Improving Memory ECC By Exploiting Unused Spare Columns

Rudrajit Datta and Nur A. Touba

Computer Engineering Research Center
Dept. of Electrical and Computer Engineering
University of Texas at Austin

Introduction

- Factors leading to memory errors
  - Transient errors due to radiation, power supply noise
  - Correction using ECC
- Memory repair methods
  - Spare columns/rows
- Need
  - Repairing defective cells increases yield greatly

Previous Work

- SEC-DED
  - Most commonly used
  - Drawbacks
    - Only detect but not correct double errors
    - Mis-corrects triple errors
  - Reduces reliability of code

Previous Work

- SEC-DAEC [Avijit 07]
  - SEC and corrects adjacent double errors
  - Adjacent double errors more likely than double errors
  - 1-5% of single event upsets (SEUs) cause multiple-bit errors (MBUs)
  - Drawbacks
    - Mis-corrects non-adjacent double bit errors

- Use spare rows/columns [Han 05]
  - Map defective cells to the spare rows/columns
  - Generally memories have leftover spare rows/columns after the repair process

Linear Block Codes

$$\overset{H}{H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}} \quad \overset{C^T}{\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}} = \overset{Syndrome}{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}}$$

$$\overset{H}{H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}} \quad \overset{C^T}{\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}} = \overset{Syndrome}{\begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}}$$

Linear Block Codes

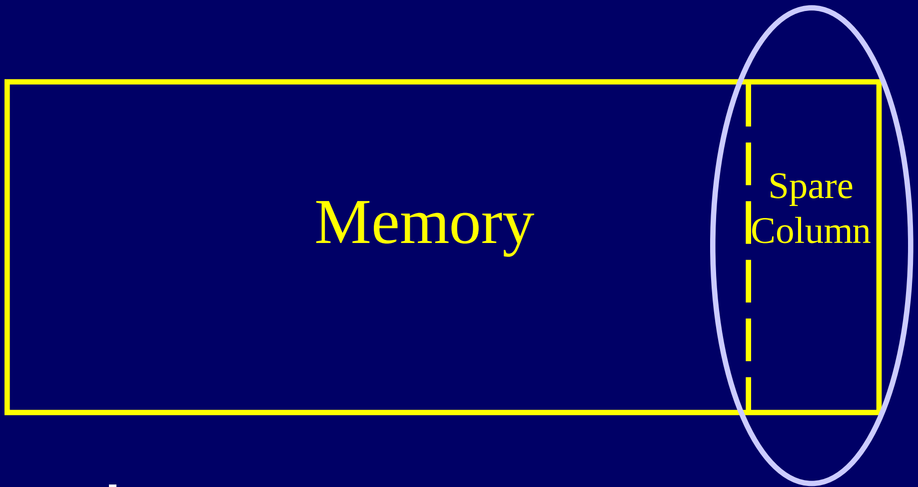
- H-matrix chosen such that
  - for all single and double bit error syndromes are unique
  - for SEC-DAEC adjacent double-bit syndromes are unique from
    - single-bit errors
    - double-bit errors

- Hsiao code [Hsiao 70]
  - All columns unique and contains odd number of 1s
  - SEC-DED

Spare Row/Column Allocation

- Spare cells arranged into rows and/or columns
- Used to mitigate small number of defects
- Optimal allocation of spare rows and columns NP complete [Kuo 86]
- Previous algorithms
  - Greedy algorithm – repair most [Tarr 84]
  - Branch and bound
  - Exhaustive search – for smaller array sizes
- Enhances memory yield

Proposed Scheme



- Unused spare columns
- Extra check bits stored in unused spare columns
- Extra check bits aid in correction and detection

Proposed Scheme

- Goal
  - Implement SEC-DED and reduce triple error mis-correction
  - Implement SEC-DAEC and reduce non-adjacent double error mis-correction
- Procedure
  - Add extra check bits
  - Increases search space for an optimal code that meets above constraints
  - Extra check bits stored in unused spare columns
  - Worst case, no unused spare columns, code should still be SEC-DED/SEC-DAEC
  - Start with a SEC-DED/SEC-DAEC code and build on top of it

Proposed Scheme

- Each extra row added by
  - exhaustive search of entire space (for smaller codes)
  - random search (bigger codes)
- Not much difference between random and/or exhaustive search
- Number of added check bits (extra rows to the H-matrix) depends on number of spares leftover after repair process
- Rows are added one at a time in a greedy fashion

Triple Error Mis-correction

Original H-matrix

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

- **Mis-corrected triple errors = 28**

Modified H-matrix

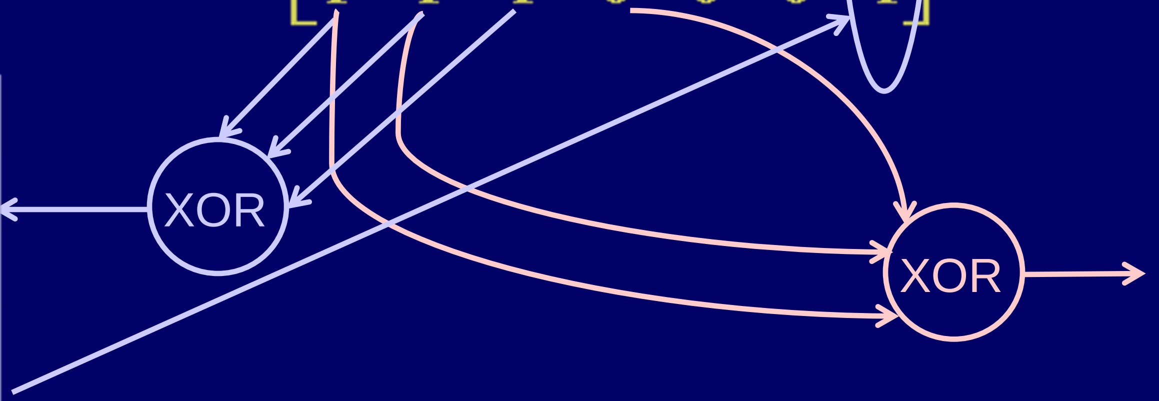
$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

- **Mis-corrected triple errors = 12**

Triple Error Mis-correction

Original H-matrix

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



$$S = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Triple error
mis-correction

$$S = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Unique syndrome

Triple Error Mis-correction

Modified H-matrix

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

XOR

$$S = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

No triple error mis-correction

Non-adjacent Double Error Mis-correction

Original H-matrix

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

Modified H-matrix

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

Non-adjacent Double Error Mis-correction

Original H-matrix

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

XOR

$$S = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

No non-adjacent double error mis-correction

XOR

XOR

$$S = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} = S = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Non-adjacent double error mis-correction

Non-adjacent Double Error Mis-correction

Modified H-matrix

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

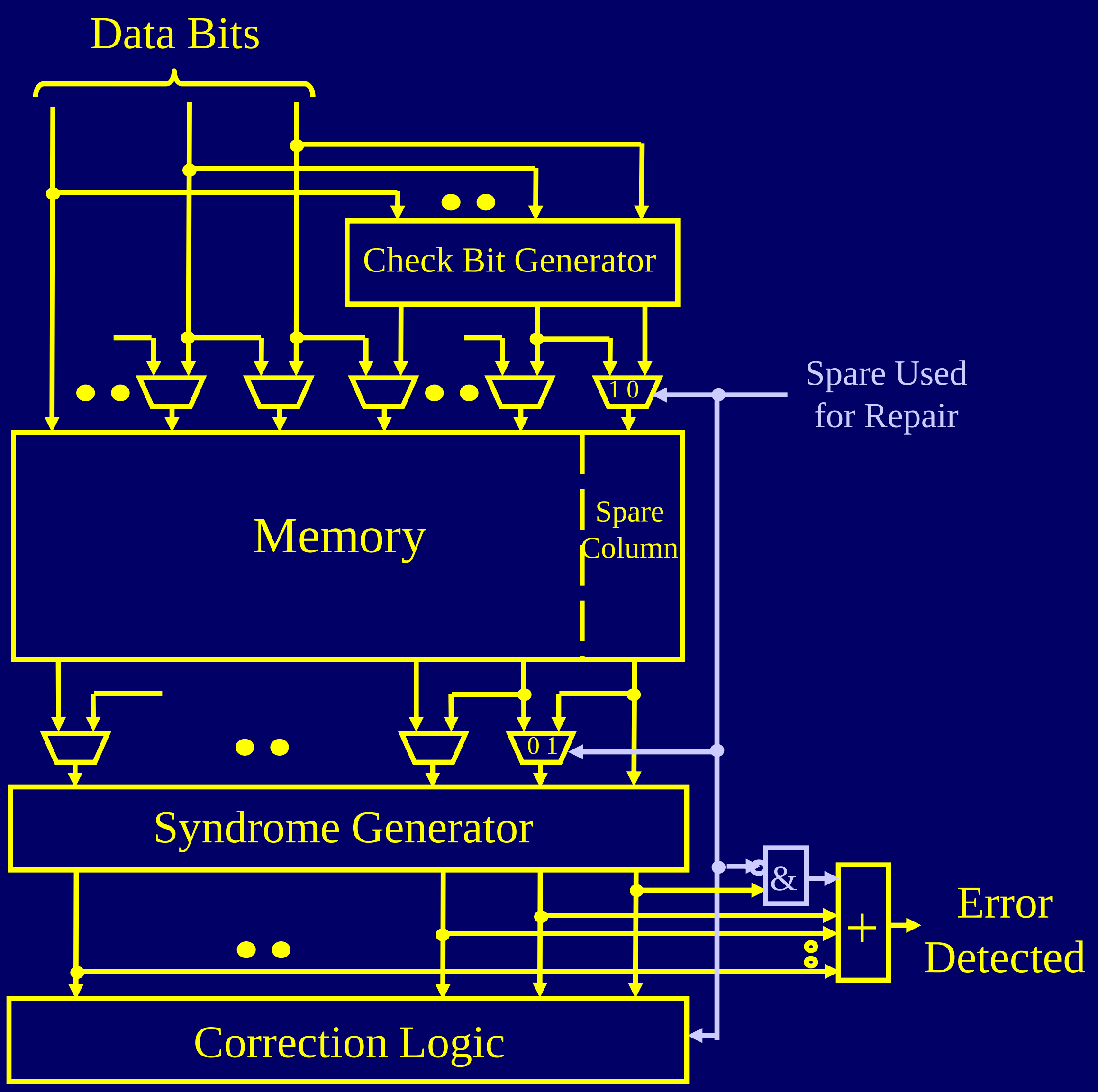
XOR

XOR

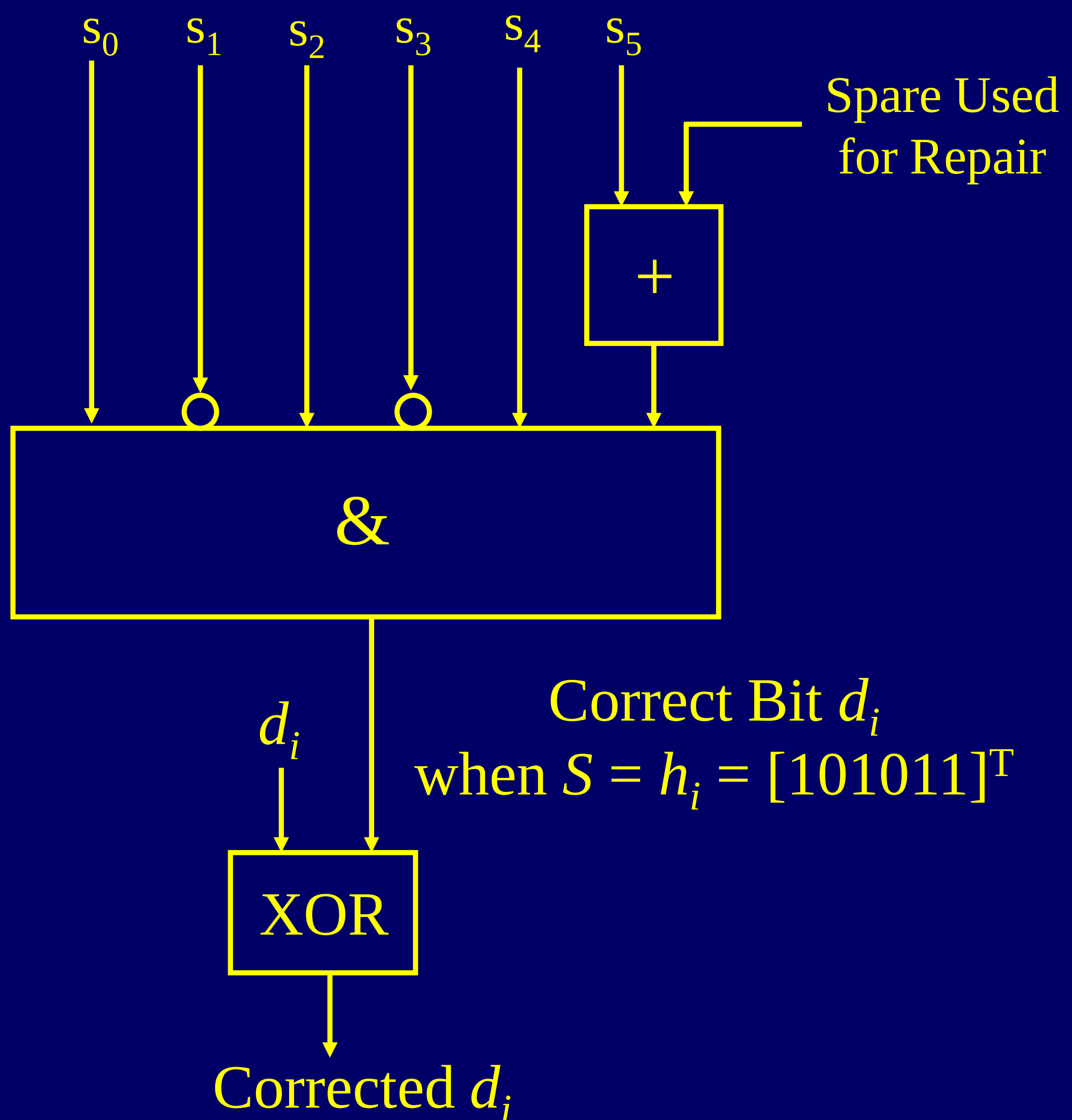
$$s = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} \neq s = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

No non-adjacent double error mis-correction

Block Diagram – 1 Spare Column



Bit Slice of Correction Logic



Results

Comparison of Triple-Error Mis-correction Probability for SEC-DED codes

Data Bits	[Hsiao 70]		[Richter 08]		Proposed Method					
					1 Spare Column		2 Spare Columns		3 Spare Columns	
	XORs	Miscorrected	XORs	Miscorrected	XORs	Miscorrected	XORs	Miscorrected	XORs	Miscorrected
16	48	1,000 (64.9%)	-	-	58	448 (25.3%)	70	176 (8.7%)	76	52 (2.3%)
32	96	5,452 (59.6%)	115	4, 284 (46.8%)	118	2,548 (25.8%)	129	1,200 (11.3%)	138	588 (5.1%)
64	181	33,568 (56.2%)	250	26,616 (44.6%)	265	16,176 (26.0%)	308	9,084 (14.1%)	351	7,392 (11.0%)

Results

Comparison of Non-Adjacent Double-Error Mis-correction Probability
for SEC-DAEC codes

Data Bits	[Dutta 07]		[Richter 08]		Proposed Method					
					1 Spare Column		2 Spare Columns		3 Spare Columns	
	XORs	Miscorrected	XORs	Miscorrected	XORs	Miscorrected	XORs	Miscorrected	XORs	Miscorrected
16	48	118 (56.2%)	-	-	55	68 (29.4%)	62	33 (13.0%)	67	24 (8.7%)
32	96	379 (53.4%)	115	274 (39.0%)	117	203 (27.4%)	130	108 (13.8%)	140	72 (8.8%)
64	224	1316 (53.0%)	250	864 (34.8%)	263	688 (26.9%)	306	469 (17.8%)	353	395 (14.6%)

Results

Comparison of Random and Exhaustive Searches for Obtaining Optimal Result

Data Bits	Triple-Error Miscorrection Probability Adding 1 Spare Row				Non-Adjacent Double-Error Miscorrection Probability Adding 1 Spare Row			
	Random Search		Exhaustive Search		Random Search		Exhaustive Search	
	XORs	Miscorrected	XORs	Miscorrected	XORs	Miscorrected	XORs	Miscorrected
16	56	453 (25.5%)	58	448 (25.3%)	56	72 (31.2%)	55	68 (29.4%)
18	63	352 (13.5%)	63	352 (13.5%)	65	51 (17.0%)	65	51 (17.0%)
20	76	496 (15.1%)	76	496 (15.1%)	73	65 (17.2%)	73	65 (17.2%)

Conclusion

- Scheme for improving reliability of SEC-DED/SEC-DAEC code
- Utilizes already existing resources
- In presence of both spare rows and columns, spare rows may be used first for the repair process
- Save more spare columns for implementing the above scheme