



# Toward Exascale Resilience

## **Part 4:**

## **Memories and networks**

Mattan Erez

The University of Texas at Austin

July 2015



## Memories

- Arrays of cells
- Peripheral circuits
- Communication

## Soft and hard faults



## Error checking and correcting (ECC)

- Redundant coding for detection and correction
- Most-efficient way to detect errors (and correct them)

## Repair

- Once fault is known, can choose to repair it
- Necessary?



# Introduction to Memory ECC

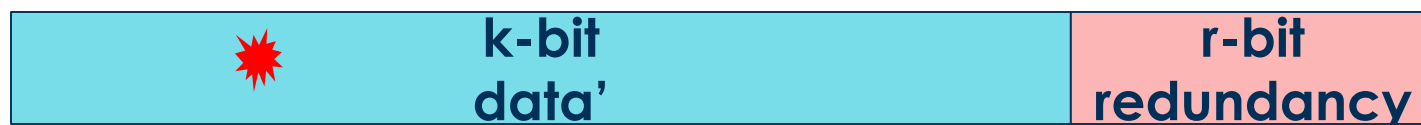
## A codeword (**CW**)

- A valid pair of {data, redundancy}



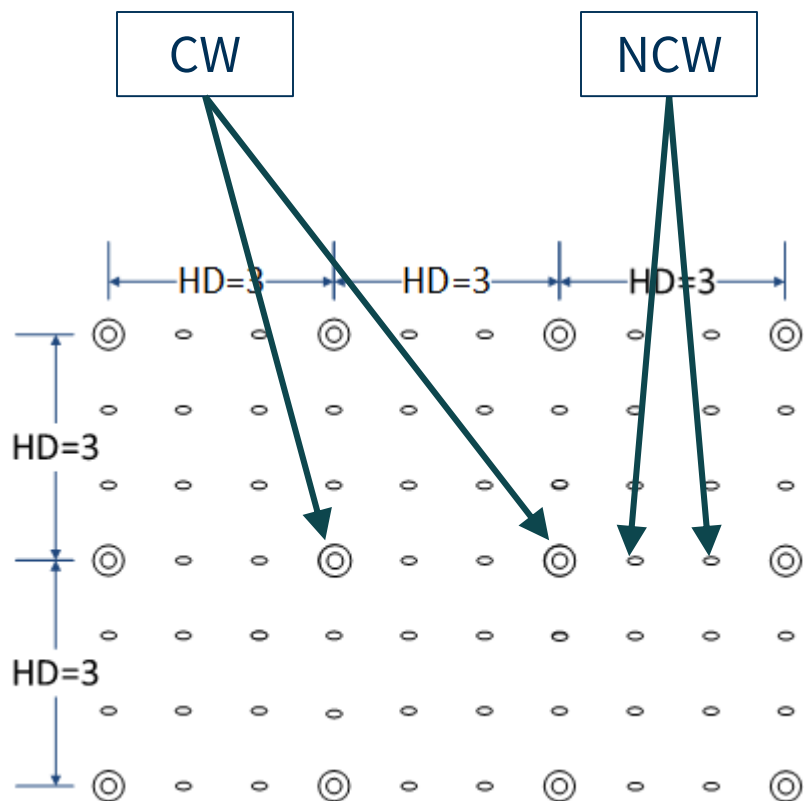
## A non-codeword (**NCW**)

- An invalid pair due to errors on CW





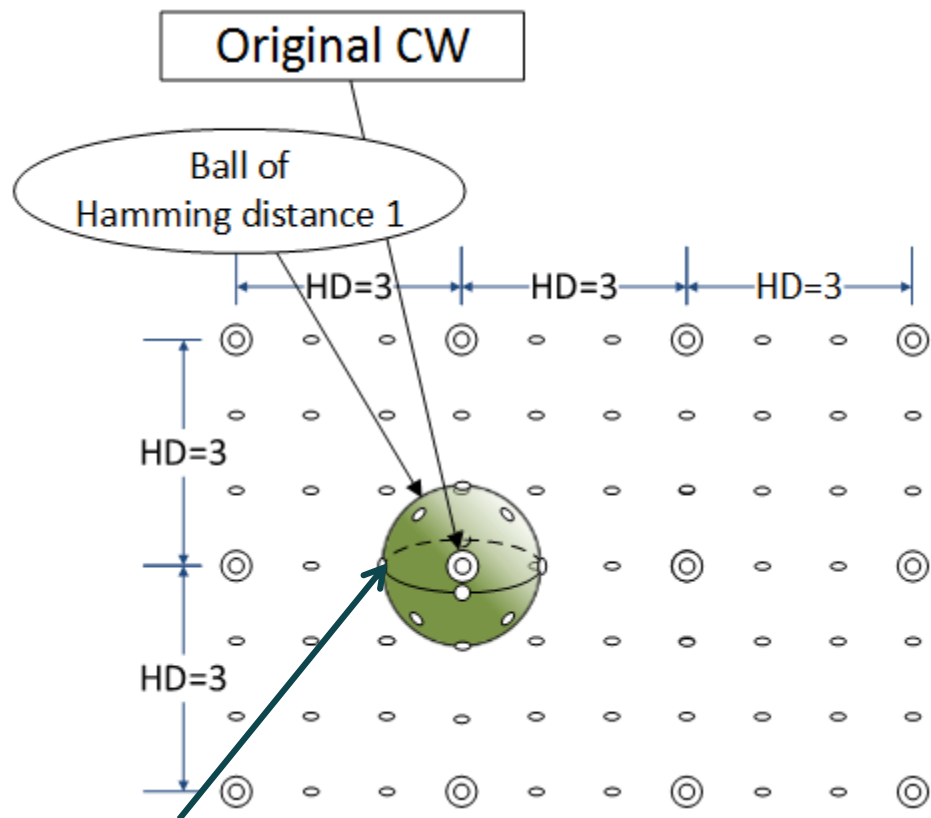
# Example: Single Symbol Correcting Code



Hamming Distance(HD):  
the number of different symbols between two words

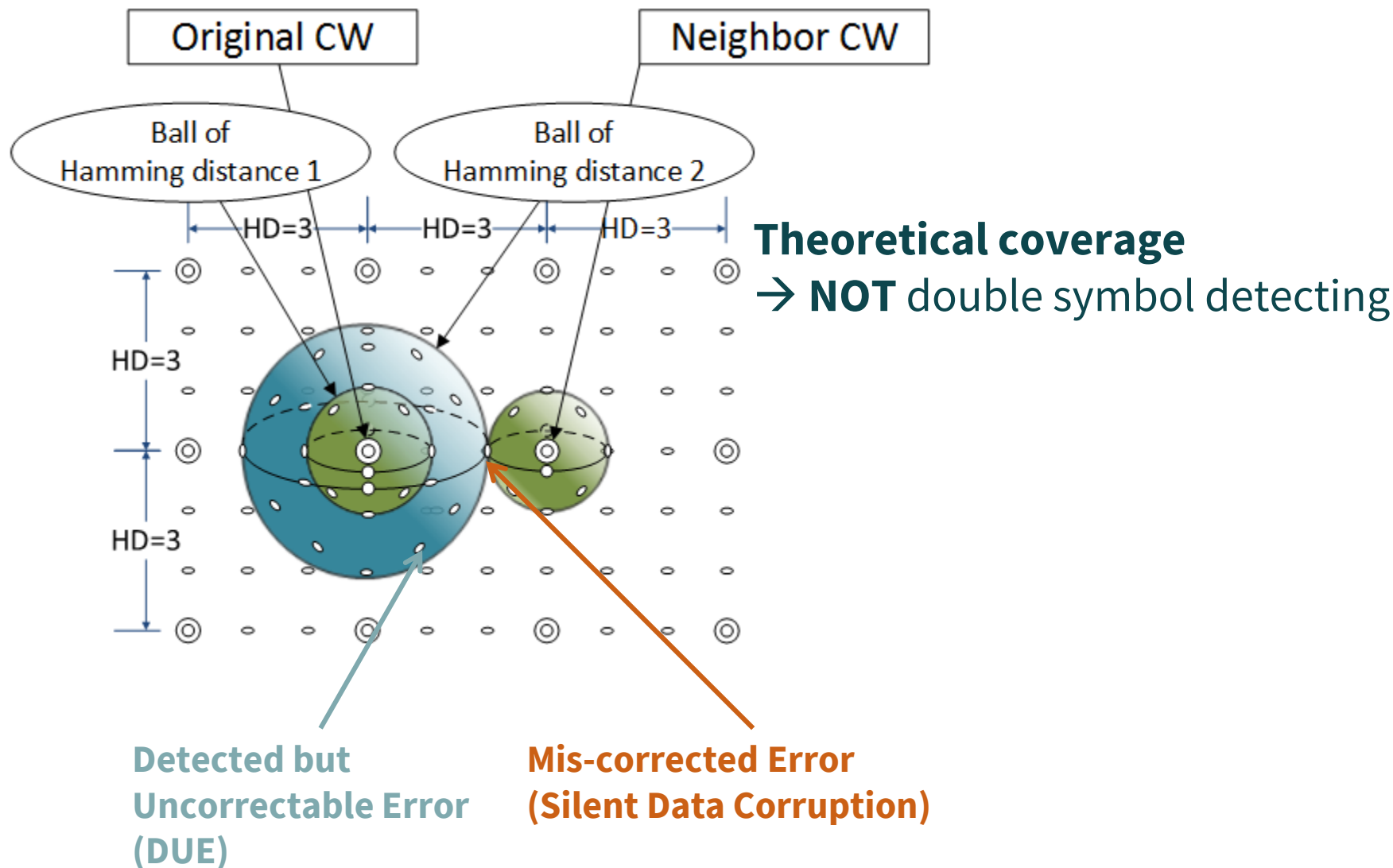


# Example: Single Symbol Correcting Code





# Example: Single Symbol Correcting Code

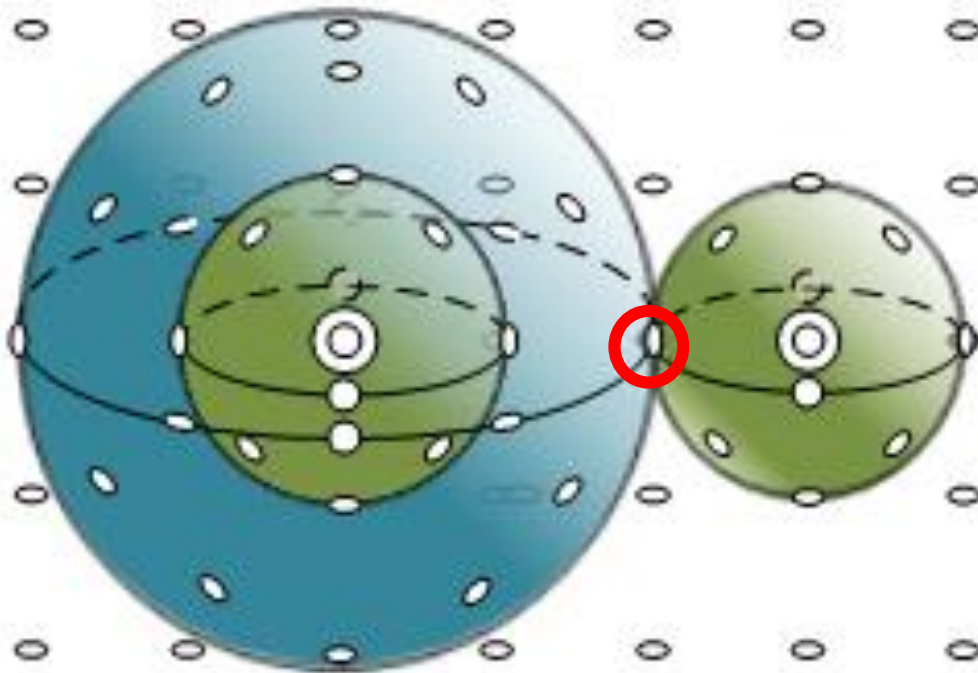




# #1 – Codeword Size

We can have **99.9999%** detection

**Longer codewords** exponentially reduce SDC





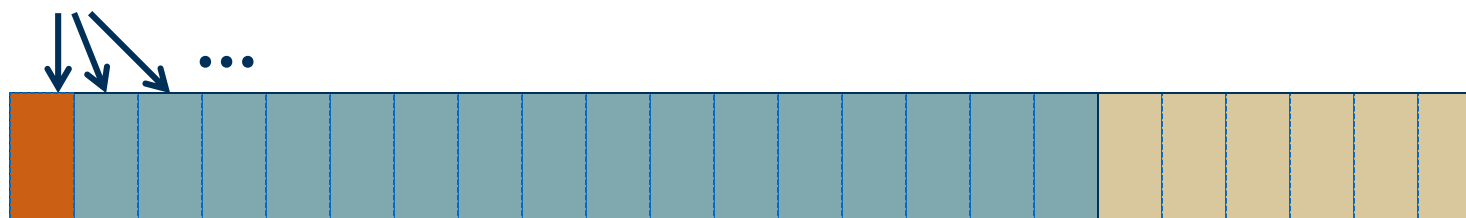


# How Much Redundancy Do We Need?

## Depends on

- Protection Strength
- Data size
- **Symbol size**


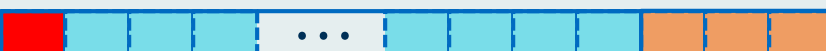
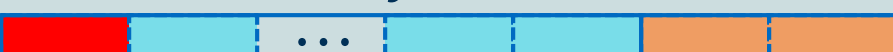
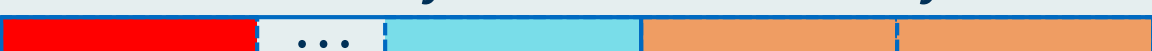
## Symbols



**A codeword**



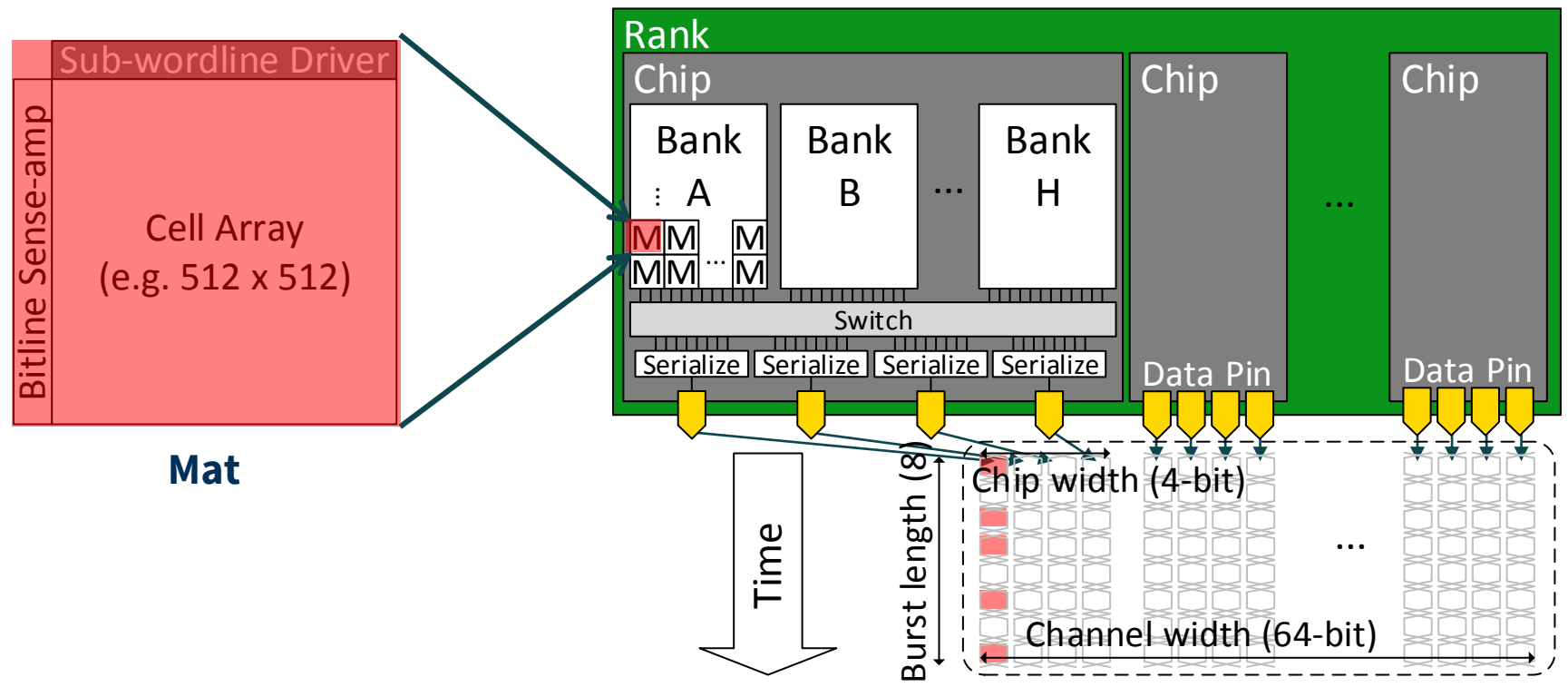
## #2 – Symbol Size

Sym size	Codeword (512-bit data)	Redun. size (bit)	Redun. ratio
<b>1-bit</b>	Data: <b>1b</b> x 512 sym      Redun: 10 sym 	10-bit	<b>10</b>
<b>4-bit</b>	Data: <b>4b</b> x 128 sym      Redun: 3 sym 	12-bit	<b>3</b>
<b>8-bit</b>	Data: <b>8b</b> x 64 sym      Redun: 2 sym 	16-bit	<b>2</b>
<b>16-bit</b>	Data: <b>16b</b> x 32 sym      Redun: 2 sym 	32-bit	<b>2</b>



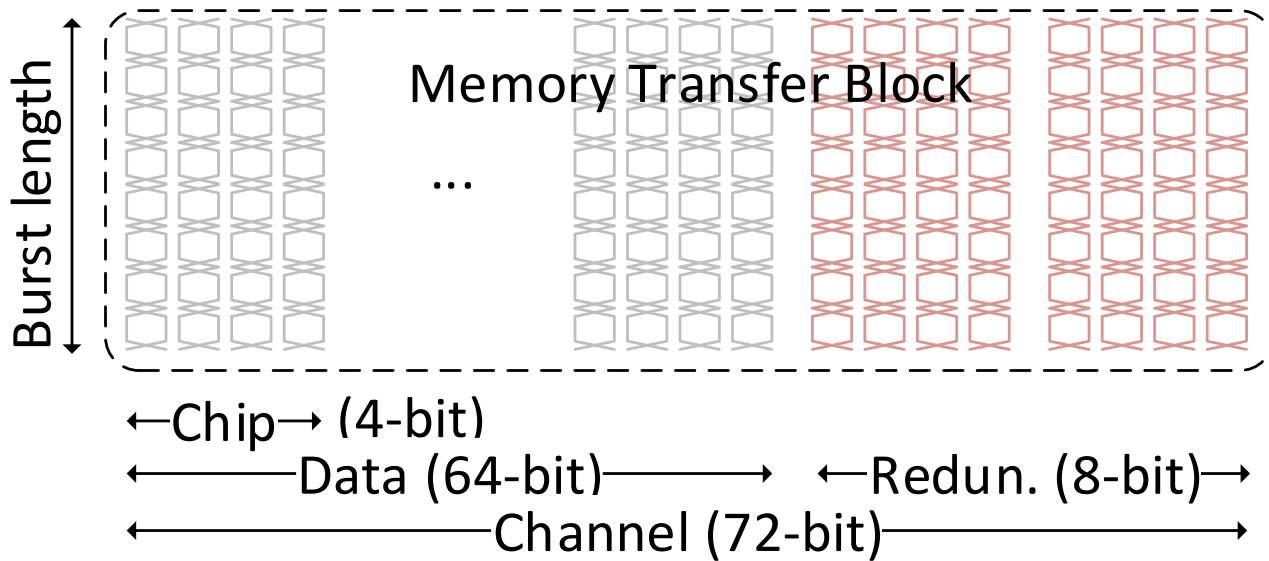
# #3 - DRAM Internal Structure

Most errors affect a **single data pin** only  
– Per-mat data go to the same data pin





# Memory Transfer Block

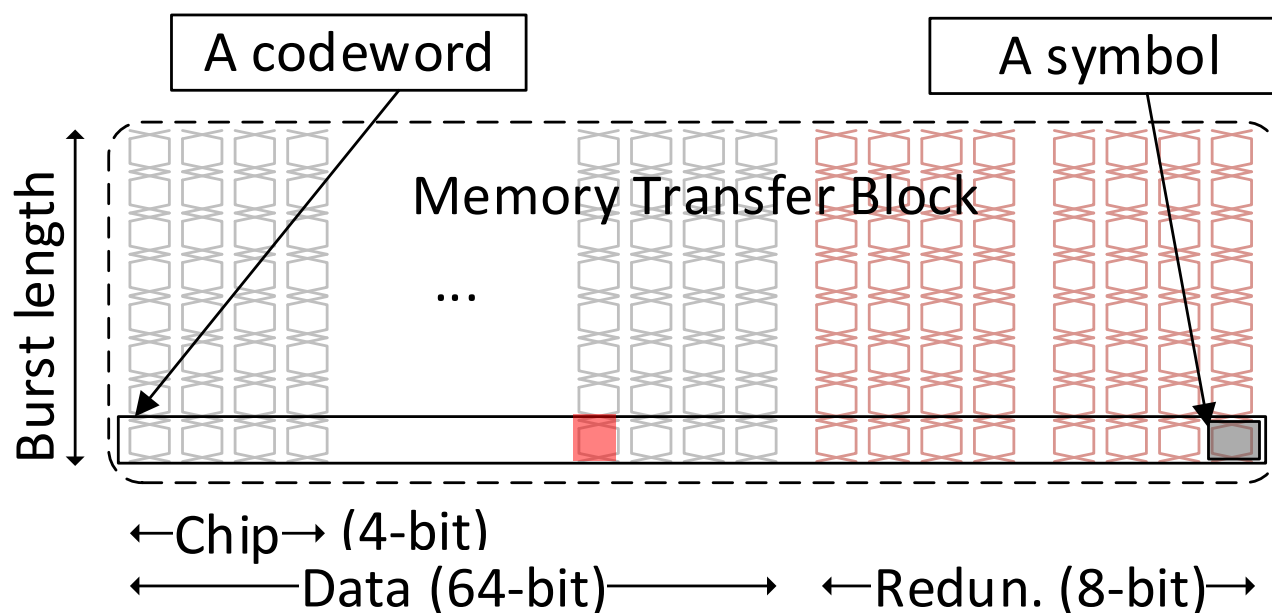




# Historical ECC

## **SEC-DED** (Single Error Correcting – Double Error Detecting)

- **Per-beat** codeword with **binary** symbols
- 12.5% redundancy on 64-bit data



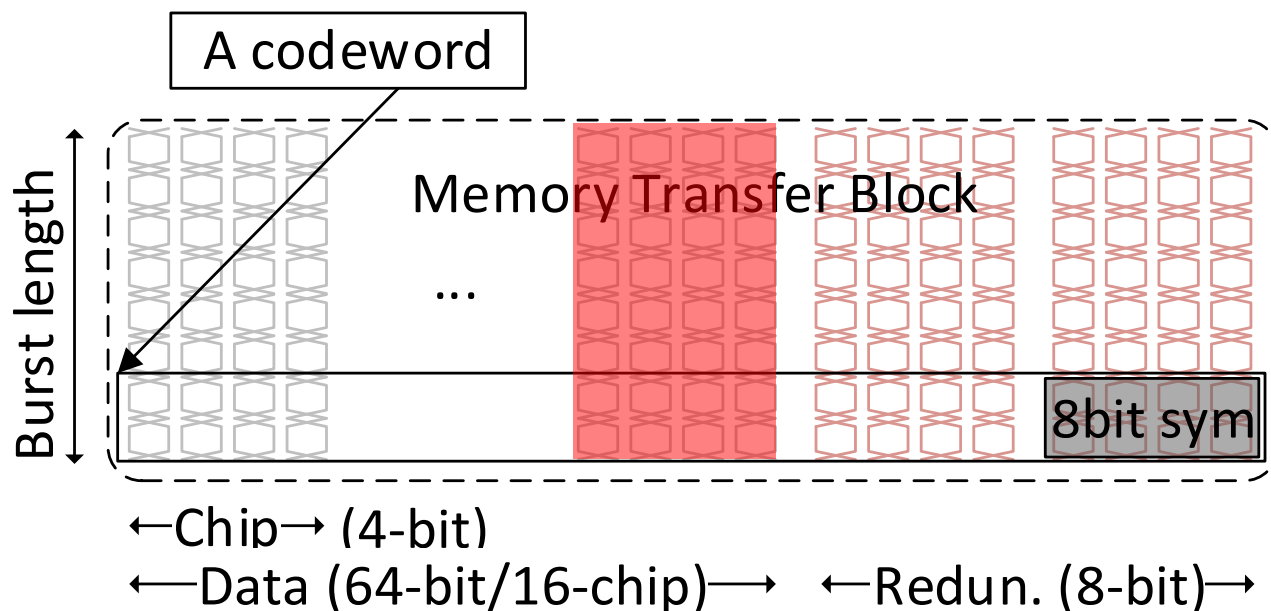


# Current ECC

## Chipkill-correct

## AMD Chipkill-correct

- **Double-bus-beat** codeword with **8-bit** symbol
- 12.5% redundancy on 64-bit data



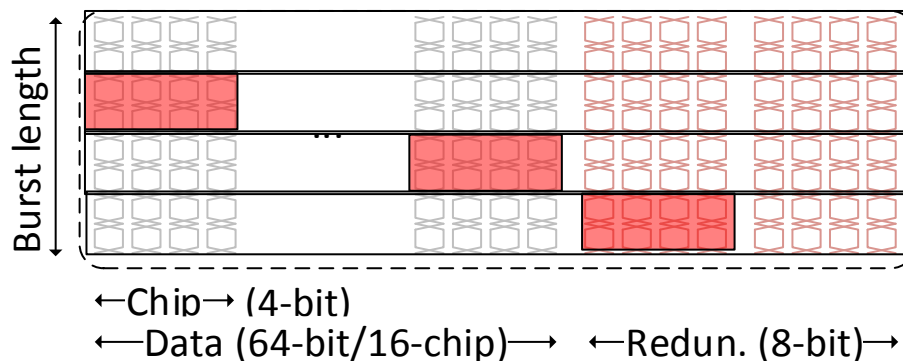


# Post-processing of Corrections

## Discard corrections if suspicious

### – AMD Chipkill

- A.k.a History-based miscorrection detection

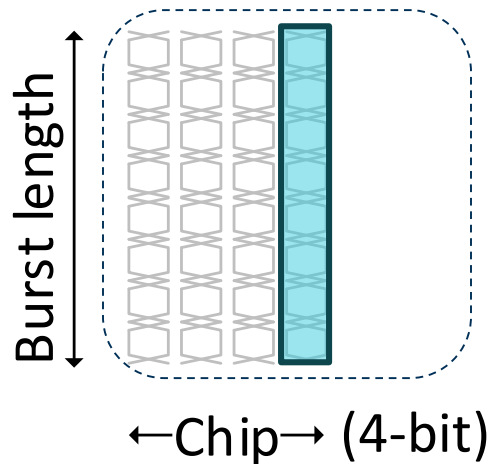




# Bamboo ECC

Based on a very simple idea

## Change ECC layout



- **Vertical symbol**
  - 8-bit symbol
  - Aligned to frequent per-pin errors
- **On a large codeword**
  - Memory transfer block

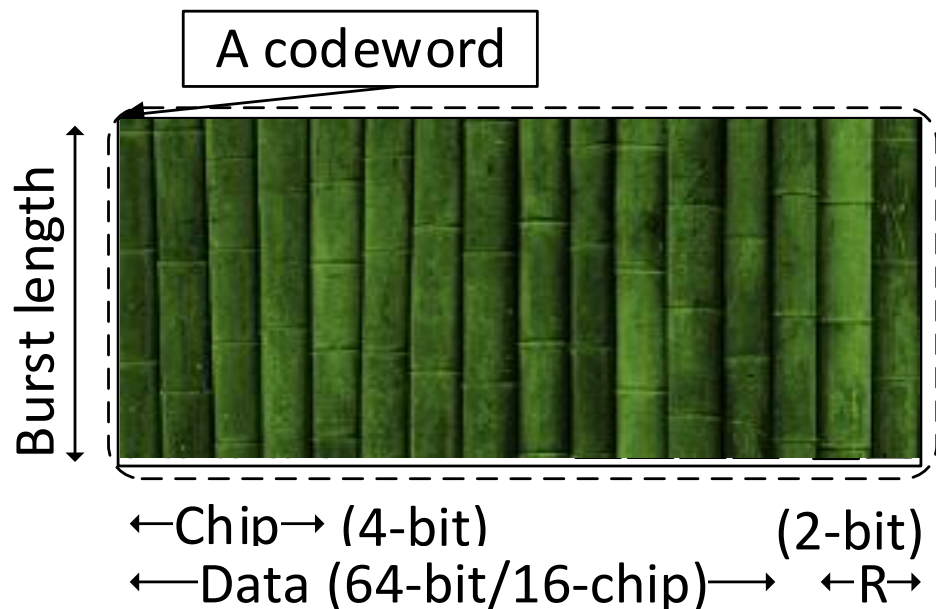




# Low-end Bamboo ECC

## Single Pin Correcting (**SPC**/1PC)

- Corrects 1 pin error using 2 pin redundancy
  - **3.1%** redundancy on 64-bit data





# Low-end Bamboo ECC

	Which one is better?	Note
Correction probability (A)	<b>SEC-DED</b>	

## SPC vs. SEC-DED

- **1/4 redundancy**
- **Lower uncorrected error rate**

Overall: uncorrected errors ( $B \times (1-A)$ )	<b>SPC</b>	~95%
---	------------	------



## Bamboo ECC: A Family of Codes

	SPC	...	SPC-TPD	...	QPC	...	OPC
Redundancy (pin)	2	...	4	...	8	...	16
Correction (pin)	1	...	1 (or 2)	...	4	...	8

Fine-grained control over redundancy

Costs are proportional to protection

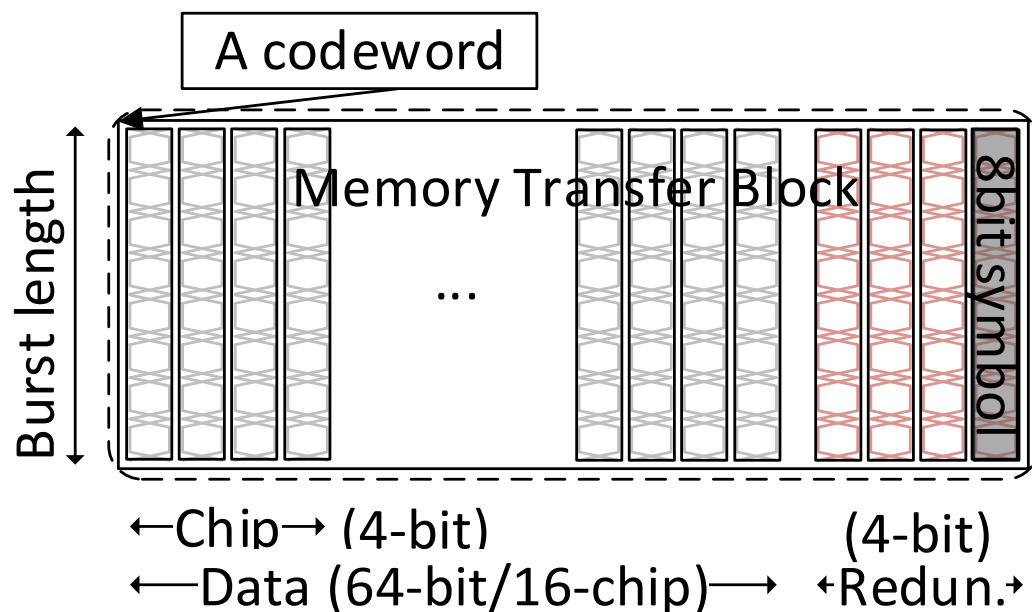


# Low-end Bamboo ECC (Cont')

## Single Pin Correcting – Triple Pin Detecting (**SPC-TPD**)

### Vs. SEC-DED

- ½ redundancy (4 pins)
- Fewer uncorrected errors
- Safer detection (up to 51% vs. 0.0004% SDC)

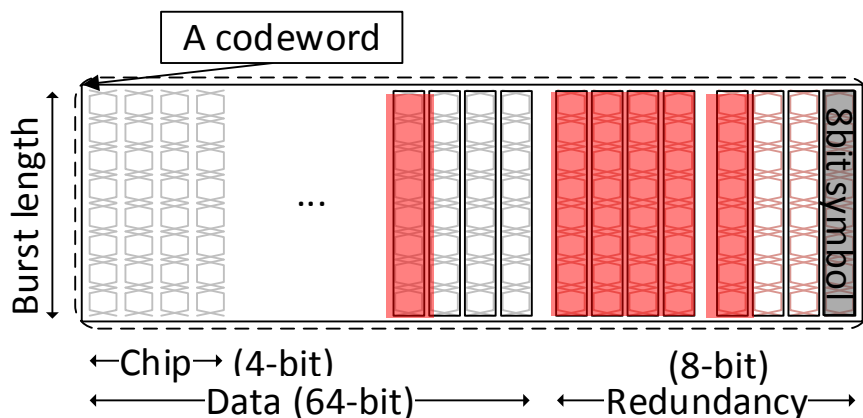




# Middle-/High-end Bamboo ECC

## Quadruple Pin Correcting (**QPC**/4PC)

- Corrects one x4 chip error, using 8 pin redundancy



### Compared to AMD Chipkill

- Same redundancy (8 pins / 12.5%)
- **Stronger correction** (e.g. two chips)
- **Safer detection**

## Octuple Pin Correcting (**OPC**/8PC)

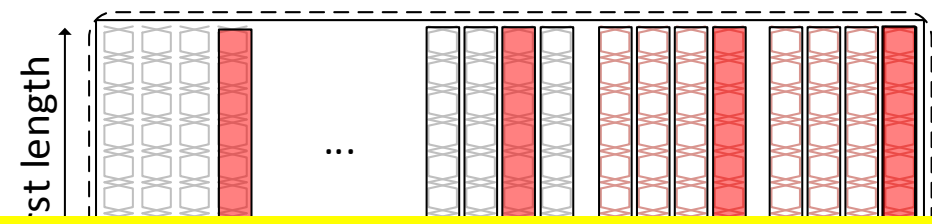
- Corrects one x8 chip error, using 16 pin redundancy



# #4: SDC Reduction

Discard corrections if suspicious

E.g. 4PC



## SDC probability (for a severe error)

- QPC:  $2^{-12}$
- OPC:  $2^{-22}$



4 pin corrections on a chip (a chip error)



2 pin corrections on 2 different chips



Cf.) AMD Chipkill's history-based miscorrection detection

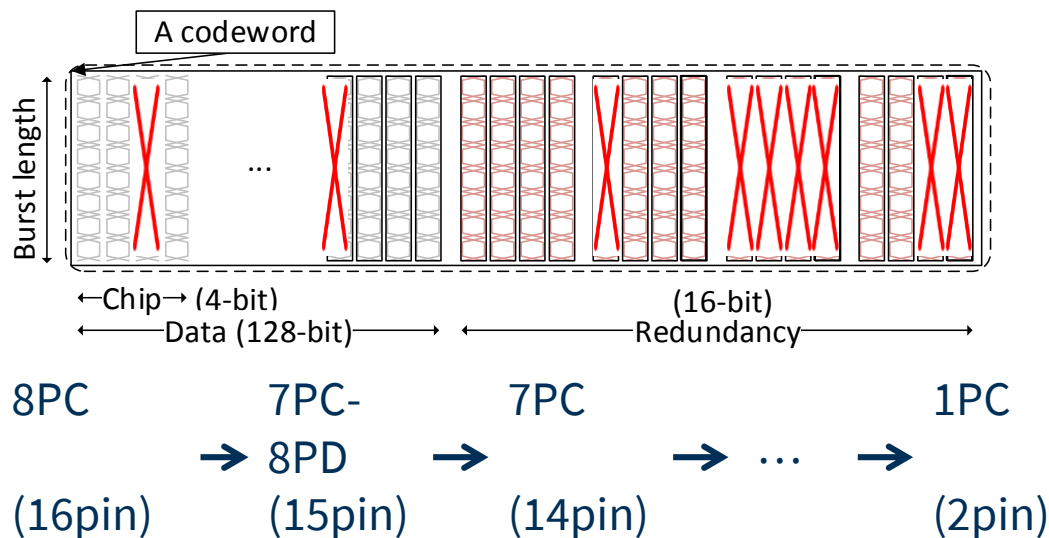
# #5: Graceful Downgrade

## Issue: **non-transient faults**

- 71% of faults / 99.9% of errors
- More severe errors / repeated corrections

## Retirement

Tolerate up to **14** pin faults (or 3 chips + 2 pins) without rebuilding data



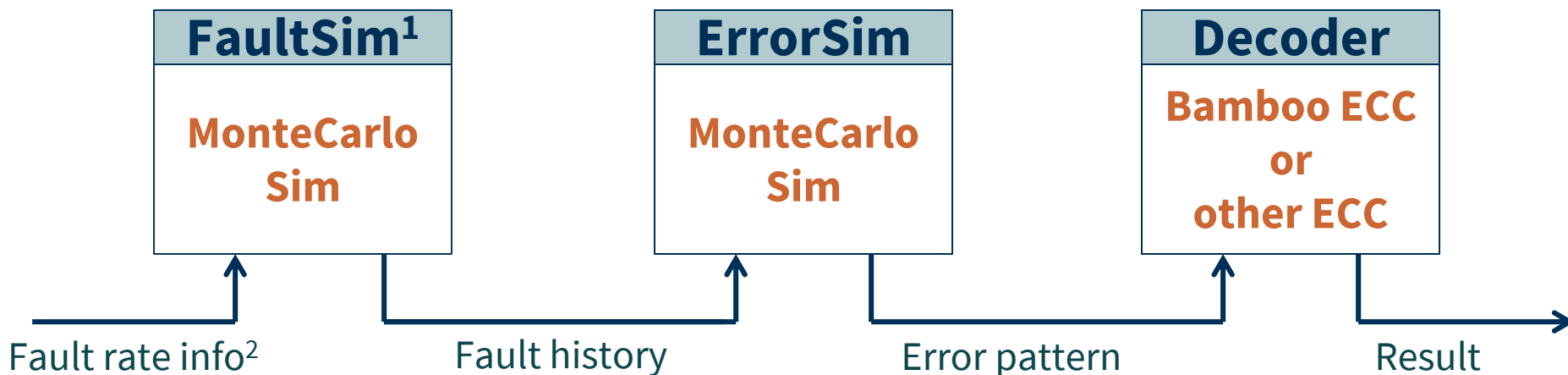


Backed up by backward recovery





# Evaluation: Reliability



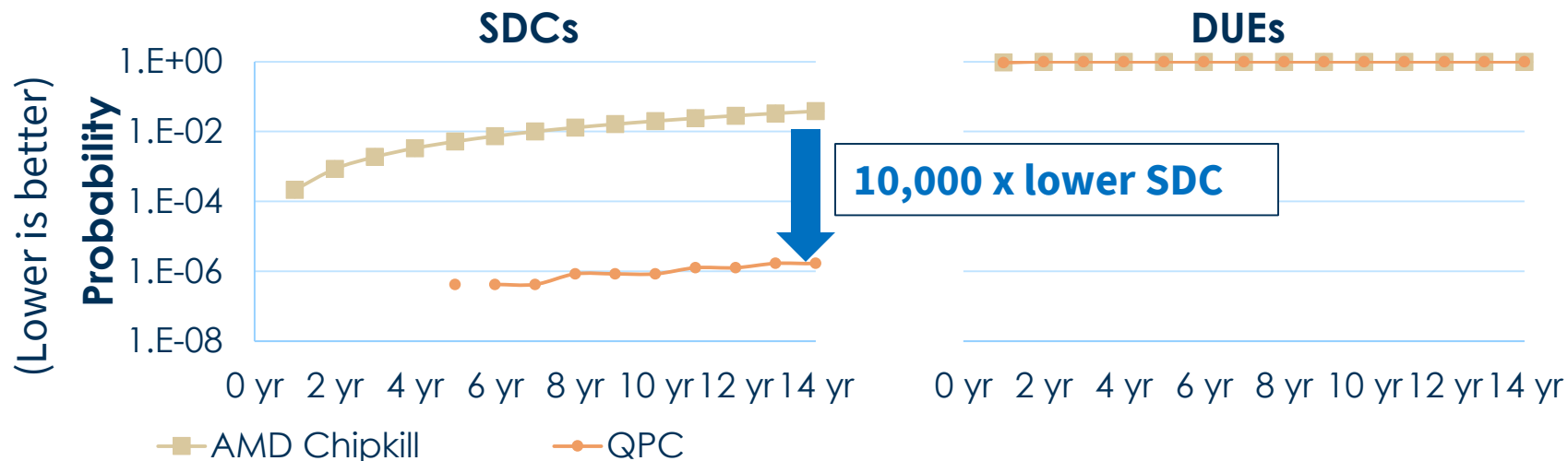
Fault mode	Fault rate (1x)	Fault rate (10x)
Single-bit	18.6	186
Single-row	8.2	82
Single-column	5.6	56
Single-bank	10	100
Multiple-bank	1.4	14

[1] "FAULTSIM: A fast, configurable memory-resilience simulator", D. Roberts, et al. The Memory Forum '14

[2] "A study of DRAM failures in the field", V. Sridharan and D. Liberty, SC'12



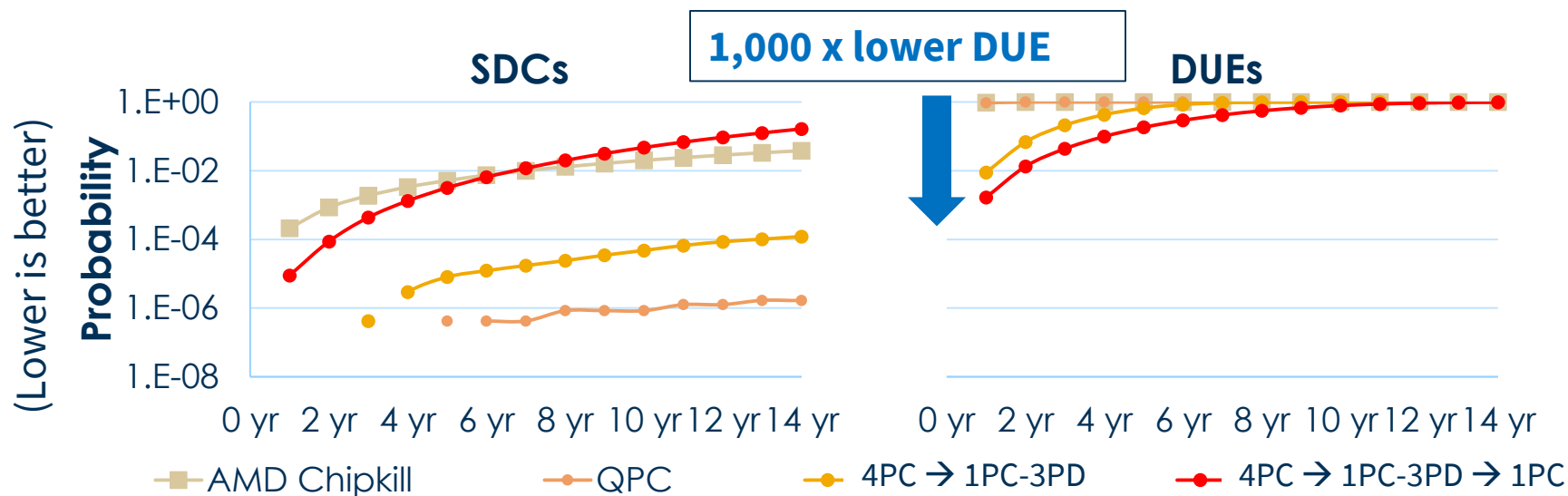
# System-level Reliability (72b channel)



A system with 100K ranks (18 x4 chips/rank)



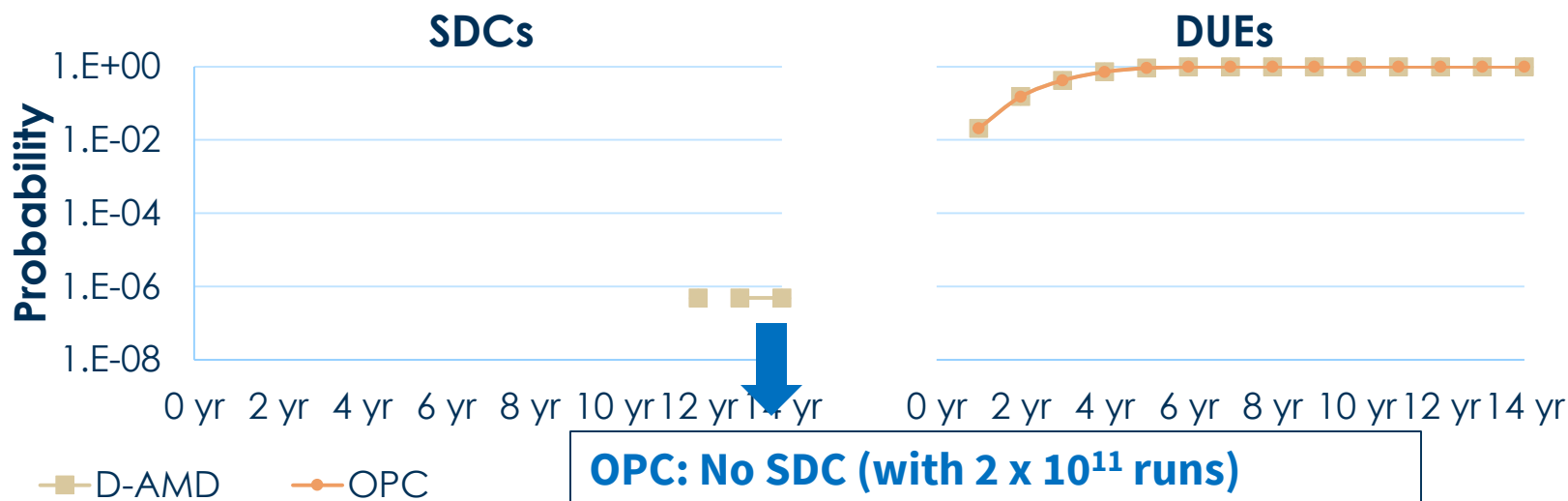
# System-level Reliability (72b channel)



A system with 100K ranks (18 x4 chips/rank)



# System-level Reliability (144b channel)



A system with 100K ranks (36 x4 chips/rank)





# Overheads

<b>Logic</b>			
	<b>AMD Chipkill</b>	<b>QPC</b>	<b>Note</b>
<b>Area</b> (NAND2 gates)	1,600	25,000	<b>16 x</b>
<b>Latency</b> (XOR2 gates)	8	10	<b>+ 2</b>

Logic overheads of encoder and decoder (error detection part), each

## Performance

- **2%** (H. mean) execution time increases<sup>1</sup>
- Due to +3 read/write latency to wait symbol transfer

## Energy

- **<1%** (H.mean) DRAM energy increases<sup>1</sup>

[1] SPECcpu2006 on Gem5 simulator (2GHz 1-core / 2M LLC / 2GB (64+8)b DDR3-1600)



## What about repair?

- Is there any benefit given strength of ECC?



## What about repair?

- Repair reduces risk of failure and SDC
- Repair can improve performance/efficiency
  - Reduce correction overhead
- Fault rates may be rising
  - Same question about vendor success as processors





## How to repair memory?

- Replace module
- Replace bits
- Disable bits / modules



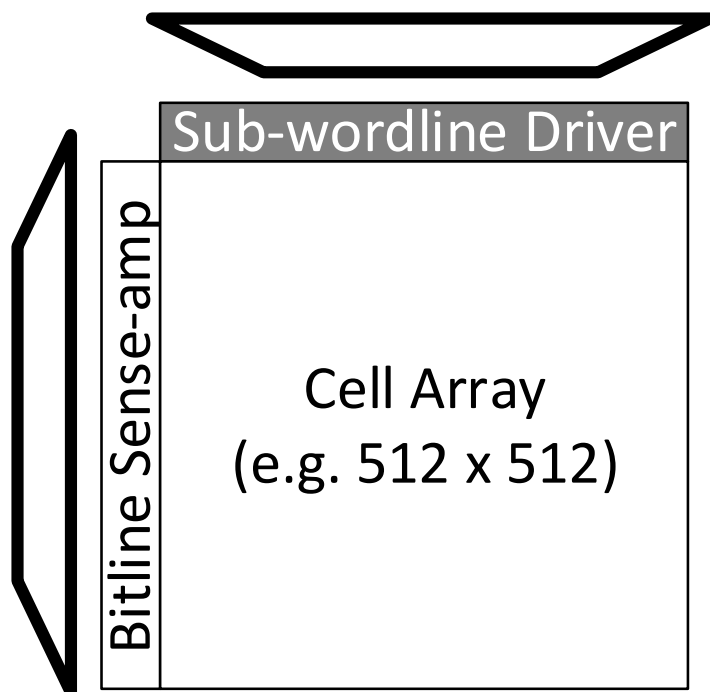
## How to replace bits?

- Remap (reconfigure)
- Redirect (cache)



## Row and column sparing

- Used in all memories for yield
- Used in some DRAMs for repair (LPDDR4)
  - Simple changes to existing decoders / muxes
- Remap at quite coarse granularity





## Bit / cache-line remapping

- Remap bits to pool of spares
- Error correcting pointers
- FREEp (cache lines)
- ArchShield



## Redirect – fault caching

- Small cache that intercedes for remapped bits (blocks)
- Can be integrated with DRAM or with processor



## Remap/redirect schemes depend on fault model

- How many bits do faults affect?
- Are those bits localized or scattered?
- Examples on board (slides maybe after class)

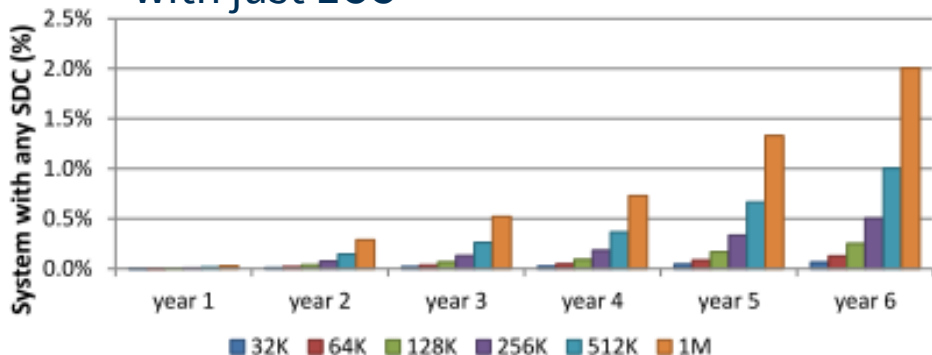


## Combining ECC with repair very effective

- Delay module replacement by years
- Reduce potential SDC rate by orders of magnitude

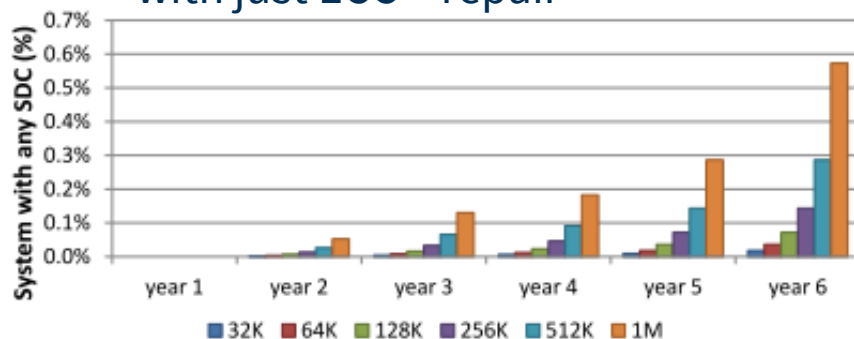


### 1 - 2% of systems will have SDCs with just ECC



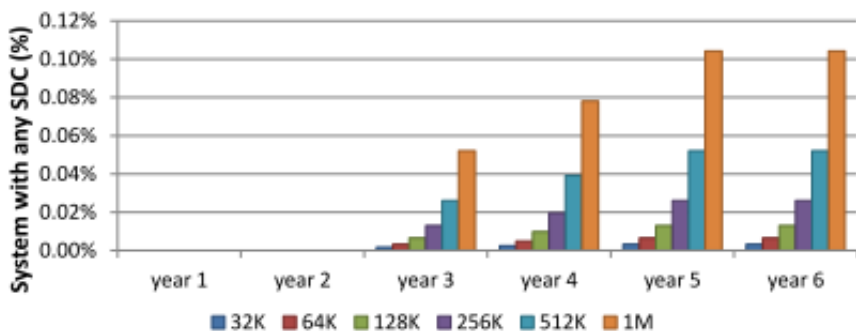
(a) ECC only (no fine-grained DRAM repair)

### .3 - .6% of systems will have SDCs with just ECC + repair



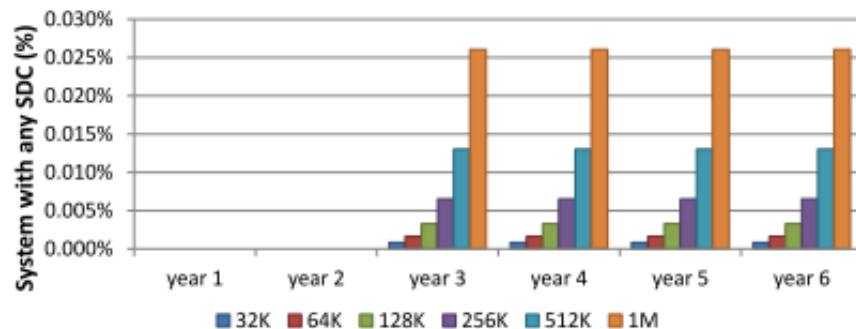
(b) ECC + 256KB DRAM repair capability per node

### .1% of systems will have SDCs with ECC + ECC downgrade



(a) ECC + spare chip per DIMM pair

### .02% of systems will have SDCs with just ECC + ECC downgrade + repair



(b) ECC + spare chip per DIMM pair + 256KB repair per node





Networks added later.