

Approximate Image Storage with Multi-level Cell STT-MRAM Main Memory

Hengyu Zhao*, Linuo Xue*¹, Ping Chi², Jishen Zhao

Department of Computer Engineering, University of California, Santa Cruz, CA 95064, USA

¹Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106, USA

²Intel Corporation, Santa Clara, CA 95054, USA

{hzhao28, jishen.zhao}@ucsc.edu, ¹linuo_xue@umail.ucsb.edu, ²ping.chi@intel.com

Abstract—Images consume significant storage and space in both consumer devices and in the cloud. As such, image processing applications impose high energy consumption in loading and accessing the image data in the memory. Fortunately, most image processing applications can tolerate approximate image data storage. In addition, multi-level cell spin-transfer torque MRAM (STT-MRAM) offers unique design opportunities as the image memory: the two bits in the memory cell require asymmetric write current – the soft bit requires much less write current than the hard bit. This paper proposes an approximate image processing scheme that improves system energy efficiency without upsetting image quality requirement of applications. Our design consists of (i) an approximate image storage mechanism that strives to only write the soft bits in MLC STT-MRAM main memory with small write current and (ii) a memory mode controller that determines the approximation of image data and coordinates across precise/approximate memory access modes. Our experimental results with various image processing functionalities demonstrate that our design reduces memory access energy consumption by 53% and 2.3× with 100% user’s satisfaction compared with traditional DRAM-based and MLC phase-change-memory-based main memory, respectively.

I. INTRODUCTION

Images consume increasingly large amount of storage space in various computer systems, whether they be cloud servers, mobile, or embedded. As such, modern image processing applications consume substantial energy on data movement between the main memory and the processor. The emergence of new applications and camera products with more powerful image recording and processing capabilities can further enforce this trend, by further pushing the system energy efficiency bottleneck from computation to memory access. Therefore, one of the most critical challenges in image processing applications is how to manipulate image data in the memory in an energy efficient manner.

In order to mitigate the energy efficiency challenge, prior studies explored image compression techniques (e.g., JPEG) and approximate storage techniques that reduce the amount of data movement. These techniques trade-off image quality with image data size. Image compression techniques are lossy, yet applied in a deterministic way such that all loss happens at encoding stage. Approximate storage techniques relax the costly reliability requirement by allowing occasional errors. However, image compression can impose substantial energy consumption in CPUs and processor-memory data movement to compress and decompress the image data. As such, image compression is typically performed in the storage devices. Most prior approximate image storage techniques are limited to the types of images where the signal inputs are already noisy, such as images generated by image sensors.

*These two authors contributed equally to this paper as first authors.
This work is supported by NSF 1652328 and NSF I/UCRC Center.

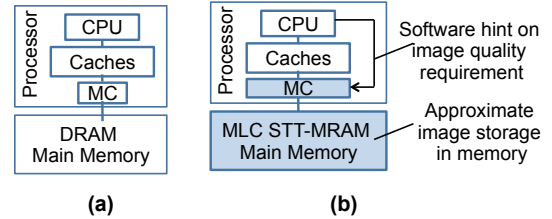


Fig. 1. System configurations with (a) traditional DRAM-based main memory and (b) MLC STT-MRAM main memory (shaded components are integrated with our hardware modifications).

The scaling of main memory process technology makes the energy efficiency issues of main memory access even worse. The continued scaling of DRAM process technology has enabled higher density DRAM by placing smaller memory cells close to each other. Unfortunately, the close proximity of cells exacerbates cell-to-cell interference, making cells susceptible to failures [1]. Furthermore, the scaling of DRAM technology imposes high leakage energy. For example, the leakage of commodity DRAM can impose up to 40% of the total memory energy consumption.

Nonvolatile RAMs, such as spin-transfer torque MRAM (STT-MRAM) [2] and phase-change memory (PCRAM) [3], offer promising opportunities as alternative technologies to implement main memory. They typically have near-zero leakage energy and have the potential to be scaled up to much higher density than DRAM [2]. Furthermore, these nonvolatile RAM technologies can be implemented with multi-level cells (MLCs), where a single memory cell can store two or more bits. As such, main memory implemented by MLC STT-MRAM and MLC PCRAM can offer several times higher capacity than traditional memory designs.

The **goal** in our paper is to achieve energy-efficient data movement in image processing applications, by leveraging MLC STT-MRAM technology. To achieve our goal, we design a set of approximate image storage and data access mechanisms with MLC STT-MRAM main memory. We do not plan to put forward any compression method to improve energy efficiency, because the introducing of compression schemes will increase the overhead [4].

Prior studies explored MLC PCRAM and single-level cell (SLC) STT-MRAM based approximate memory techniques [5], [6], [7]. Yet compared with these memory technologies, MLC STT-MRAM offers several unique properties that make more promising for implementing approximate main memory to store image data. First, MLC STT-MRAM typically has much lower dynamic energy and access latency compared with MLC PCRAM technologies. Furthermore, MLC PCRAM has even longer write latency and higher write energy, due to its program-and-verify process (i.e., iteratively applies partial set pulse and verifies if a specified precision criterion is met) in write

operations [6]. Finally, 2-bit MLC STT-MRAM cells have a unique property – writing to the two bits (a soft bit and a hard bit) can be performed by asymmetric write current. This used to be considered as an issue in MLC STT-MRAM design [8], [9]. However, we found it a property that can be leveraged to reduce the energy consumption in image data access with our design.

Our design consists of two components. First, we design an approximate image memory mechanism, which strives to only write to the soft bits in the MLC STT-MRAM cells by writing an approximate image into the main memory. Our mechanism determines the level of acceptable approximation based on application’s hint provided by our software interface. Second, we develop an approximate mode controller integrated with the memory controller. The mode controller coordinates among precise and approximate reads/writes to the main memory. In particular, this paper makes the following contributions.

- We design an approximate image storage mechanism that substantially reduces the write access in the main memory. Our mechanism also leverages the asymmetric write current in MLC STT-MRAM cells to reduce the energy consumption of image data access.
- We design a memory access mode control mechanism that coordinates across precise and approximate image data access in the memory.
- We develop a set of lightweight hardware implementations in MLC STT-MRAM banks and the memory controller to facilitate our approximate image processing scheme. We also develop a software interface for applications to communicate their image quality requirement to our design mechanisms.
- We compare our design with various memory designs with DRAM and PCRAM. We also evaluate the yielded image quality based on a user’s satisfaction model. We show that our design significantly improves system energy efficiency, while meeting user’s satisfaction.

II. BACKGROUND

In this section, we introduce the background of approximate computing and STT-MRAM technologies.

A. Approximate Techniques

Approximate computing and storage techniques are built based on the observation that many applications, such as image processing, can tolerate certain levels of errors in both data storage and computation [10], [11], [12], [13], [14], [6]. The goal of approximate computing techniques is to improve system performance and reduce energy consumption. Approximate storage techniques may also target at reducing storage capacity of the data, beyond performance and energy improvement. Approximate techniques leverage the computation and data storage patterns of applications to ensure that applications are highly likely to produce a result that is sufficiently close to accurate results. In general, approximate techniques can be categorized into two classes: coarse-grained approximate technique and fine-grained approximate technique. In particular, our design explores fine granularity approximate storage techniques in the main memory, in order to improve system performance and energy consumption.

Coarse-grained approximate techniques. An approximate technique is coarse-grained, if it affects entire blocks, functions, or data structures. As such, coarse-grained approximate computing may execute only a the subset of programs, while approximate image storage techniques can apply a global approximation to the entire images. One typical example of coarse-grained approximate techniques is *loop perforation* [15], which can transform loops to execute a subset of their iterations.

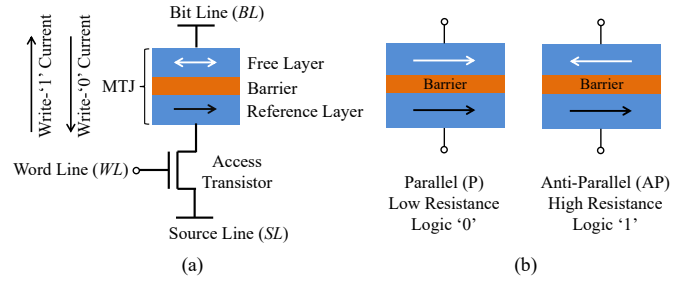


Fig. 2. Single-level cell (SLC) STT-MRAM. (a) 1T1J SLC STT-MRAM cell structure; (b) in-plane MTJs in low and high resistance states.

Fine-grained approximate techniques. Fine-grained approximate techniques apply to individual memory locations, instructions, or data packets. Such approximation can be very general – they can potentially apply to any computation or memory locations in a program [16]. Recently proposed MLC PCRAM-based approximate image storage [6] is an example of approximate storage techniques in this category.

B. SLC STT-MRAM

STT-MRAM is a type of emerging nonvolatile memory technology. Figure 2(a) shows a conventional one-transistor-one-MTJ (1T1J) cell structure. The resistance of an MTJ is determined by the relative magnetization directions of the reference layer and the free layer. When they are in parallel, the MTJ is in a low resistance state and represents logic “0”; when they are anti-parallel, the MTJ is in a high resistance state and denotes logic “1”, as shown in Figure 2(b). Writing “0” or “1” is determined by the direction of the applied switching current, as illustrated in Figure 2(a). Each STT-MRAM cell only stores one logic bit. Therefore, such STT-MRAM technologies are called single-level cell (SLC) STT-MRAM.

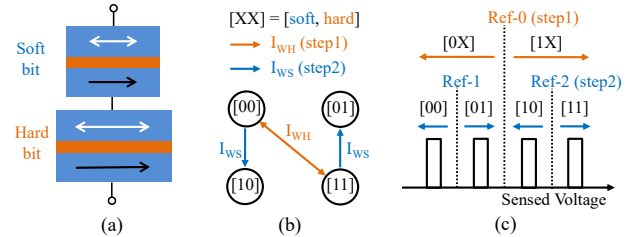


Fig. 3. Multi-level cell STT-MRAM. (a) Series MLC MTJ structure; (b) Two-step write operation; (c) Two-step read operation.

C. MLC STT-MRAM

Multi-level cell (MLC) STT-MRAM cells store multiple logic bits (typically two bits) in each cell, increasing the density of STT-MRAM (Figure 3). There are two categories of MLC MTJ designs, series [17] and parallel [18]. Series MLC STT-MRAM has two MTJs stacked together to achieve four resistance states by the combinations of their magnetization directions, as shown in Figure 3(a). In parallel MLC MTJs, the free layer has two domains to represent two logic bits. Our design leverages the series MLC STT-MRAM. Series MLC STT-MRAM has been demonstrated to be more feasible than the parallel implementation, because series design is compatible with advanced MTJ technologies, such as perpendicular MTJ and has overwhelming advantages in read and write reliability. Instead, the parallel implementation is only applicable to in-plane MTJ technology [19].

Read and write operation of MLC STT-MRAM cell. In Figure 3(a), the two MTJs of series MLC MTJ structure have different

areas to distinguish the two logic bits. The bit stored in the smaller MTJ is a *soft bit*; the bit in the bigger MTJ is a *hard bit*. Given a constant resistance-area product and a critical switching current density (I_C), the soft bit has a higher resistance than the hard bit. As such the soft bit is typically the more significant bit (MSB), and it requires a smaller switching current, i.e. $I_{C,soft} < I_{C,hard}$. The write and read operations of series MLC STT-MRAM are illustrated in Figure 3(b) and (c), both of which contain two steps. As an example, in a write operation, in the first step a large current I_{WH} ($I_{WH} > I_{C,hard}$) is applied to switch the hard bit; at the same time the soft bit gets the same value as the hard bit; in the second step a smaller current I_{WS} ($I_{C,soft} < I_{WS} < I_{C,hard}$) is used to flip only the soft-bit. Figure 3(c) illustrates a two-step read operation based on voltage sensing, which requires three reference voltages (Ref-0, Ref-1, and Ref-2) and two comparisons. In the first step, the soft bit is detected by comparing the sensing voltage with Ref-0. In the second step, the hard bit is read by comparing the sensing voltage with either Ref-1 or Ref-2 based on the result of the first step.

Issues and opportunities with MLC STT-MRAM cell. Due to the two-step read and write operations, MLC STT-MRAM has even longer read and write latencies and consumes higher read and write energy than SLC STT-MRAM. This can degrade system performance and energy efficiency, despite MLC STT-MRAM's density benefits. However, we observe that, reading the soft bit only takes one step, while writing the soft bit only requires a small switching current that will not flip the hard-bit. Thus, MLC STT-MRAM can perform in a similar way as SLC by only accessing the soft-bits, which improves both access speed and the energy efficiency. Furthermore, this also results in value depended write current in MLC STT-MRAM cells: If both bits in a cell need to be changed and the new value is either "00" or "11", only a one-step write with large write current is necessary; Otherwise, if the new values of the soft bit and the hard bit are different from each other (i.e., writing "01" or "10"), the aforementioned two-step write is required. Such opportunities motivate us to explore the use of MLC STT-MRAM in approximate image storage, because the two bits in each cell can have the same value can be written at one time.

MLC STT-MRAM as main memory. STT-MRAM has been considered as a promising replacement of DRAM in main memory system design. Compared with DRAM, STT-MRAM has many advantages, such as lower leakage power and no radiation-induced soft errors. With the MLC technology, the density can be doubled compared with SLC STT-MRAM. MLC PCRAM also offers high density and low leakage benefits. However, the technology impose much longer access latency and higher dynamic energy than DRAM and MLC STT-MRAM. Furthermore, the writing mechanisms of MLC PCRAM cells do not have the unique properties as MLC STT-MRAM has. As such, we adopt MLC STT-MRAM as our main memory technology in our design.

III. OUR DESIGN

The goal of our design is to reduce the energy consumption and improve the performance of memory access in image processing applications. To achieve our goal, we propose an approximate image storage and access scheme, by leveraging the aforementioned unique properties of MLC STT-MRAM technology. Our scheme is developed on top of two design principles. First, our approximate image memory mechanism reduces the number of write operations and the write current required to store image data in the main memory. To this end, our mechanism skips the writes of neighbor pixels that are sufficiently similar (e.g., the differences of their R, G, B values are smaller than

a given threshold determined by user's satisfaction metrics). Second, we adopt four memory access modes to improve the energy efficiency of MLC STT-MRAM access by exploiting application's memory access behaviors. The hardware support and software interface of our design will be discussed in Section IV.

A. Approximate Image Storage in Memory

A color digital image typically consists of a matrix of pixels, which are made of combinations of primary colors represented by a series of digital code. For instance, a pixel from a standard digital camera image can have red (R), green (G), and blue (B) color codes (also defined as "channels"). Each channel – R, G, or B – is typically represented by a 8-bit binary number, i.e., each pixel is stored as three consecutive bytes in the memory. Pixels in the same row of an image are typically stored in consecutive memory locations. As such, storing or modifying a pixel typically requires to write a 3-byte number into the memory, consuming substantially high energy with large images or frequent writing.

In order to reduce the energy consumption of storing and updating images in the main memory, we propose an approximate image storage mechanism. Figure 1(b) shows our system configuration, which adopt a MLC STT-MRAM as the main memory. By exploiting the similarity across neighbor pixels in images, our design only requires to write one soft bit of one MLC STT-MRAM cell among the neighbor pixels with sufficiently similar R, G, B values.

Approximate image memory algorithm. Figure 4 illustrates the key idea of our approximate image storage mechanism. Assuming four neighbor pixels A, B, C, and D (Figure 4(a)) have similar channel code values (determined by the similarity policy described below), precise image storage would require that the code of all pixels are stored in the MLC STT-MRAM array. Figure 4(b) shows an example mapping of the pixels to a memory bank. Even though the pixel values are similar, we are still highly likely to write different bit values, "01" or "10", to the same memory cells. Each of such writes require two write operations, leading to high energy consumption. Yet, users may not be aware of the small differences between these pixels. Furthermore, image processing applications do not necessarily access all pixels of an image [6]; writing accurate values of each pixel can be a waste of energy.

Our approximate image storage mechanism only stores the accurate code of one representative pixel (e.g., the first pixel stored in the memory), while allowing its neighbor pixels to have a copy of the same values. As shown in Figure 4(c), we only store the code of pixel A into the soft bits of each MLC STT-MRAM cell with small write current, when the image is initially written or updated in the memory. We reserve the bits for the other three pixels, but do not write them. We update these neighbor pixels, when they are read by applications, by copying the values from pixel A. Such value copies only require an additional single write operations to write "00" or "11" values in memory cells. Furthermore, we perform the copy operation with memory bank's local sense amplifiers and write drivers without involving the CPU (Section IV). As such, we also proactively perform the copies before the pixels are read by applications, while the particular memory bank is idle. This effectively hide the latency of the copy operations.

Leveraging the asymmetric write current of MLC STT-MRAM.

MLC STT-MRAM cells have the write disturbance issue caused by the asymmetric MTJ sizes in a cell [8], [9] – the soft bit has to be restored after writing to the hard bit (Section II). With our design we either (i) write the soft bit or (ii) write both the soft and the hard bit in the cell. This effectively hides the disturbance issue.

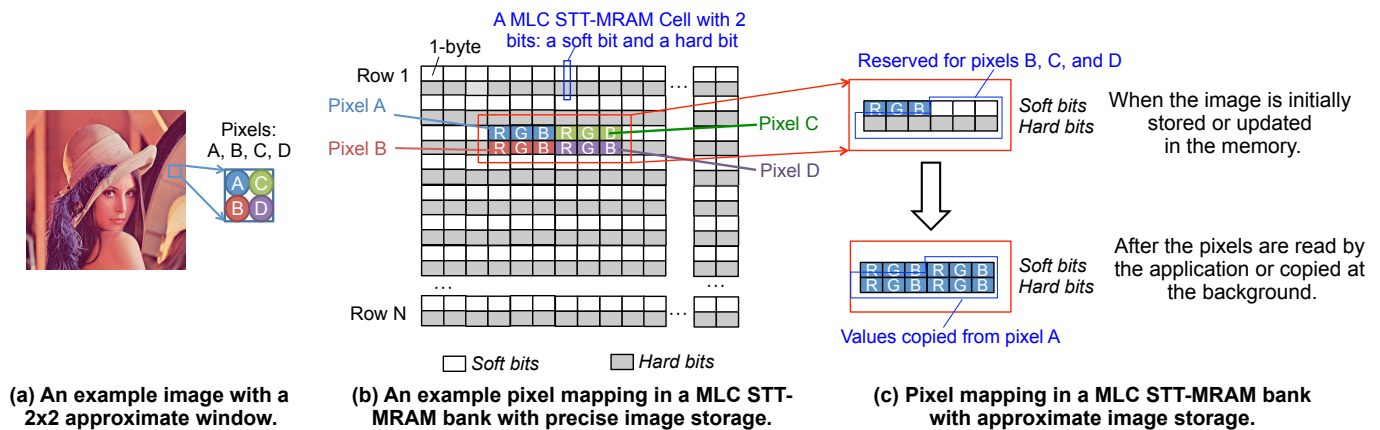


Fig. 4. Approximate image storage mechanism.

Determining the similarity of neighbor pixels. We allow image processing applications to provide hints on the levels of allowed approximation in images. The hints will indicate image quality requirements by using our software interface (Section IV). Based on the image quality requirement, we calculate the threshold (θ), which is the maximum allowed difference among R, G, B values between similar neighbor pixels. For example, we define the R, G, B values of two neighbor pixels A and B are $R_A, G_A, B_A, R_B, G_B, B_B$, respectively. Pixels A and B are similar, if

We also define approximate window as the matrix of all similar neighbor pixels. For example, the approximate window, which consists of pixels A, B, C, and D in Figure 4(b), has a size of 2×2 .

In order to achieve energy-efficient memory access, we adopt four memory access modes controlled by a mode controller integrated with the memory controller.

representative pixel (“00” or “11” will be written back to each cell). Step-ii only requires single write operations, because both bits in each memory cell have the same value. Second, we can also proactively perform the copies whenever the memory bank is idle, because both steps can be performed with local sense amplifiers and bus drivers without involving CPU. Most traditional image encoding/compression algorithms (e.g., JPEG) requires decoding operations during processing of the image data. Yet, our copy mode does not require the CPU to decode the image, hence can consume much lower energy and achieve higher performance.

In this section, we discuss hardware and software implementation details of our design.

Figure 5 illustrates an MLC STT-MRAM bank incorporated with the components that implement our design mechanisms. We only add multiplexers to switch the data flow across different memory access modes. The rest of the memory bank organization stays the same as conventional MLC STT-MRAM designs. The data flow of precise write and read modes remains the same as normal memory access in conventional memory designs. Figure 5 shows the data flow of the other two modes. In approximate write mode, input data will be written into soft bits through bus drivers. In copy mode, the soft bit values will be sensed in sense amplifiers. These values will then be written into both soft and hard bits via the bus drivers.

We implement a mode controller integrated with the memory controller in the processor. The mode controller determines the modes of memory access and generates the mode control signals. As shown in Figure 6, the CPU writes application's image quality requirement (i.e., similarity thresholds of certain memory regions) into a set of special registers in the mode controller. Based on the information, the mode controller constructs a threshold table. During the application execution, the mode controller determines the mode of each memory request based on the logic control flow as shown in Figure 6.

Our design determines the pixel similarity thresholds, based on application’s hints on image quality requirement written into the image quality registers in the mode controller. To this end, we provide

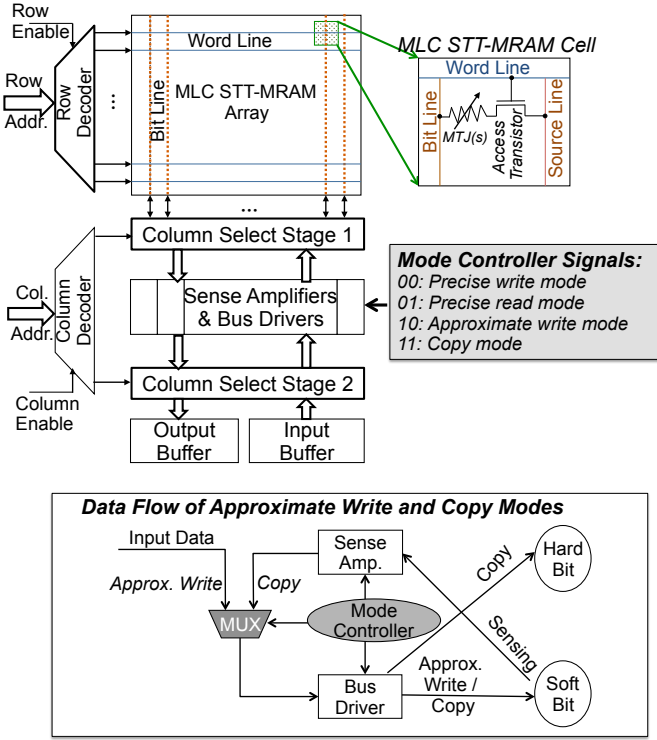


Fig. 5. MLC STT-MRAM bank organization and the data flow in approximate write and copy modes.

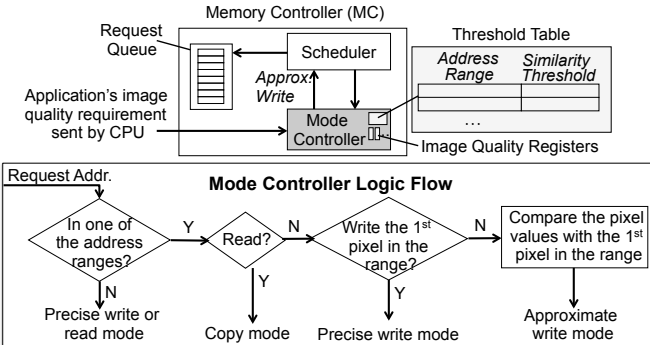


Fig. 6. Mode controller design integrated with the memory controller.

the following software interface for applications to deliver the image quality requirements during the execution:

```
load_image(quality_level)
```

where *quality_level* is an integer number. The function is used when the application initializes an image in the memory. Our design offers at most six image quality level options. As such, the *quality_level* will range from 1 to 6. These are corresponding to six levels of similarity thresholds. As shown in our evaluation results (Section V), the six levels of image quality requirements can yield a variety of image qualities as measured by standard image quality metrics such as signal-to-noise ratio (PSNR) and structural similarity (SSIM) [20], as well as memory write energy during application execution.

D. Discussion

Summary of hardware overhead. Hardware overhead of our design includes the added multiplexers in memory banks, the mode controller logic and storage, and signal wiring. In particular, the mode controller adopts a the threshold table with 16 bytes per entry – 8-byte address ranges and 8-byte thresholds.

User’s satisfaction model: an alternative image quality metric. An alternative mechanism to determine the image quality requirement is adopt inputs from users of the target image processing products. Instead of relying on application developers to provide the software hint, we can obtain the image quality requirement information by leveraging previous user’s satisfaction models [21]. Such models sample user’s response on example applications with various image qualities. The response is then feed into a statistical model to generate the most likely image quality requirement of target users. Such user’s satisfaction can also be used as a metric to evaluate the effectiveness of our approximate image storage mechanism (Section V).

Working with image storage encoding and compression mechanisms. Images are rarely stored in raw format long-term, e.g., in hard drives and SSD. Instead, they are typically encoded in compressed form to save space. Image compression techniques, such as JPEG, trade-off image quality against image size via quantization techniques [6]. Our design does not employ any image encoding or compression mechanisms applied to the in-memory image data. Therefore, our approximate image memory mechanism can be combined with existing image storage encoding and compression algorithms. For instance, applications can decode or decompress the image data, when loading the data from disks or flash. The image data is then stored in the MLC STT-MRAM main memory with our approximate image memory mechanism. MLC STT-MRAM has much higher density than traditional DRAM-based main memory. Therefore, our main memory offers much larger capacity for storing decompressed image data.

V. EVALUATION

In this section, we describe our experimental setup and report the evaluation results of our design.

A. Experimental Setup

We simulate image processing workloads to study the performance and energy consumption of our design. We also perform image quality analysis, and user’s satisfaction study to evaluate our design.

Workloads. We evaluate *VIPS*, which is an image processing benchmark in PARSEC benchmark suite [22]. *VIPS* incorporates a wide range of image processing functionalities, including filters, arithmetic operations, color processing, histograms, and geometric transforms. We select *VIPS* because of its state-of-the-art performance and wide range of supported encoding formats. In addition to the widely used image encoding formats, such as JPEG, TIFF, PNG, SVG, PDF, GIF, and WebP, it also supports scientific formats such as FITS, OpenEXR, Matlab, Analyze, PFM, Radiance, OpenSlide and DICOM (via lib-Magick). Compared with similar libraries, *VIPS* performs faster and requires less memory capacity. We investigate nine input images that offer a variety types of features in the images [23]. Such diversified features they include make them ideal for benchmarking image encoding algorithms. All quality measurements are based on raw images at 512×512 pixels resolution. *VIPS* with higher-resolution input images with more pixels would have similar behavior, but yields even higher memory capacity and bandwidth usage. As such, our experimental results provide a conservative lower bound on the

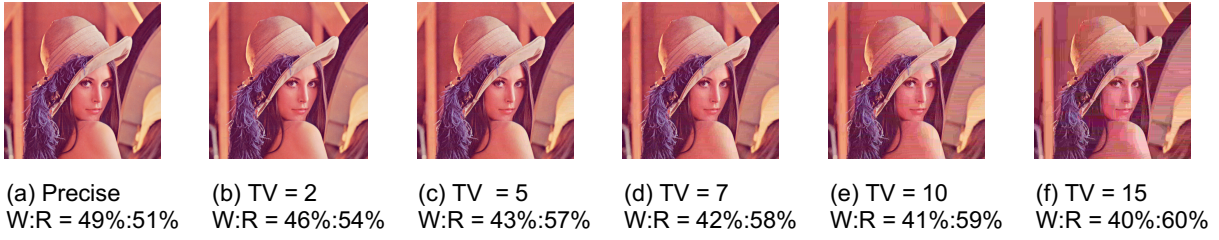


Fig. 7. The ratio of write energy and read energy of image *Lena* across various approximate levels.

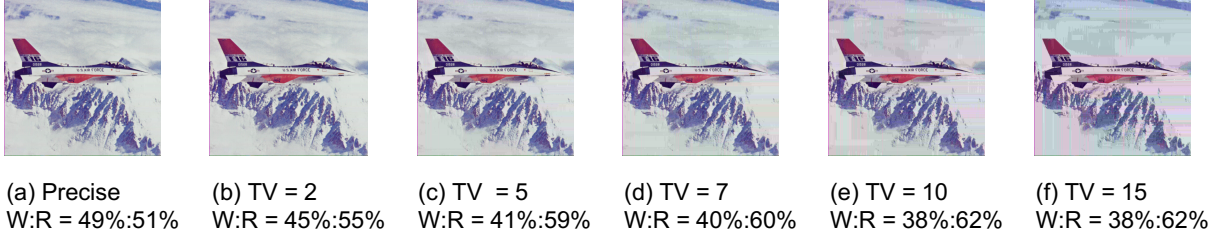


Fig. 8. The ratio of write energy and read energy of image *Airplane* across various approximate levels.

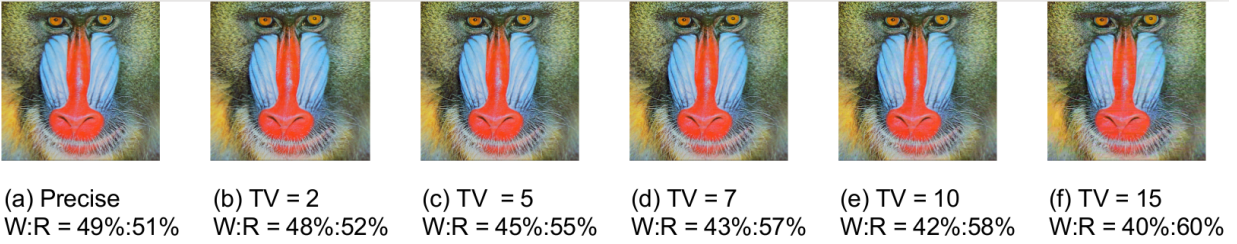


Fig. 9. The ratio of write energy and read energy of image *Baboon* across various approximate levels.

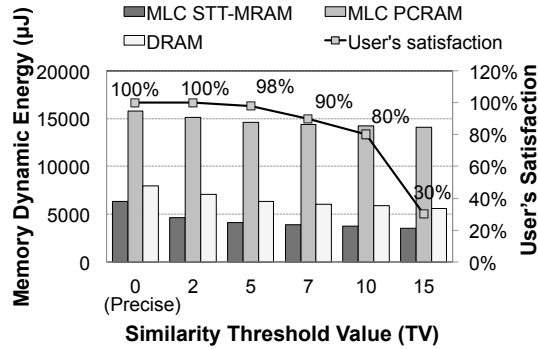


Fig. 10. Memory access energy and user's satisfaction with MLC STT-MRAM, MLC PCRAM, and DRAM main memory.

benefits of the cooperatively designing the approximate computing algorithm and main memory hardware. We report our performance and energy results generated from ten billion instructions in the region of interest.

Simulation framework. In order to simulate the performance of our design, we developed a pintool on top of PIN [25], a widely used dynamic binary instrumentation framework developed by Intel. The pintool models our approximate image memory storage algorithm and memory controller design integrated with our mode controller. The pintool can only obtain the address of each memory access. The corresponding image data associated with the memory access addresses is obtained from each input image. The pintool also

TABLE I
MEMORY ACCESS ENERGY PARAMETERS OF VARIOUS MEMORY TECHNOLOGIES (PJ/BIT) [24], [3].

Energy	MLC STT-MRAM	PCRAM	DRAM
Read	0.38	2	1.56
Write	Soft Transition: 1.92 Hard Transition: 3.192	SET: 13.5 RESET: 19.2	0.39
Peripheral	0.47	0.47	-

generates a trace of memory accesses in the workloads. Based on the trace, we calculate dynamic memory energy consumption using the parameters listed in Table I. In order to evaluate our design across various image quality requirements, we adopt six approximate levels represented by six different similarity threshold values (TV = 0 – 15), which are the maximum allowed R, G, B value differences among the pixels in each approximate window. We compare our design with MLC STT-MRAM with other two main memory configurations – MLC PCRAM and DRAM.

Metrics. We adopt a user's satisfaction model [21] to evaluate a combination of image quality and performance generated by the application with our design. The user's satisfaction model provides a lower bound of image processing applications by ensuring that the users can obtain the results with sufficiently accuracy at a sufficiently low delay. Because our approximate image memory storage mechanism does not involve computation of CPU, we do not observe visible changes in processor energy consumption compared with using traditional main memory designs. Therefore, we only

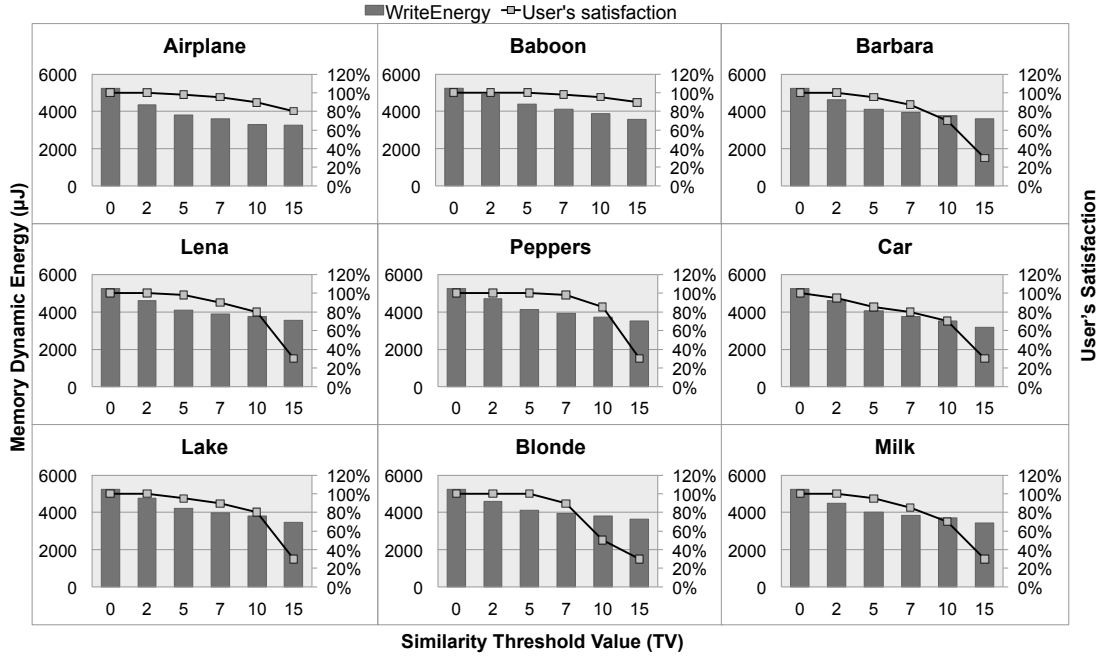


Fig. 11. Memory access energy and user's satisfaction with our design based on MLC STT-MRAM.

report the dynamic memory energy consumption in this paper. We also adopt bit flip rate (BFT) in the MLC STT-MRAM cells across the main memory to evaluate the dynamic energy consumption. In addition, we employ PSNR and SSIM metrics to quantify the image quality generated by our design.

B. Results

Image quality loss. In order to measure the image quality degradation when we implemented our algorithm in MLC STT-MRAM, we choose nine 512×512 well known test images, such as 'Lena', 'Peppers', etc [23]. Figure 7 8 9 show that three images perceived quality at different TVs. We can hardly see any errors or artificial effects when TV is 2 and 5. Interestingly, when TV is larger than 10, different images present different levels of quality loss. For image Baboon, even when TV reaches 15, we still cannot find any obvious artificial effects, because this image contains more high frequency areas. For image Airplane, there are much more low frequency areas than Baboon's. When TV is larger than 10, we can see some artificial effects, because human eyes are more sensitive to the low frequency area in images. In order to ensure the perceived quality of most images, we consider use image Lena to represent most of images, so we let TV is equal to 7 to conduct our user's satisfaction experiments, for when TV is 7 the quality of image Lena can be accepted by most people. By the way, users also can choose the TV they hope to use to achieve different energy savings with different quality loss, which is usually related to the contents of different images. Table II lists the PSNR and SSIM comparison between original image and processed image. When the threshold value is 7, the PSNR and SSIM results are as high as 31.96dB and 0.922, and the bit flip rate can reach to 25.91%. Although the bit flip rates when threshold rate is 10 and 15 are a little bit larger than 25.91%, the image quality gets heavily decreased.

Energy consumption and user's satisfaction. To study the energy efficiency of our approximate image processing scheme, we evaluate user's satisfaction and dynamic energy consumption in the memory.

Figure 11 depicts the user's satisfaction and write energy consumption across various approximation levels across the nine input images. Due to the limited space, we only depicted the energy and user's satisfaction comparison among various memory technologies with input image *Lena*. With the rest of input images, we only show MLC STT-MRAM results. But we also report the average numbers compared with other memory technologies – MLC PCRAM with our approximate image processing scheme and DRAM with conventional image processing schemes.

We make three major observations from these results. First, when $TV \leq 5$, we can substantially improve system energy consumption, while ensuring user's satisfaction higher than 98%. In particular, with $TV=2$, our design can achieve 100% user's satisfaction, while reduces memory dynamic energy by $2.3\times$ and 53% on average across the input images compared with MLC PCRAM and DRAM main memory, respectively. By relaxing the user's satisfaction to be 98%, we can further reduce memory dynamic energy consumption by $2.5\times$ and 55% compared with MLC PCRAM and DRAM, respectively. Figure 10 shows the detailed comparison among various memory technologies with input image *Lena*. Clearly, MLC STT-MRAM outperforms other memory technologies. Furthermore, PCRAM-based design does not appear to save more energy, when we degrade user's satisfaction with larger TVs. Second, MLC STT-MRAM dynamic energy degrades linearly, when TV is increased from 2 to 10. Yet the energy consumption remains almost the same, when we further increase TV beyond 10. Finally, the sensitivity of user's satisfaction to approximate level will depend on the types of images. For example, *Airplane* and *Baboon* have large areas with similar colors. Therefore, the user's satisfaction does not degrade significantly with larger TVs.

VI. RELATED WORK

Previous studies [7] explored opportunities in fine granularity approximate technique with SLC STT-MRAM memory. The study identified and characterized mechanisms in STT-MRAM bit-cells that yield trade-offs between energy and image quality. However,

TABLE II
PSNR AND SSIM RESULTS AT FOUR DIFFERENT THRESHOLD VALUES.

Images	TV=2			TV=5			TV=7			TV=10			TV=15		
	PSNR	SSIM	BFT	PSNR	SSIM	BFT	PSNR	SSIM	BFT	PSNR	SSIM	BFT	PSNR	SSIM	BFT
Airplane	42.91	0.990	17.18%	35.42	0.955	27.24%	32.20	0.921	31.01%	29.58	0.871	37.19%	26.17	0.817	38.04%
Baboon	42.51	0.999	6.34%	35.20	0.987	16.59%	31.88	0.972	21.66%	28.61	0.942	26.21%	25.20	0.886	31.94%
Barbara	42.78	0.992	12.03%	34.85	0.959	21.77%	32.17	0.939	25.07%	29.05	0.899	28.12%	25.50	0.832	31.15%
Lena	42.68	0.993	12.07%	34.49	0.945	22.05%	31.89	0.907	25.74%	28.89	0.845	28.53%	25.47	0.767	32.13%
Peppers	43.36	0.994	10.32%	34.46	0.950	21.50%	31.72	0.908	25.52%	28.85	0.843	28.79%	25.57	0.745	32.75%
Car	43.14	0.994	12.50%	34.78	0.964	22.73%	31.66	0.936	28.47%	28.73	0.897	33.25%	25.23	0.829	39.18%
Lake	43.96	0.995	8.83%	34.57	0.962	19.61%	31.66	0.929	24.08%	28.79	0.876	27.63%	25.45	0.788	33.56%
Blonde	43.05	0.992	12.15%	35.03	0.946	21.34%	32.28	0.906	24.78%	29.45	0.849	27.32%	26.02	0.761	30.69%
Milk	42.38	0.987	14.08%	34.76	0.924	23.64%	32.16	0.881	26.90%	29.16	0.816	29.43%	26.11	0.741	34.49%
Average	42.97	0.994	11.72%	34.84	0.955	21.83%	31.96	0.922	25.91%	29.01	0.871	29.61%	25.64	0.796	33.77%

the energy improvements are at the cost of small probabilities of read/write failures. In particular, the write failures stems from reduced write current or duration, which are two main factors to guarantee the successful write operation. approximate techniques enable every bit-cell. Other studies [7] proposed MLC PCRAM and solid DRAM used as the approximate image memory. There are two mechanisms to implement the approximate techniques. The first mechanism allows errors in multi-level cells by reducing the number of programming pulses used to write them. The second mechanism mitigates wear-out failures and extends memory endurance by mapping approximate data onto blocks that have exhausted their hardware error correction resources. However, such approximate techniques can be difficult to implement in the hardware due to the potentially large hardware implement overhead.

VII. CONCLUSION

In this paper, we propose an approximate image processing scheme achieves high energy efficiency while meeting user's satisfaction on image quality and system performance. Our design leverages the unique asymmetric write current between the soft and hard bits in each MLC STT-MRAM cell. This property has been considered as a downside of MLC STT-MRAM technologies, because of the write disturbance issues. Yet our design exploits the benefit of this property in approximate image memory storage. In addition, our design only requires lightweight hardware modifications in memory banks and the memory controller. Therefore, our design is feasible in implementation and easy to adopt.

REFERENCES

- [1] K. Kim, "Technology for sub-50nm DRAM and NAND flash manufacturing," in *IEDM*, pp. 323–326, 2005.
- [2] E. Kultursay, M. Kandemir, and A. Sivasubramaniam et al., "Evaluating STT-RAM as an energy-efficient main memory alternative," in *ISPASS*, pp. 256–267, 2013.
- [3] B. C. Lee, E. Ipek, and O. Mutlu et al., "Architecting phase change memory as a scalable DRAM alternative," in *ISCA*, pp. 2–13, 2009.
- [4] T. Acharya and P.-S. Tsai, *JPEG2000 standard for image compression: concepts, algorithms and VLSI architectures*. John Wiley & Sons, 2005.
- [5] A. Sampson, J. Nelson, and K. Strauss et al., "Approximate storage in solid-state memories," *ACM TOCS*, vol. 32, no. 3, p. 9, 2014.
- [6] Q. Guo, K. Strauss, and L. Ceze et al., "High-density image storage using approximate memory cells," in *ACM SIGPLAN Notices*, vol. 51, pp. 413–426, ACM, 2016.
- [7] A. Ranjan, S. Venkataramani, and X. Fong et al., "Approximate storage for energy efficient spintronic memories," in *DAC*, p. 195, ACM, 2015.
- [8] X. Chen, N. Khoshavi, and J. Zhou et al., "AOS: adaptive overwrite scheme for energy-efficient MLC STT-RAM cache," in *DAC*, p. 170, ACM, 2016.
- [9] X. Chen, N. Khoshavi, and R. F. DeMara et al., "Energy-aware adaptive restore schemes for MLC STT-RAM cache," *IEEE Transactions on Computers*, 2016.
- [10] A. Sampson, W. Dietl, and E. Fortuna et al., "EnerJ: Approximate data types for safe and general low-power computation," in *ACM SIGPLAN Notices*, vol. 46, pp. 164–174, 2011.
- [11] J. Park, H. Esmailzadeh, and X. Zhang et al., "Flexjava: Language support for safe and modular approximate programming," in *Proceedings of the Joint Meeting on Foundations of Software Engineering*, pp. 745–757, 2015.
- [12] P. Stanley-Marbell and M. Rinard, "Lax: Driver interfaces for approximate sensor device access," in *HotOS*, 2015.
- [13] H. Esmailzadeh, A. Sampson, and L. Ceze et al., "Architecture support for disciplined approximate programming," in *ACM SIGPLAN Notices*, vol. 47, pp. 301–312, ACM, 2012.
- [14] H. Esmailzadeh, A. Sampson, and L. Ceze et al., "Neural acceleration for general-purpose approximate programs," in *MICRO*, pp. 449–460, IEEE Computer Society, 2012.
- [15] S. Sidiropoulos-Douskos, S. Misailovic, and H. Hoffmann et al., "Managing performance vs. accuracy trade-offs with loop perforation," in *Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering*, pp. 124–134, ACM, 2011.
- [16] T. Moreau, J. S. Miguel, and M. W. et al., "A taxonomy of approximate computing techniques," in *UW CSE Technical Report*, pp. 1–5, 2016.
- [17] T. Ishigaki et al., "A multi-level-cell spin-transfer torque memory with series-stacked magnetotunnel junctions," in *VLSIT*, 2010.
- [18] X. L. et al., "Demonstration of multilevel cell spin transfer switching in MgO magnetic tunnel junctions," *Applied Physics Letters*, 2008.
- [19] Y. Z. et al., "Multi-level cell STT-RAM: Is it realistic or just a dream?," in *ICCAD*, 2012.
- [20] Z. Wang, A. C. Bovik, and H. R. S. et al., "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [21] J. Park, E. Amaro, and D. Mahajan et al., "AxGames: Towards crowdsourcing quality target determination in approximate computing," *ACM SIGPLAN Notices*, vol. 51, no. 4, pp. 623–636, 2016.
- [22] C. Bienia, S. Kumar, and J. P. S. et al., "The PARSEC benchmark suite: Characterization and architectural implications," Tech. Rep. TR-811-08, Princeton University, January 2008.
- [23] H. Zhao, H. Sun, and Q. Yang et al., "Exploring the use of volatile STT-RAM for energy efficient video processing," in *ISQED*, pp. 81–87, IEEE, 2016.
- [24] Y. C. et al., "Access scheme of multi-level cell spin-transfer torque random access memory and its optimization," in *MWSCAS*, 2010.
- [25] C.-K. Luk, R. Cohn, and R. Muth et al., "Pin: Building customized program analysis tools with dynamic instrumentation," in *PLDI*, pp. 190–200, 2005.