Implementing Rank Modulation

Yue Li[1], **Yanjun Ma**[2], **Eyal En Gad**[1], **Mina Kim**[3], **Anxiao (Andrew) Jiang**[4] and **Jehoshua Bruck**[1]
[1]*California Institute of Technology* {yli, eengad, bruck}@caltech.edu, [2]*Intellectual Ventures* yma@intven.com
[3]*Binghamton University* mkim22@binghamton.edu, [4]*Texas A&M University* ajiang@cse.tamu.edu

I. Introduction

The density of NAND flash grows at the price of significantly reduced reliability. To enable the continued scaling of flash memory more fundamental changes are needed. In this paper, we show that flash is much more reliable by adopting the *rank modulation* (RM) scheme [2]. RM encodes data using the relative orders of memory cell voltages, which is inherently resilient to asymmetric errors. However, the practical performance of RM is still unknown due to the lack of implementation studies. This work has multiple contributions on the practical aspects of RM, which include: 1) Adaptations that make RM implementable using existing NAND flash memories. 2) Characterization of the first RM implementation using 20nm MLC from two vendors. 3) An application of RM to archival storage, and its performance evaluation using 16nm MLC. 4) An implementation-friendly VLSI architecture of an NAND flash memory based on RM.

II. Rank Modulation for Existing Flash

We propose a simple approach to program and read RM codewords for cells with r-level. Denote the states of an r-level cell by $\mathrm{P1}, \mathrm{P2}, \cdots, \mathrm{Pr}$. A length-$N$ RM codeword with r ranks is stored in cells by programming the cells of rank i to state Pi. In practice, this requires sequentially programming the $\log_2 r$ pages sharing the same N cells with the binary bits mapped from the desired cell states using the Gray code. (We assume r is an integer power of 2.) The cells storing an RM codeword are ranked using read-retry (RR). The multiple reference voltages provided by RR divides the whole threshold voltage interval into many bins. Each of the pages sharing the cells is read multiple times with different reference voltages. The reading results are combined to determine the bin of each cell, and we thus rank cells by comparing their bin indices.

III. Characterizing Rank Modulation

We characterized the RM scheme in 20nm MLC using our implementation. Our experiments used a commercial NAND flash tester and three kinds of MLC flash that can be purchased on the market from two different vendors A and B . All the chips provide 8 RR options. Due to space limitation, we only presents the results for one of the MLCs.

Fig 1 shows the average raw cell error rates (RCERs) of RM and conventional MLC under different types of noise. RCER measures the probability of adjacent rank switches for RM, and the probability that the bits stored by a cell has at least an error. Here we used balanced rank modulation [1] with 4 ranks and each rank has 64 cells. All the results suggest that RM provides high reliability when (1) errors are strongly asymmetric, and (2) cells carry a small number of P/E cycles. On the other hand, RM is at least as reliable as MLC at larger P/E cycles. When comparing the RCERs across different types of noise, we found the RCERs under data retention are much higher than those under interference and read disturb at the same P/E cycles. The characterization suggests that RM is an excellent approach to trade off endurance for data retention time.

IV. Rank Modulation Based Archival Storage

We show that RM enables the adoption of high density NAND flash by archival storage systems. Archival data once written are not likely to be read in the future, and conventional archival storage systems thus use tape and hard disk drive (HDD) as storage media to provide high capacity, long retention time. As the density of NAND flash grows, SSD is rapidly catching up with HDD on capacity and price. More importantly, flash memory naturally offers random data access, which enables very efficient *data deduplication* and *data compression* algorithms for even higher capacity. In flash-based archival storage, write and read traffic mainly comes from memory scrubbing since data are rarely accessed by user. With a reasonable scrubbing frequency, a memory cell only needs to survive a small number of P/E cycles. Observing that RM provides much lower RCERs at small P/E cycles, we apply RM to archival storage.

We experimentally studied the performance of RM and conventional MLC using two common scrubbing schemes for archival storage. The first scheme is periodic scrubbing, which refreshes a block with a fixed frequency. The second scheme refreshes block adaptively, by increasing frequency at higher P/E cycles. Here we show the results for the latter case. The focus of our experiments is to compare the maximum data retention time provided by RM and conventional MLC using the same amount of coding redundancy.

Figure 2 shows the performance of adaptive scrubbing in MLC and RM. The flash memories used in the experiments of this section are 64Gb MLC flash from vendor B manufactured on 16nm technology node. Each curve in the figure shows the RBER at different

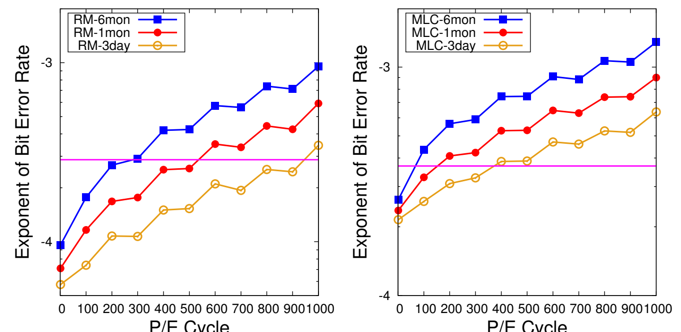

Fig. 2. The adaptive scrubbing performance of RM (left) and MLC (right).

P/E cycles with fixed scrubbing periods. Three scrubbing periods are used: 3 days, 1 month and 6 months. For RM, we let the user data be protected by a BCH code correcting 10 errors. The correction capability of the BCH code for MLC is set to 11 errors for same redundancy comparisons. The horizontal lines in the figure mark the RBER that each ECC is able to correct. If a curve is below a horizontal line for a range of P/E cycles, then the corresponding scrubbing frequency can be used for reliable data recovery. We let adaptive scrubbing always use the lowest frequency that is available. From the results, we can see that for RM 6-month scrubbing period is used from 0 to 300 P/E cycles, 1-month period is used from 301 to 530 P/E cycles, and 3-day period is used from 151 to 940 P/E cycles, therefore, the total retention period is about 172.6 years. Compared to MLC which only covers 39.3 years, RM provides 4.4x longer data retention time with the same coding redundancy.

V. A VLSI Architecture for Rank Modulation

To fully achieve the advantage of the RM scheme, we need to enable adaptive cell programming for higher capacity and reliability.

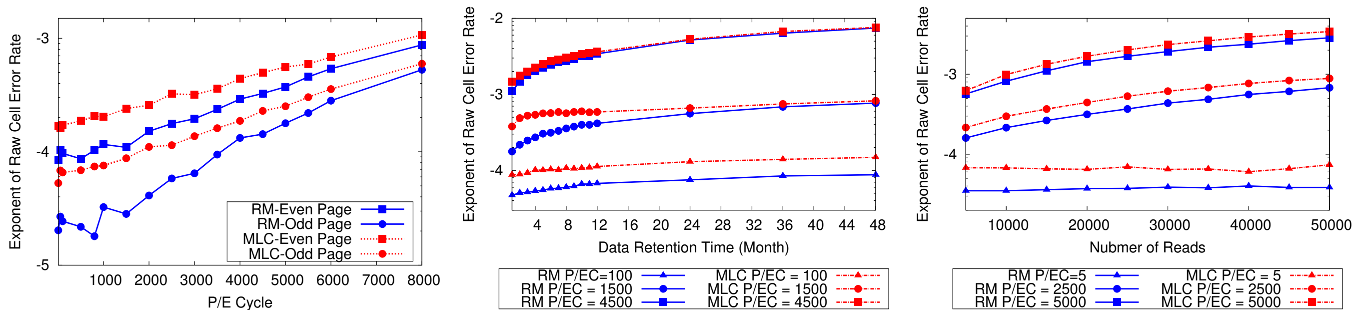

Fig. 1. The RCERs of BRM and MLC under programming interference (left), charge leakage (middle), and read disturb (right).

Also desirable is to speed up the reading of an RM codeword, preferable in one reading cycle, to increase memory read throughput needed to see the adoption of RM in general SSD applications. This section presents a VLSI implementation of RM to achieve these requirements. We then compare this implementation with the state-of-the-art flash architectures and show that this design can be implemented with only minor changes.

Fig. 3 shows the architecture based on the conventional FG array. The array, sense amps, data buffers are standard components that can

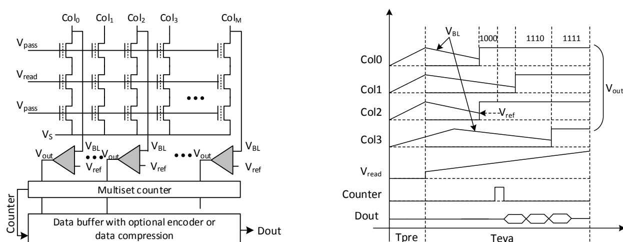

Fig. 3. Left: Architecture of an RM memory. Right: A illustrative timing diagram for the reading of memory cells according to the relative sense timing.

be found in all conventional flash memory chips. The multiset counter is added to count the number of cells belong to the same rank. The method for reading a RM set in one sensing cycle is illustrated by the timing diagram in Fig. 3, using an example of four cells. The unselected rows are biased by a constant voltage (V_{pass}) to allow the unselected transistors to turn on. The reading process starts by pre-charging all BLs in a RM set and then control the discharging of these BLs s by ramping the control gate voltage of the selected row, indicated by the V_{read} signal in Fig. 3. When V_{read} exceeds the threshold voltage of a selected transistor, it turns on and starts to discharge that particular BL. As the BL voltage decreases to a reference voltage, V_{ref}, the sense amp on that BL trips and increments the multiset counter. As the counter reaches a preset multiplicity number, e.g. 2 when Col_0 and Col_2 trips in Fig. 3, it triggers the latching of the data present on all BLs to the data buffer and streaming out of the data, D_{out}. In the meantime the counter rets and remaining BLs continue to discharge until the next BL reaches V_{ref} trips another round of counter increment, latching and buffering, until selected row of all BLs have been ranked.

Note that this reading method only records the relative order when the sense amps are tripped. The rate of discharge depends on the ramp rate of V_{read} and the threshold voltage of the flash cells. Since digital circuitry in the submicron technology can easily achieve sub-nanosecond speed, the ranking can take place within one typical sensing cycle of the conventional NAND flash memory [3].

For programming, the standard Incremental Step Pulse Programming (ISPP) method [5] may be adapted to the program the cells in multiset RM scheme using a program-verify process. The verification step compares cells to each other rather than to absolute reference levels, as is the case in the conventional NAND flash memory.

Note that in this process the total tunneling time is about the same as the tunneling time in programming the conventional NAND flash memory, on the order of about 1ms. Since the tunneling time is usually the dominant factor in the programming of NAND flash memory cells, the verification step usually takes about $25\mu s$, the programming time for the RM memory is thus on the same order of magnitude as the programming time of conventional flash memory.

The sensing and programming methods can be implemented with very simple, mostly digital, modifications in current flash chips. All current NAND flash chips use the same basic circuit blocks as shown in Fig. 3, with the exception of the multiset counter. Consider the Micron 20nm MLC architecture in [4]. The reading is performed by pre-charging all BLs and monitoring the discharge of BLs, by stepping selected wordline voltage through each desired read voltages, that is provided by precision voltage references [4]. This design uses ramp wordline voltage during the sensing operations, which is generated by the analog bias generator block. The page buffer array block contains the sensing amplifiers. Since the basic architectures of RM and [4] are very similar, it is fairly straightforward to modify the Micron architecture to implement the RM scheme by adding a multiset counter that triggers the data latching.

Since current flash chips discern different logic state by comparing cell threshold voltage to predetermined reference threshold voltages, they usually contains complex circuits that provide precision voltage reference, DACs and ADCs, and temperature compensation functions. Since the RM readout method senses only the relative timing, it is much simpler to implement with no need for these circuitry. Elimination of these circuits should save areas for the RM memory as well as make the RM chips more energy efficient.

References

[1] E. En Gad, A. Jiang, and J. Bruck, “Trade-offs between instantaneous and total capacity in multi-cell flash memories,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2012, pp. 990–994.

[2] A. Jiang, R. Mateescu, M. Schwartz, and J. Bruck, “Rank modulation for flash memories,” *IEEE Trans. on Inform. Theory*, vol. 55, no. 6, pp. 2659–2673, Jun. 2009.

[3] M. Kim, M. Shaterian, and C. Twigg, “Rank determination algorithm by current comparing for rank modulation flash memories,” in *IEEE Int. Midwest Symp. on Circuits and Systems*, Aug 2013, pp. 1354–1357.

[4] G. Naso and other, “A 128Gb 3b/cell NAND flash design using 20nm planar-cell technology,” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb 2013, pp. 218–219.

[5] K.-D. Suh *et al.*, “A 3.3 V 32 Mb NAND flash memory with incremental step pulse programming scheme,” *IEEE Journal of Solid-State Circuits*, vol. 30, no. 11, pp. 1149–1156, Nov 1995.