DRAM Faults: Data from the Field

VILAS SRIDHARAN
RAS ARCHITECTURE

SEPTEMBER 18, 2013
PUBLIC

INTRODUCTION

- **Architectural & micro-architectural approaches to reliability**

- **To get it right, you must know the faults you expect**

- This presentation focuses on fault modeling in DRAM

INTRODUCTION

- **Dynamic random-access memory (DRAM)**
  - – Used for almost all computer main memory
  - – Single-capacitor memory
    - Charged = Logic 1
    - Discharged = Logic 0
  - – Reads are destructive – must rewrite data after read (“precharge”)
  - – Capacitors lose charge over time – must periodically rewrite data (“refresh”)

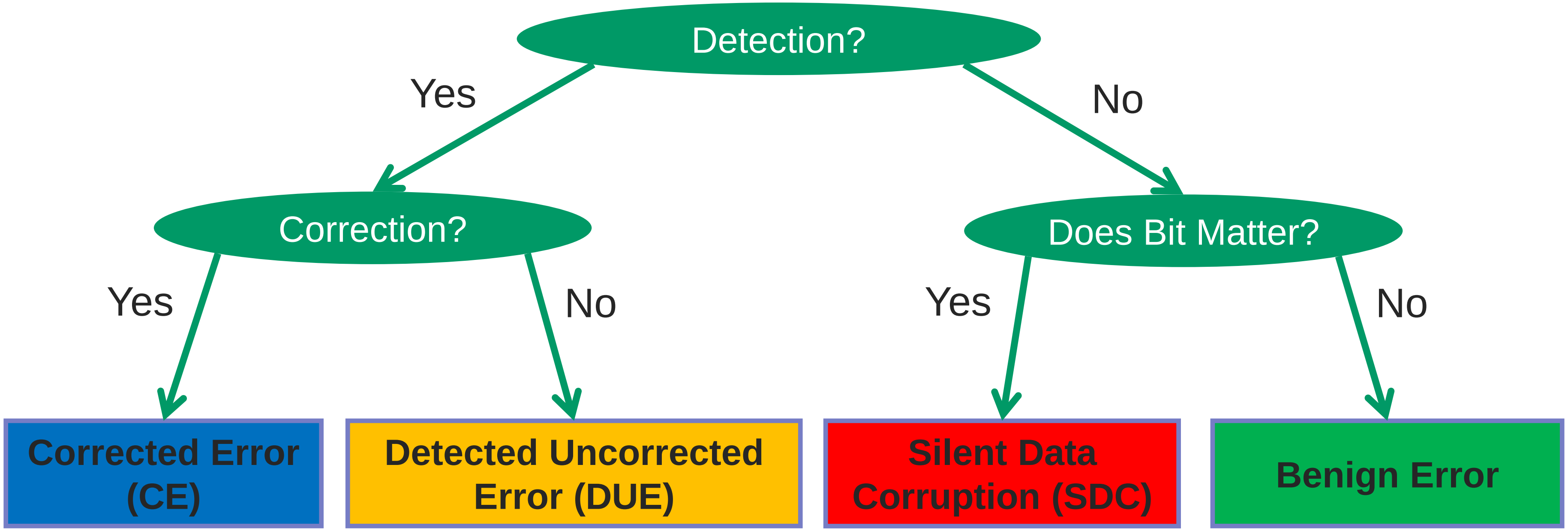
- **DRAM reliability is important today**
  - – Laptop: O(1-10 GB) of DRAM
  - – Petascale supercomputer: O(10-100 TB) of DRAM

- **DRAM reliability will be critical in the future**
  - – Exascale supercomputer: O(1-100 PB) of memory
  - – In-package (die-stacked) DRAM

- Understanding DRAM faults is critical to providing appropriate levels of reliability

TERMINOLOGY

- **Fault**
  - The underlying cause of an error, such as a stuck-at bit or high-energy particle strike
- **Error**
  - An incorrect state resulting from an active fault, such as an incorrect value in memory
- **Correction vs. repair**



Avizienis, IEEE TDSC, Jan.-Mar. 2004

STUDIES OF DRAM FAULTS

Type	Title	Authors	Publication	Year
Lab	**Alpha-particle induced soft errors in dynamic memories**	May and Woods	*IEEE Transactions on Electron Devices*	1979
	Comparison of accelerated DRAM soft error rates measured at component and system level	Borucki et al.	*IEEE Reliability Physics Symposium*	2008
Small-scale field studies	**A large-scale study of failures in high-performance computing systems**	Schroeder and Gibson	*Dependable Systems and Networks (DSN)*	2006
	A Memory Soft Error Measurement on Production Systems	Li et al.	*USENIX*	2007
	A Realistic Evaluation of Memory Hardware Errors and Software System Susceptibility	Li et al.	*USENIX*	2010
Large-scale field studies	**DRAM Errors in the Wild: A Large-Scale Field Study**	Schroeder et al.	*SIGMETRICS*	2009
	Cosmic Rays Don't Strike Twice: Understanding the Nature of DRAM Errors and the Implications for System Design	Hwang et al.	*ASPLOS*	2012
	A Study of DRAM Failures in the Field	Sridharan and Liberty	*SuperComputing! (SC12)*	2012
	Feng Shui of Supercomputer Memory: Positional Effects in DRAM and SRAM Faults	Sridharan et al.	*SuperComputing! (SC13)*	2013

- We have lots of data on DRAM faults
- Be careful when interpreting / comparing different studies

EXECUTIVE SUMMARY

- **Fault rates**
  - Fault rate *per bit* is trending down with each technology generation
  - Fault rate *per device* is roughly flat

- **Fault modes**
  - Faults occur in logical / physical entities
  - Many *single points of failure* in a DRAM system

- **Choice of DRAM vendor matters quite a bit**
  - Large (4x) differences in observed fault rate among vendors

- **Memory channel reliability strongly affected by DRAM organization**
  - Need to match the mitigation technique to the channel layout
  - Expect unexpected faults

- **Current mitigation techniques won’t suffice in the future**
  - Die-stacking, lower-power, wider interfaces, etc.

OUTLINE

- Introduction
- **Fault types**
- **Fault modes**
- **Inter-vendor effects**
- **Mitigation techniques**
- **Future trends**

FAULT TYPES

FAULT TYPES

- **Transient fault**
  - Return incorrect data until overwritten
  - Random and not indicative of device damage

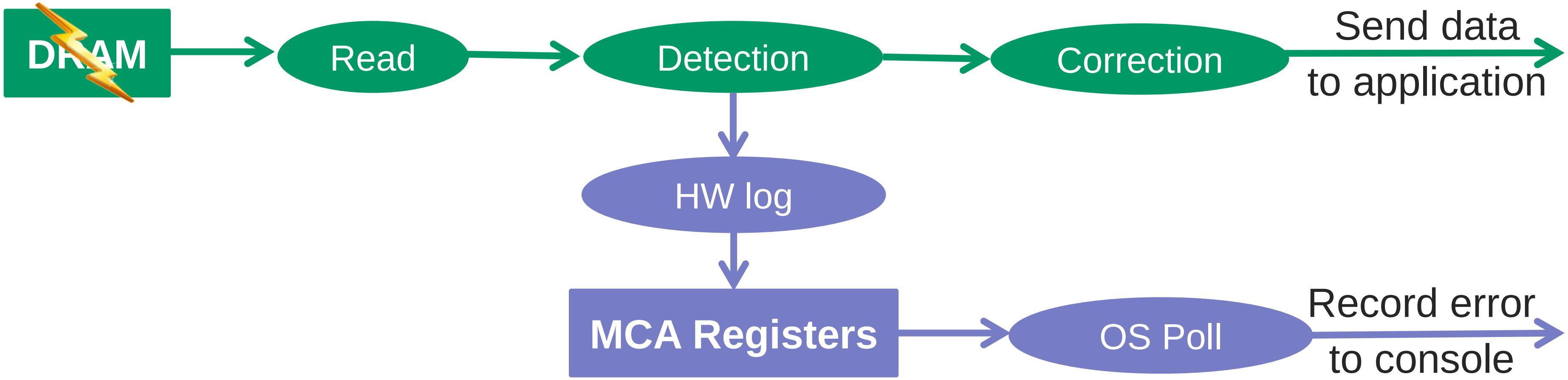
- **Hard fault**
  - **Consistently** return an incorrect value
  - Repair by disabling or by replacing the faulty device

- **Intermittent fault**
  - **Sometimes** return an incorrect value
  - Under specific conditions such as elevated temperature
  - Indicative of device damage or malfunction

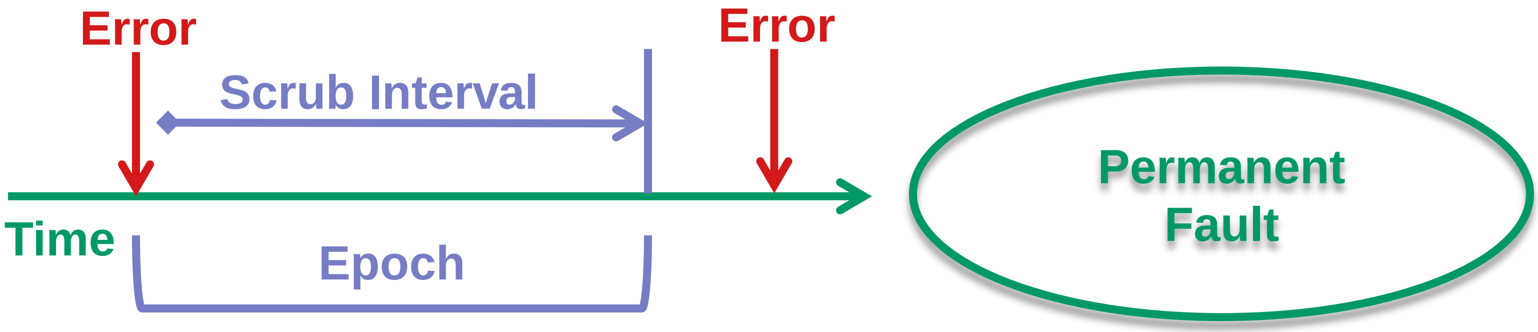
“Permanent” faults

IDENTIFYING FAULT TYPES IN THE FIELD

- **Data collection**

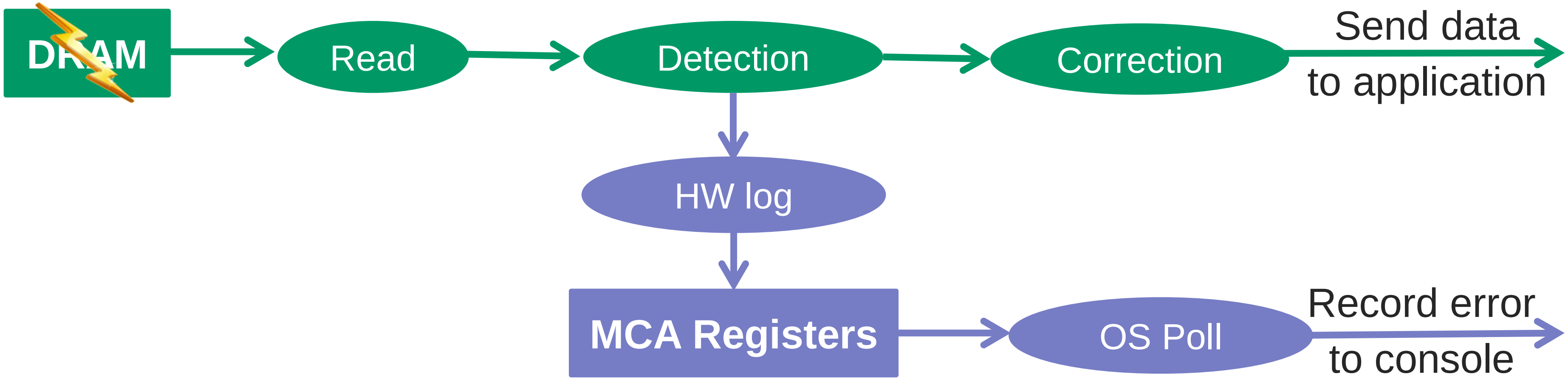


- **Identifying permanent faults**
  - HW scrubber periodically reads each DRAM location, corrects any errors found, writes corrected data back to DRAM

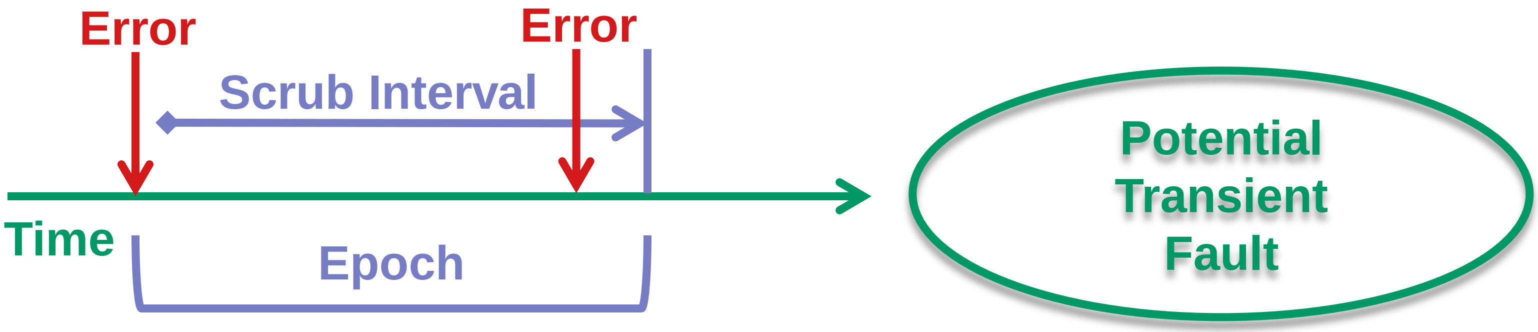


IDENTIFYING FAULT TYPES IN THE FIELD

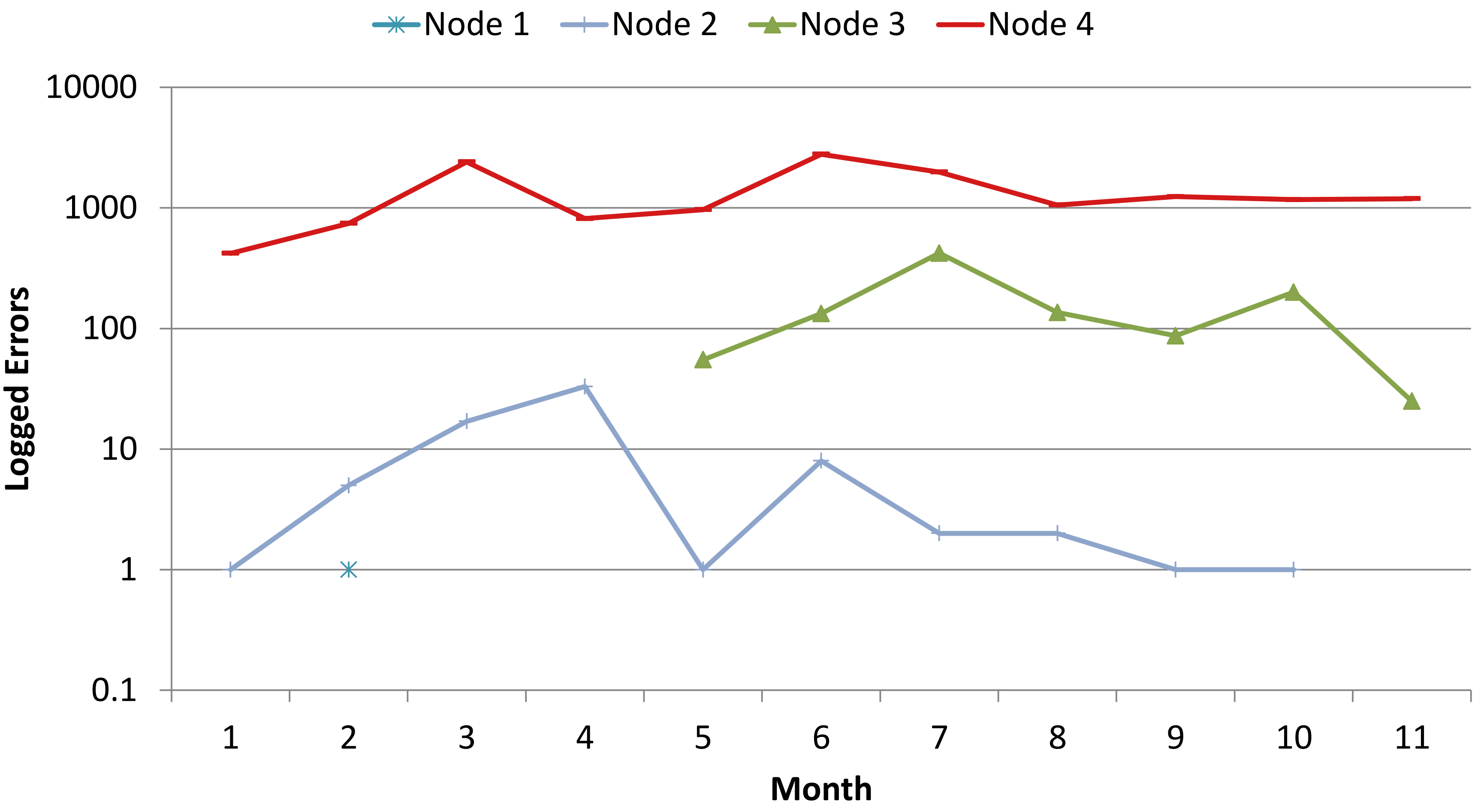
- **Data collection**



- **Identifying permanent faults**
  - HW scrubber periodically reads each DRAM location, corrects any errors found, writes corrected data back to DRAM



ERROR PATTERNS

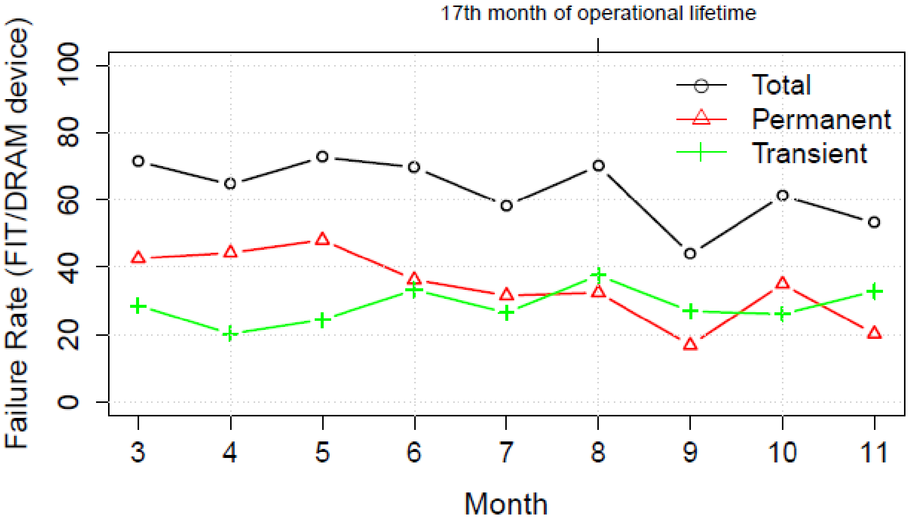


- DRAM experiences transient and permanent faults

Sridharan and Liberty, *A Study of DRAM Failures in the Field*, SC 2012

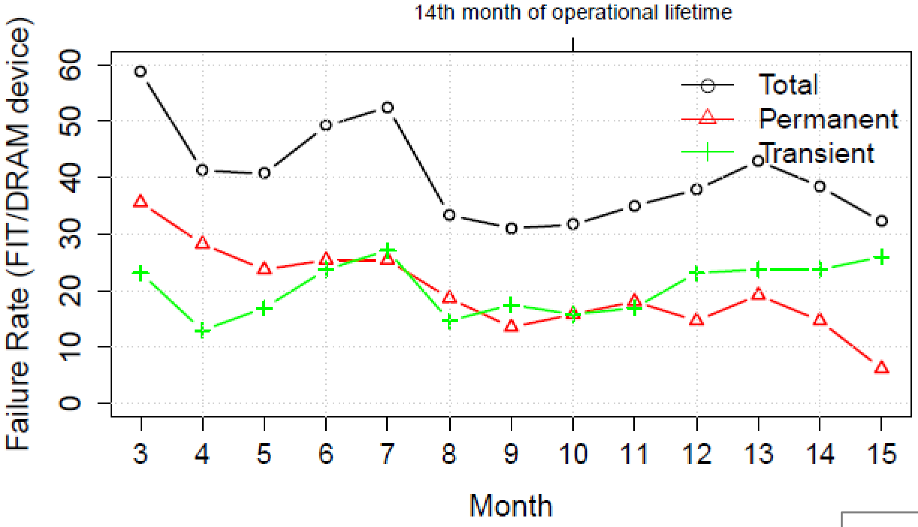
FAULT RATE AND FAULT TYPES

DDR-2



% Faulty DRAM devices	0.09%
% Faulty DIMMs	1.6%
Fault Rate (FIT/Mbit)	0.066
Fault Rate (FIT/device)	66.1

DDR-3

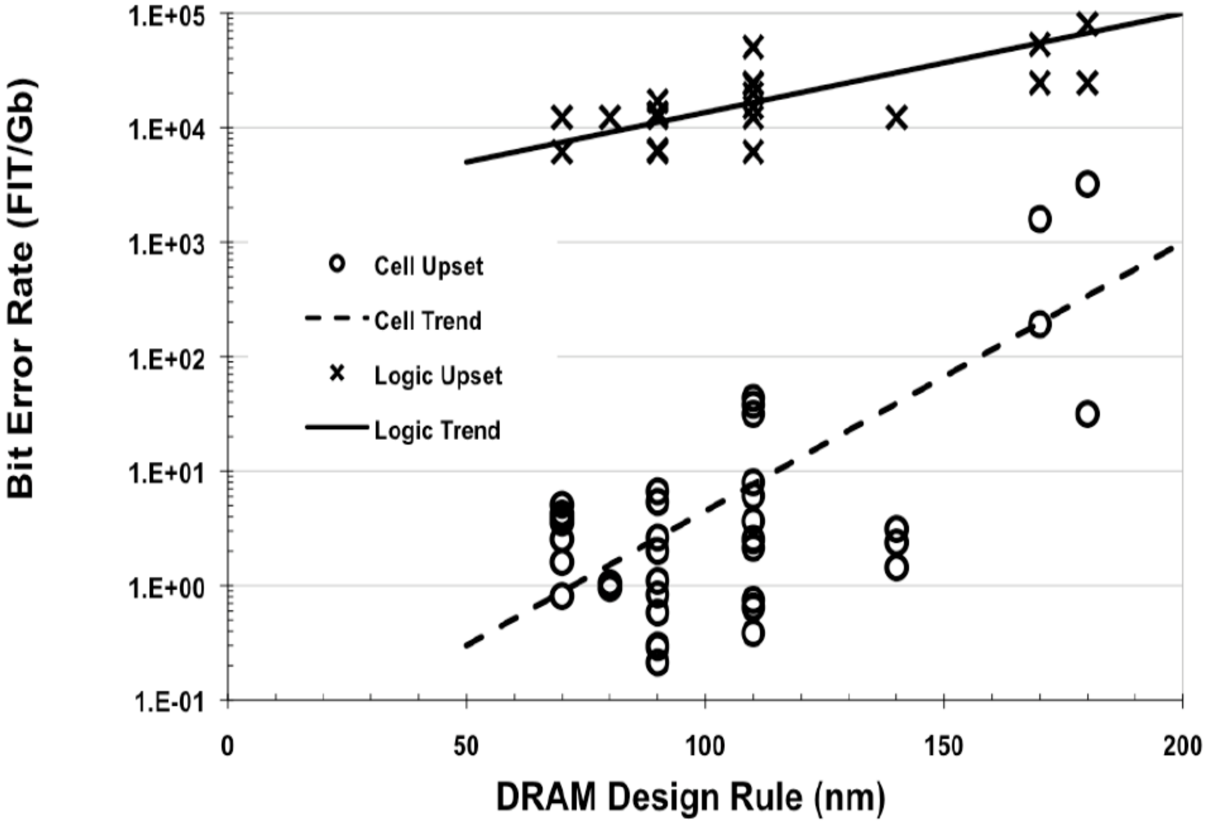


% Faulty DRAM devices	0.038%
% Faulty DIMMs	1.32%
Fault Rate (FIT/Mbit)	0.044
Fault Rate (FIT/device)	40.3

- Declining permanent fault rate over time
- Approx. constant transient fault rate

Sridharan et al., *Feng Shui of Supercomputer Memory*, SC 2013

TRANSIENT FAULT TREND



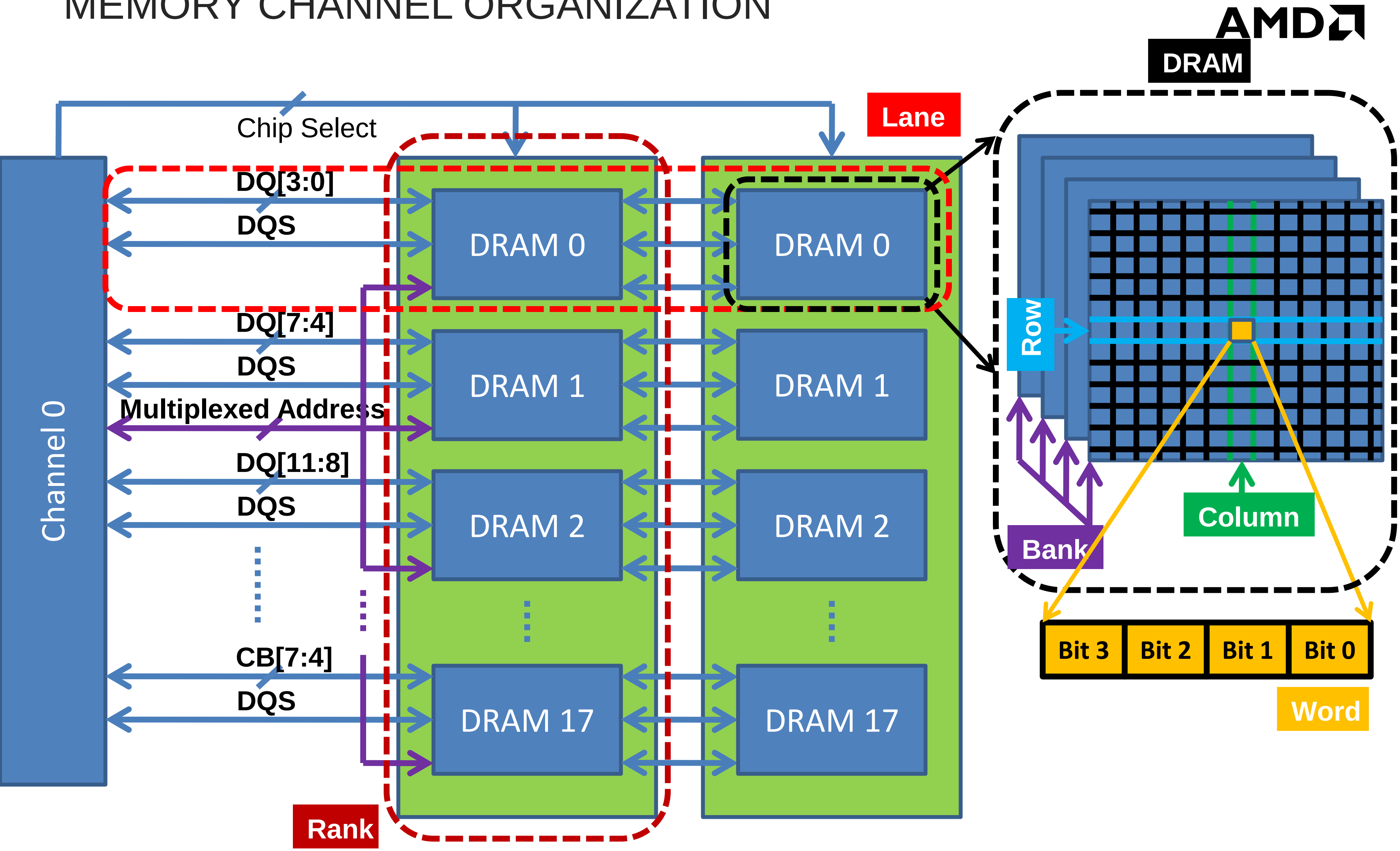
- DRAM cell upset rate trending downwards because Qcrit flat but cell area shrinking

- For DRAM faults, FIT/device is a better model than FIT/bit at a system level
  - Control logic upsets are becoming more significant

Slayman, *Soft Error Trends and Mitigation in Memory Devices*, RAMS 2011

FAULT MODES

MEMORY CHANNEL ORGANIZATION



- Each logical entity (e.g. row, rank) shares control logic
- Control logic is a single point of failure for bits in each logical entity

FAULT MODES

DDR-2

Fault Mode	% Faulty DRAMs
Single-bit	49.7%
Single-word	2.5%
Single-column	10.6%
Single-row	12.7%
Single-bank	16.3%
Multi-bank	2.5%
Multi-rank	5.5%

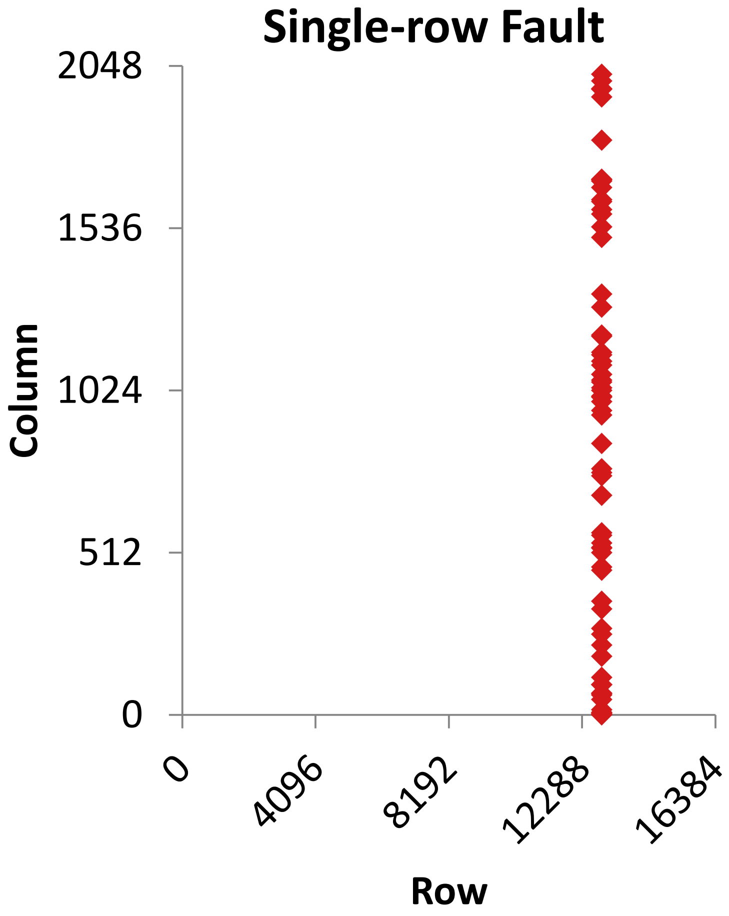
DDR-3

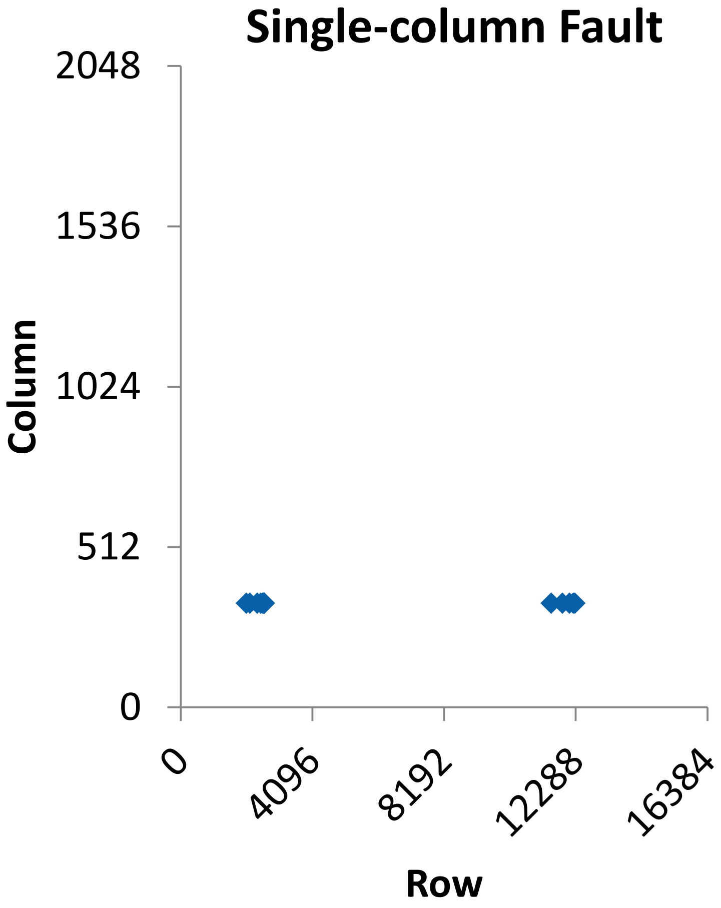
Fault Mode	% Faulty DRAMs
Single-bit	67.7%
Single-word	0.2%
Single-column	8.7%
Single-row	11.8%
Single-bank	9.6%
Multi-bank	1.0%
Multi-rank	1.1%

- DDR-2 and DDR-3 experience similar fault modes
- Virtually all logical entities experience faults

Sridharan and Liberty, *A Study of DRAM Failures in the Field*, SC 2012
Sridharan et al., *Feng Shui of Supercomputer Memory*, SC 2013

A CLOSER LOOK AT MULTI-BIT FAULTS

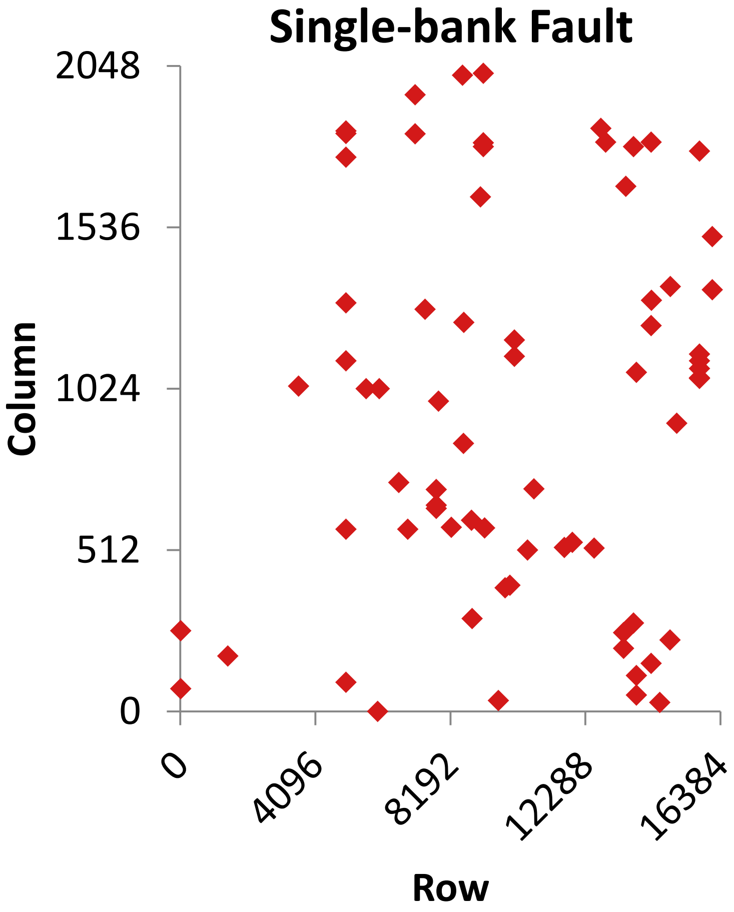


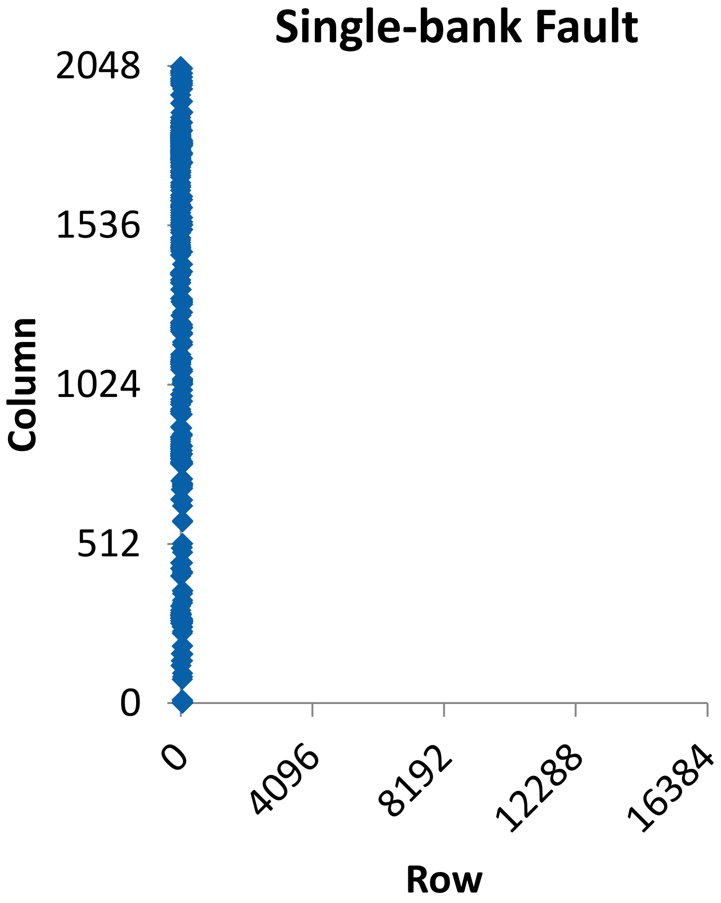


- Row faults affect whole row; column faults affect sub-columns

Sridharan and Liberty, *A Study of DRAM Failures in the Field*, SC 2012

A CLOSER LOOK AT MULTI-BIT FAULTS





- Two distinct bank fault patterns: “Spread” and “Row-cluster”

Sridharan and Liberty, *A Study of DRAM Failures in the Field*, SC 2012

A CLOSER LOOK AT MULTI-BIT FAULTS

Fault Mode	Faulty DQs			
	1	2	3	4
Single-column	85.8%	3.3%	0.8%	10.0%
Single-row	31.1%	66.8%	1.4%	0.7%
Single-bank	55.5%	23.0%	3.8%	17.8%
Multi-bank	17.5%	33.3%	3.5%	45.6%
Multi-rank	7.5%	7.1%	1.8%	83.6%

Two fault modes

One or two fault modes?

Suggests strobe (DQS) fault

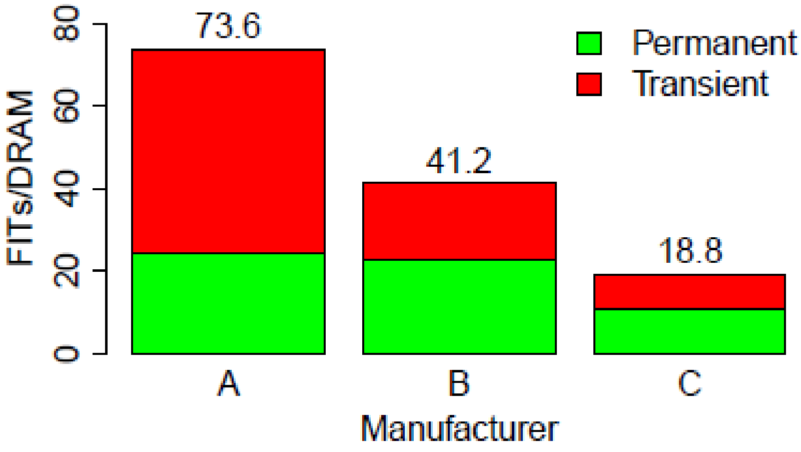
- Multi-bit faults often affect multiple data pins
- Appears to be multiple fault modes even within a logical entity

Sridharan and Liberty, *A Study of DRAM Failures in the Field*, SC 2012

VENDOR EFFECTS

VENDOR EFFECTS

Fault Mode	Vendor A	Vendor B	Vendor C
Single-bit	64.6%	69.5%	58.4%
Single-word	0%	0.3%	0%
Single-column	8.7%	8.8%	11.9%
Single-row	12.2%	10.6%	14.9%
Single-bank	13.5%	7.8%	9.9%
Multiple-bank	1.3%	0.7%	2.0%
Multiple-rank	1.3%	3.0%	3.0%

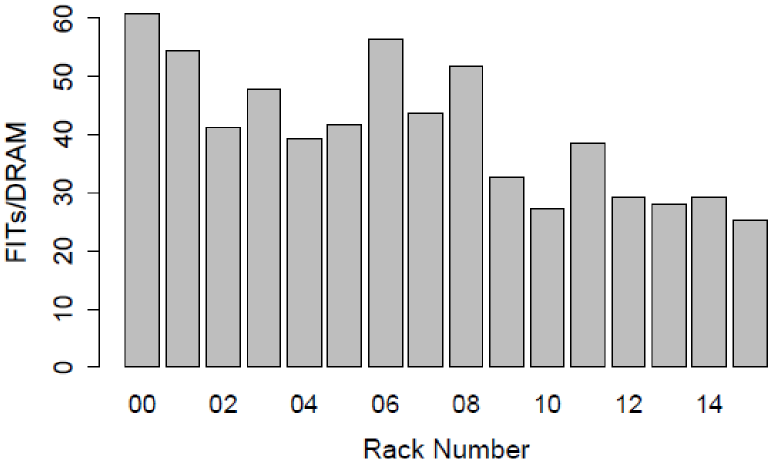


- Overall fault rate per vendor

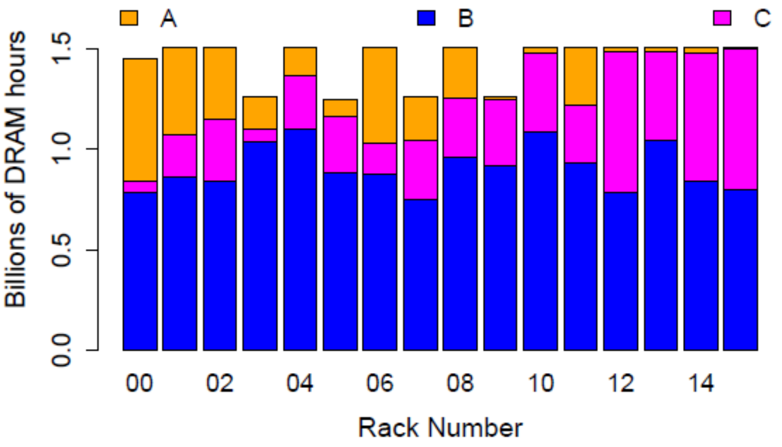
- Fault modes are present across vendors
- Fault rates differ significantly by vendor

Sridharan et al., *Feng Shui of Supercomputer Memory*, SC 2013

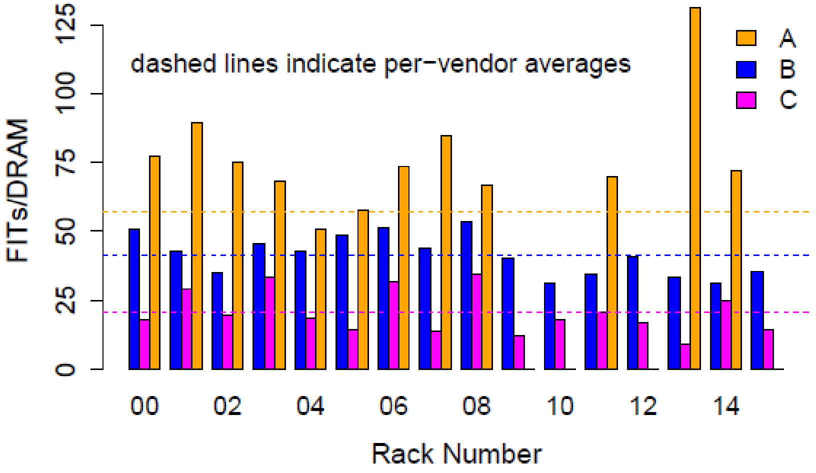
VENDOR EFFECTS



- A correlation to physical location...



- ...is due to non-uniform distribution of vendor...

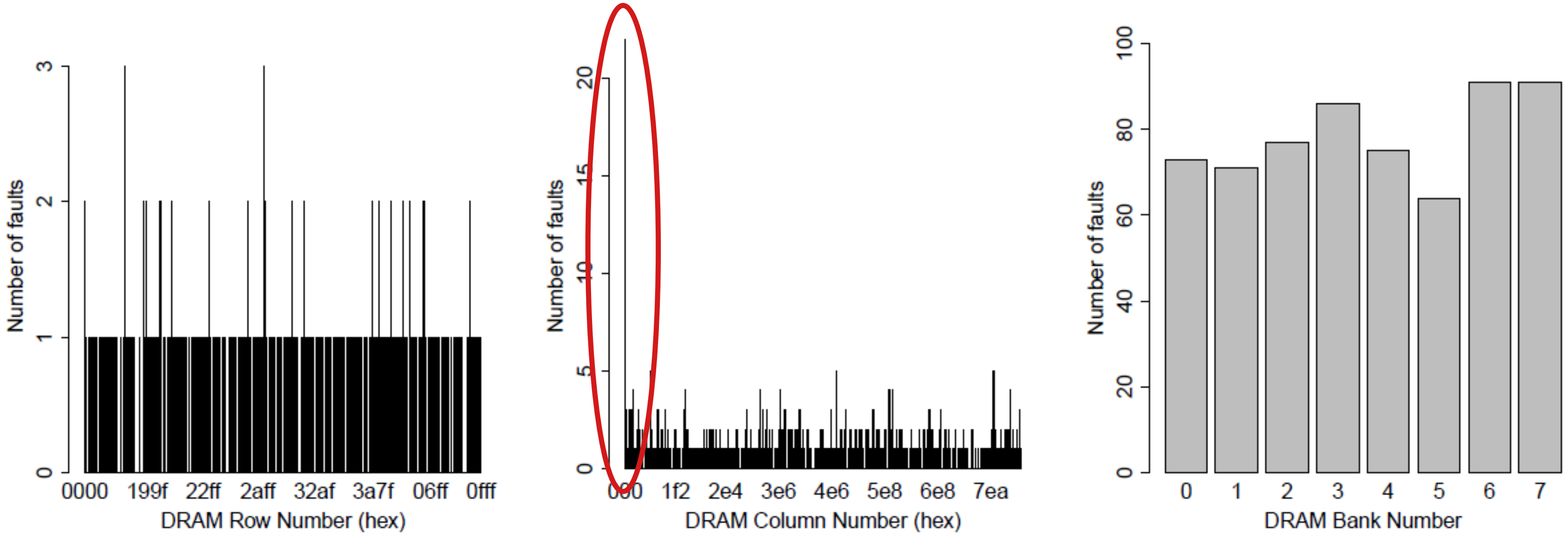


- ...and disappears when examined by vendor.

- DRAM reliability studies must account for DRAM vendor or risk inaccurate results

Sridharan et al., *Feng Shui of Supercomputer Memory*, SC 2013

FAULT DISTRIBUTION WITHIN A DEVICE (DDR-3)

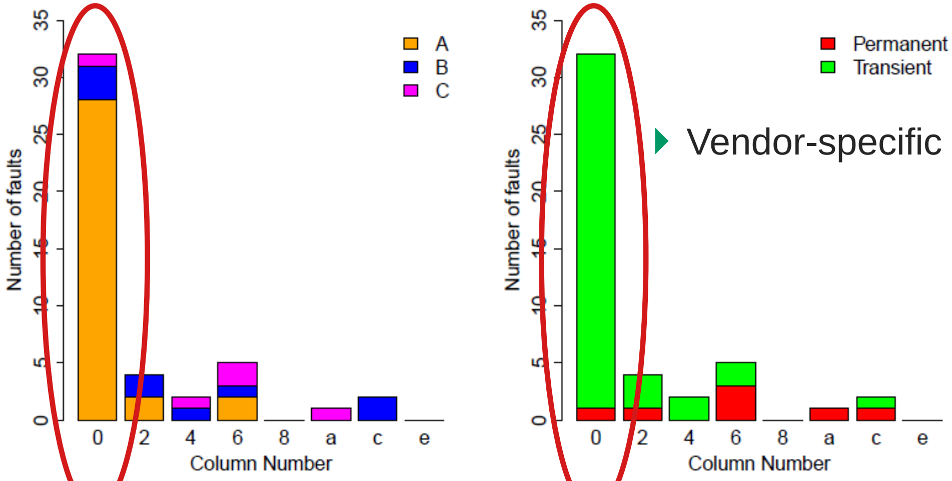


(a) Distribution over rows

(b) Distribution over columns

(c) Distribution over banks

- Distribution of single-bit faults within a DRAM device



- Vendor-specific effect

(a) Fault rate by vendor

(b) Fault rate by fault type

- No other correlation between fault rate and DRAM location

Sridharan et al., *Feng Shui of Supercomputer Memory*, SC 2013

MITIGATION TECHNIQUES

OVERVIEW OF STANDARD MITIGATION TECHNIQUES

Correction

- **Single-error Correction / Double-error Detection (SEC-DED ECC)**
  - Corrects any single-bit error, detects any double-bit error

- **Chipkill**
  - Corrects any error in one (single-chipkill) or two (double-chipkill) DRAM devices
  - Uses linear block (symbol-correction) ECCs such as Reed-Solomon codes

Repair

- **Scrubber**
  - Periodically reads each DRAM location, corrects any errors found, write data to DRAM
  - Repairs transient faults before a second fault (transient or permanent) can occur

- **Sparing**
  - Provides spare memory (e.g., rank, chip) that can be swapped in when a fault is detected

CHIPKILL: BASICS

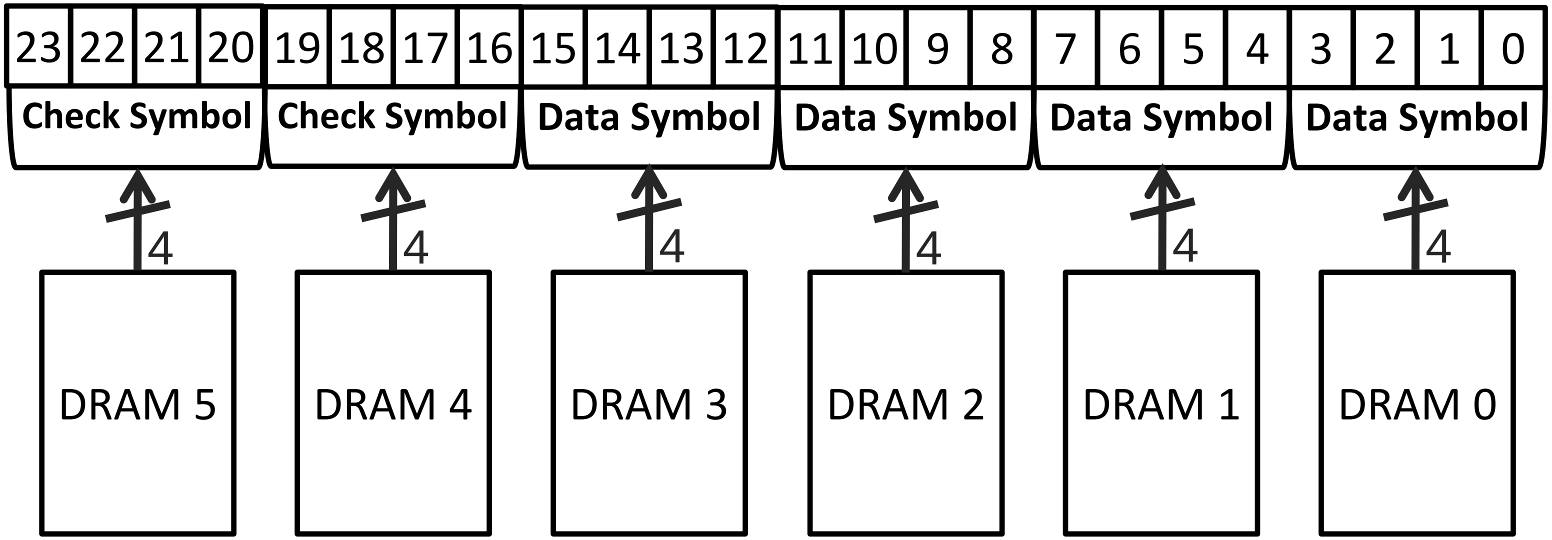
- **Chipkill uses linear block (*symbol-correction*) error-correcting codes**
  - Group each data word into N *data symbols* (blocks) of M adjacent bits
  - Add K *check symbols* to the data word

- **One example: Reed-Solomon code**
  - Detects up to *K* symbols in error, corrects up to *K/2* symbols in error
  - *Erases* up to *K* symbols in error if locations are already known

- **Memory sub-system layout**
  - Each DRAM contributes exactly one symbol to a data word
  - *k*-device fault → *k*-symbol error
  - If $k \leq K$ → detected
  - If $k \leq K/2$ → corrected

- **Overhead**
  - Single-symbol correction → K=2
  - Double-symbol detection → K=3
  - Multiple cycles for encode/decode

CHIPKILL: BASICS



23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

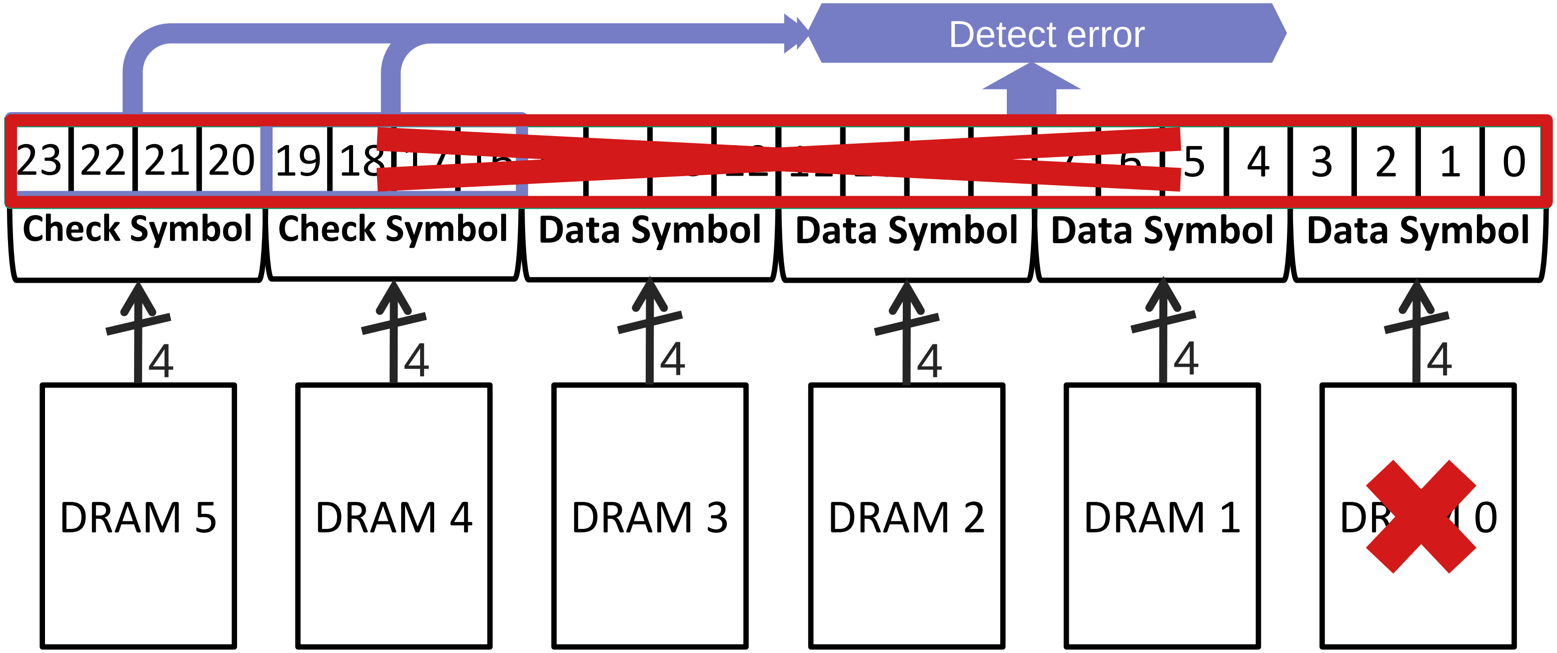
Check Symbol | Check Symbol | Data Symbol | Data Symbol | Data Symbol | Data Symbol

M = 4

N = 4

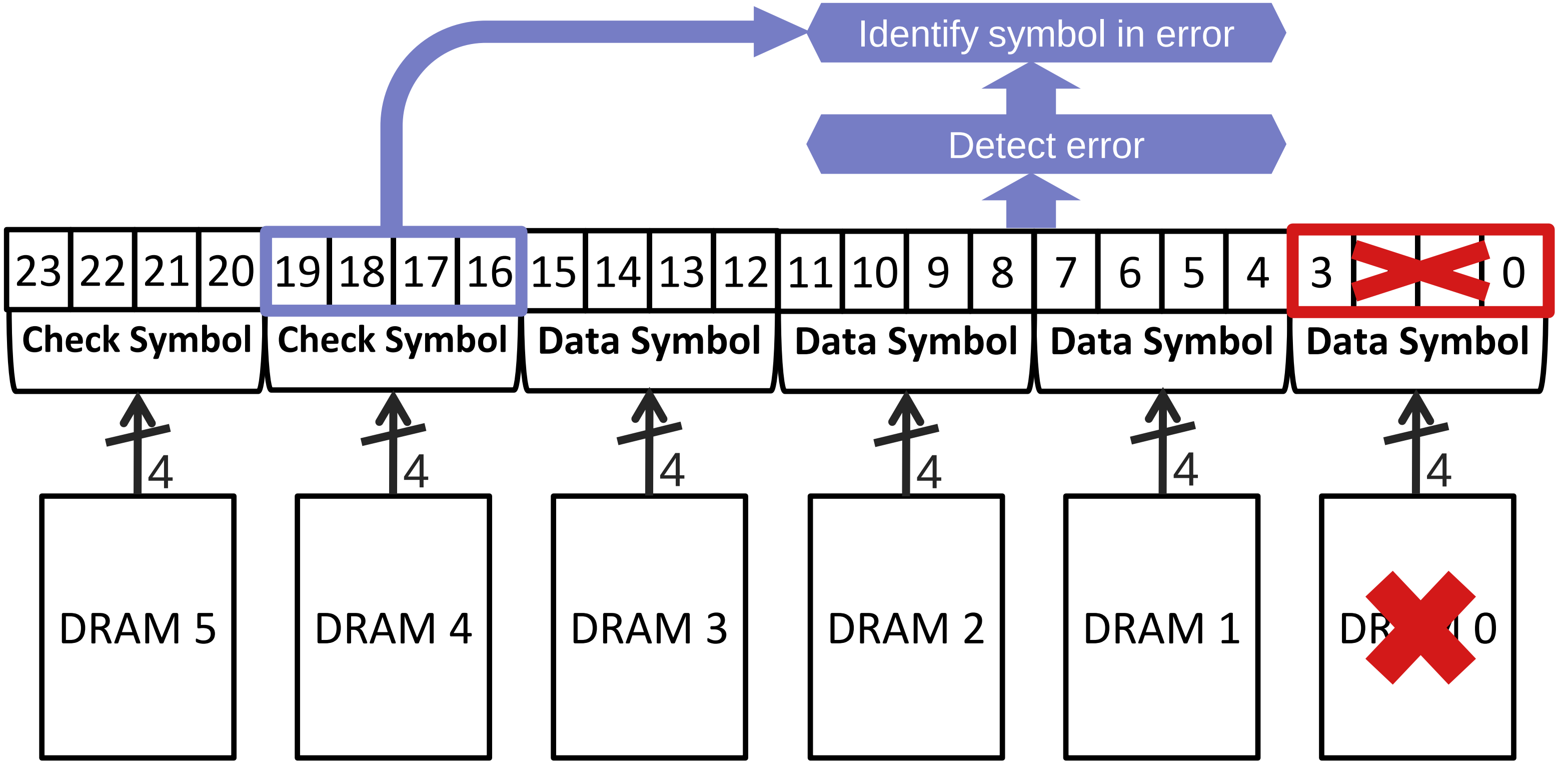
K = 2

CHIPKILL: BASICS



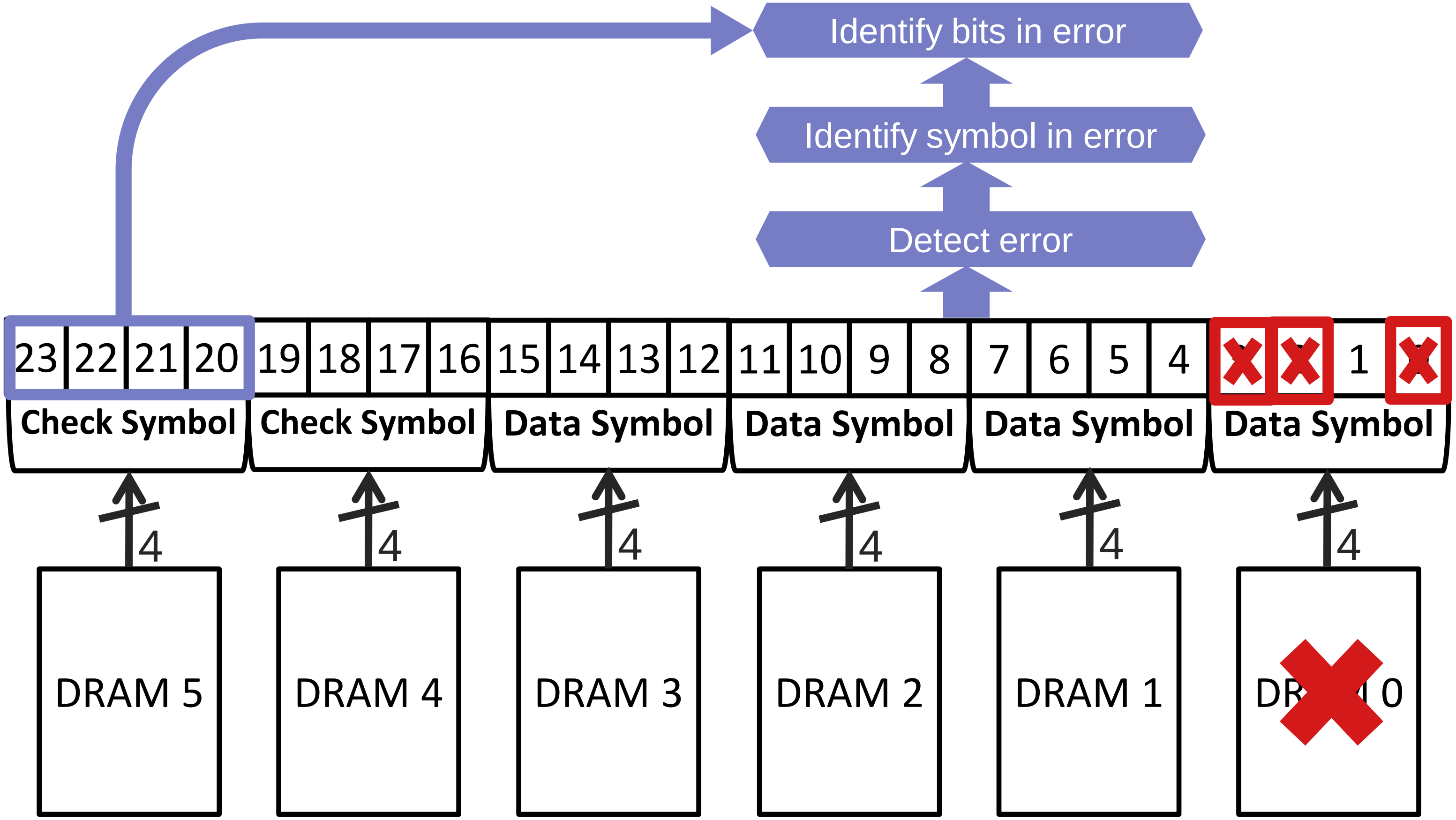
M = 4
N = 4
K = 2

CHIPKILL: BASICS



M = 4
N = 4
K = 2

CHIPKILL: BASICS



Identify bits in error

Identify symbol in error

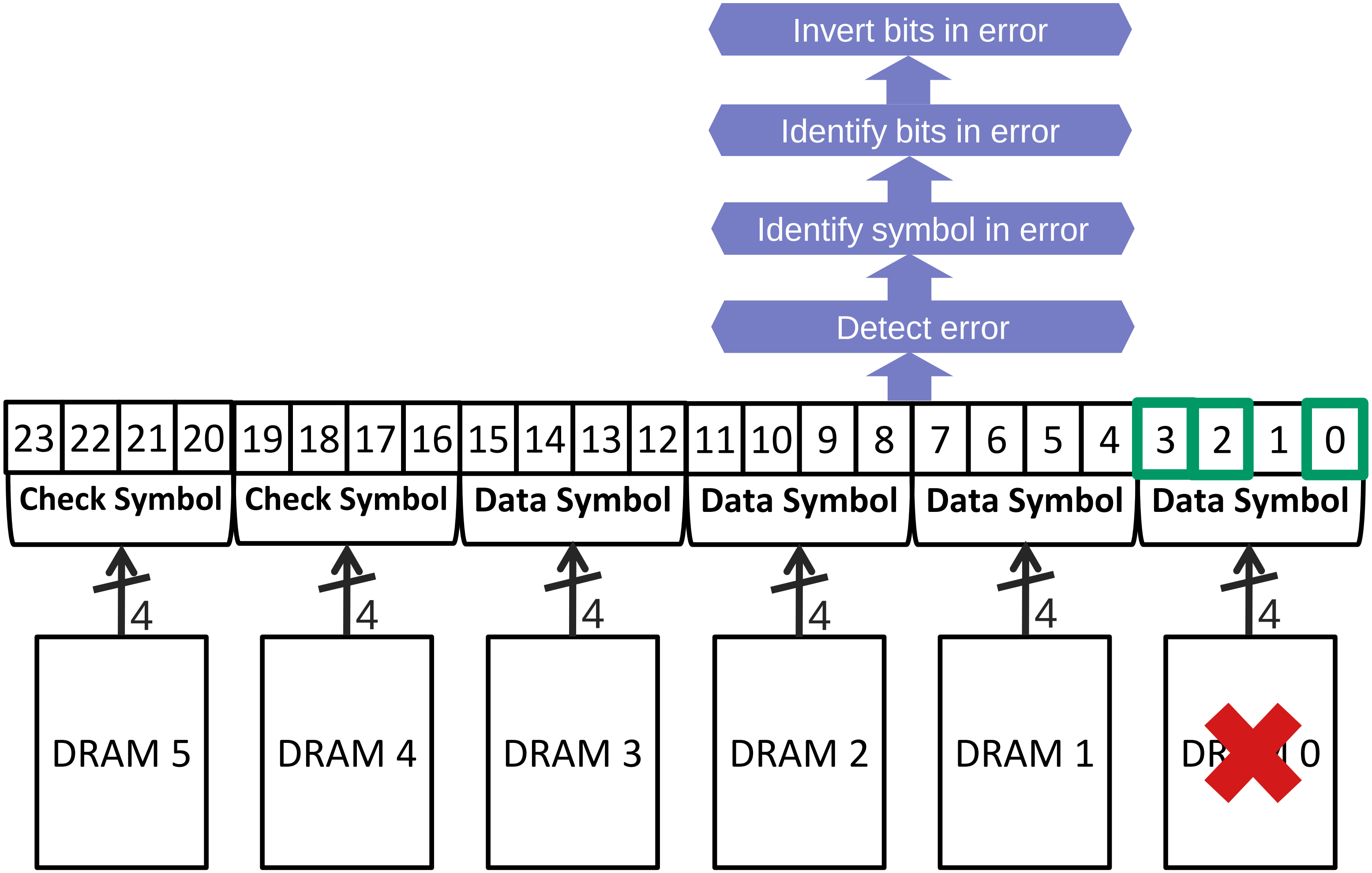
Detect error

M = 4

N = 4

K = 2

CHIPKILL: BASICS



Invert bits in error

Identify bits in error

Identify symbol in error

Detect error

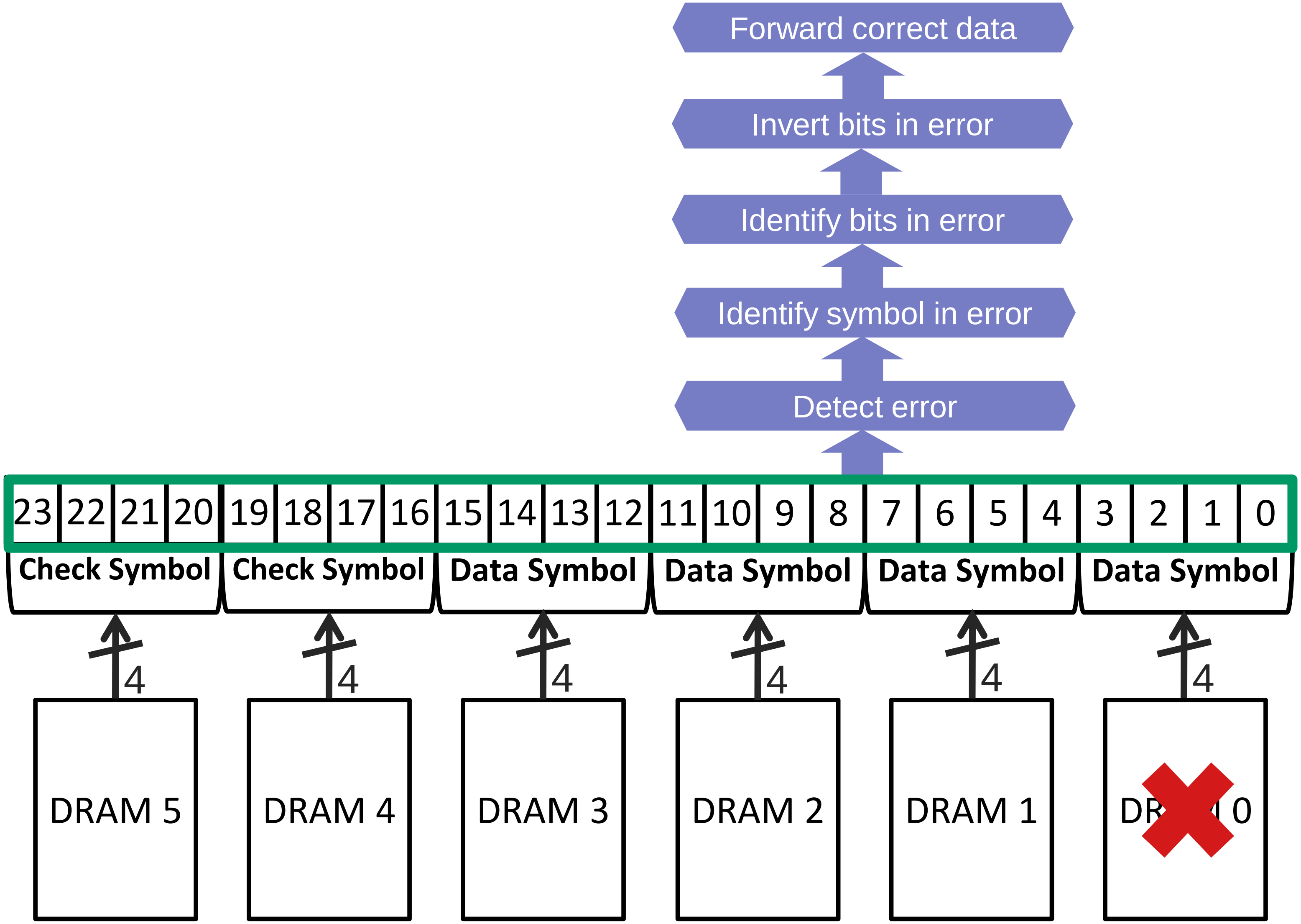
23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Check Symbol | Check Symbol | Data Symbol | Data Symbol | Data Symbol | Data Symbol

DRAM 5 | DRAM 4 | DRAM 3 | DRAM 2 | DRAM 1 | DRAM 0

M = 4
N = 4
K = 2

CHIPKILL: BASICS



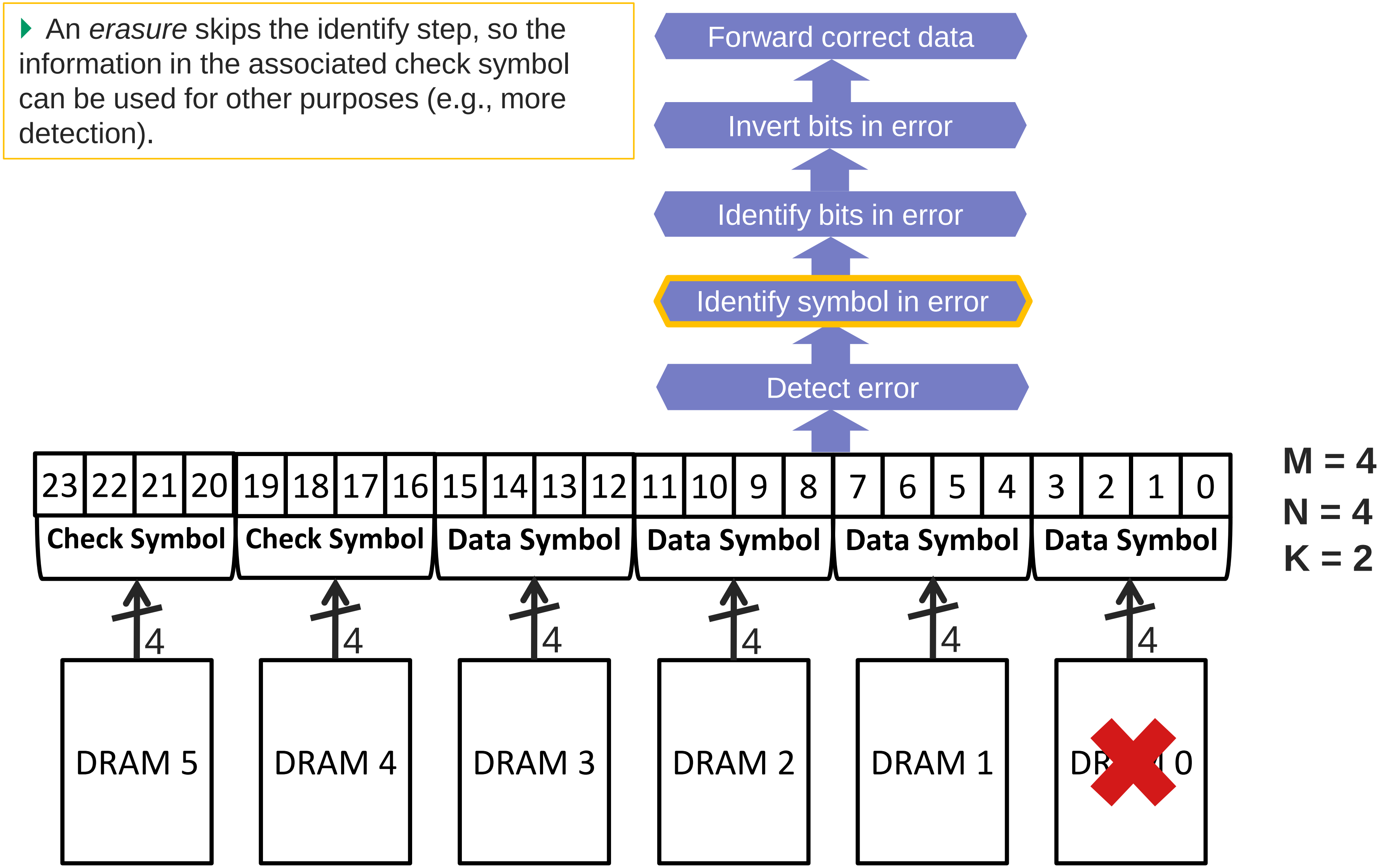
M = 4

N = 4

K = 2

CHIPKILL: BASICS

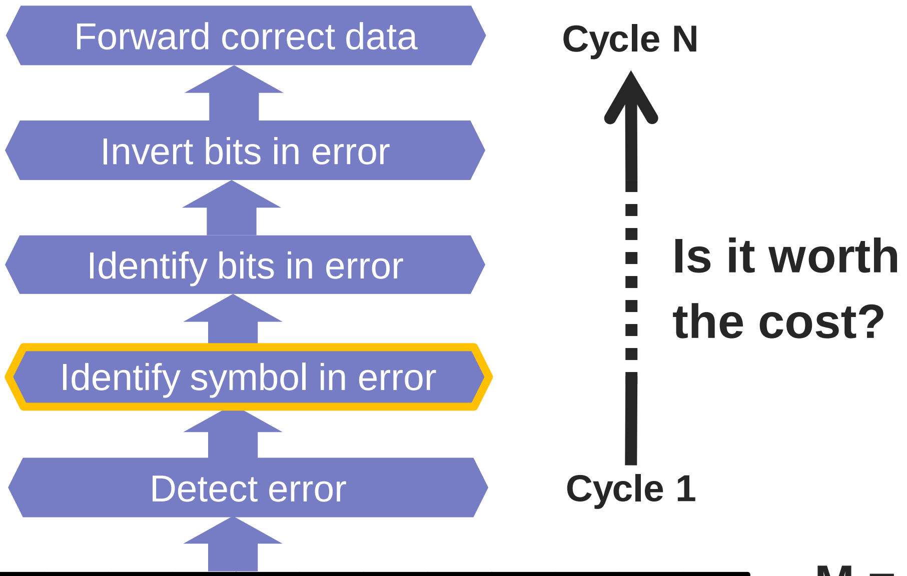
- An *erasure* skips the identify step, so the information in the associated check symbol can be used for other purposes (e.g., more detection).



CHIPKILL: BASICS

- An *erasure* skips the identify step, so the information in the associated check symbol can be used for other purposes (e.g., more detection).

- Actual implementation of code + logic is more complicated. This is just for illustration purposes.



Forward correct data

Invert bits in error

Identify bits in error

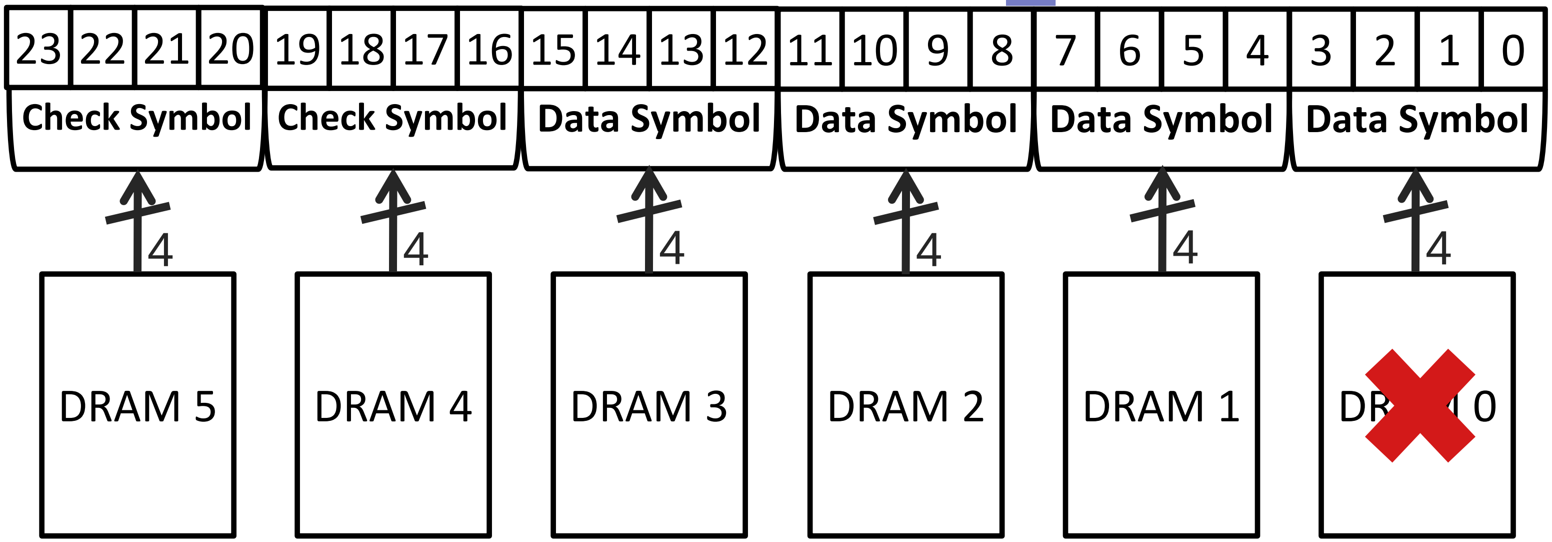
Identify symbol in error

Detect error

Cycle N

Cycle 1

Is it worth the cost?



23 22 21 20 | 19 18 17 16 | 15 14 13 12 | 11 10 9 8 | 7 6 5 4 | 3 2 1 0

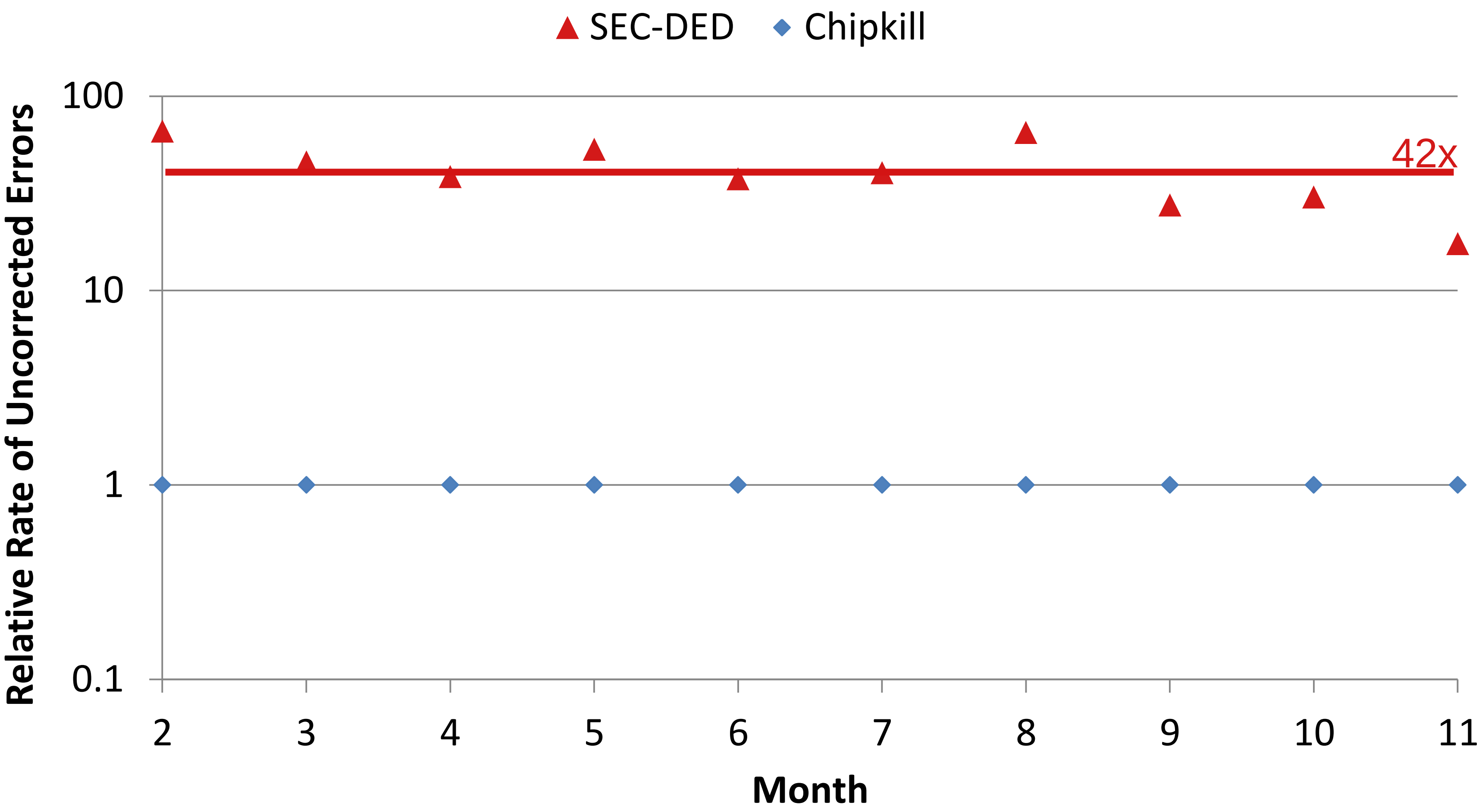
Check Symbol | Check Symbol | Data Symbol | Data Symbol | Data Symbol | Data Symbol

4 | 4 | 4 | 4 | 4 | 4

DRAM 5 | DRAM 4 | DRAM 3 | DRAM 2 | DRAM 1 | DRAM 0

M = 4
N = 4
K = 2

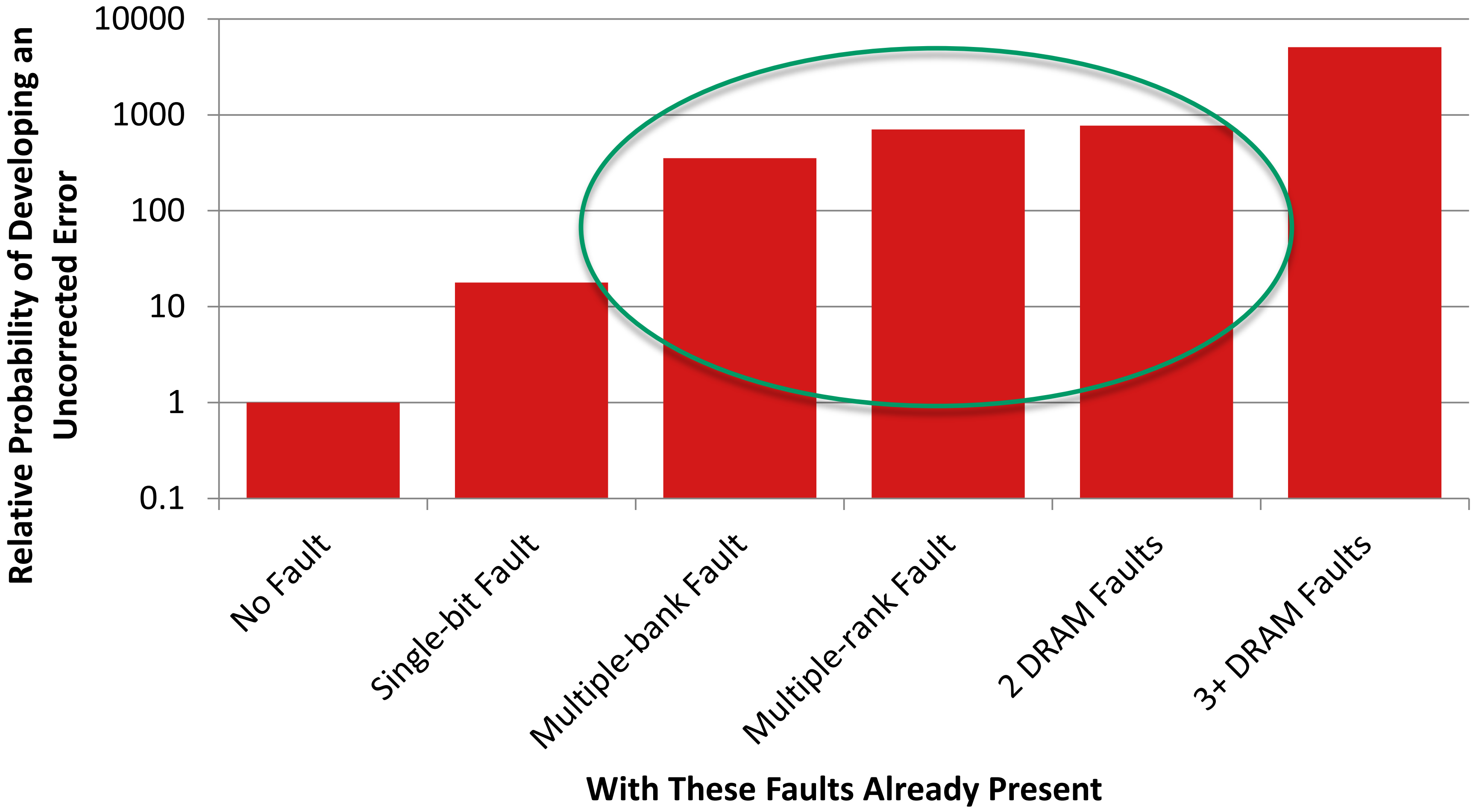
CHIPKILL: BENEFIT



- Chipkill improves uncorrected error rate by 42x compared to SEC-DED ECC

Sridharan and Liberty, *A Study of DRAM Failures in the Field*, SC 2012

CHIPKILL: EFFECT OF FAULTS ON FUTURE ERRORS



- Multi-bank / multi-rank faults act like multiple device faults

Sridharan and Liberty, *A Study of DRAM Failures in the Field*, SC 2012

FUTURE TRENDS

FUTURE TRENDS: CHIPKILL AT LOWER POWER

- **Goal: Provide chipkill while reducing memory power consumption**
  – Reduce the number of DRAM devices required per access
  – Most proposed techniques provide independent detection and correction resources

- **Virtual ECC (VECC)**
  – Store linear block code for detection only in dedicated DRAM devices
  – Store the correction portion of check code in main memory space
  – Doubles the tolerable symbol size for a given level of protection
  – On detected error, fetch additional correction resources

- **Localized and Tiered ECC (LOT-ECC)**
  – Provide error detection within each DRAM chip rather than across multiple chips
  – Provide separate error-correction resources within each chip
  – Tailored to observed fault modes: some single-device faults will result in SDC

- **Multi-line Error Correction (Multi-ECC)**
  – Provide detection (Reed-Solomon code) for each cache line
  – Amortize correction (column checksums) across multiple cache lines
  – On detected error, use checksums to identify chip(s) in error
  – Reprocess cache line using *erasure decoding* of the Reed-Solomon code

Yoon and Erez, ASPLOS 2010; Udipi et al., ISCA 2012; Jian et al., SC 2013

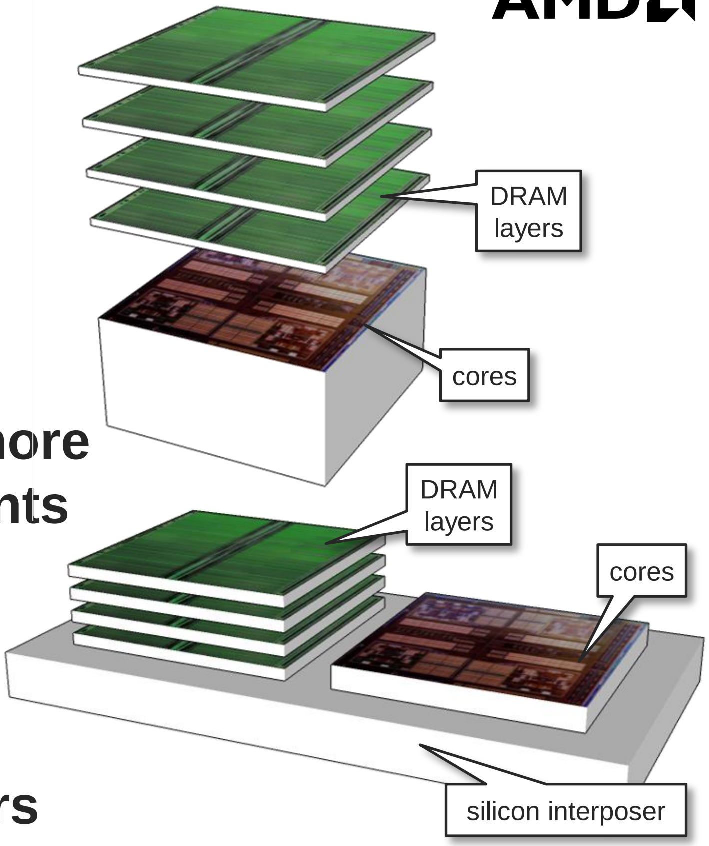
FUTURE TRENDS: DIE-STACKED DRAM

- **Die stacking is coming along (esp. DRAM)**
  - JEDEC Wide-IO and HBM standards
  - Micron Hybrid Memory Cube™

- **For the time being, likely to be utilized in more expensive and higher-end product segments**
  - HPC, data center, gaming / enthusiast GPU

- **May not be able to stack enough DRAM for target capacities in HPC and datacenters**
  - Perhaps for some segments or cloud application classes but not for others (e.g., HPC)

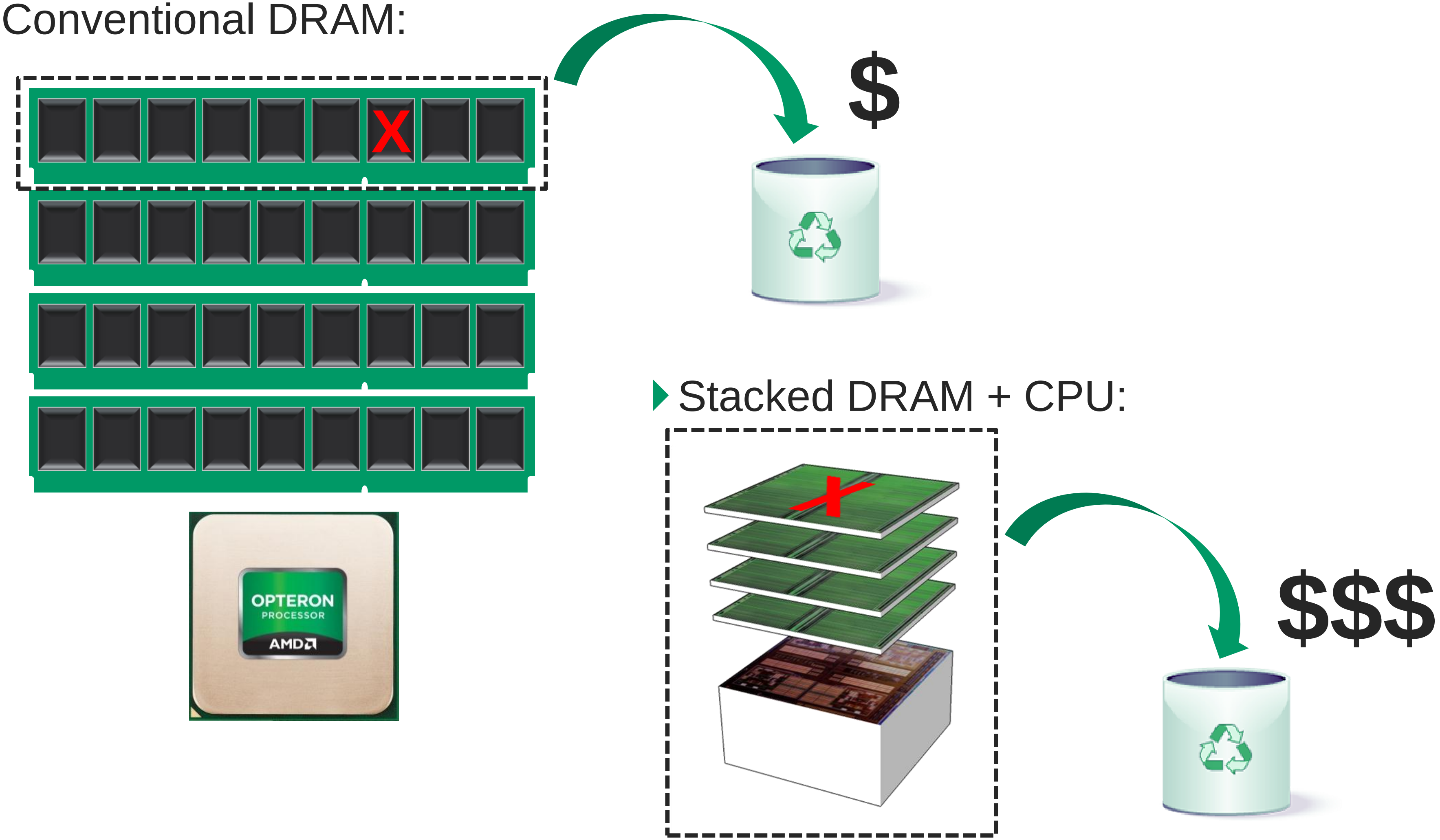
- **Past work has explored how to utilize stacked DRAM as a large software-transparent cache**



Sim et al., *Resilient Die-Stacked DRAM Caches*, ISCA 2013

DIE-STACKED DRAM: SERVICEABILITY

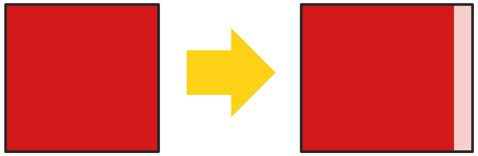
- Die-stacked DRAM changes the ***serviceability*** of memory
- Conventional DRAM:



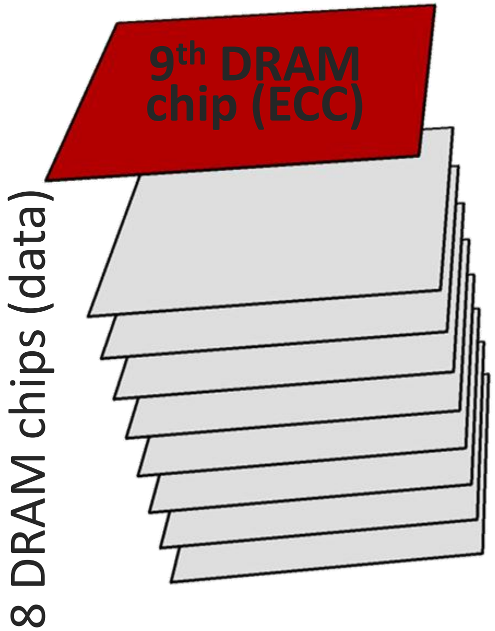
- Stacked DRAM + CPU:

Sim et al., *Resilient Die-Stacked DRAM Caches*, ISCA 2013

HOW TO ADD 12.5% STORAGE TO 3D DRAM?



- Make row 12.5% wider? 👎
  - Memory vendor must support two different ICs
- Add more chips?



Must have 8 chips 👎

Not practical;
Stacking concerns 👎

Wasteful
Extra cost 👎

Sim et al., *Resilient Die-Stacked DRAM Caches*, ISCA 2013

RESILIENT DIE-STACKED DRAM CACHES: SUMMARY

- **Comprehensive error detection and correction for DRAM caches**
  - Basic error correction: all single-bit upsets correctable
  - Coarse-grain error correction: 99.9993% of single-column, -row, and -bank failures
  - Superior SDC protection: 5 orders of magnitude reduction in SDC FIT rate compared to basic SEC-DED ECC

- **Many existing techniques synthesized specifically for DRAM caches**
  - Basic SEC (no extra detection) for common single-bit upsets
  - CRC for very strong multi-bit detection
  - Address / Index hashing for decoder failure detection
  - Duplicate-on-write: RAID1-like approach for correction

- **Low-cost, flexible, and general**
  - Works with **non-ECC**, unmodified stacked DRAM
    - HBM-like interface used in this study
  - Optional enablement for different market segments / needs
  - A specific design is presented, but this is a ***general framework*** to support RAS in DRAM caches

Sim et al., *Resilient Die-Stacked DRAM Caches*, ISCA 2013

OTHER DIRECTIONS

- **What to do after a bank / channel / stack goes down?**
  - Sparing mechanisms
  - Remapping of cache around failed resource(s) – i.e., capacity reduction
- **DiRT for limiting dirty data to reduce duplicate-on-write impact**
  - Dirty region tracker mechanism proposed by Sim et al. that bounds the amount of dirty cachelines in the DRAM$ [Sim+ MICRO'12]
- **Scrubbing / "Rinsing"**
  - Periodic error correction to prevent accumulations of multiple-bit errors
  - Periodic write-back of modified data to reduce DOW impact
- **System interactions**
  - Allow OS or other software to specify pages (or other granularity) to protect or not protect (or at what level)
  - If error corrections happen too often, alert OS and possibly reconfigure the RAS mechanisms for higher levels of protection
- **RAS for non-cache 3D DRAM?**
  - This work focused on die-stacked DRAM as a cache; for some markets, the entirety of main memory may be stacked … how to provide RAS support?

Sim et al., *Resilient Die-Stacked DRAM Caches*, ISCA 2013

DISCLAIMER & ATTRIBUTION

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions and typographical errors.

The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION.

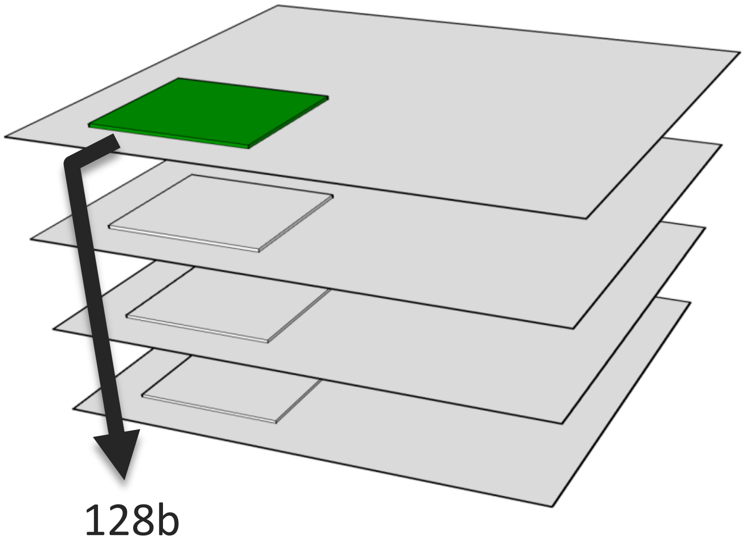
AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY DIRECT, INDIRECT, SPECIAL OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

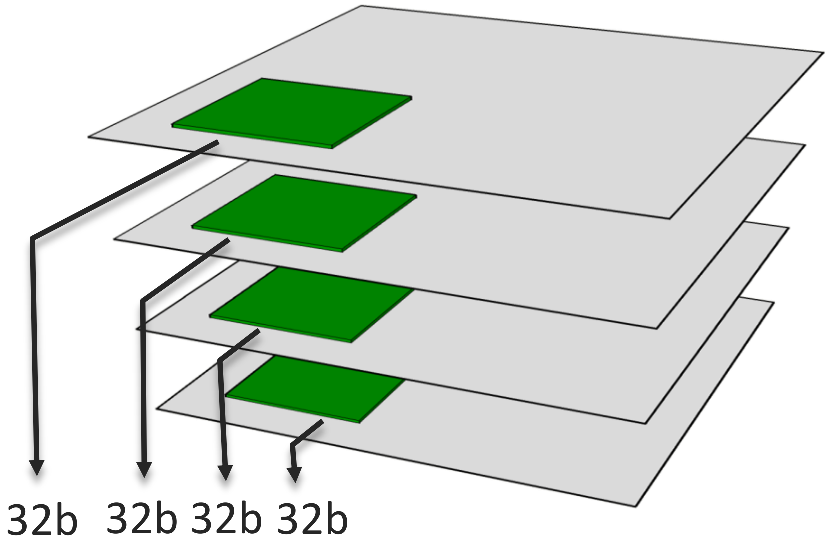
ATTRIBUTION

BACKUP

DIE-STACKED DRAM ORGANIZATION

- "DIMM-style" ECC for stacked DRAM is expensive





- 3D DRAM uses wide buses
  - E.g., JEDEC HBM has eight independent 128-bit channels
- Single point of failure
  - Chipkill works because data are spread over multiple physical units of failure

- Accessing multiple chips
  - Wastes channel/command bandwidth
  - Reduces opportunities for bank-level parallelism
  - Wastes power (multiple activations)

Sim et al., *Resilient Die-Stacked DRAM Caches*, ISCA 2013

REFERENCES

- V. Sridharan and D. Liberty, *A Study of DRAM Failures in the Field*, Proceedings of the International Conference for High-Performance Computing, Networking, Storage, and Analysis (SC12), November 2012.
- V. Sridharan, J. Stearley, N. DeBardeleben, S. Blanchard, and S. Gurumurthi, *Feng Shui of Supercomputer Memory: Positional Effects in DRAM and SRAM*, Proceedings of the International Conference for High-Performance Computing, Networking, Storage, and Analysis (SC13), November 2013.
- C. Slayman, *Soft Error Trends and Mitigation Techniques in Memory Devices*, Proceedings of the Reliability and Maintainability Symposium (RAMS), January 2011.
- D. H. Yoon and M. Erez, *Virtualized ECC: Flexible Reliability in Main Memory*, Proceedings of the Fifteenth Symposium on Architectural Support for Programming Languages and Operating Systems (ASPLOS), March 2011.
- A. N. Udipi, N. Muralimanohar, R. Balsubramonian, A. Davis, and N. P. Jouppi, *LOT-ECC: Localized and Tiered Reliability Mechanisms for Commodity Memory Systems*, Proceedings of the 39th Annual International Symposium on Computer Architecture (ISCA), June 2012.
- X. Jian, H. Duwe, J. Sartori, V. Sridharan, and R. Kuman, *Low-power, Low-storage-overhead Chipkill Correct via Multi-line Error Correction*, Proceedings of the International Conference for High-Performance Computing, Networking, Storage, and Analysis (SC13), November 2013.
- J. Sim, G. H. Loh, V. Sridharan, M. O’Connor, *Resilient Die-Stacked DRAM Caches*, Proceedings of the 40th Annual International Symposium on Computer Architecture (ISCA), June 2013.
- A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, *Basic Concepts and Taxonomy of Dependable and Secure Computing*, IEEE Transactions on Dependable and Secure Computing, Volume 1, Issue 1, Jan.-Mar. 2004.