peptides," *Nature Biotechnol.*, vol. 33, no. 7, pp. 743–749, Jul. 2015.

[22] M. Kim et al., "A draft map of the human proteome," *Nature*, vol. 509, no. 7502, pp. 575–581, May 2014.

[23] R. J. Chalkley et al., "Proteome informatics research group (iPRG)_2012: A study on detecting modified peptides in a complex mixture," *Mol. Cellular Proteomics*, vol. 13, no. 1, pp. 360–371, Jan. 2014.

[24] S. Pinge et al., "SpecHD: Hyperdimensional computing framework for FPGA-based mass spectrometry clustering," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2024, pp. 1–6.

[25] C. W. M. Ma and H. Lam, "Hunting for unexpected post-translational modifications by spectral library searching with tier-wise scoring," *J. Proteome Res.*, vol. 13, no. 5, pp. 2262–2271, May 2014.

[26] N. Selevsek et al., "Reproducible and consistent quantification of the Saccharomyces cerevisiae proteome by SWATH-mass spectrometry," *Mol. Cellular Proteomics*, vol. 14, no. 3, pp. 739–749, Mar. 2015.

[27] M. Wang, J. Wang, J. Carver, B. S. Pullman, S. W. Cha, and N. Bandeira, "Assembling the community-scale discoverable human proteome," *Cell Syst.*, vol. 7, no. 4, p. 412, Oct. 2018.

[28] M. Saberi, R. Lotfi, K. Mafinezhad, and W. A. Serdijn, "Analysis of power consumption and linearity in capacitive digital-to-analog converters used in successive approximation ADCs," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 8, pp. 1736–1748, Aug. 2011.

[29] X. Wu et al., "Understanding interface-controlled resistance drift in superlattice phase change memory," *IEEE Electron Device Lett.*, vol. 43, no. 10, pp. 1669–1672, Oct. 2022.

VI. Supplementary Material

S.A Dataset

All spectra are pre-processed following the existing methodology in [6], [7]. All MS data, spectral libraries, pre-processed spectra, and identification results are available for download from the MassIVE repository under the dataset identifier MSV000091183 [2].

For clustering, the following two datasets are employed: 1) a small-scale dataset, PXD001468 [21], which contains 1.1 million spectra from kidney cell samples with a total size of 5.6 GB, and 2) a large dataset, PXD000561 [22], which contains a draft map of the human proteome with 21.1 million spectra totaling 131 GB. For DB search, we use: 1) the small-scale iPRG2012 [23] as the query (total spectra: 15,867) and the human HCD yeast library [26] as the reference database (total spectra: 1,162,392), and 2) the larger-scale HEK293 (Human Embryonic Kidney 293) [27], which includes multiple different subsets b1906∼ 1931 (total spectra per subset: 46,665 on average), as the query and the human spectral library [28] (total spectra: 2,992,672) as the reference library.

S.B Experiment setup

Hardware Configurations: For the ASIC blocks, we initially implement the logic in C++ and subsequently generate the RTL code using high-level synthesis (HLS). The RTL code is then synthesized using Cadence Genus with the CMOS 40 nm process PDK to meet the target clock frequency of 500 MHz. The ASIC area for the encoder is 44 μm^2 and 69 μm^2 for other components, both of which are negligible (less than 0.5%) compared to that of the memory arrays. Each 2T2R cell has an area of $0.5\mu m^2$. We use the power and area data of DAC in [29]. For all the other blocks, the schematic and layout are designed using 40 nm CMOS technology in Cadence Virtuoso, and their corresponding power and area are measured from the post-layout simulations. The component-level energy and area are summarized in Table S3. Most operations of the components listed in Table S3 complete within one cycle, whereas the programming of a PCM array takes 20 ns (10 cycles).

PCM device: PCM devices in this work have TiN bottom electrodes with 40 nm diameter (Fig. S2) [13]. Phase change superlattices are deposited using magnetron sputtering (detailed fabrication process can be found in [13]). Device noise and resistance drift are measured based on protocols described in [30]. Resistance drift coefficient is extracted from resistance vs. time measurement using power-law model.

Noise model: We fit PCM resistance measurements to a normal distribution to derive the standard deviation (σ). This normal distribution with σ models the noise effect on the weights stored in the memory. In the simulation, noise adjustment is applied to validate the impact of the noise on the performance, e.g., $\hat{W} = W \times (1 + \eta)$, where $\eta \sim \mathcal{N}(0, \sigma^2)$. Here, W represents the error-free stored value, and $\hat{W}$ denotes the erroneous read value, with $\mathcal{N}$ is a normal distribution.

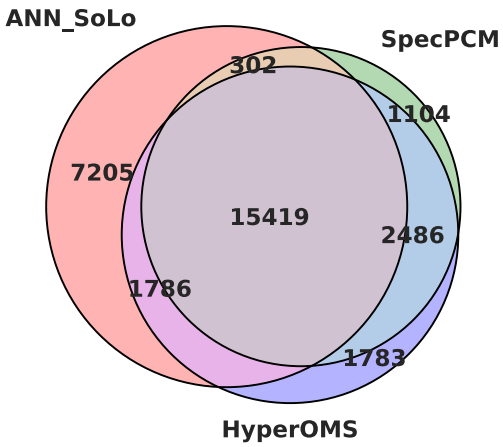

FIGURE S1: DB search result: an example of identified peptides in Venn diagram with MLC3 for the dataset HEK293 b1931. The majority of peptides detected by SpecPCM can also be found by other tools, indicating reliability of the SpecPCM results.

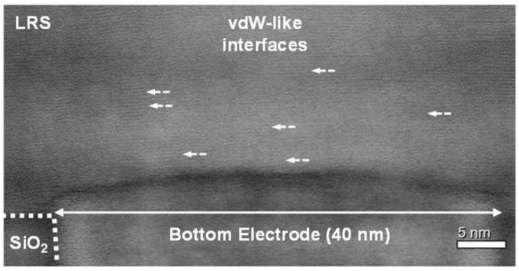

FIGURE S2: Transmission Electron Microscopy (TEM) image of a $TiTe_2$/$Ge_4Sb_6Te_7$ device used in this work, showing 40 nm bottom electrode and superlattice-like phase change materials.

TABLE S1: Measured parameters of PCM device technology.

Technology	Sb_2Te_3 / $Ge_4Sb_6Te_7$	$TiTe_2$ / $Ge_4Sb_6Te_7$
Programming current (μA)	80	160
Programming voltage (V)	0.7	0.9
Programming energy (pJ)	1.12	2.88
Retention at 105°C (hour)	30	$> 10^5$
Low resistance state (kOhm)	30	10
Resistance on/off ratio	150	100

TABLE S2: Instruction Set Architecture (ISA) for IMC control.

Instructions	Description
STORE_HV (data, arr_idx, col_addr, row_addr, MLC_bits, write_cycles)	PCM[arr_idx, col_addr, row_addr] ← data. **write_cycles** defines the number of write verify cycles **MLC_bits** defines the number of bits used by dimension packing for MLC
READ_HV (data_size, arr_idx, col_addr, row_addr, MLC_bits)	buffer← PCM[arr_idx, col_addr, row_addr] **MLC_bits** is configured same as above
MVM_COMPUTE (row_addr, num_activated_row, ADC_bits, MLC_bits)	Compute distance through Matrix-Vector Multiplication (MVM) at PCM[row_addr] **num_activated_row** defines size of activated weight matrix **ADC_bits** defines the resolution of the Flash ADC

TABLE S3: Power and area of components at 40 nm CMOS process.

Component	Unit Power (μW)	Unit Area (μmm^2)	Total Power (mW)	Total Area (mm^2)
PCM Array	0.22	0.5	3.58	0.0082
Flash ADC	320	920	5.12	0.0147
DAC	6.56	32	0.84	0.0041
SL Gen / Drive	52.5	72.47	3.36	0.0046
Read Gen	-	-	0.51	0.0018
WL Decode / Drive	4.05	10.68	1.04	0.0027
Sense Amp	20	75.9	0.64	0.0024
Selectors	-	-	0.50	0.0017
Total	**-**	**-**	**15.59**	**0.0402**

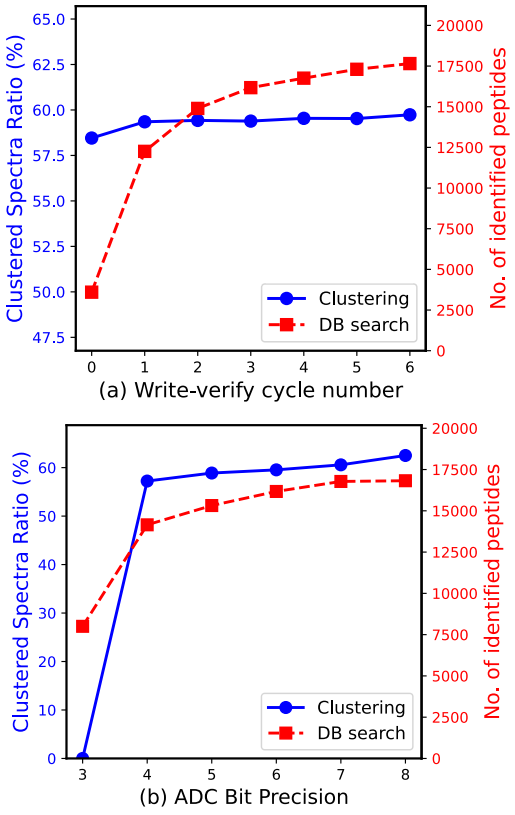

FIGURE S3: Accuracy and efficiency trade-offs. (a) quality vs. write-verify cycle number and (b) quality vs. ADC bit precisions.

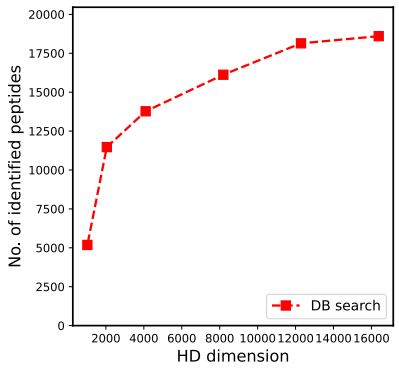

FIGURE S4: DB search quality vs. HD dimension.

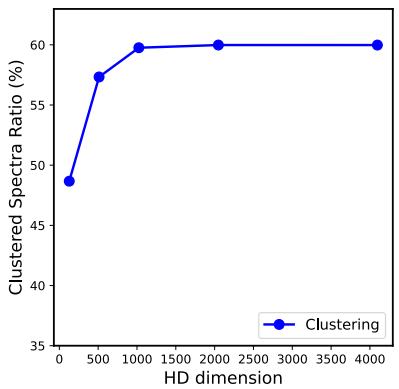

FIGURE S5: Clustering quality vs. HD dimension.