ISOCC 2023
20th International SoC Design Conference

SCC: Efficient Error Correction Codes for MLC PCM

Yujin Lim[1], Dongwhee Kim[2], and Jungrae Kim[3]

[1]Dept. of Electrical and Computer Engineering, [2]Dept of Semiconductor and Display Engineering,
[3]Dept. of Semiconductor Systems Engineering, Sungkyunkwan University

Abstract

Phase Change Memory (PCM) has presented as a potential replacement of DRAM due to its better scalability. However, **Multi-Level Cell (MLC)** PCM encounters reliability problems such as resistance drift error and cell wear-out. To improve reliability, implementing strong **Error Correction Codes (ECC)** is required but this incurs substantial overhead to memory system. Existing ECC schemes have trade-offs between error correction capability and overhead. This paper proposes ***Single Cell Correction (SCC)*** codes, an efficient ECC scheme which can correct single cell errors in MLC PCM with small overhead.

Introduction

DRAM are reaching scalability limits, making Phase Change Memory (PCM) a promising alternative. PCM, a non-volatile memory, utilizes the resistance difference in chalcogenide glass. While PCM's MLC storage boosts density, it has reliability issues like resistance drift error and cell wear-out. Gray encoding is typically utilized in MLC PCM to ensure that the level shift only causes a single bit error. However, wear-out can induce up to a 2-bit error in a single cell. Techniques such as wear-leveling have been proposed to minimize this issue but this can be challenging if PCM is used as main memory. ECC have been employed to address these, but many are either too costly or inefficient. In this paper, we introduce an efficient ECC that corrects MLC PCM errors with lesser overhead and greater correction capability than existing schemes.

Conclusion

MLC PCM can improve storage density without scalability limitations, but it faces significant reliability issues. Using conventional ECC schemes can enhance reliability, while it comes with a substantial overhead. Given the frequent error patterns in MLC PCM, SCC is the optimal solution to protect MLC PCM memory.

Proposed Method

SCC uses 7-bit redundancy over 64-bit data, like SEC. This allows for 127 non-zero syndromes, of which 106 (71 single-bit error + 35 single-cell error) are mapped to single bit or single cell error cases. Syndromes were generated over GF(2^7) with a specific polynomial, and 106 were chosen as column vectors in the H-matrix to meet the following conditions. (Examples can be found at the bottom)

- ✓ All column vectors are nonzero.
- ✓ All column vectors are unique.
- ✓ For two adjacent column vectors, with the left in an odd column and the right in an even, their sum must be unique.

Result

Scheme	Error Scenario	Reliability (%)			Redundancy Ratio (%)
		CE	*DUE*	*SDC*	
SEC	1-bit error	100.00	0.00	0.00	10.94
	1-cell error	0.00	8.57	91.43	
	2-bit error	0.00	33.03	66.97	
SEC-DED	1-bit error	100.00	0.00	0.00	12.50
	1-cell error	0.00	100.00	0.00	
	2-bit error	0.00	100.00	0.00	
SEC-DAEC	1-bit error	100.00	0.00	0.00	12.50
	1-cell error	100.00	0.00	0.00	
	2-bit error	2.95	45.07	51.98	
SCC (Proposed)	1-bit error	**100.00**	0.00	0.00	**10.94**
	1-cell error	**100.00**	0.00	0.00	
	2-bit error	0.00	16.14	83.86	

$$H = \begin{bmatrix}
0&0&0&1&0&1&0&0&0&1&0&0&1&1&0&0&1&0&0&1&1&0&0&1&1&1&1&0&1&1&1&0&1&0&0&1&0&1&1&0&0&1&1&1&0&0&0&1&0&1&1&1&1&0&0&0&0&1&1&1&1&1&0&1&1&0&0&0&0&0\\
0&1&1&1&0&0&1&1&1&0&1&0&1&0&1&1&0&1&0&1&0&0&0&1&0&0&0&1&1&1&0&1&1&1&1&0&1&0&1&1&0&1&0&0&1&1&1&0&0&1&0&0&0&0&1&0&0&0&0&1&1&0&1&0&0&0&0\\
0&0&1&1&0&0&0&0&1&0&0&1&1&0&1&1&1&1&1&1&0&0&1&1&0&1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&1&1&1&0&0&1&0&1&0&0&1&0&1&1&0&0&1&0&1&0&0&0&1&0&1&1&1&0&0&0&0&1&0&0&0&0\\
1&0&0&0&1&1&1&0&0&0&1&0&1&1&1&1&1&1&1&1&1&0&0&1&1&0&1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&1&0&1&0&0&1&0&1&1&0&0&1&0&1&0&0&0&1&0&1&1&1&0&0&0&0&1&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&1&0&1&1&1&1&1&1&1&0&0&1&1&0&1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&1&0&1&0&0&1&0&1&1&0&0&1&0&1&0&0&0&1&0&1&1&1&0&0&0&0&1&0&0\\
0&0&0&0&1&0&1&0&0&1&1&1&0&1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&1&0&1&0&0&1&0&1&1&0&0&1&0&1&0&0&0&1&0&1&1&1&0&0&0&0&1&0&1&1&1&1&0&1&1&0&0&0&0&0&1&0\\
0&1&0&1&0&1&0&1&0&0&0&1&1&0&0&1&0&0&1&1&0&0&1&1&1&1&0&0&1&1&0&1&0&0&1&0&1&1&0&0&1&0&1&0&0&0&1&0&1&1&1&0&0&0&0&1&1&1&1&1&1&0&1&1&0&0&0&0&0&1
\end{bmatrix}$$