Attaché: Towards Ideal Memory Compression by Mitigating Metadata Bandwidth Overheads

Seokin Hong*, Prashant J. Nair*
Bulent Abali, Alper Buyuktosunoglu,
Kyu-Hyoun Kim, and Michael B. Healy

IBM Thomas J. Watson Research Center

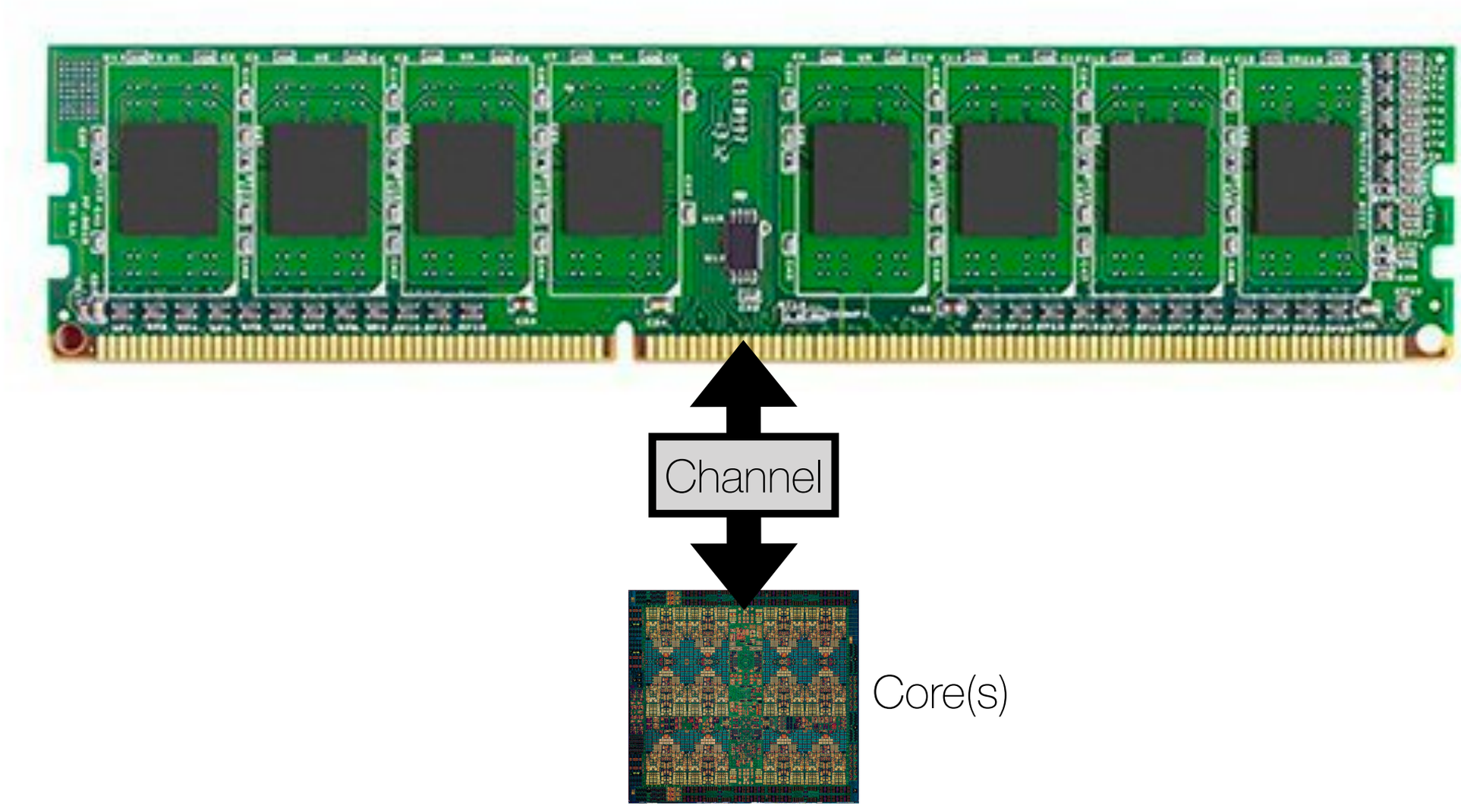
*both authors contributed equally

MICRO-51

Hotel Grand Hyatt
Oct 20th - 24th, Fukuoka, Japan

Introduction

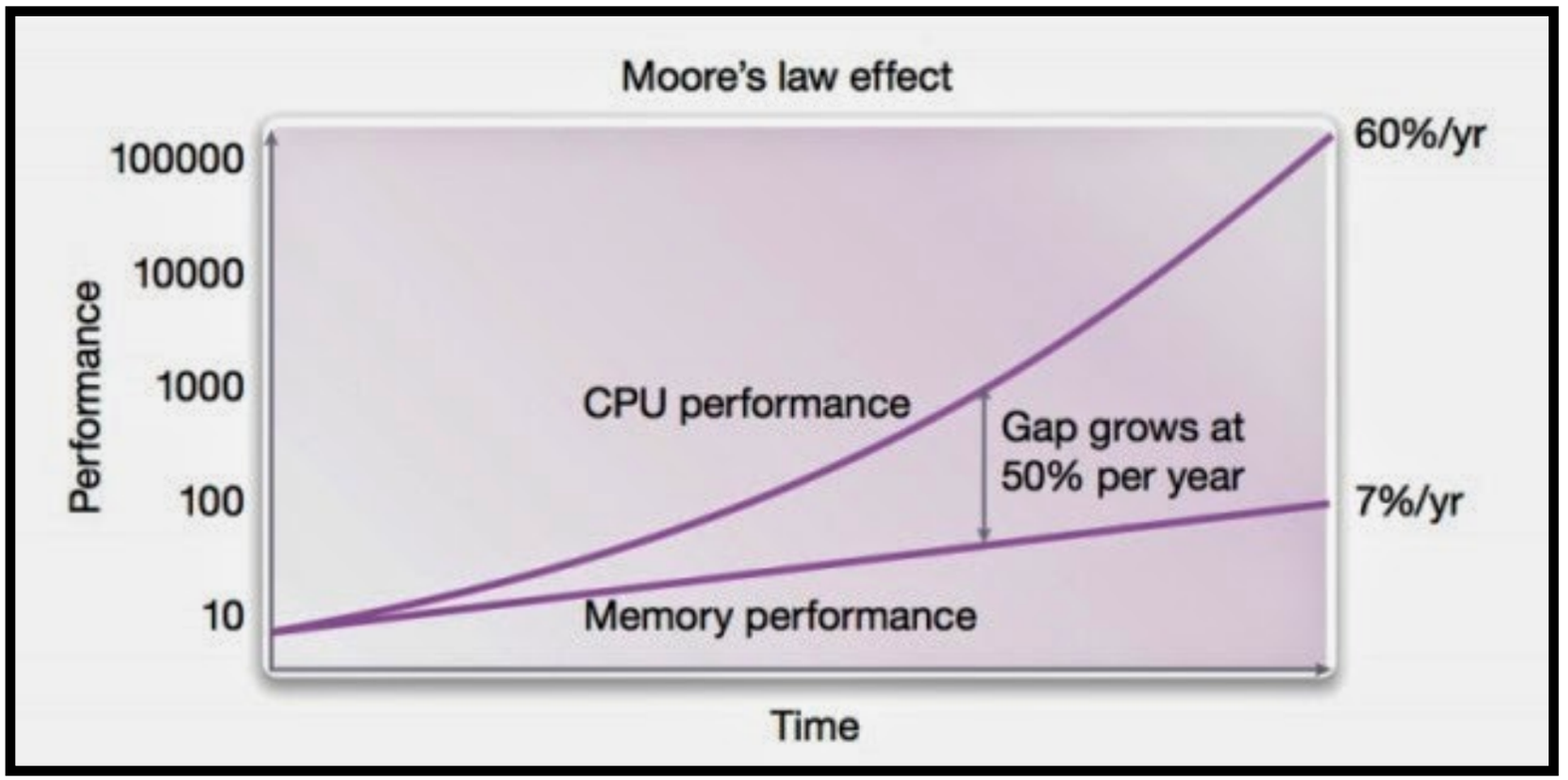
Memory systems provide data to the processing cores



Introduction

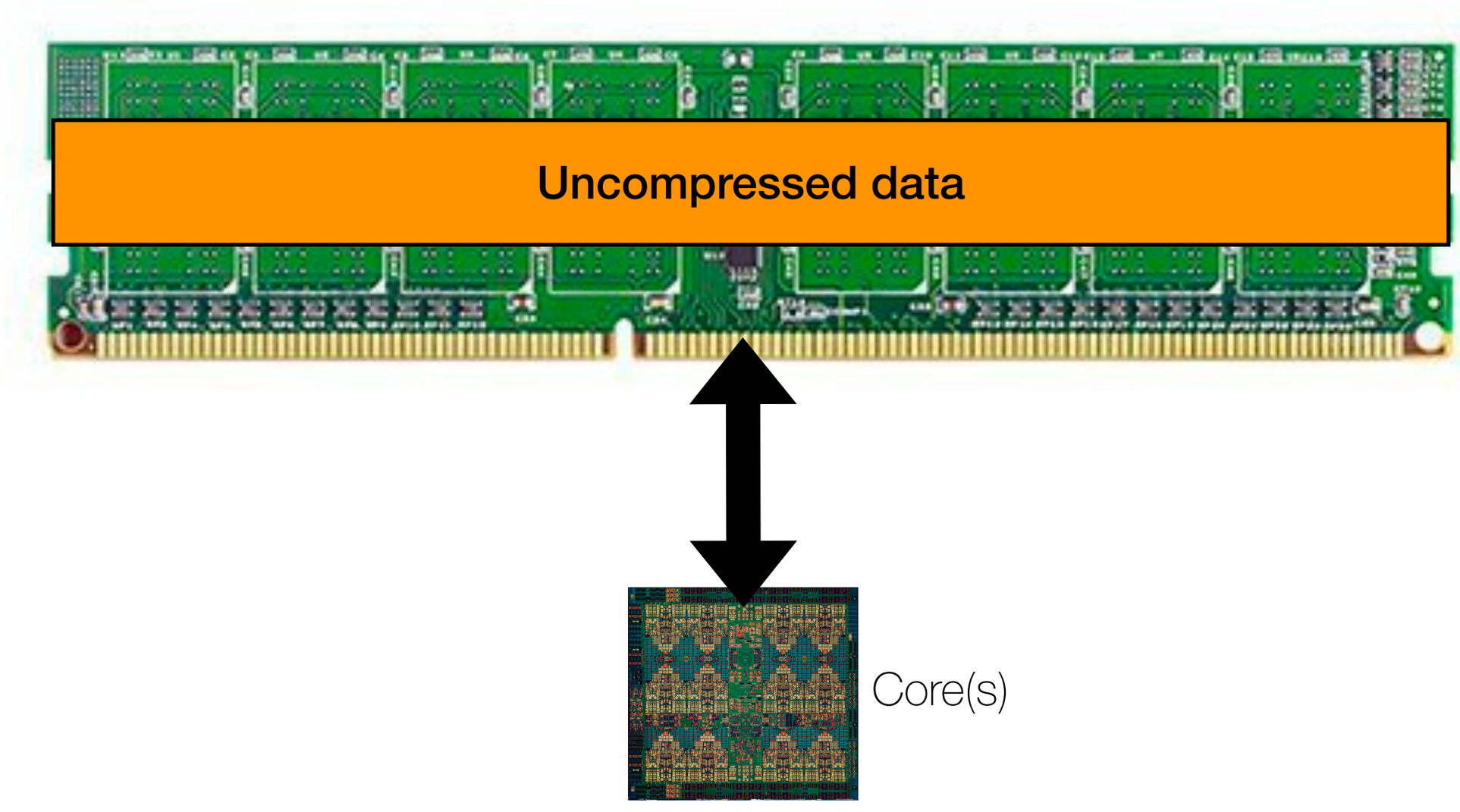
Memory systems provide data to the processing cores

However, memory bandwidth is not keeping up with core performance



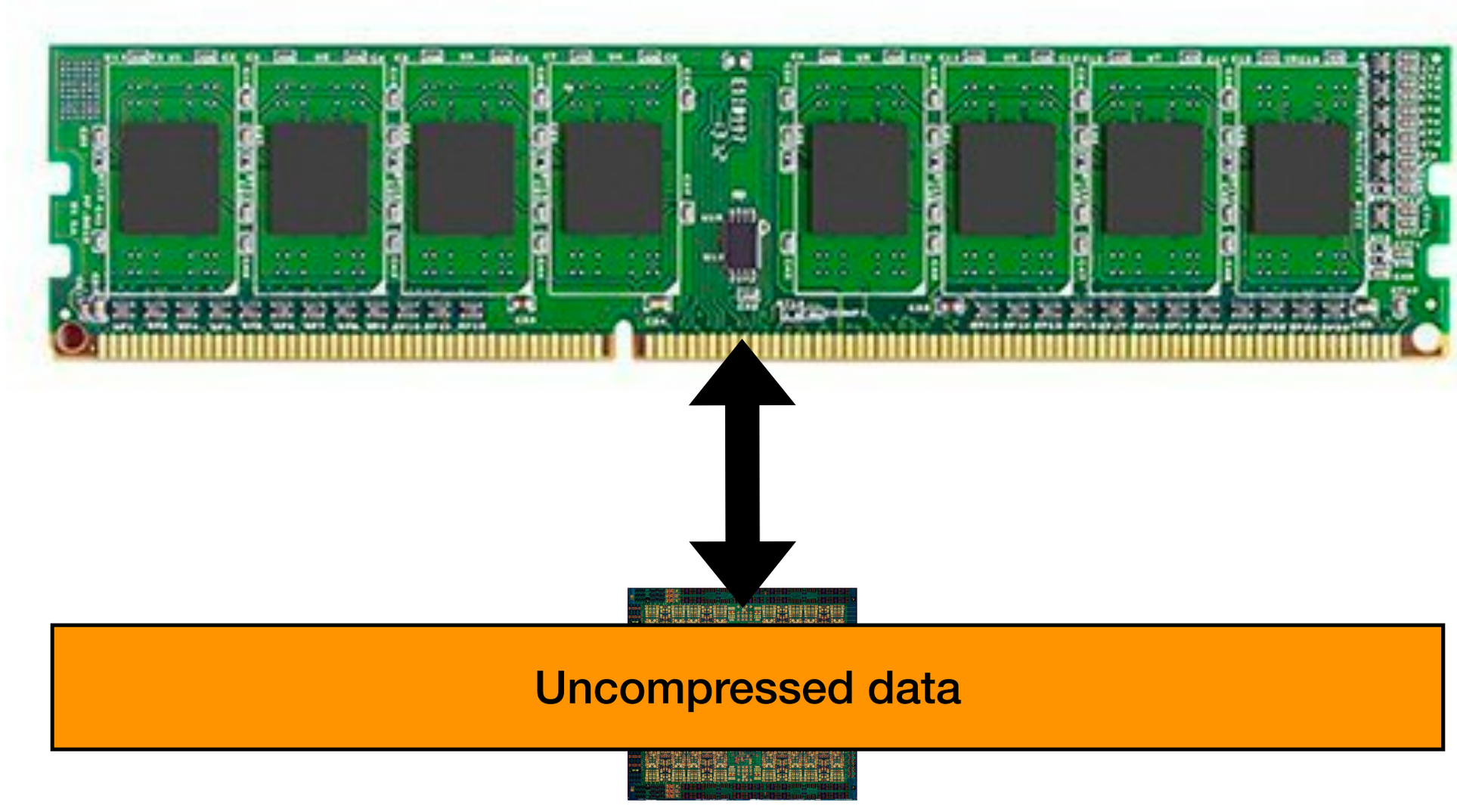
Introduction

Data compression: A simple technique to improve bandwidth



Introduction

Data compression: A simple technique to improve bandwidth

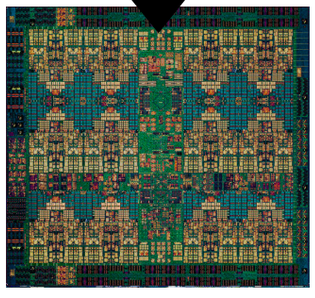


Introduction

Data compression: A simple technique to improve bandwidth

Main Memory

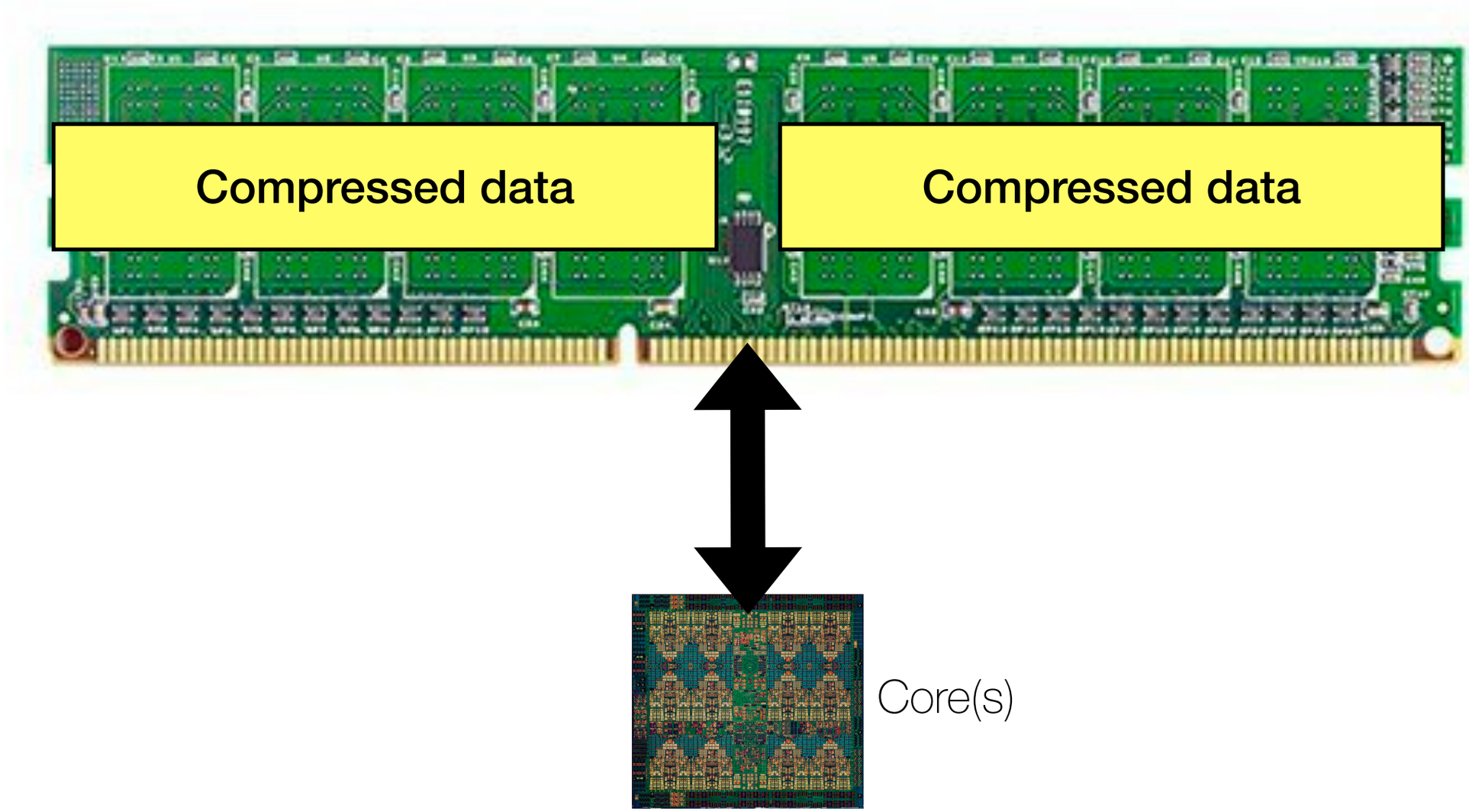
1x bandwidth



Core(s)

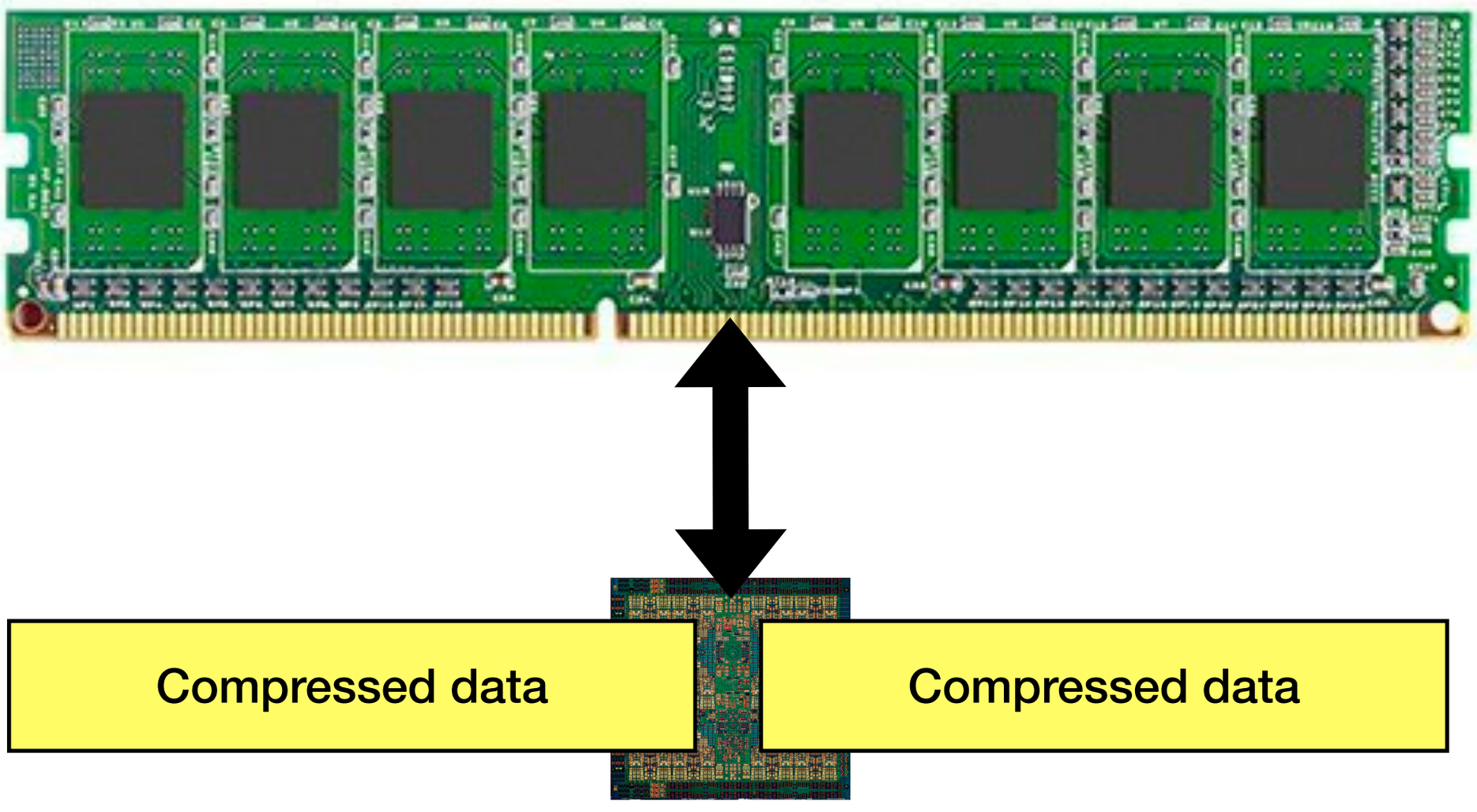
Introduction

Data compression: A simple technique to improve bandwidth



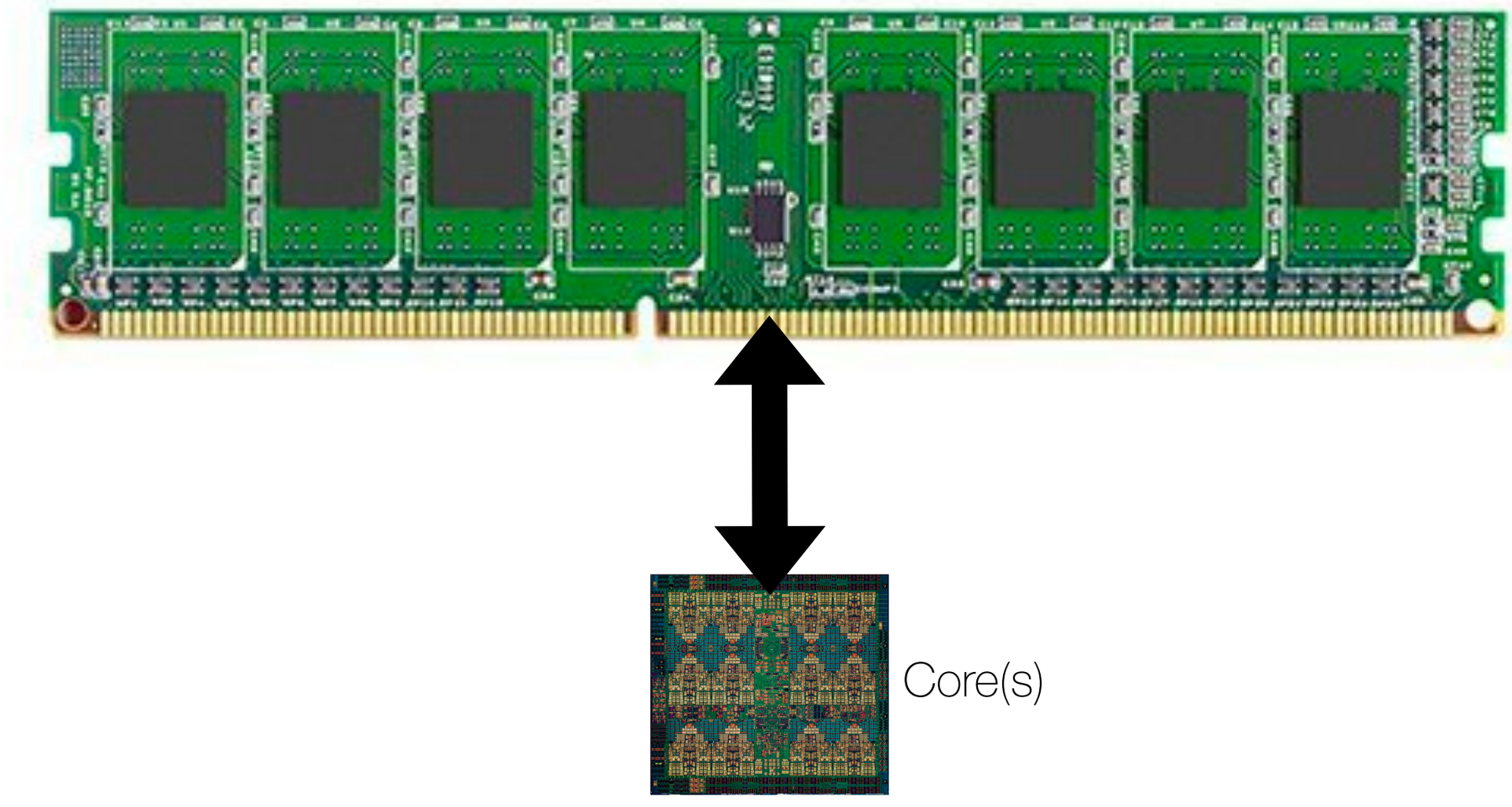
Introduction

Data compression: A simple technique to improve bandwidth



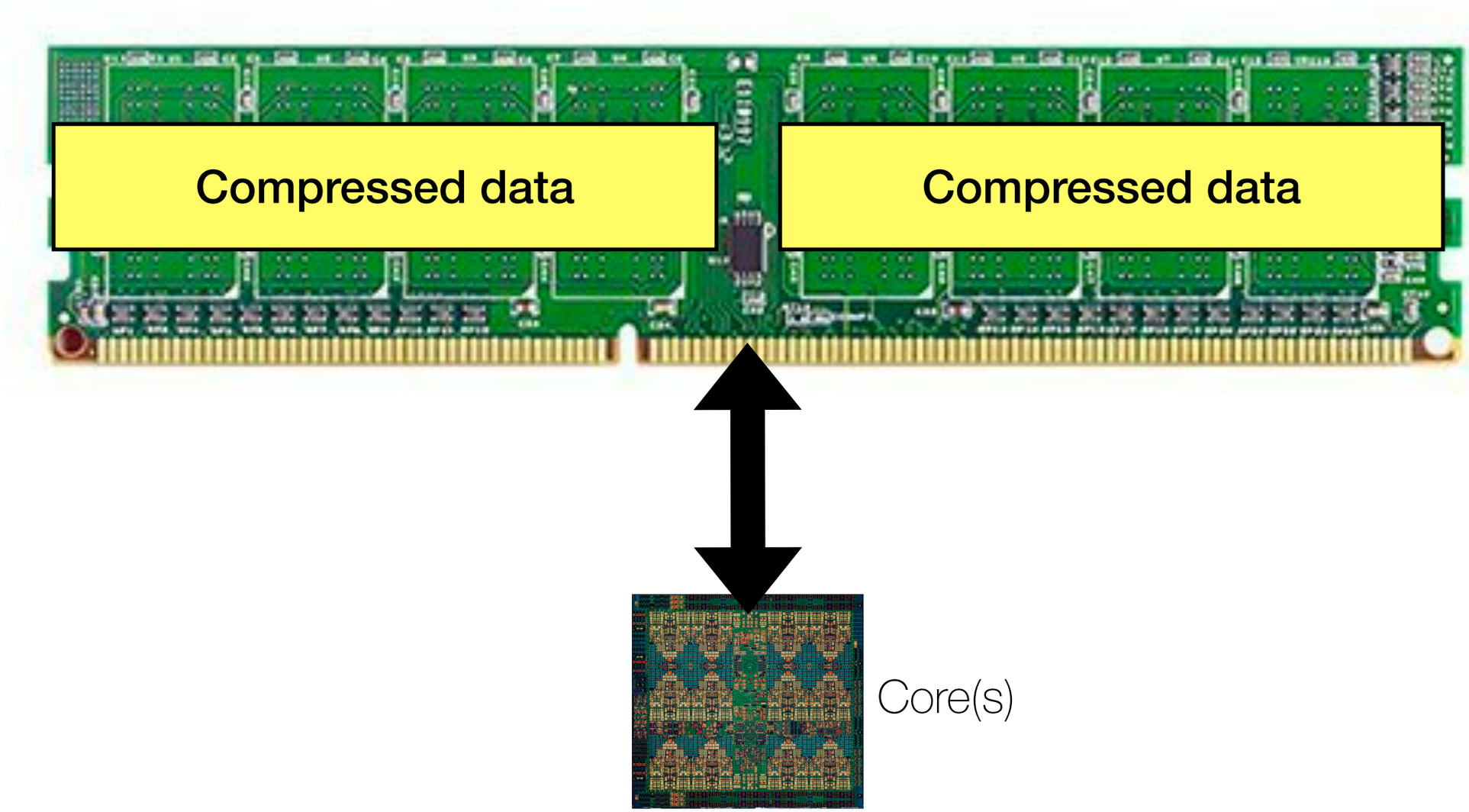
Introduction

Data compression: A simple technique to improve bandwidth



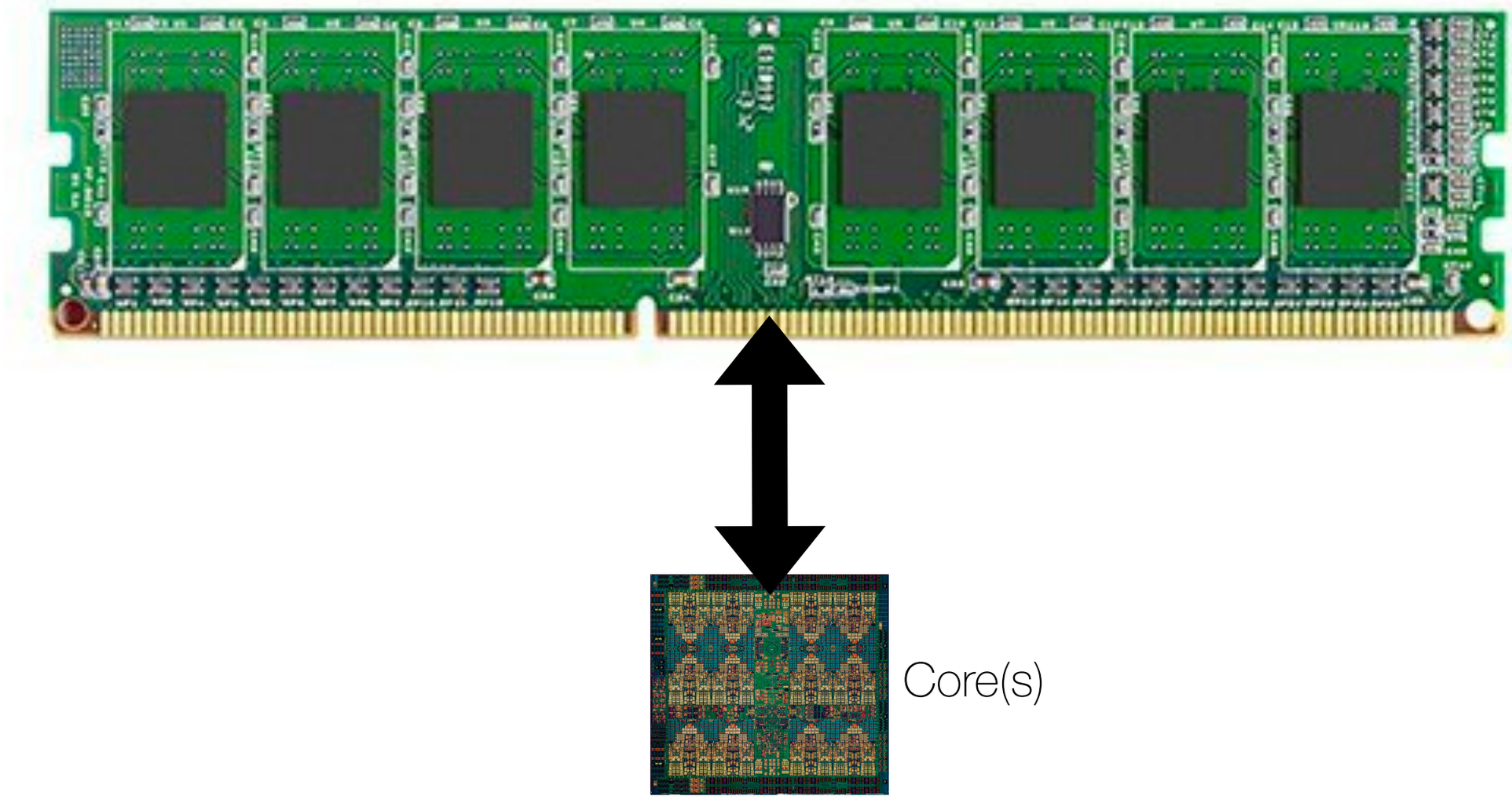
Introduction

Data compression: A simple technique to improve bandwidth



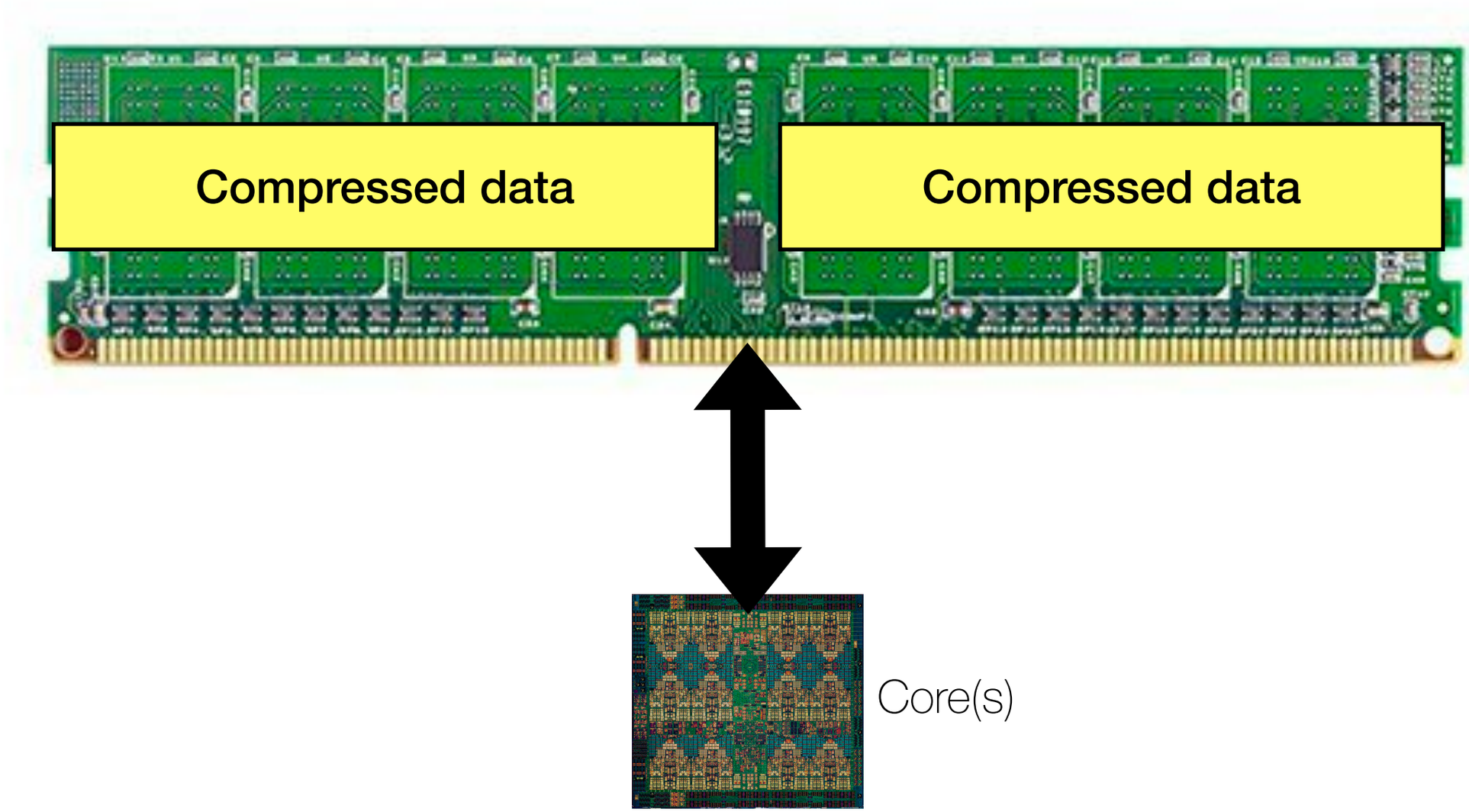
Introduction

Data compression: A simple technique to improve bandwidth



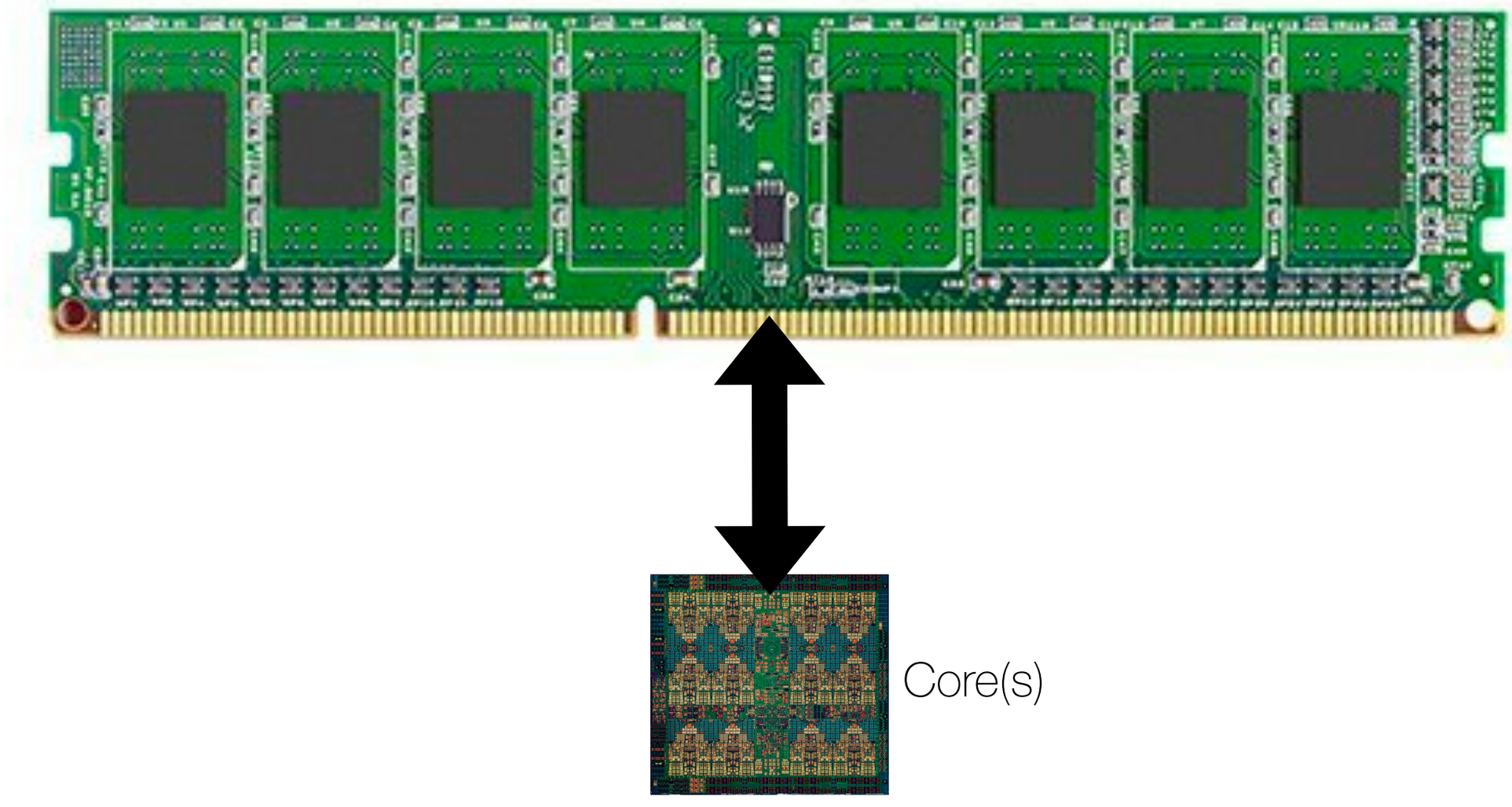
Introduction

Data compression: A simple technique to improve bandwidth



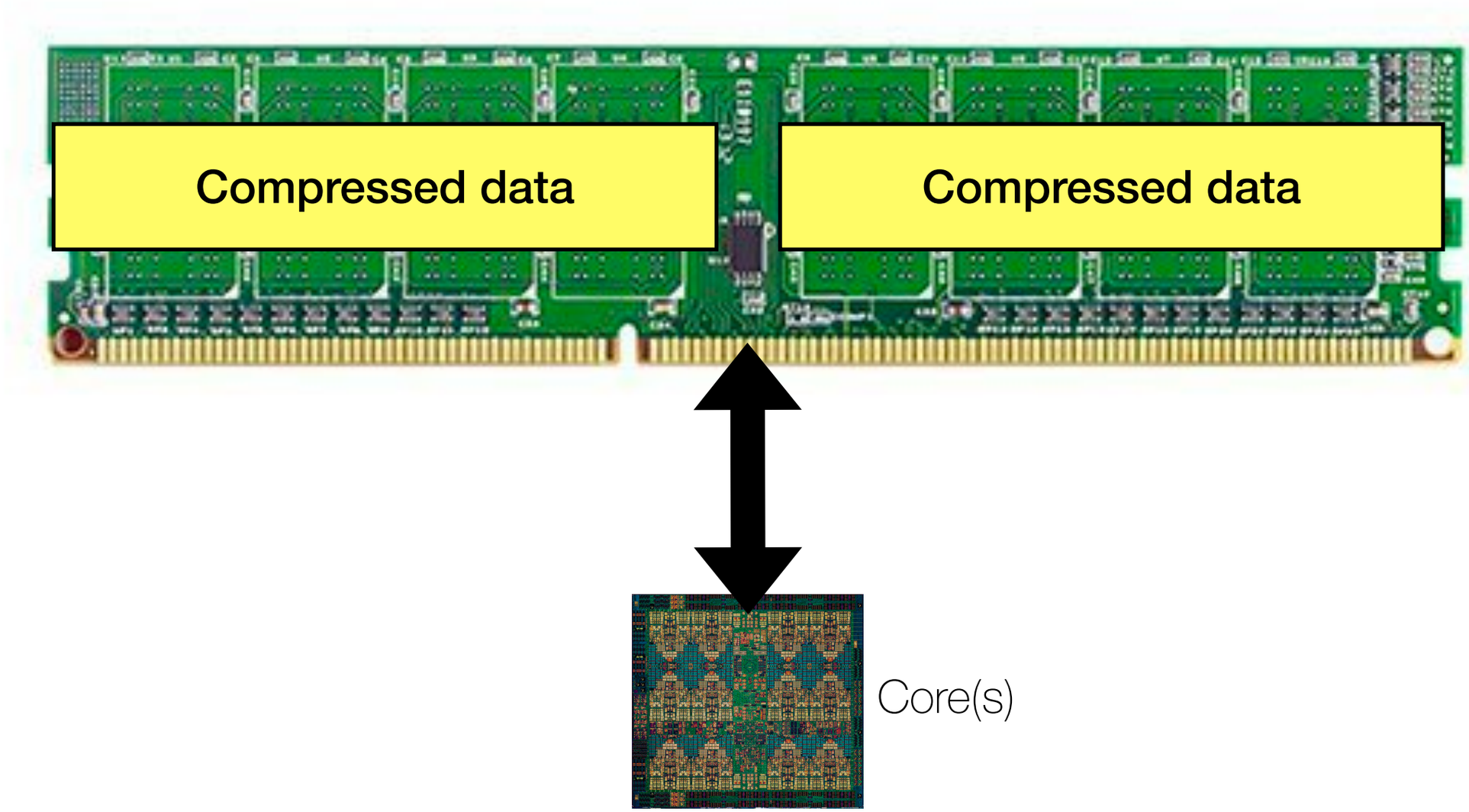
Introduction

Data compression: A simple technique to improve bandwidth



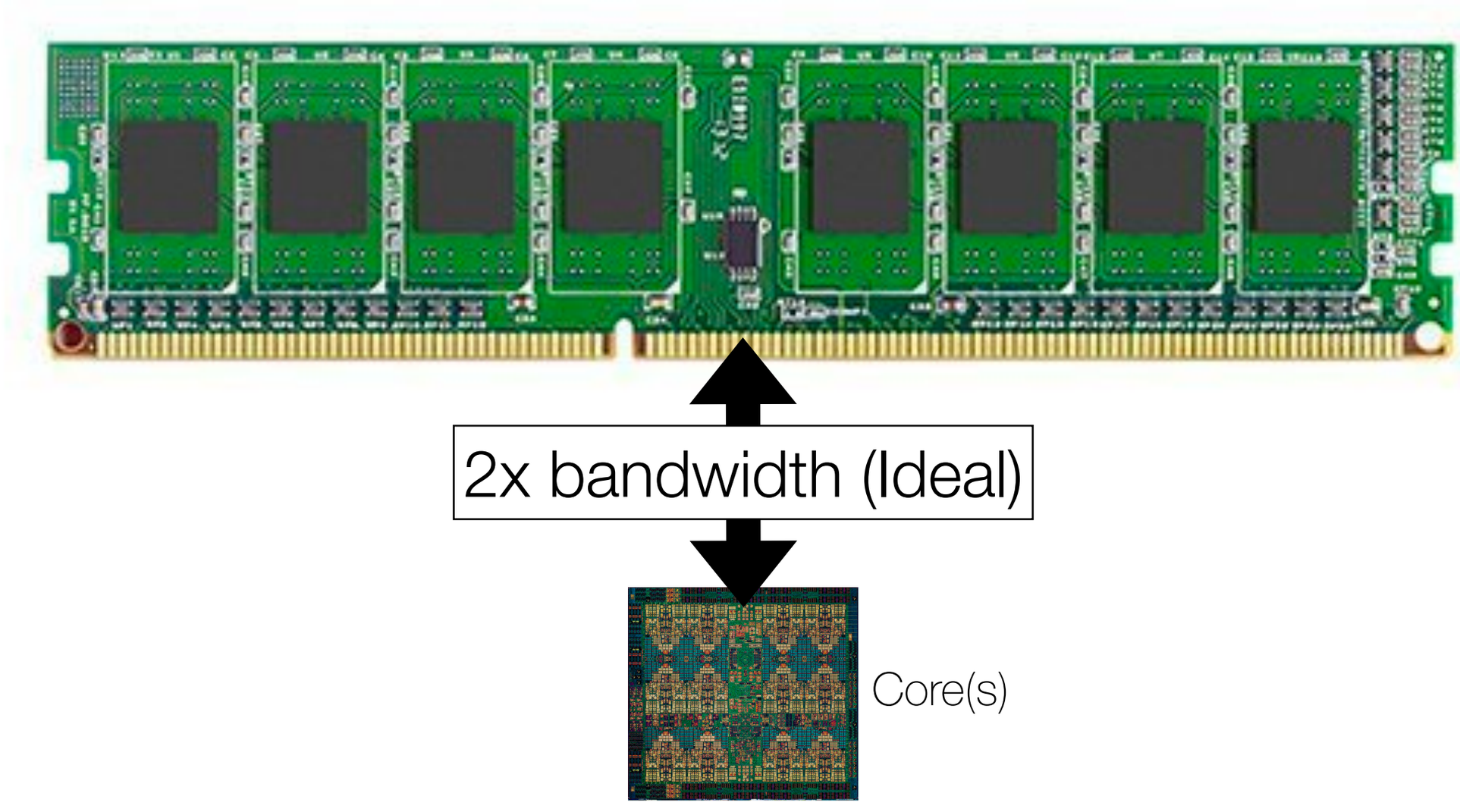
Introduction

Data compression: A simple technique to improve bandwidth



Introduction

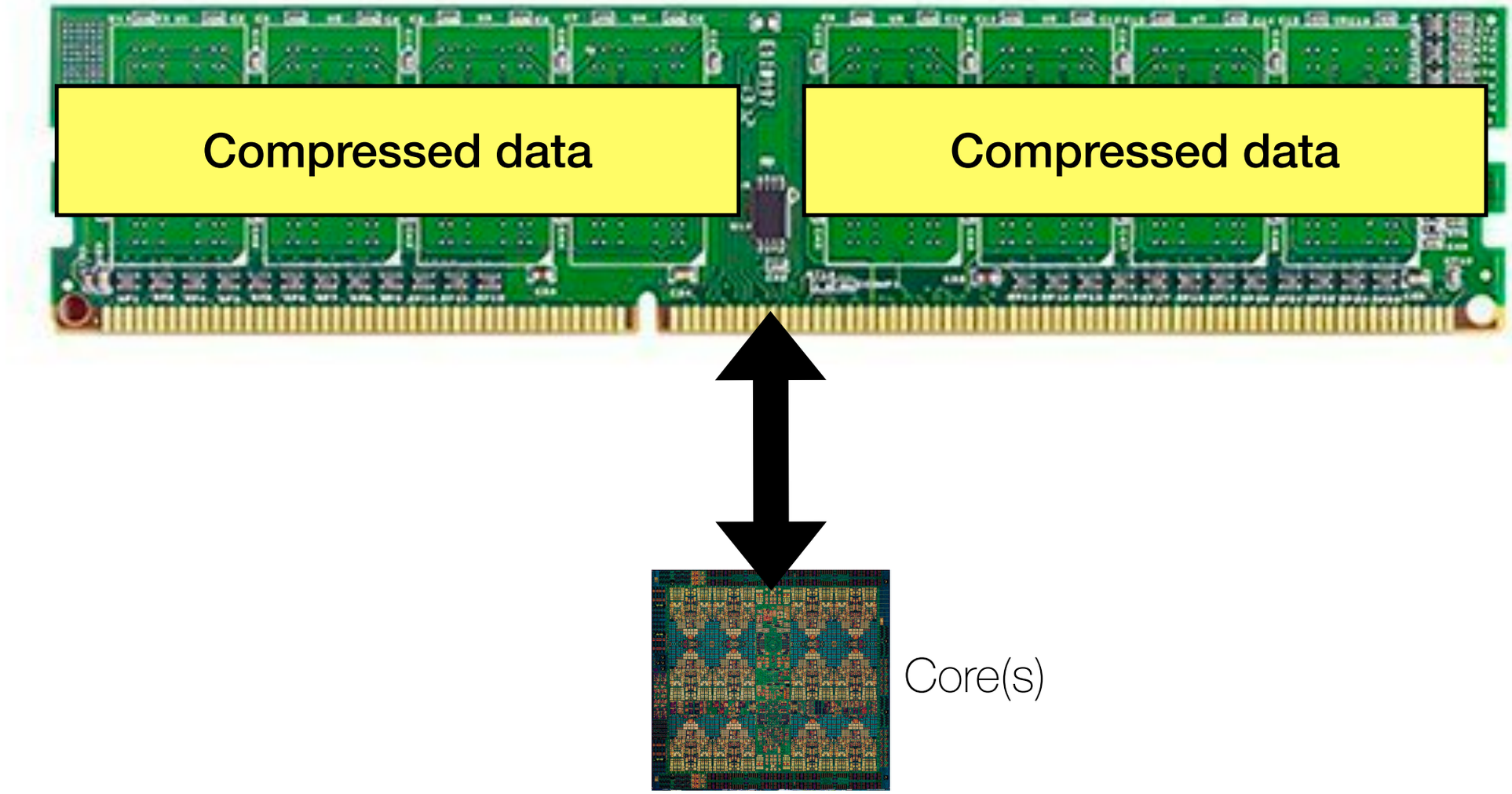
Data compression: A simple technique to improve bandwidth



Introduction

Data compression: A simple technique to improve bandwidth

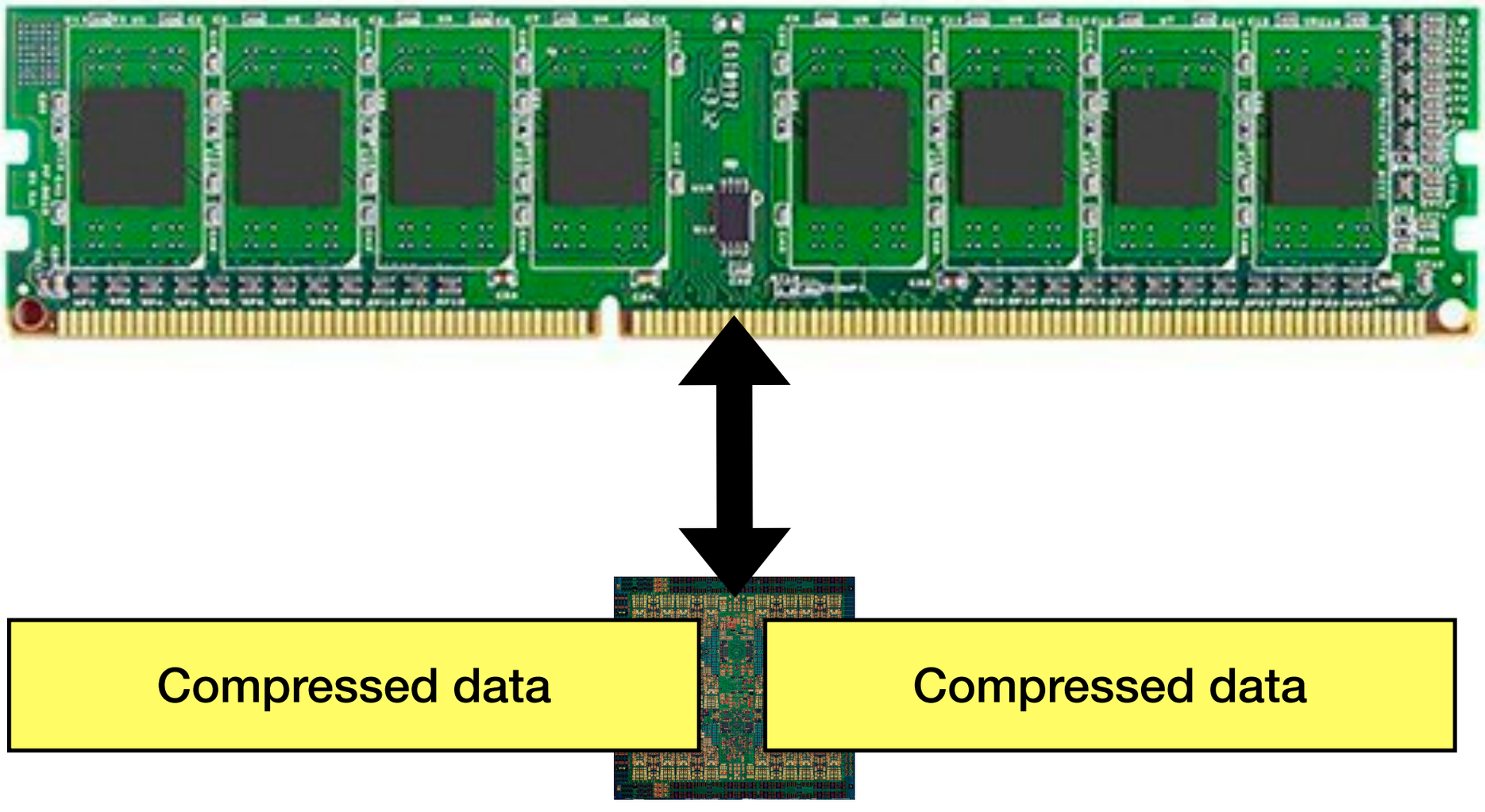
Reading metadata, to identify compressed lines reduces the benefits of compression



Introduction

Data compression: A simple technique to improve bandwidth

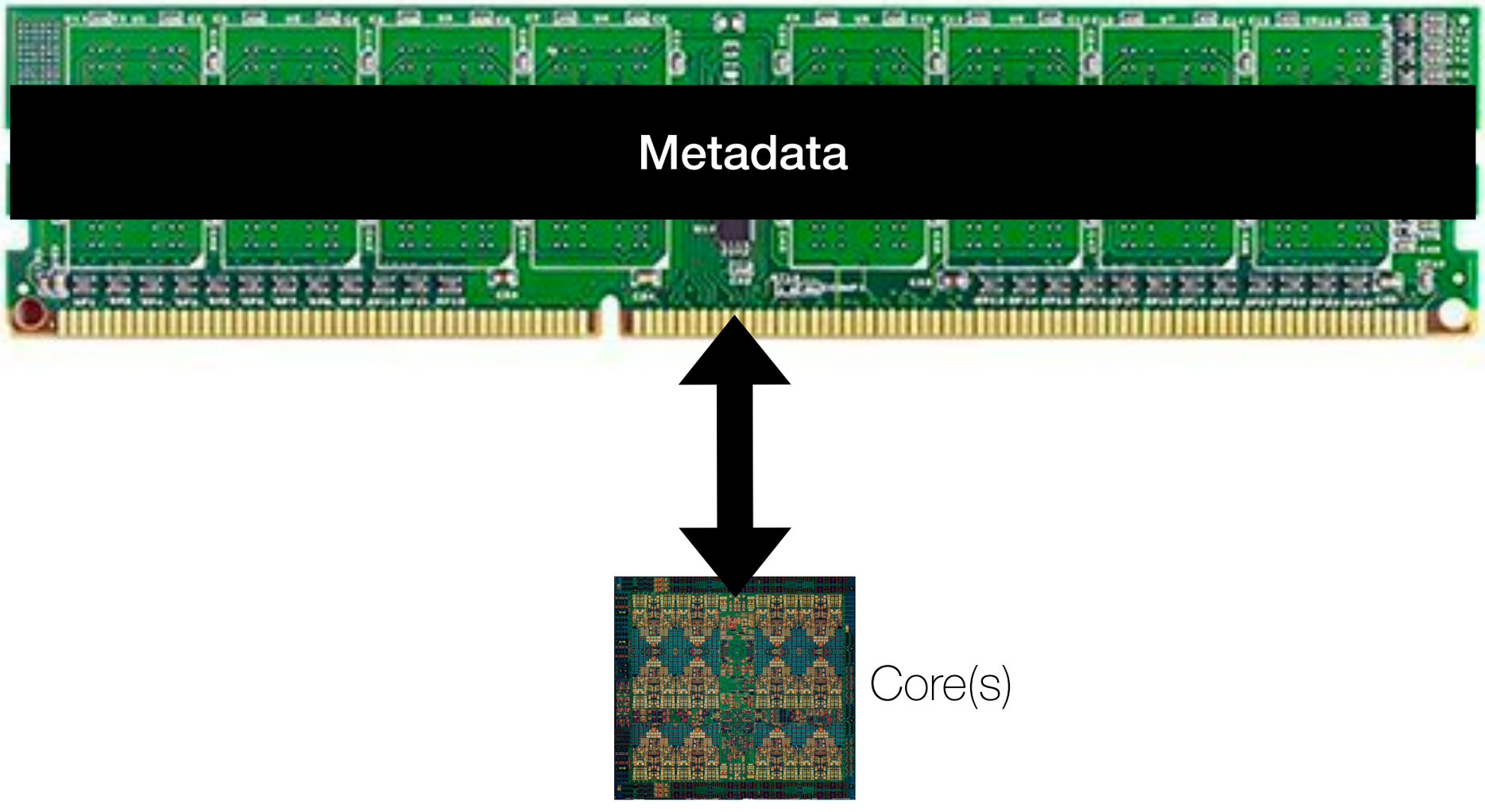
Reading metadata, to identify compressed lines reduces the benefits of compression



Introduction

Data compression: A simple technique to improve bandwidth

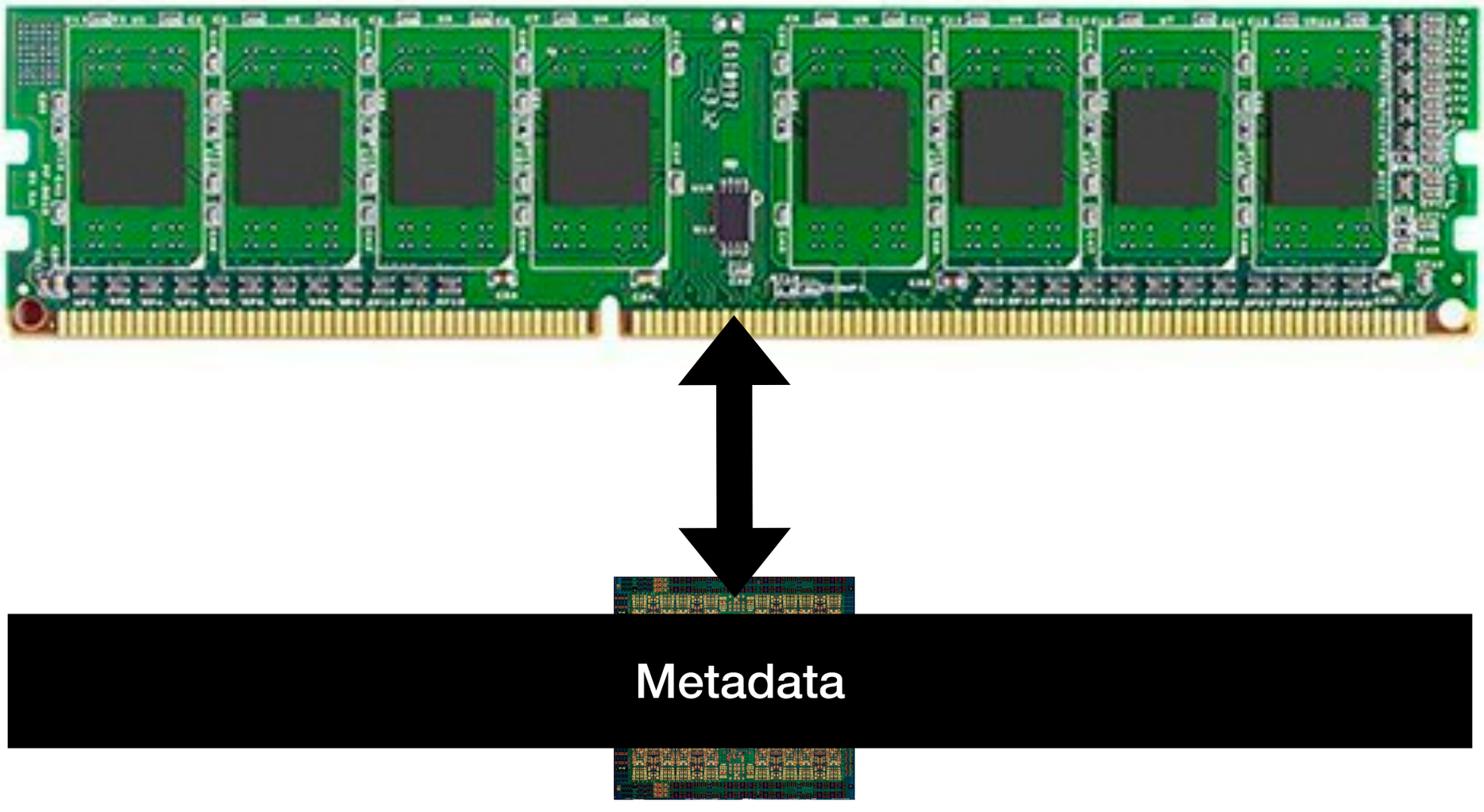
Reading metadata, to identify compressed lines reduces the benefits of compression



Introduction

Data compression: A simple technique to improve bandwidth

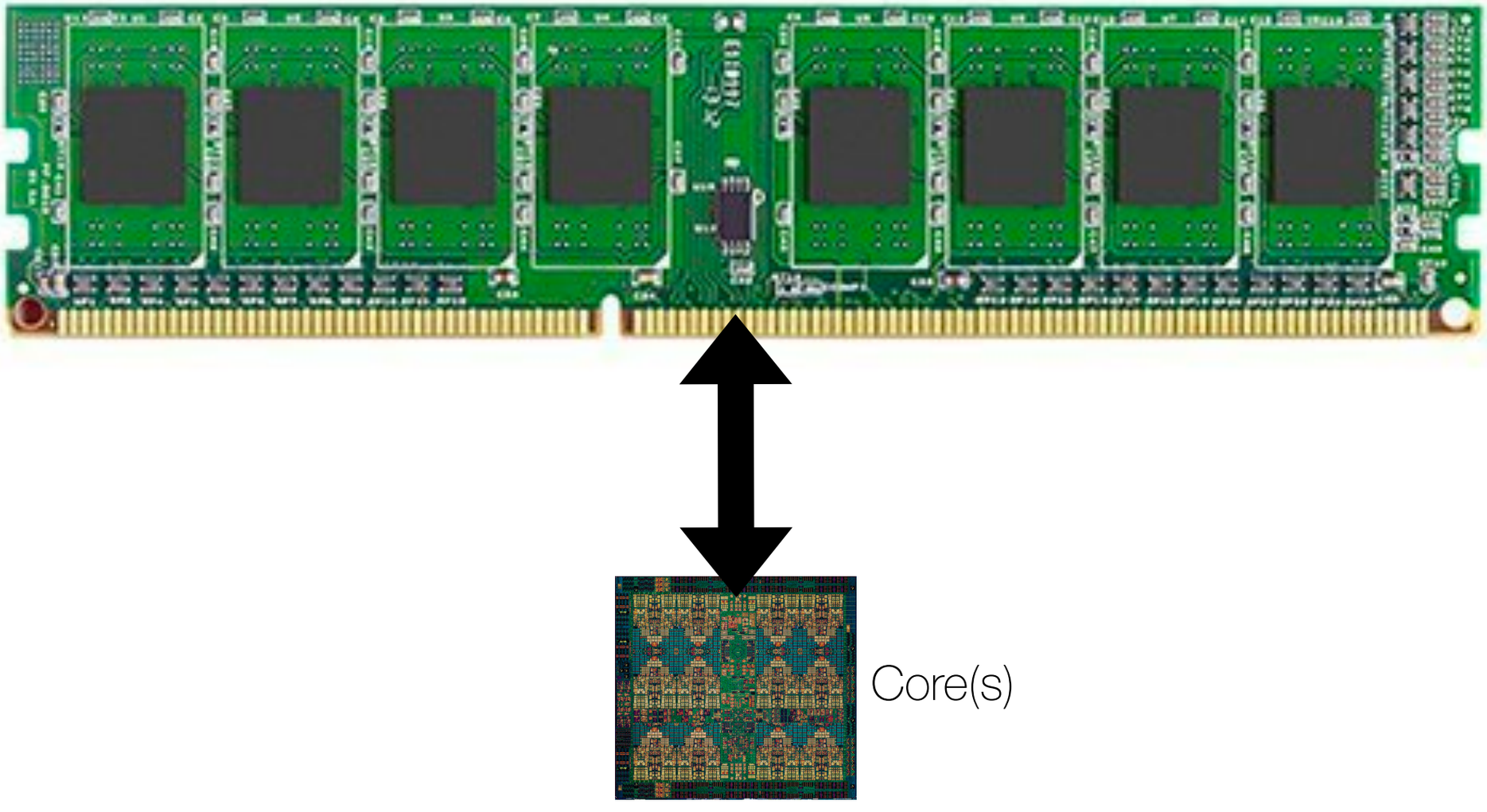
Reading metadata, to identify compressed lines reduces the benefits of compression



Introduction

Data compression: A simple technique to improve bandwidth

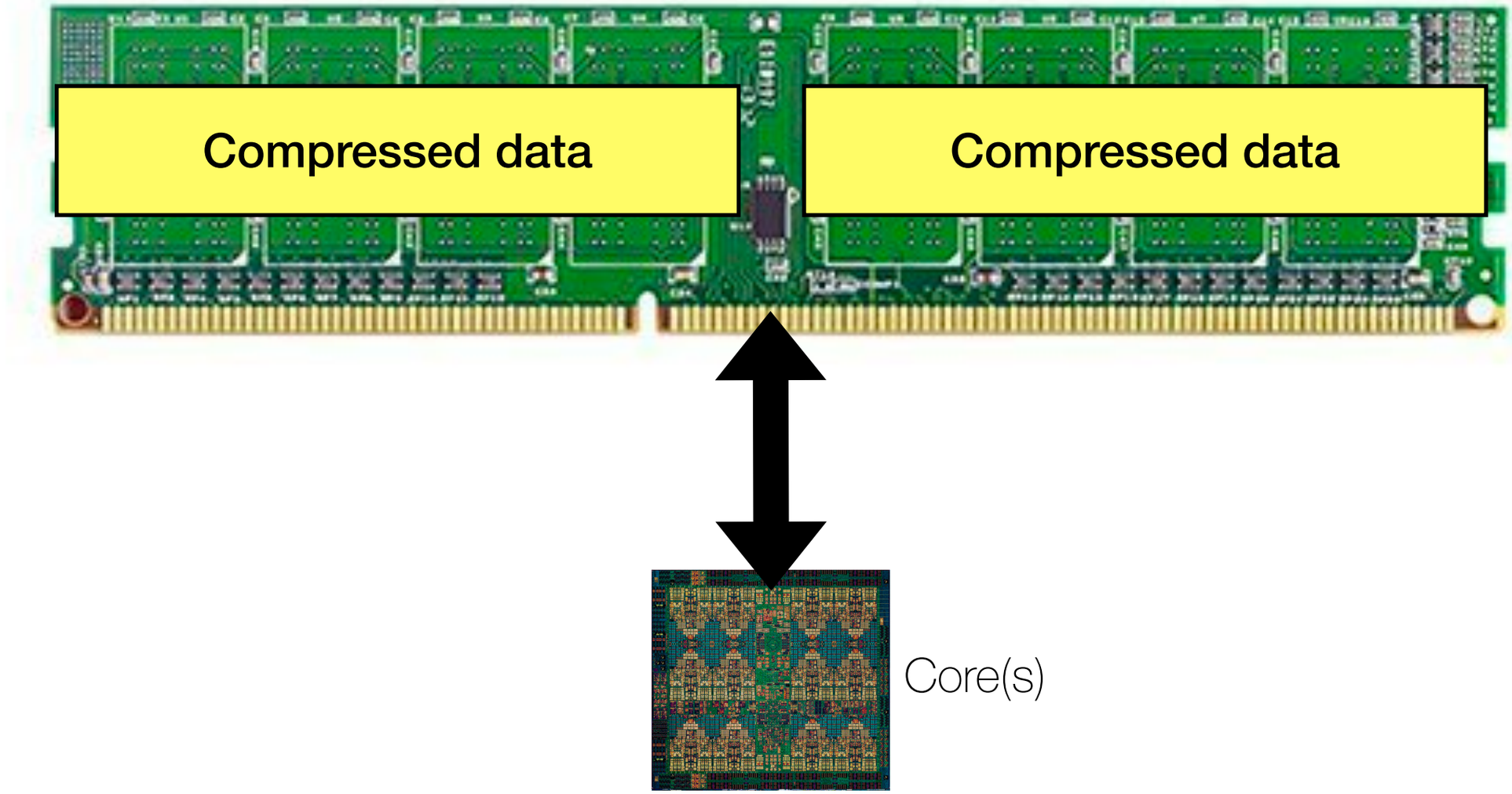
Reading metadata, to identify compressed lines reduces the benefits of compression



Introduction

Data compression: A simple technique to improve bandwidth

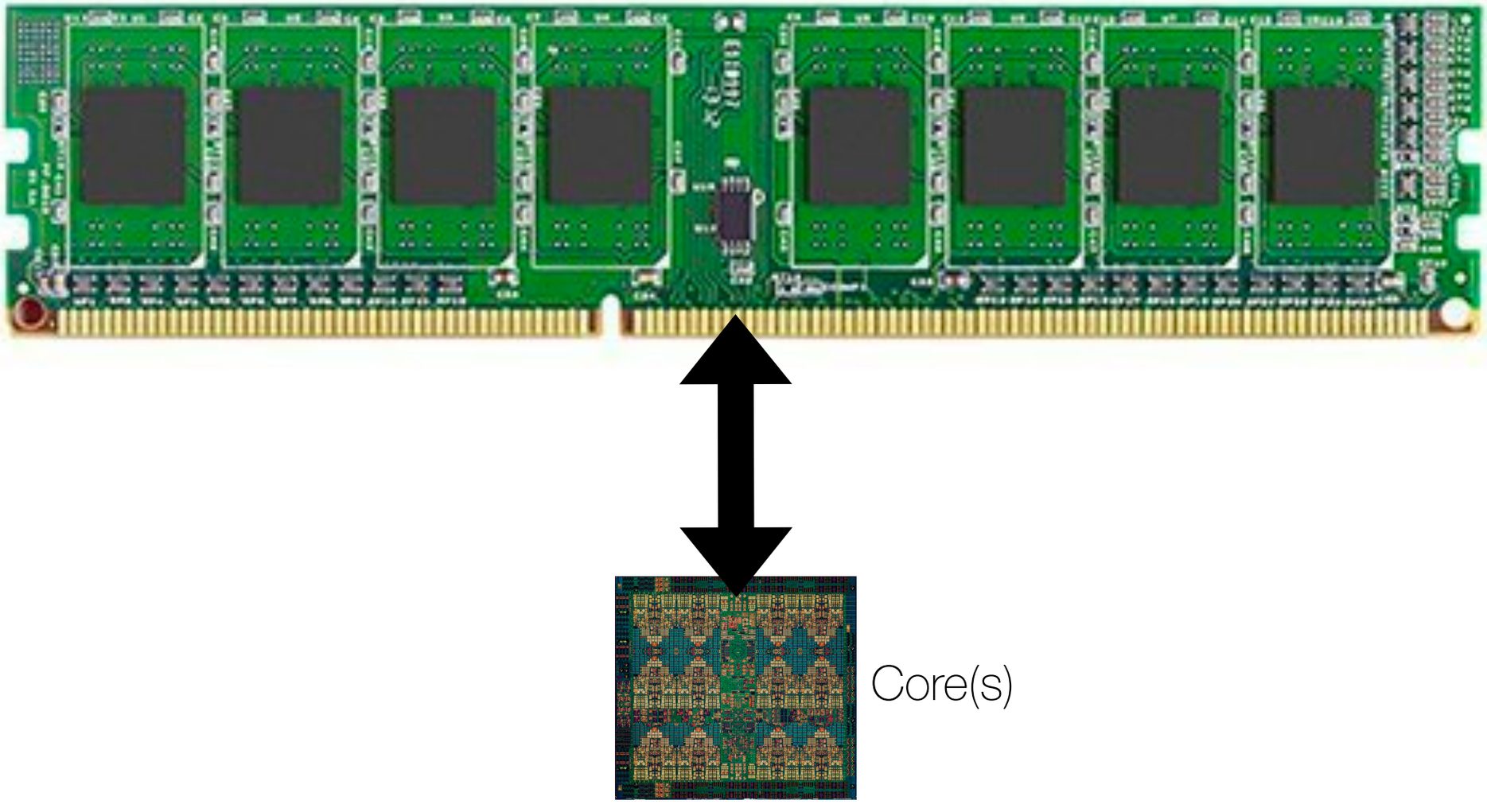
Reading metadata, to identify compressed lines reduces the benefits of compression



Introduction

Data compression: A simple technique to improve bandwidth

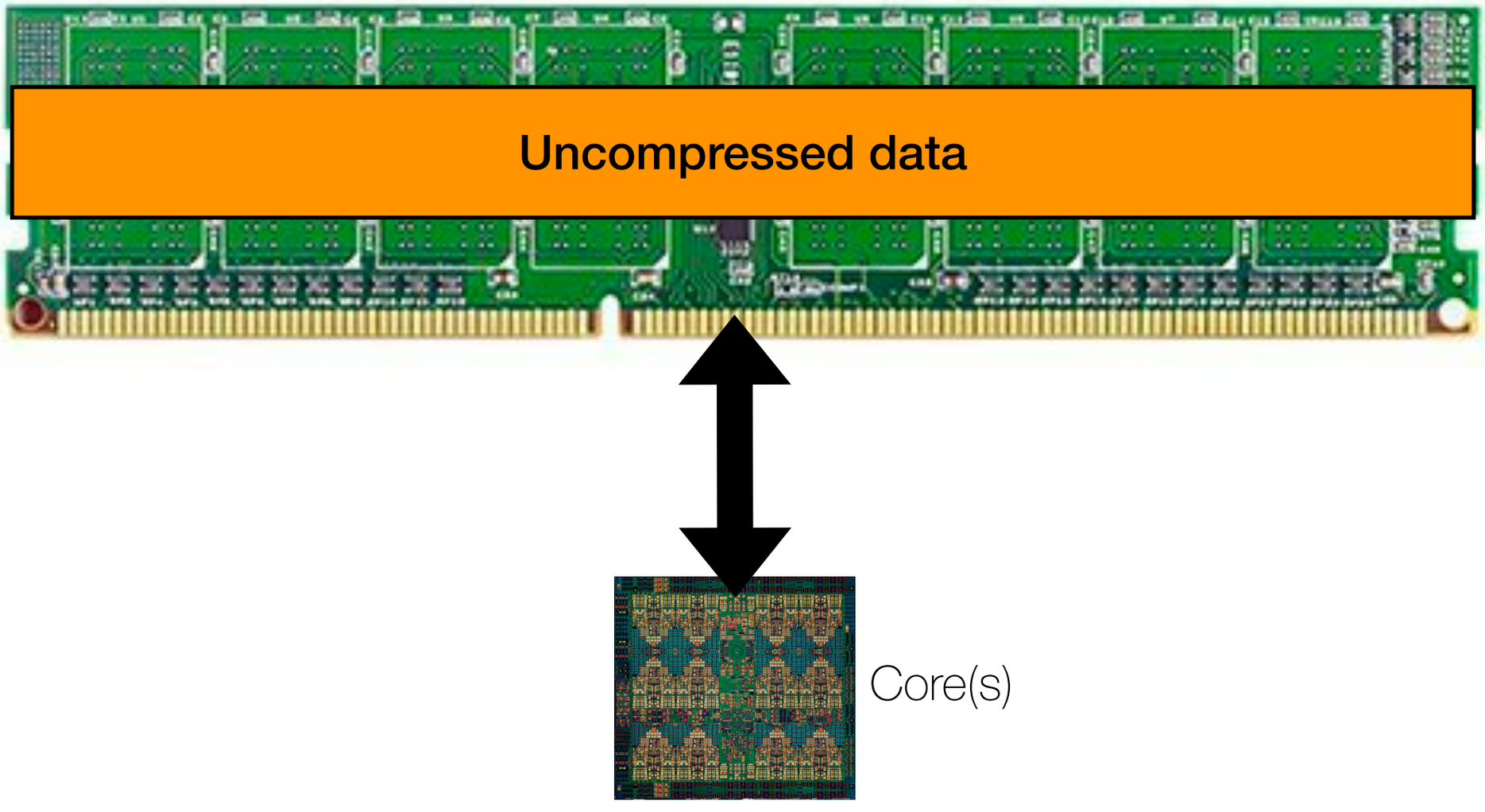
Reading metadata, to identify compressed lines reduces the benefits of compression



Introduction

Data compression: A simple technique to improve bandwidth

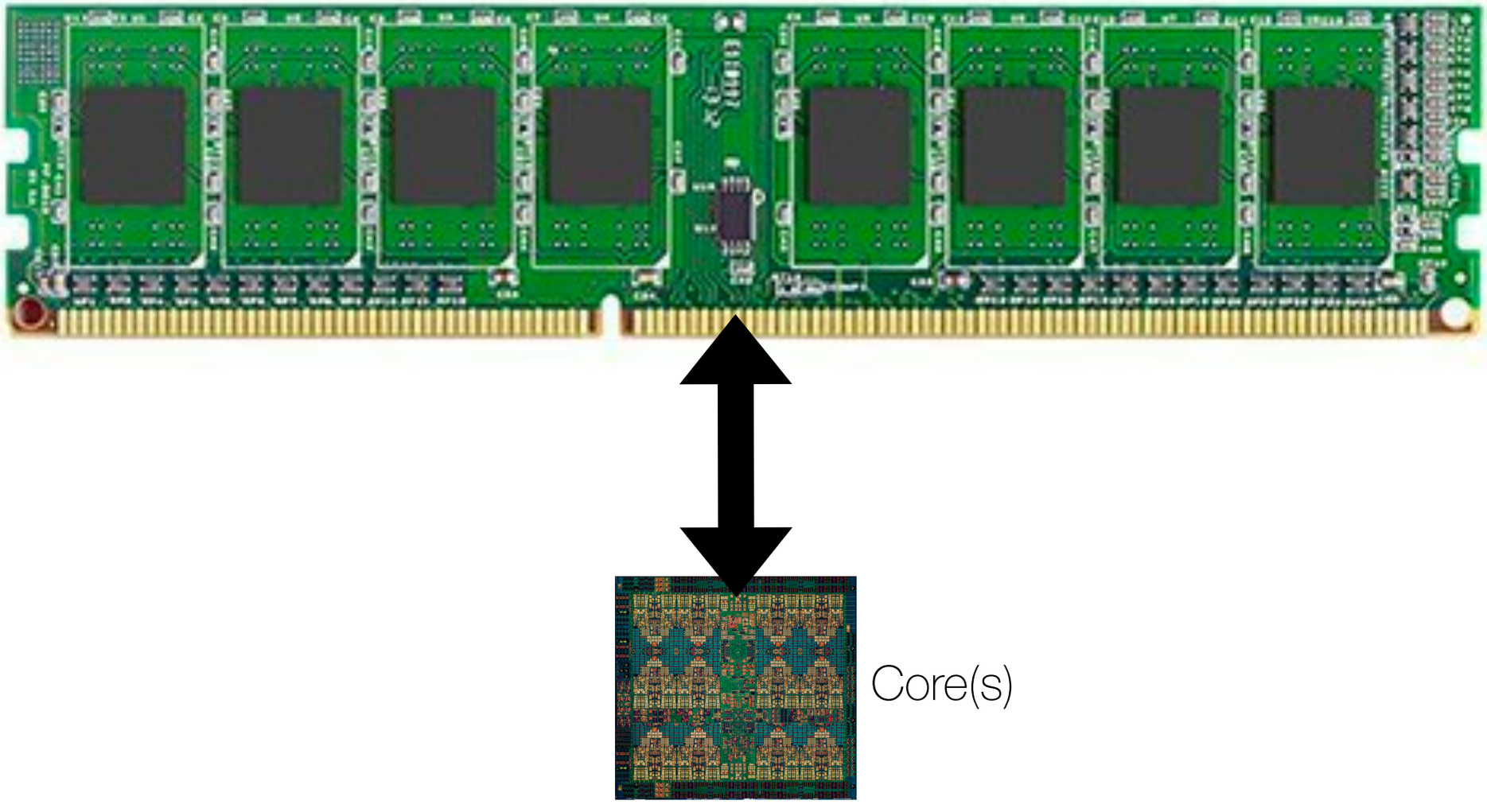
Reading metadata, to identify compressed lines reduces the benefits of compression



Introduction

Data compression: A simple technique to improve bandwidth

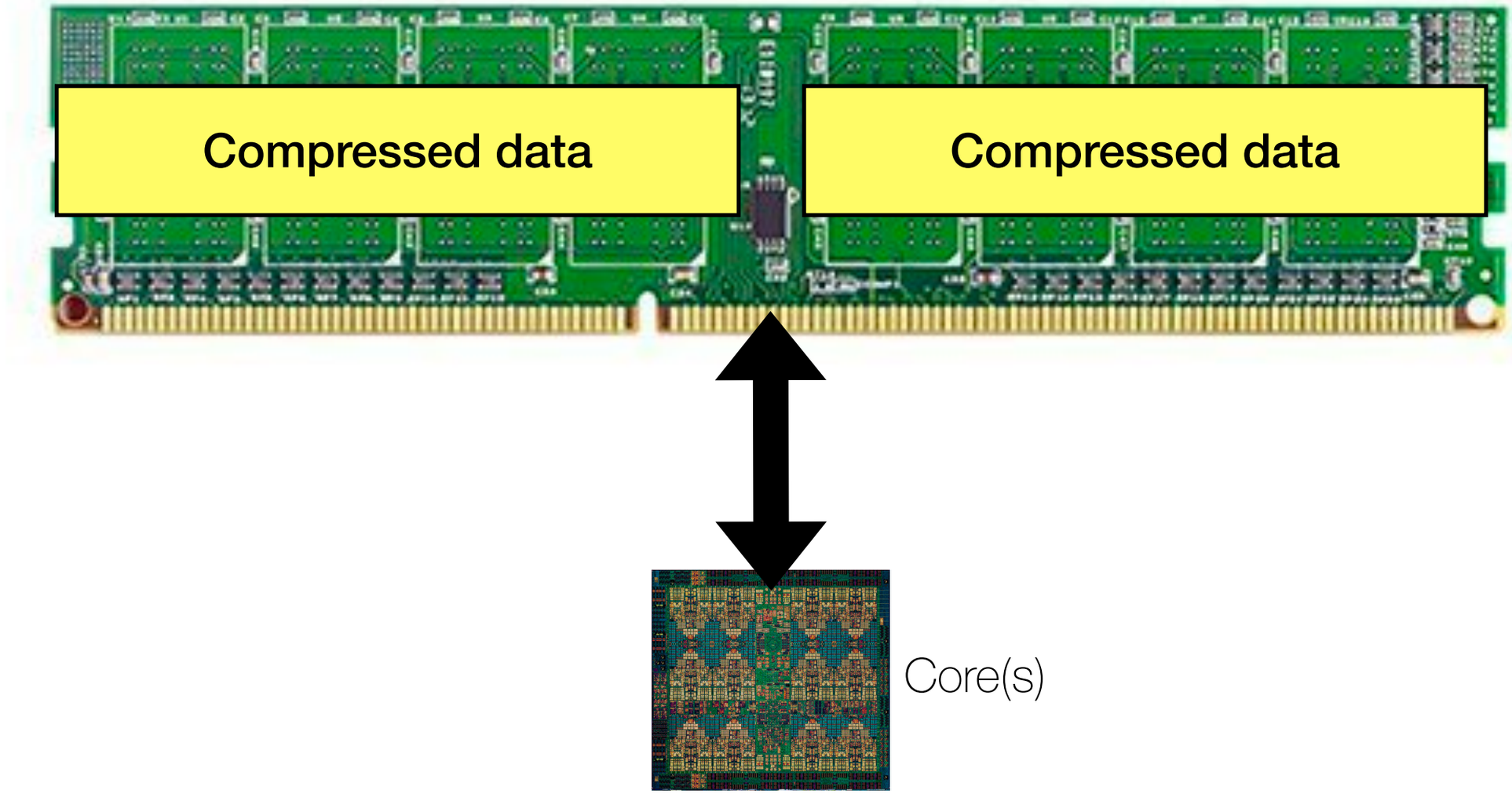
Reading metadata, to identify compressed lines reduces the benefits of compression



Introduction

Data compression: A simple technique to improve bandwidth

Reading metadata, to identify compressed lines reduces the benefits of compression



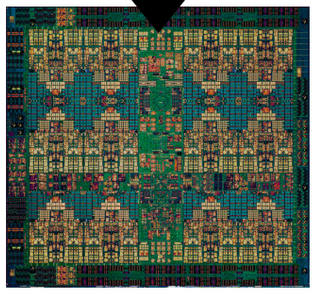
Introduction

Data compression: A simple technique to improve bandwidth

Reading metadata, to identify compressed lines reduces the benefits of compression

Main Memory

< 2x bandwidth (in practice)



Core(s)

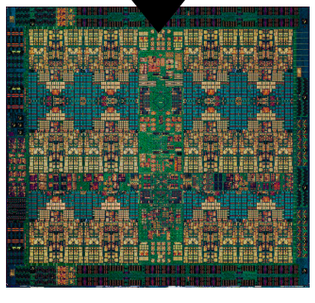
Introduction

Data compression: A simple technique to improve bandwidth

Reading metadata, to identify compressed lines reduces the benefits of compression

Main Memory

< 2x bandwidth (in practice)



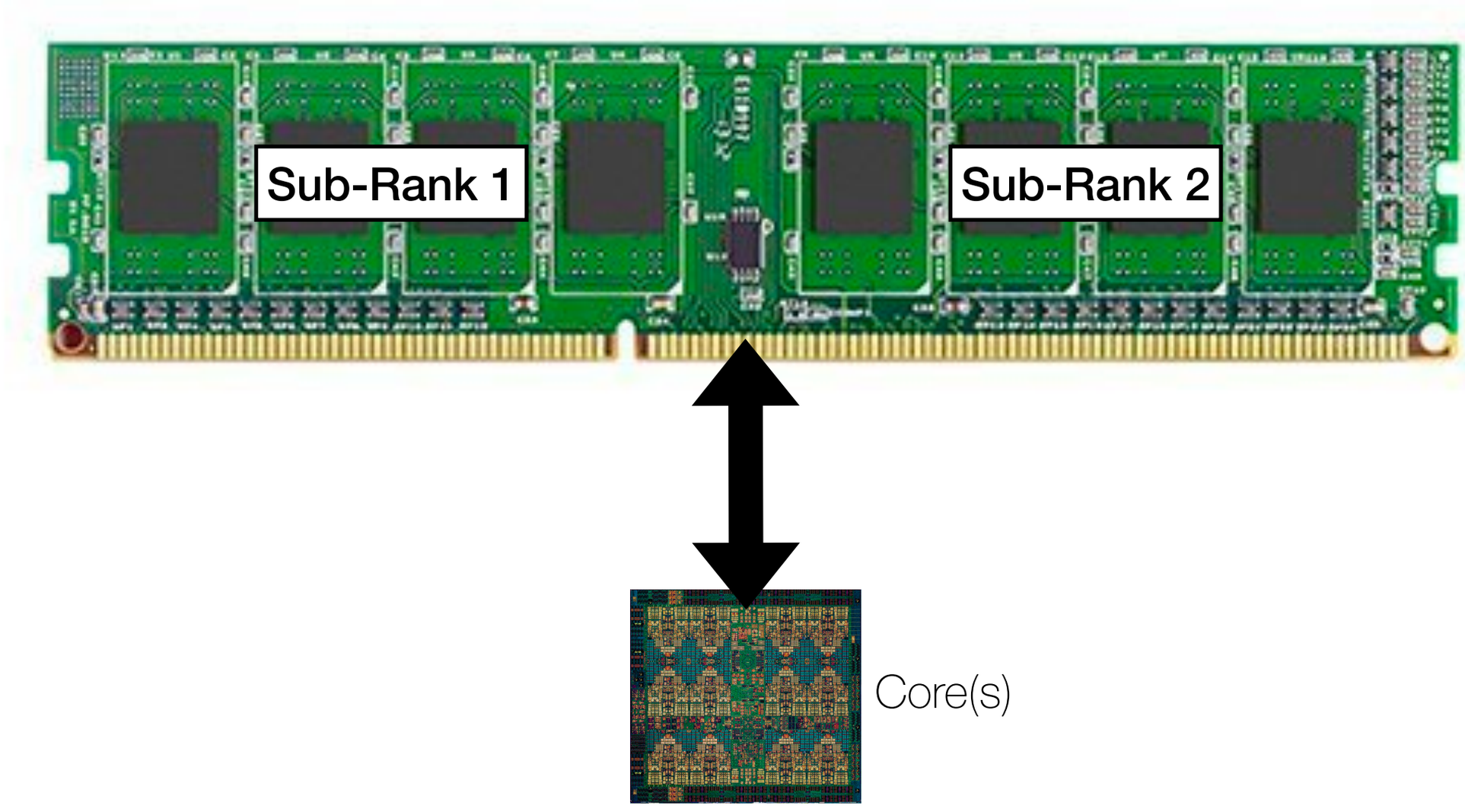
Core(s)

Need to mitigate additional bandwidth overheads from Metadata

- ✦ Introduction
- ✦ Background and Motivation
- ✦ Goal
- ✦ Attaché
  - ● *Blended Metadata*
  - ● *Compressibility Predictor*
- ✦ Results
- ✦ Summary

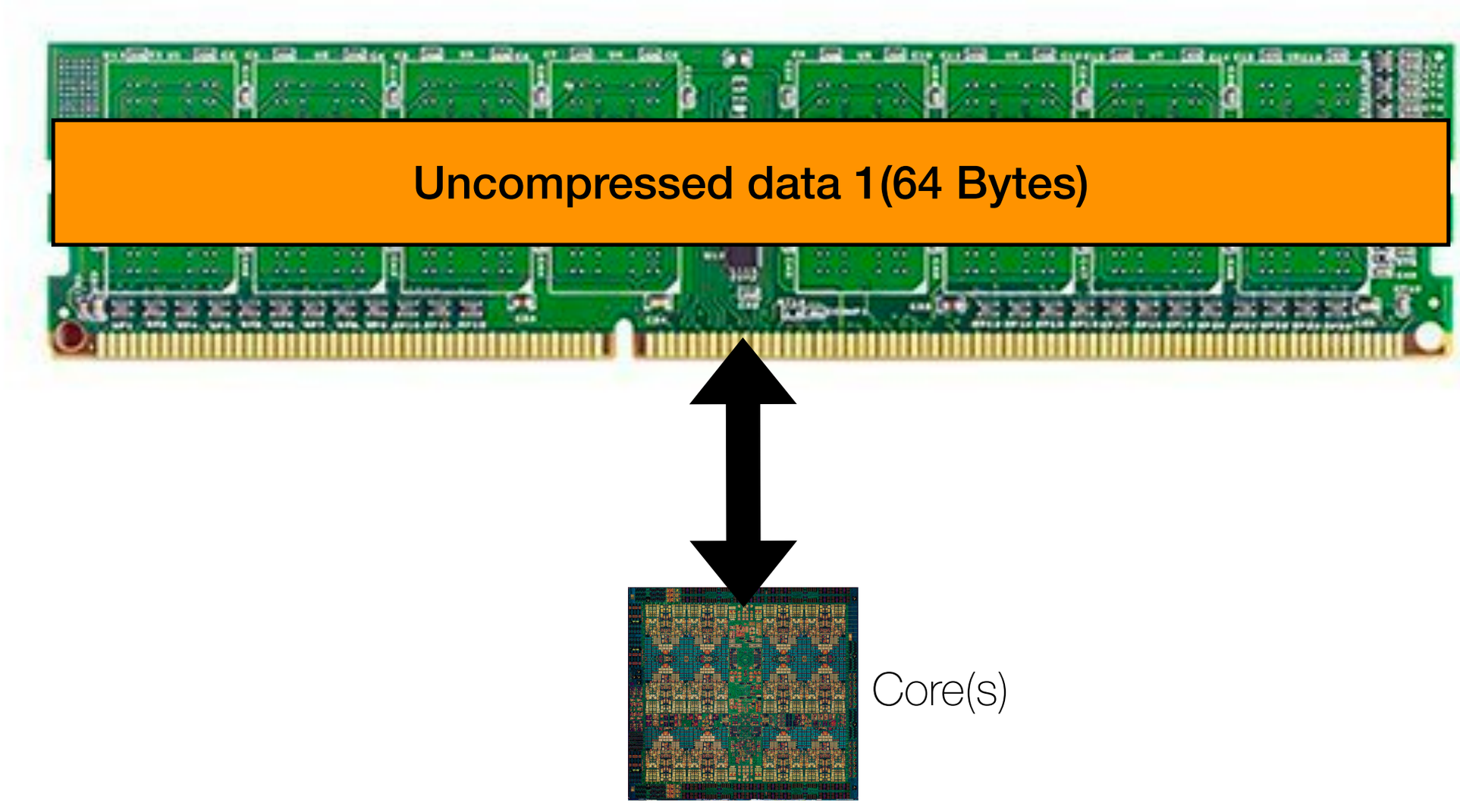
Background

Sub-Ranking: Split the memory module into smaller channels



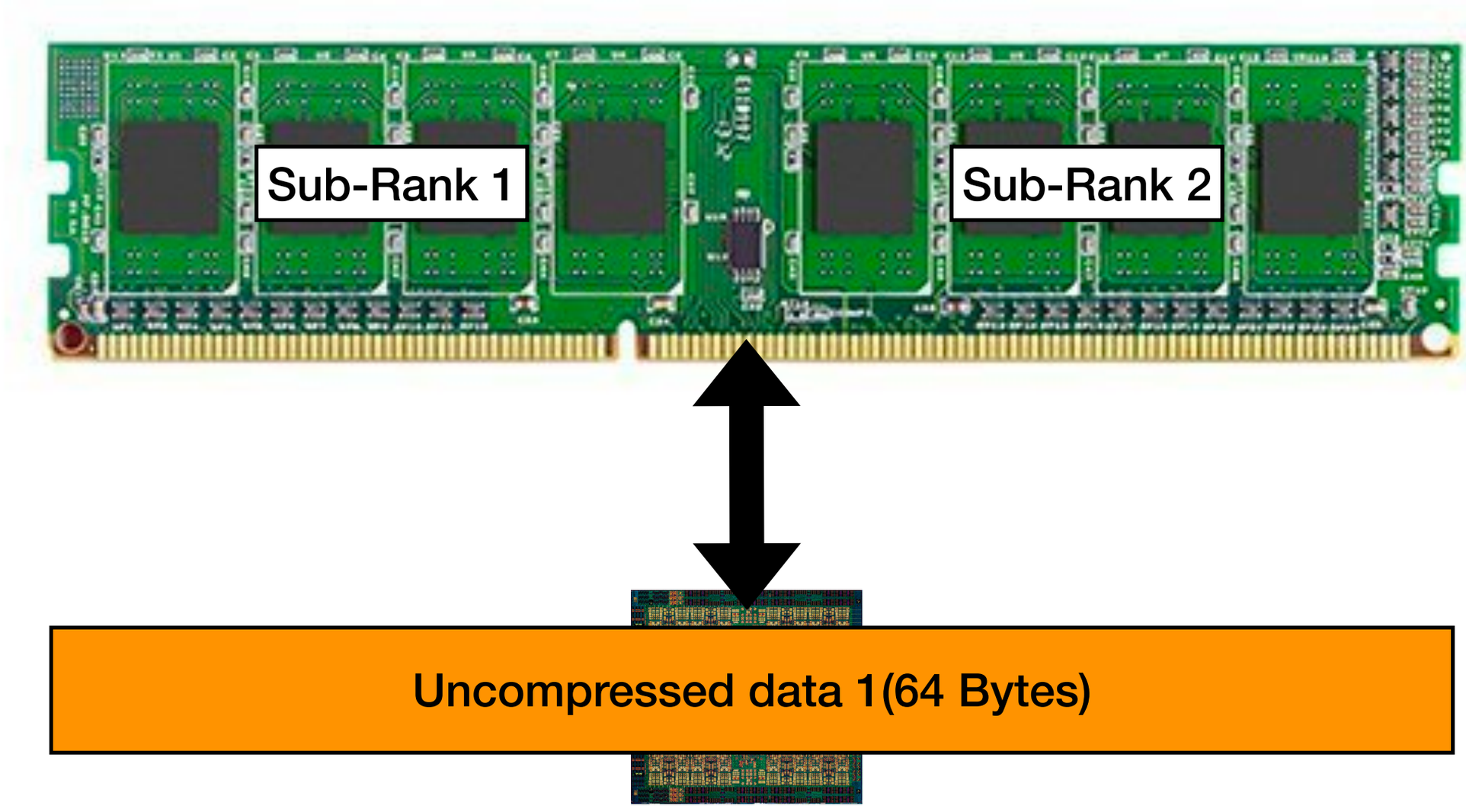
Background

Sub-Ranking: Split the memory module into smaller channels



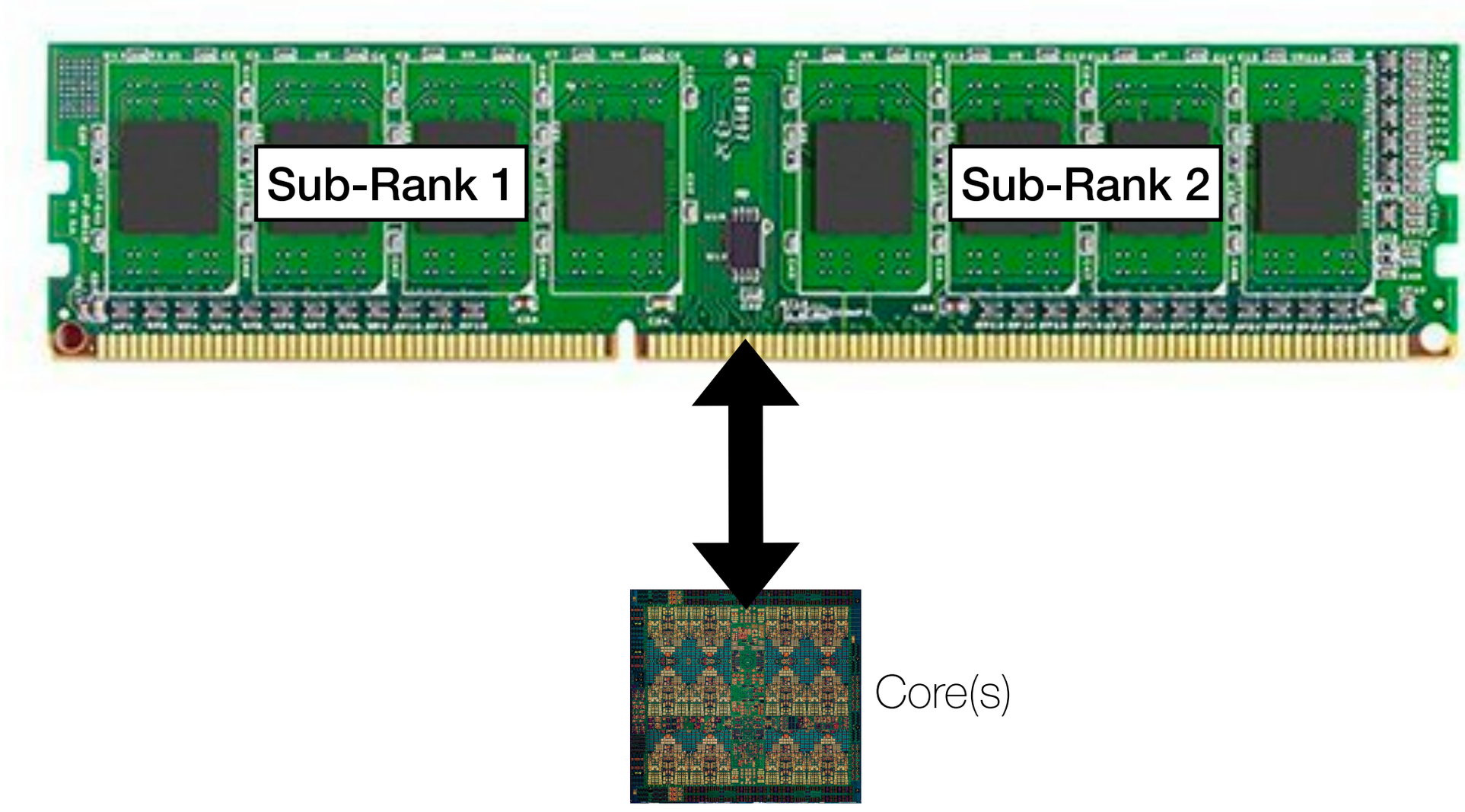
Background

Sub-Ranking: Split the memory module into smaller channels



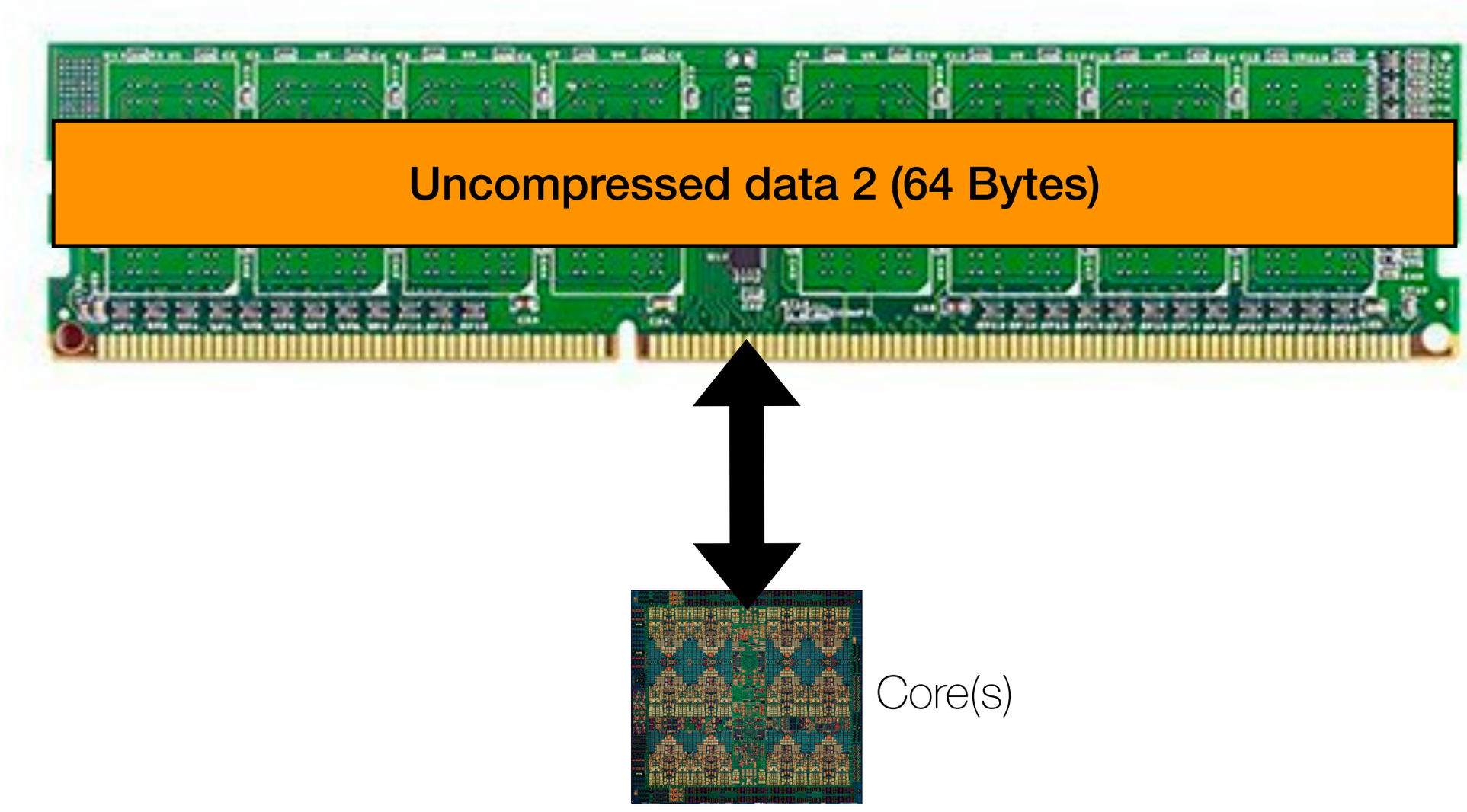
Background

Sub-Ranking: Split the memory module into smaller channels



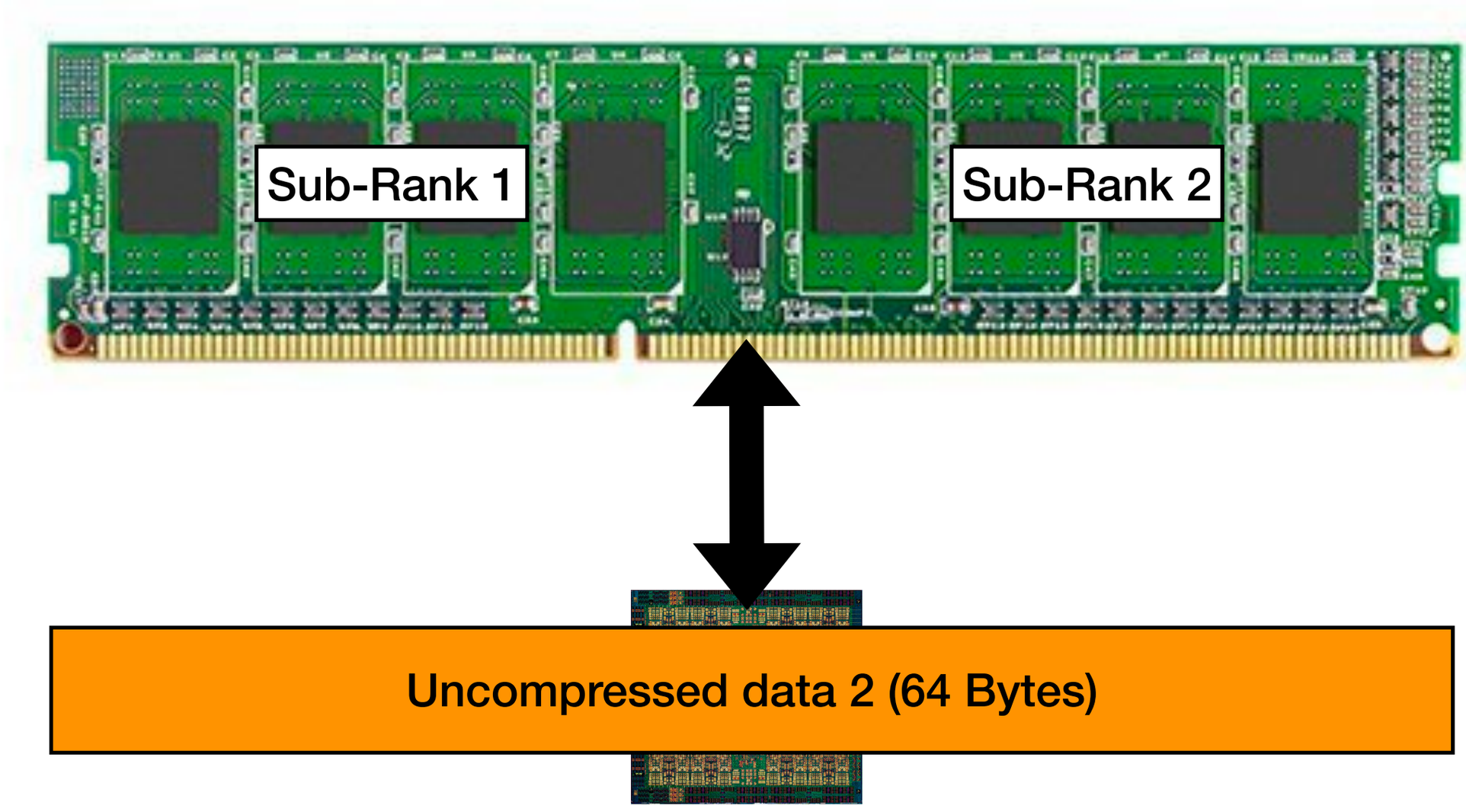
Background

Sub-Ranking: Split the memory module into smaller channels



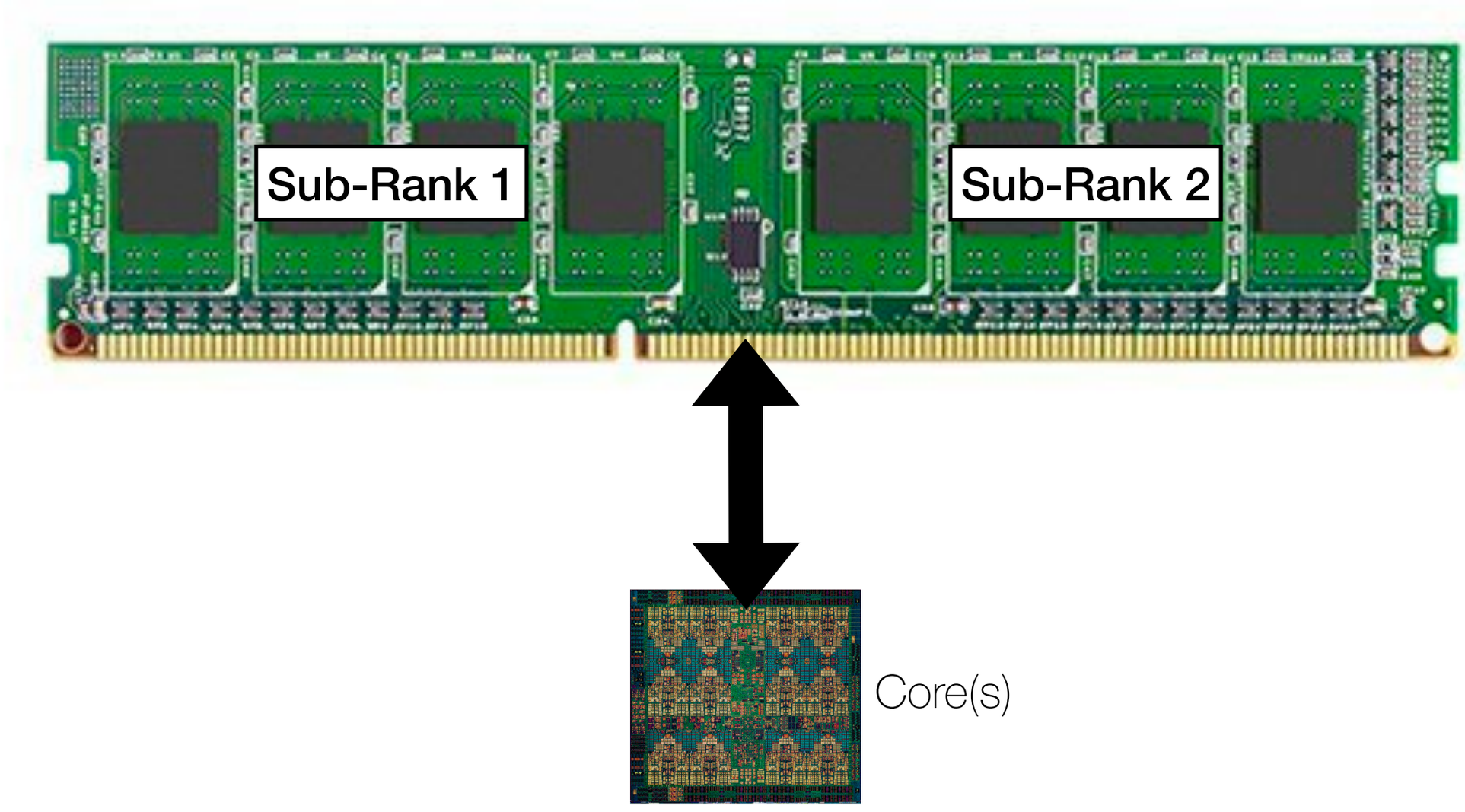
Background

Sub-Ranking: Split the memory module into smaller channels



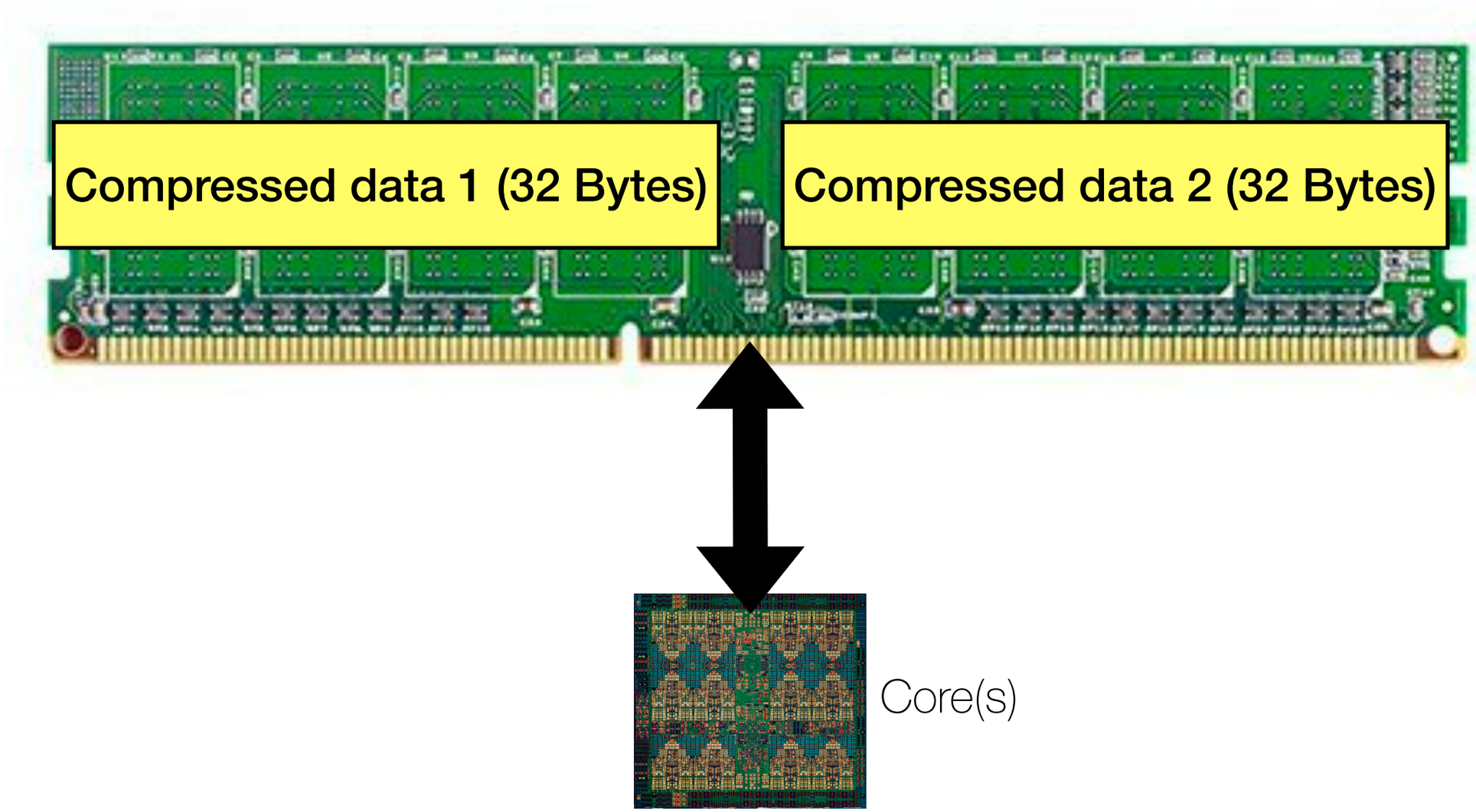
Background

Sub-Ranking: Split the memory module into smaller channels



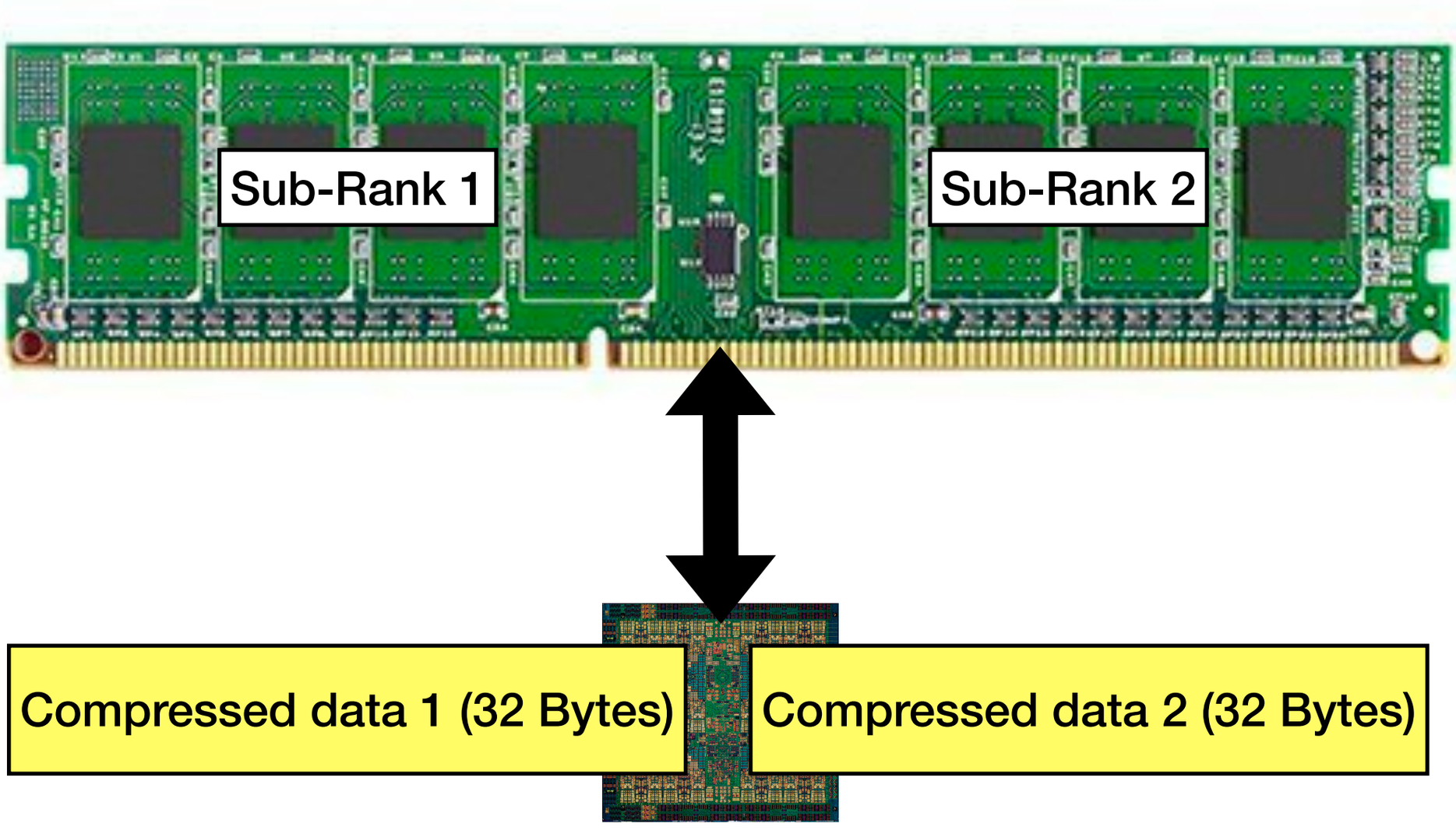
Background

Sub-Ranking: Split the memory module into smaller channels



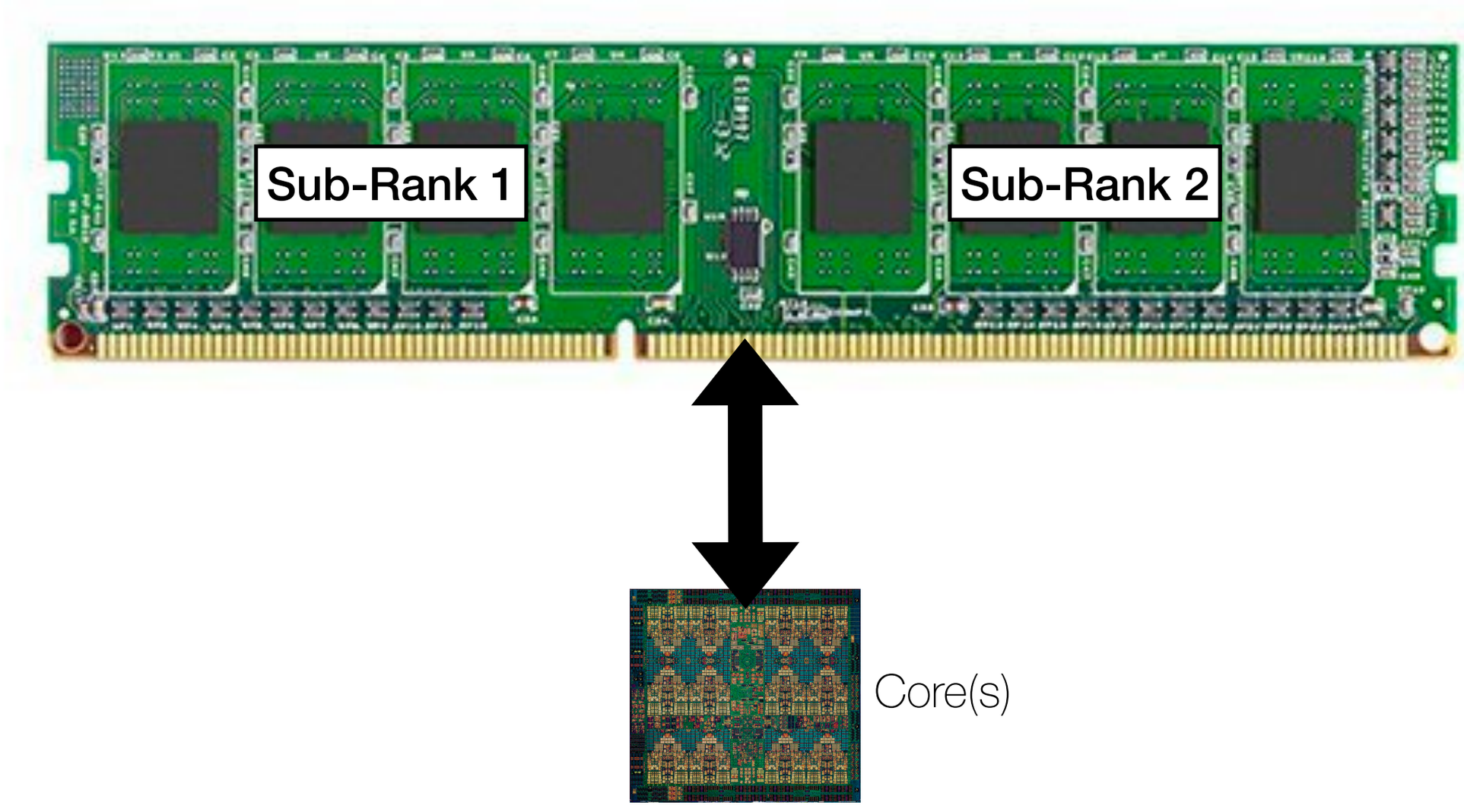
Background

Sub-Ranking: Split the memory module into smaller channels



Background

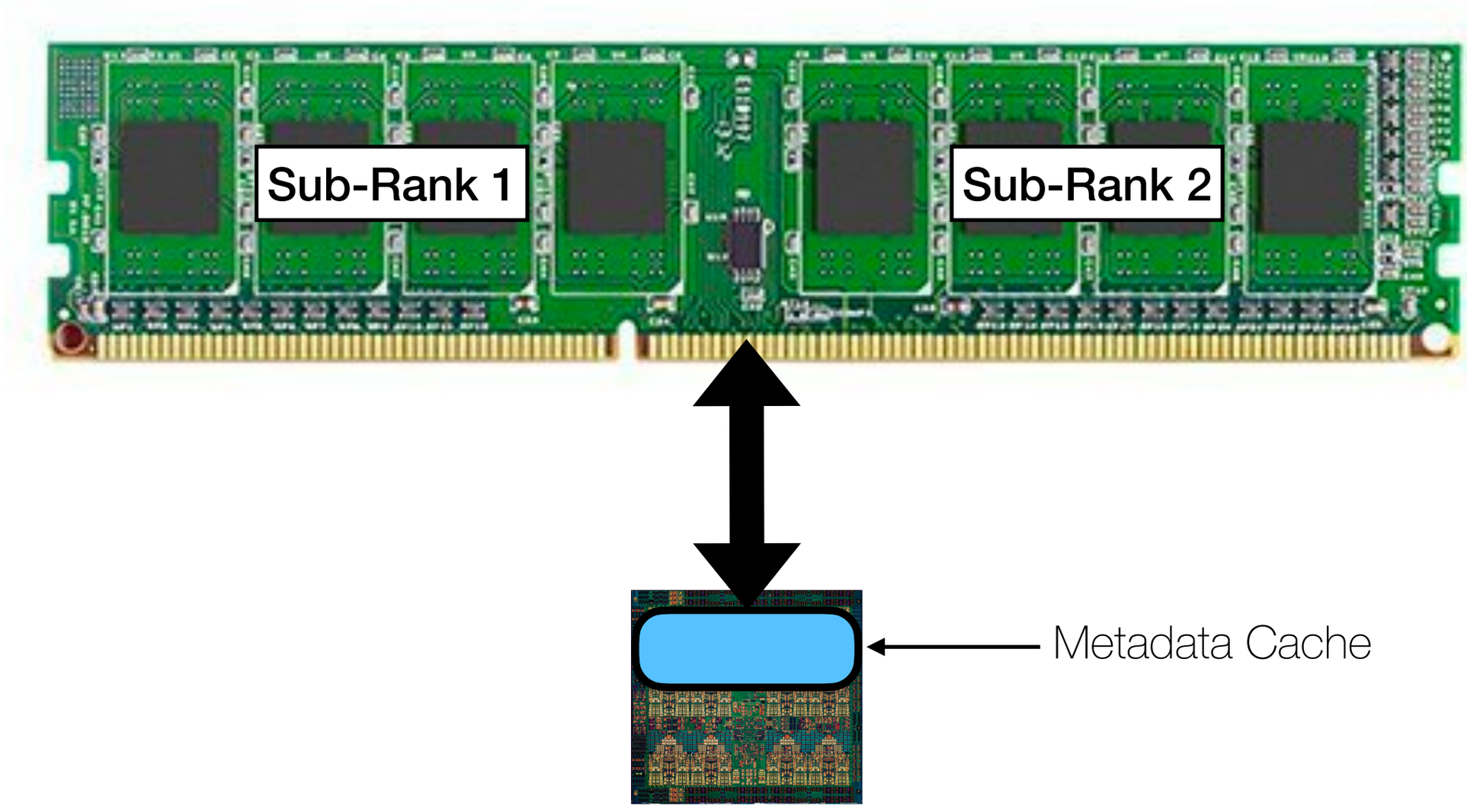
Sub-Ranking: Split the memory module into smaller channels



Sub-Ranking helps read and write smaller blocks of data → Improve bandwidth

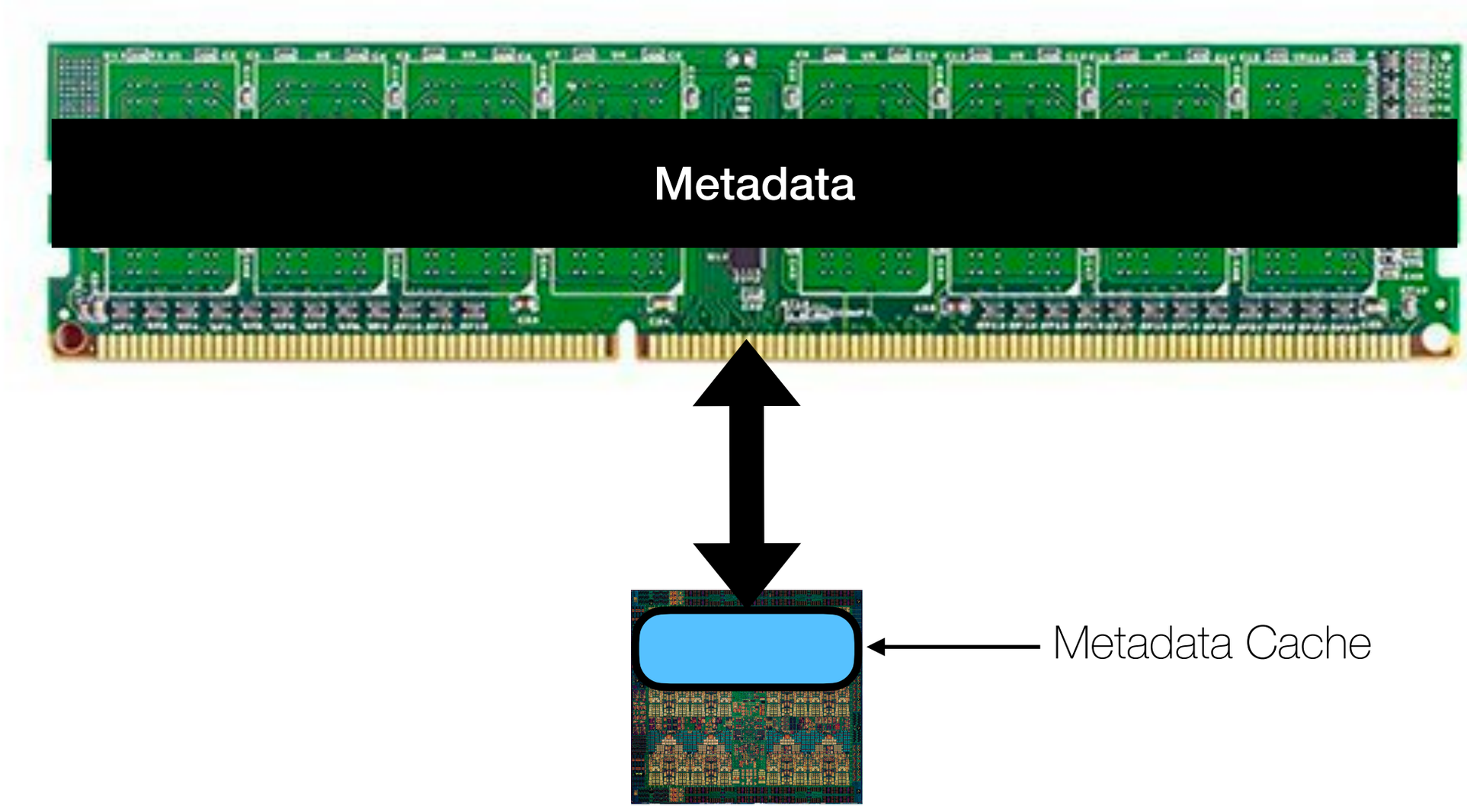
Background

Metadata Cache: Store metadata within a cache on the memory controller



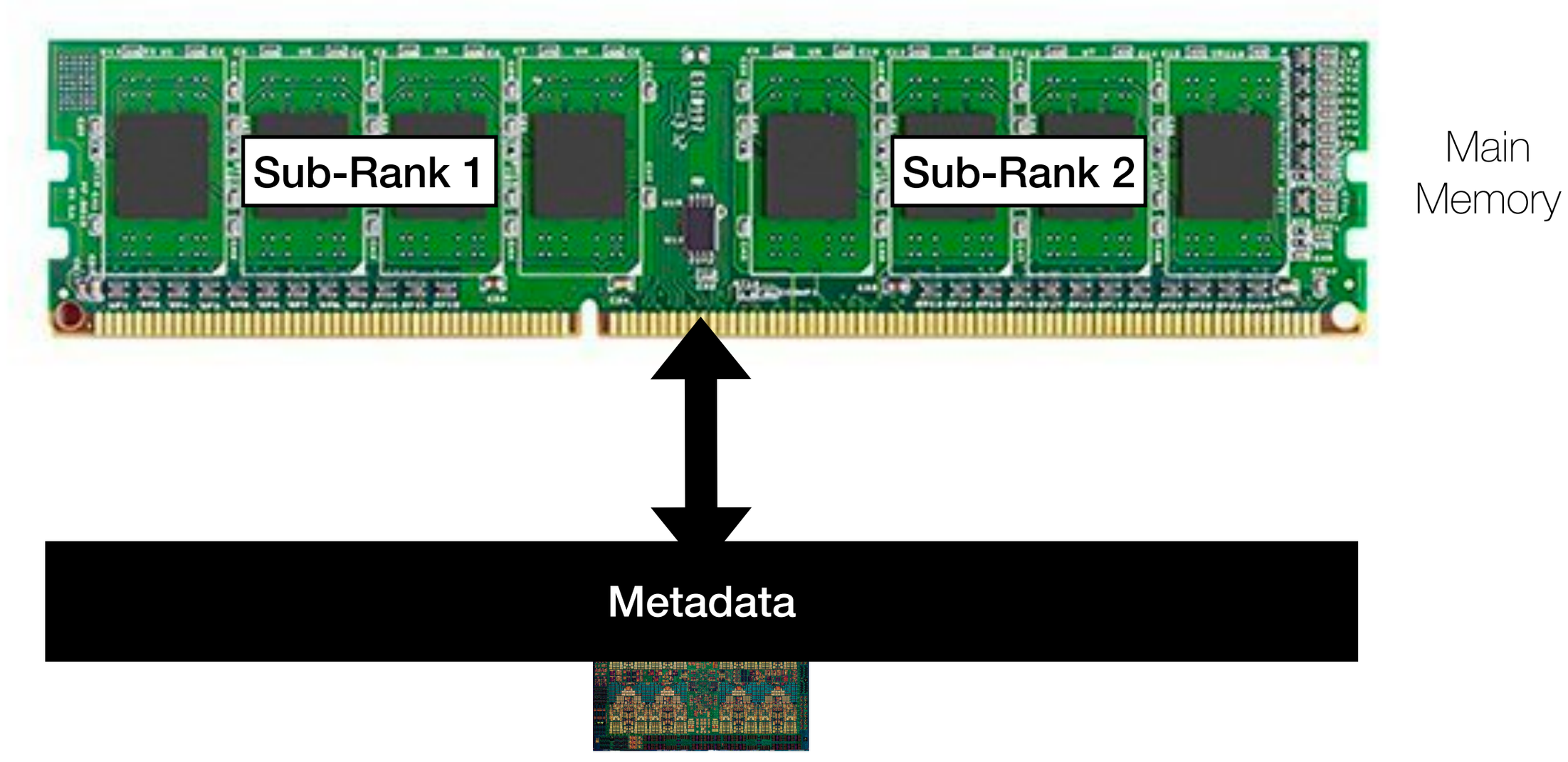
Background

Metadata Cache: Store metadata within a cache on the memory controller



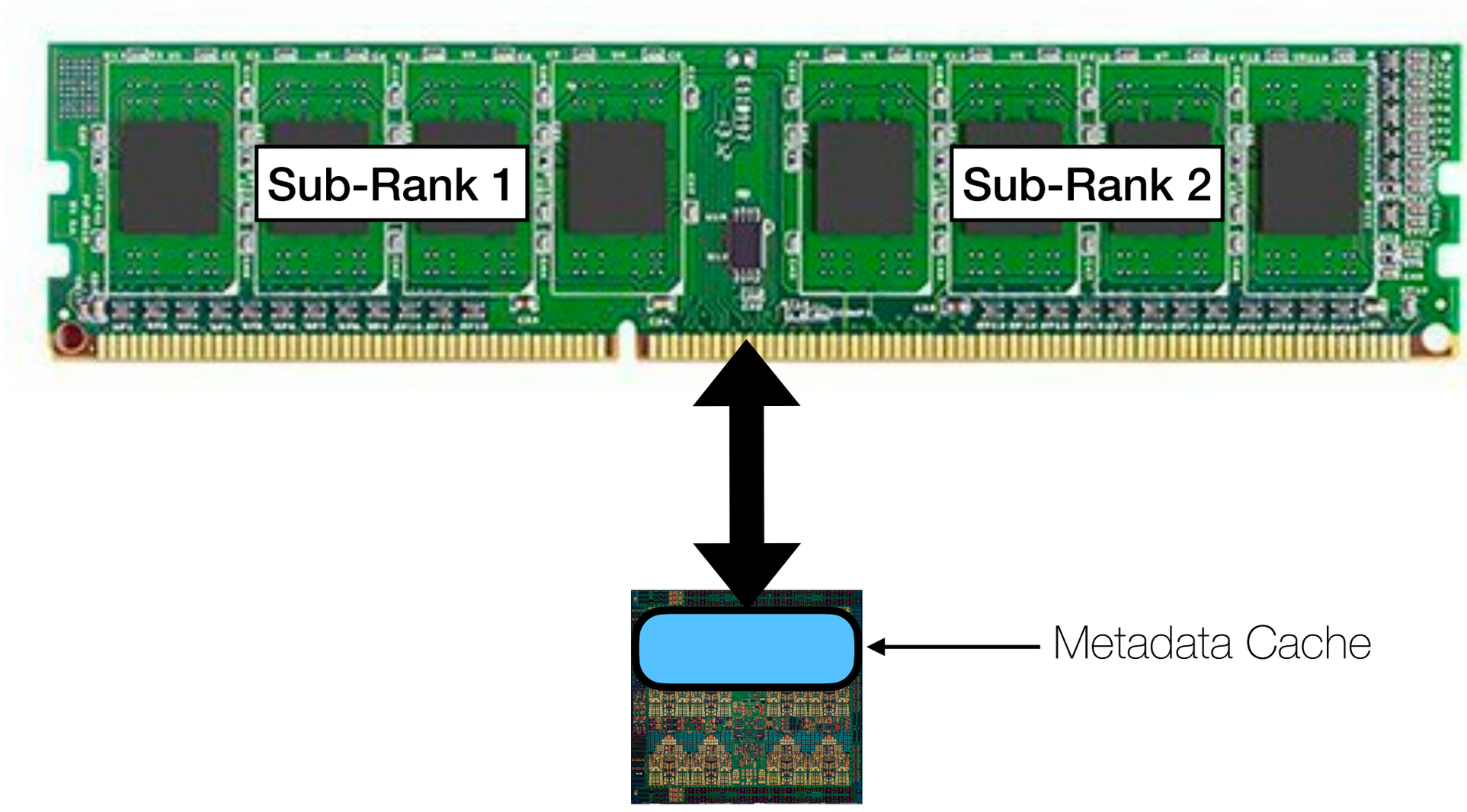
Background

Metadata Cache: Store metadata within a cache on the memory controller



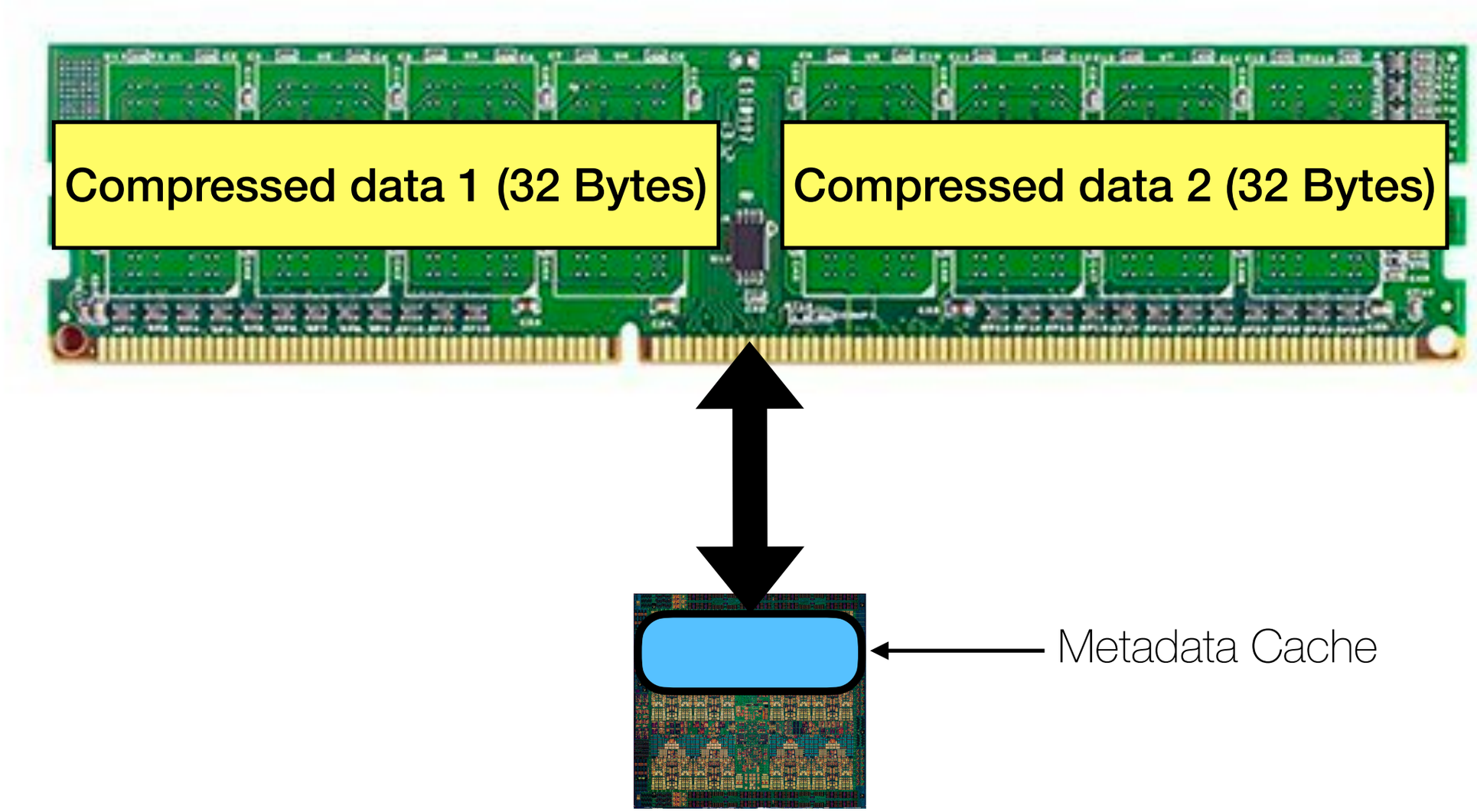
Background

Metadata Cache: Store metadata within a cache on the memory controller



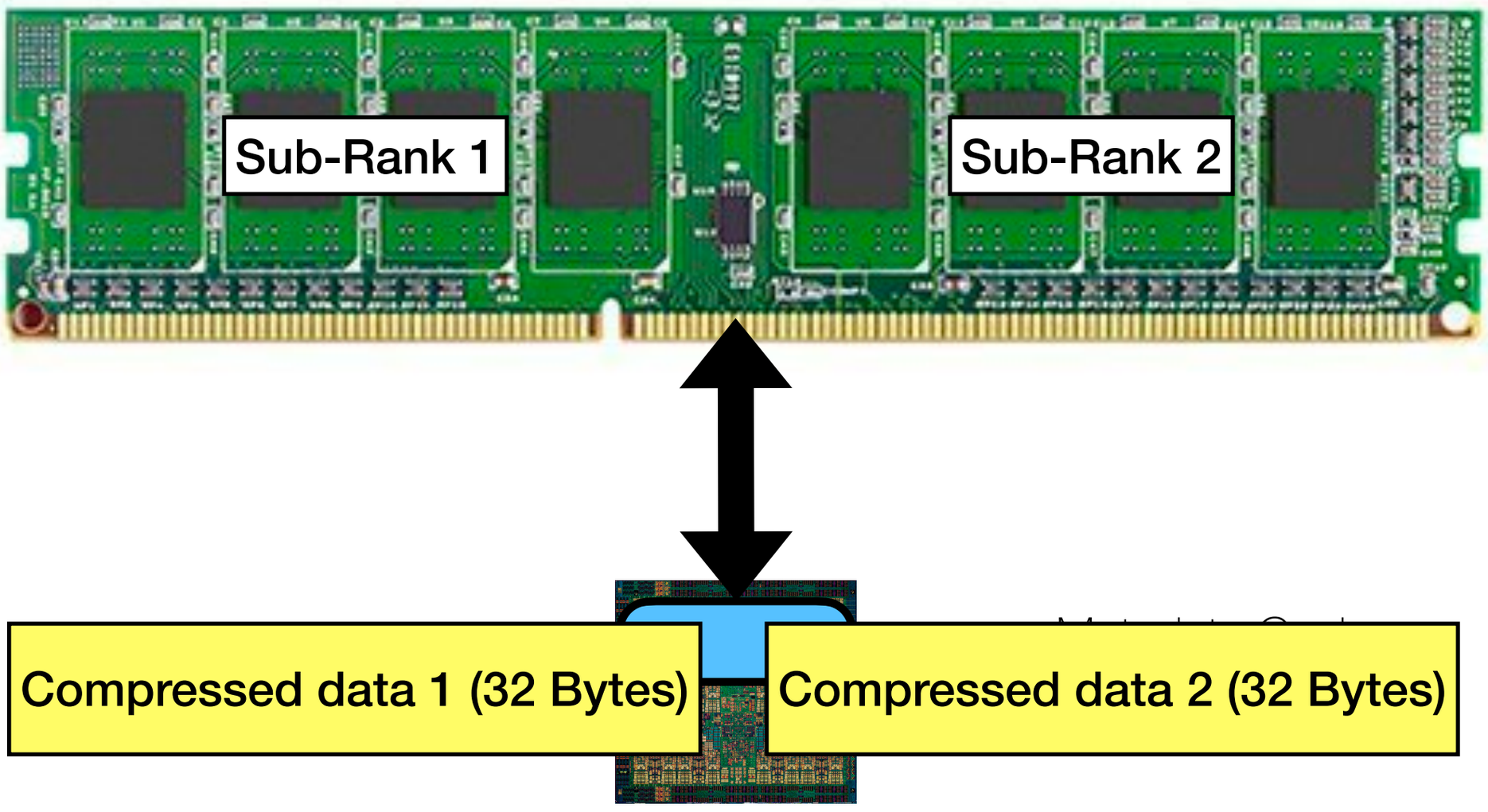
Background

Metadata Cache: Store metadata within a cache on the memory controller



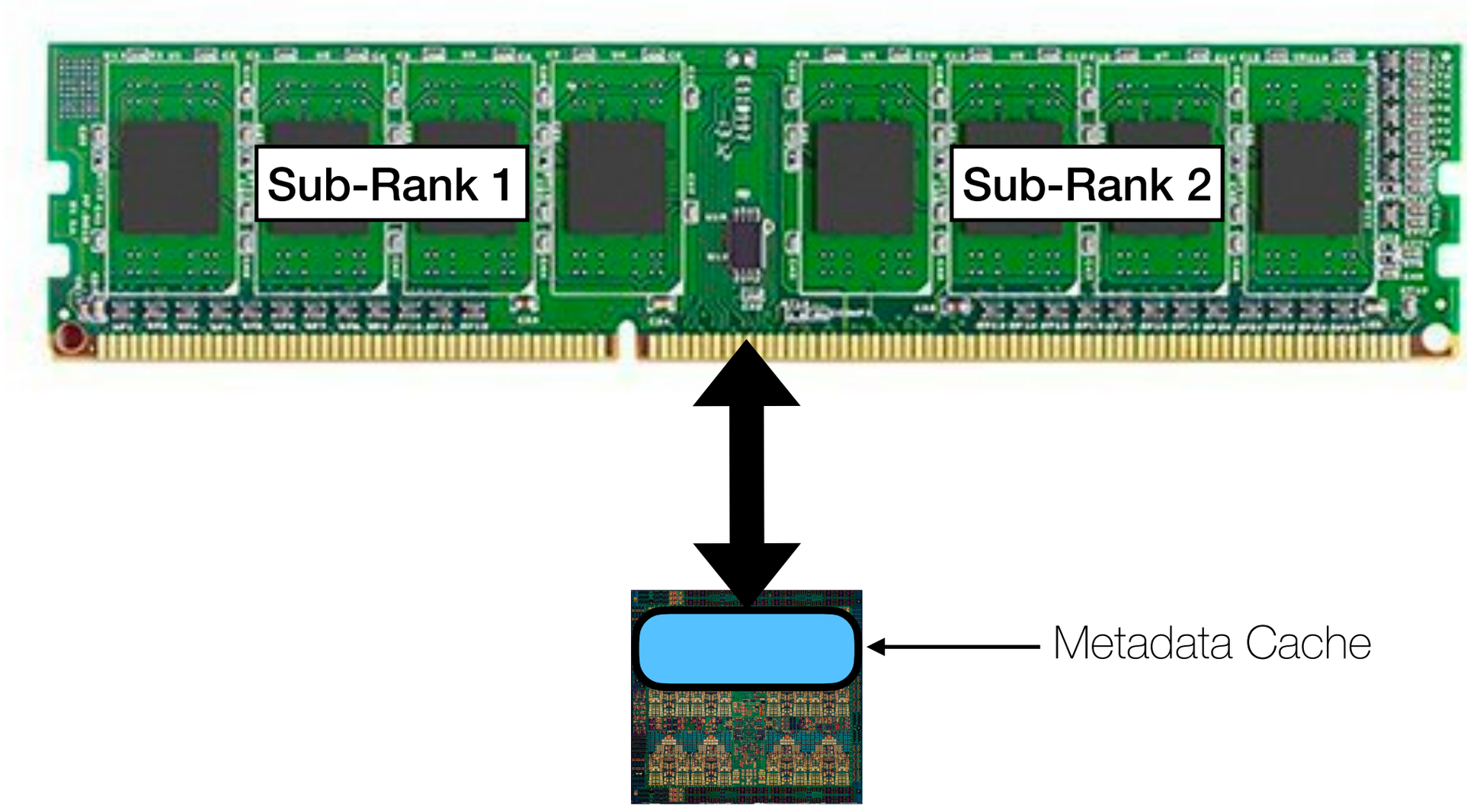
Background

Metadata Cache: Store metadata within a cache on the memory controller



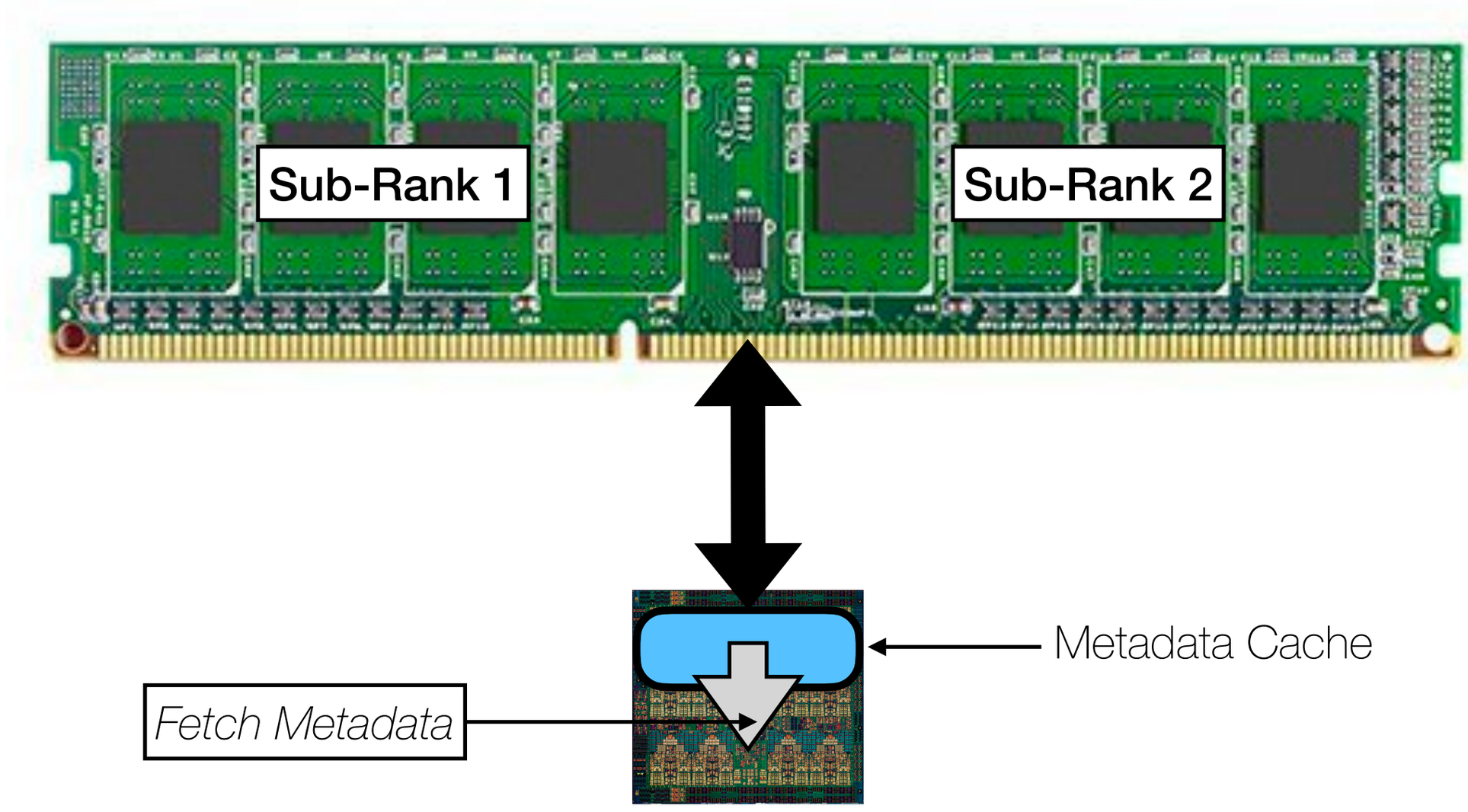
Background

Metadata Cache: Store metadata within a cache on the memory controller



Background

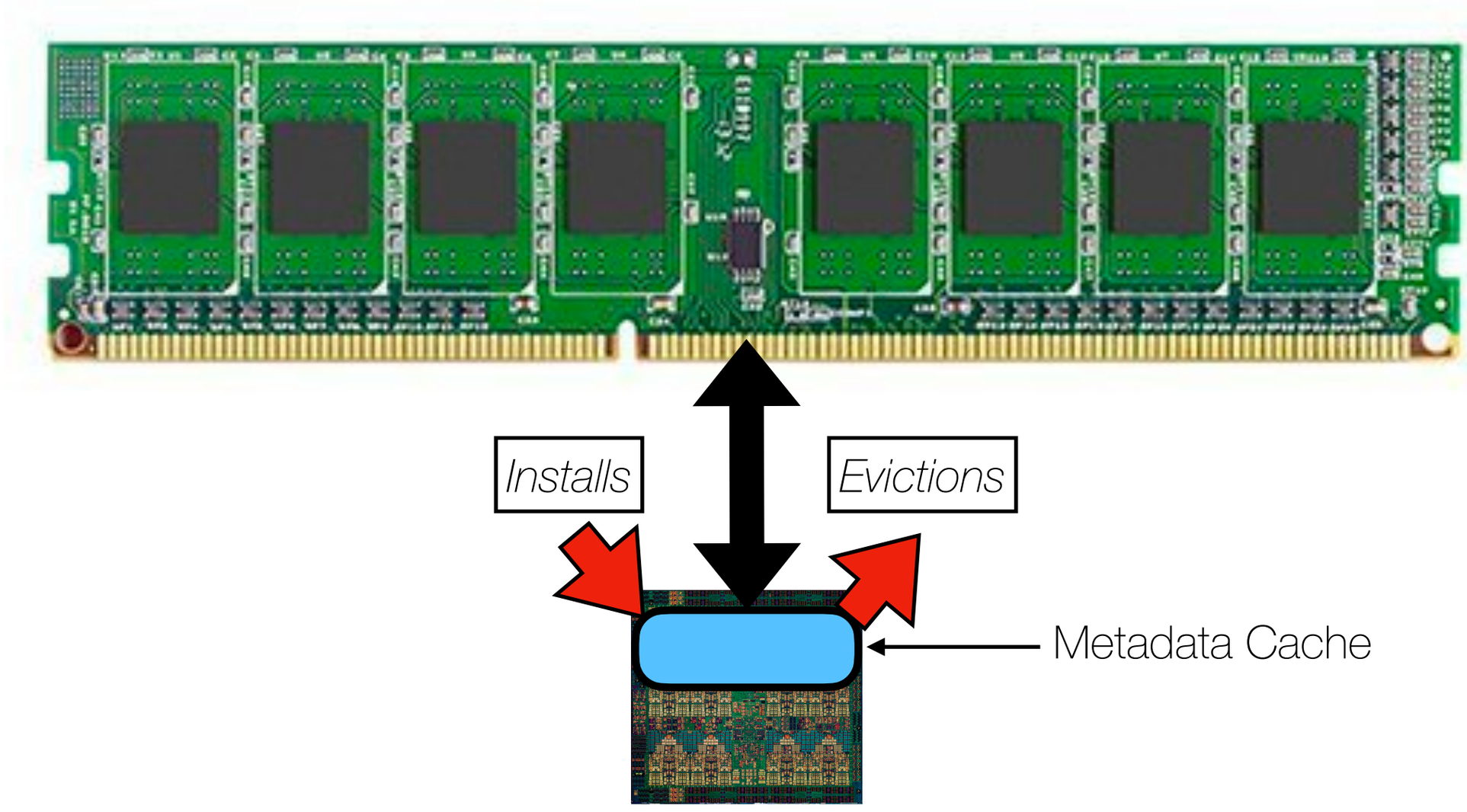
Metadata Cache: Store metadata within a cache on the memory controller



Reduce the bandwidth overheads of accessing Metadata

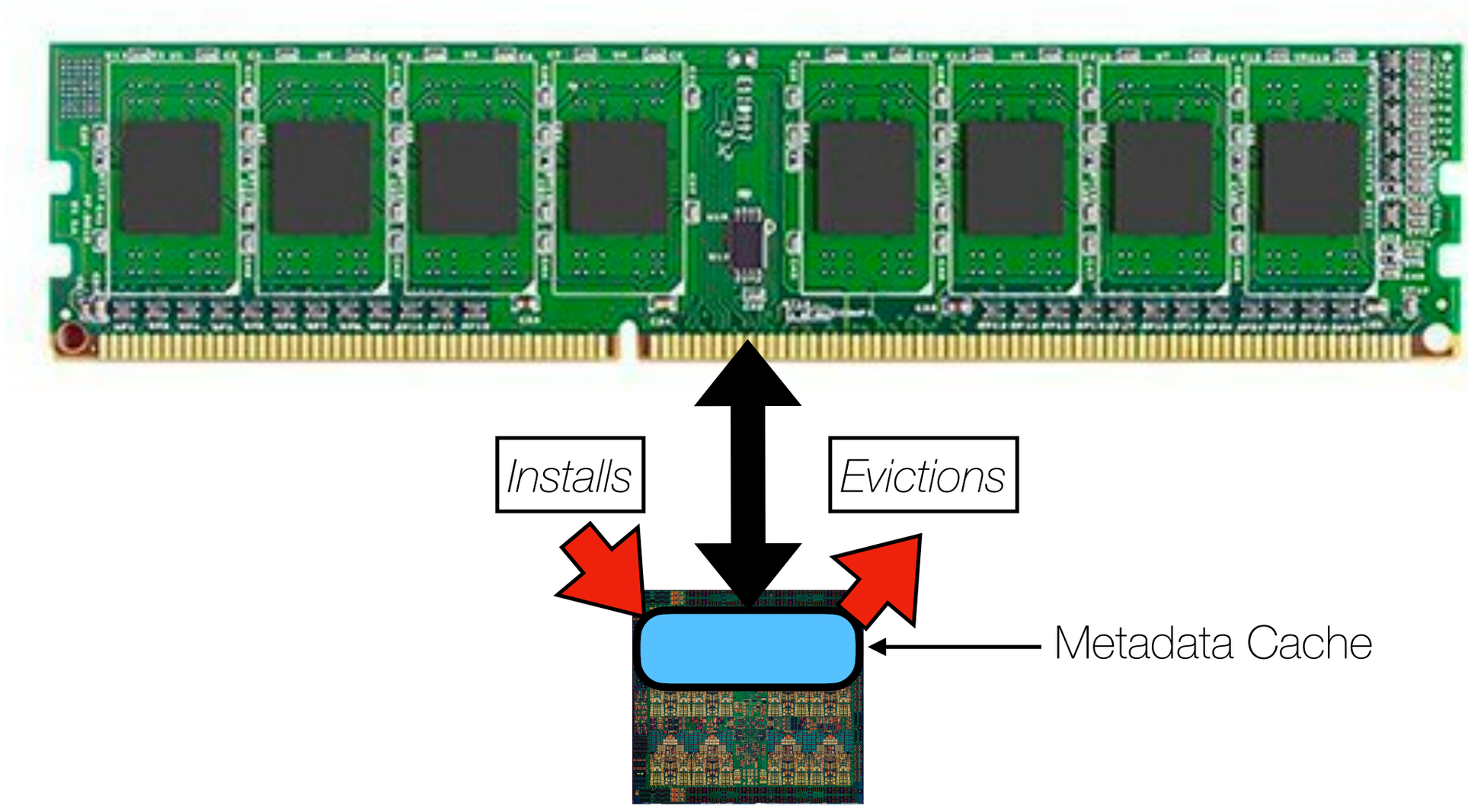
Background

Metadata Cache can be counter-productive



Background

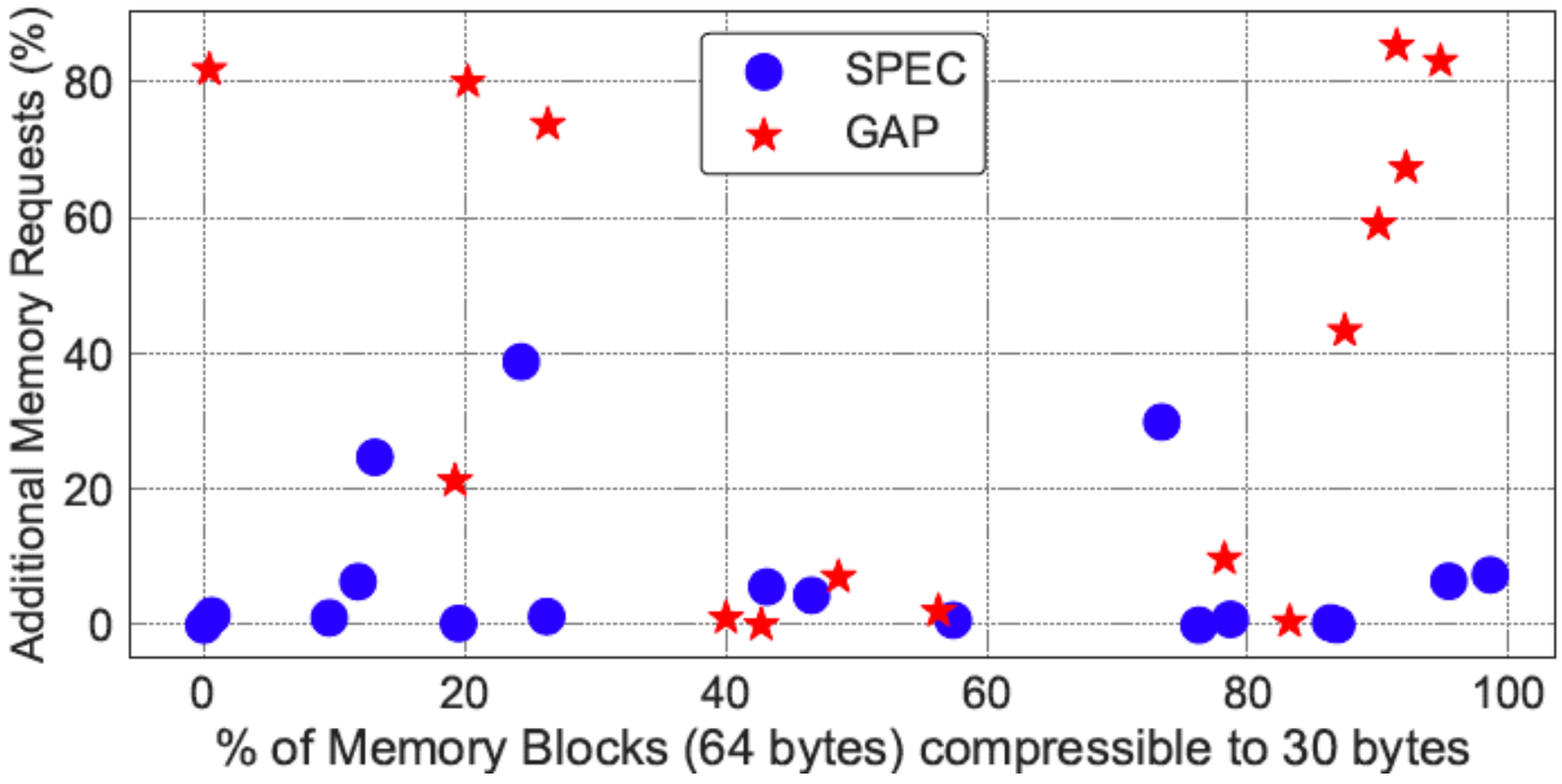
Metadata Cache can be counter-productive



Evictions and installs consume bandwidth and reduce the benefits of theMetadata Cache

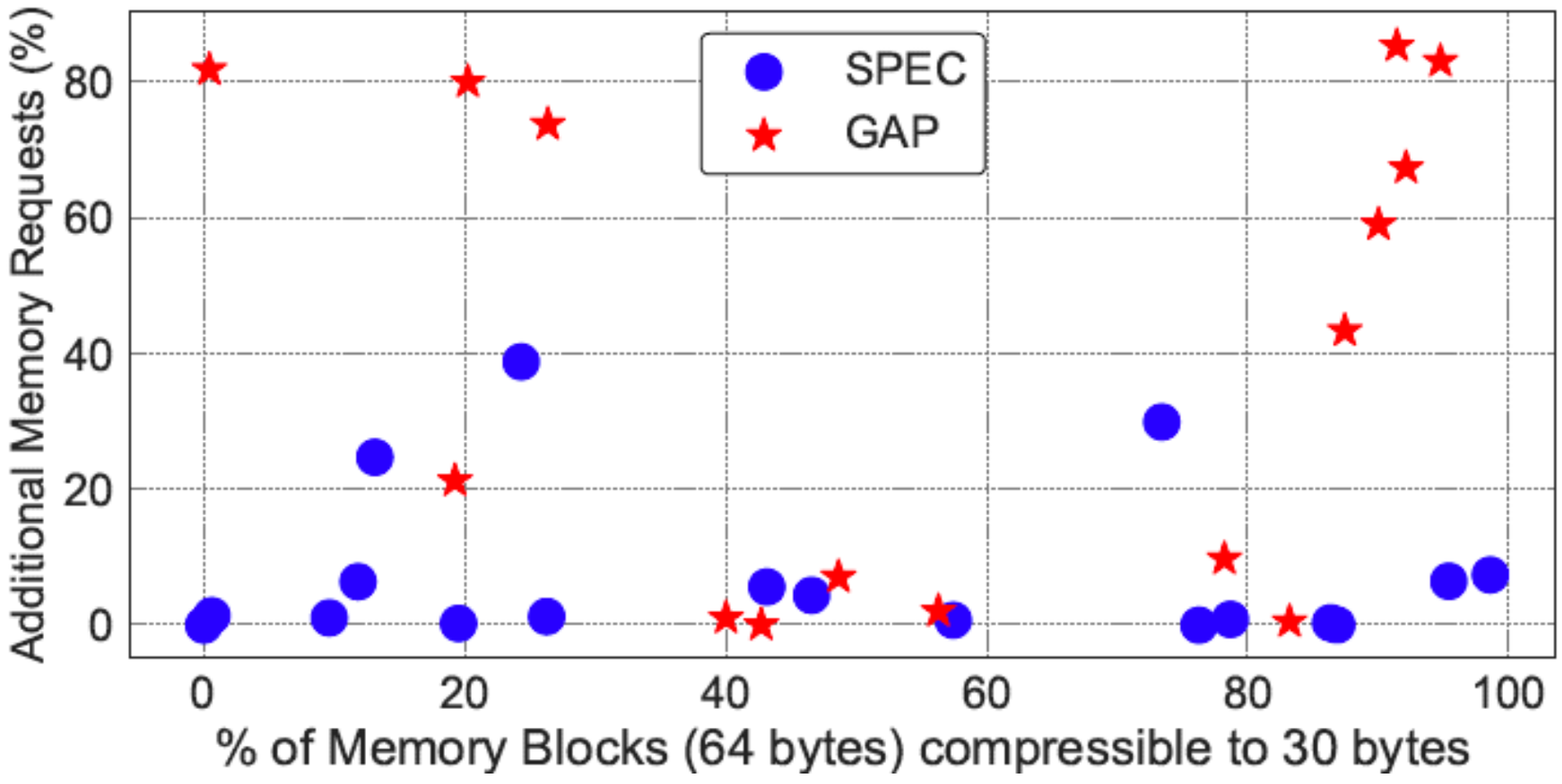
Motivation

Analysis with a 1MB Metadata Cache



Motivation

Analysis with a 1MB Metadata Cache



Upto 85% additional metadata traffic across a wide variety of workloads

- Introduction
- Background and Motivation
- Goal
- Attaché
  - *Blended Metadata*
  - *Compressibility Predictor*
- Results
- Summary

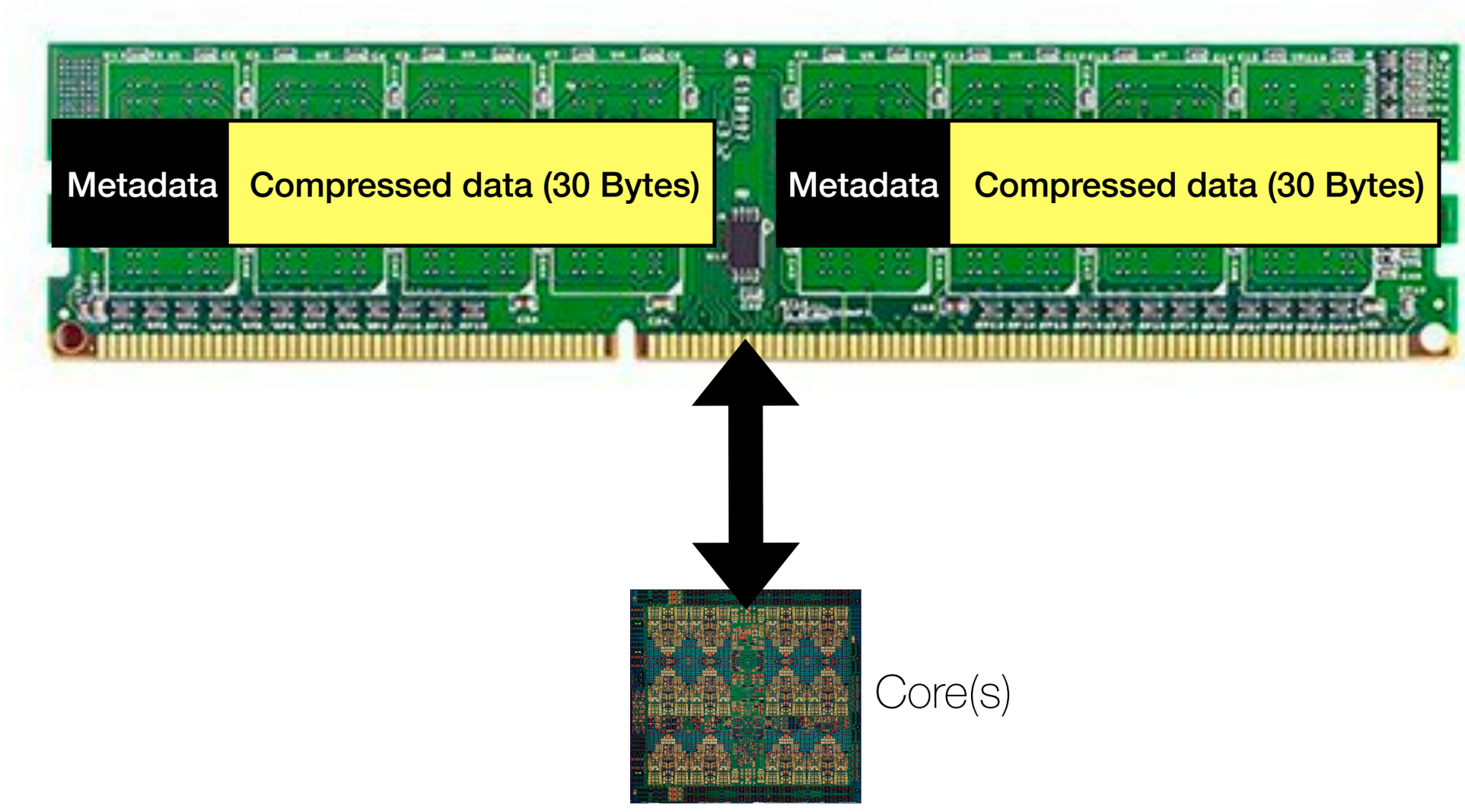
Goal

Eliminate almost **all additional metadata accesses** to get **near-ideal benefits** from data compression

- ✦ Introduction
- ✦ Background and Motivation
- ✦ Goal
- ✦ Attaché
  - *Blended Metadata*
  - *Compressibility Predictor*
- ✦ Results
- ✦ Summary

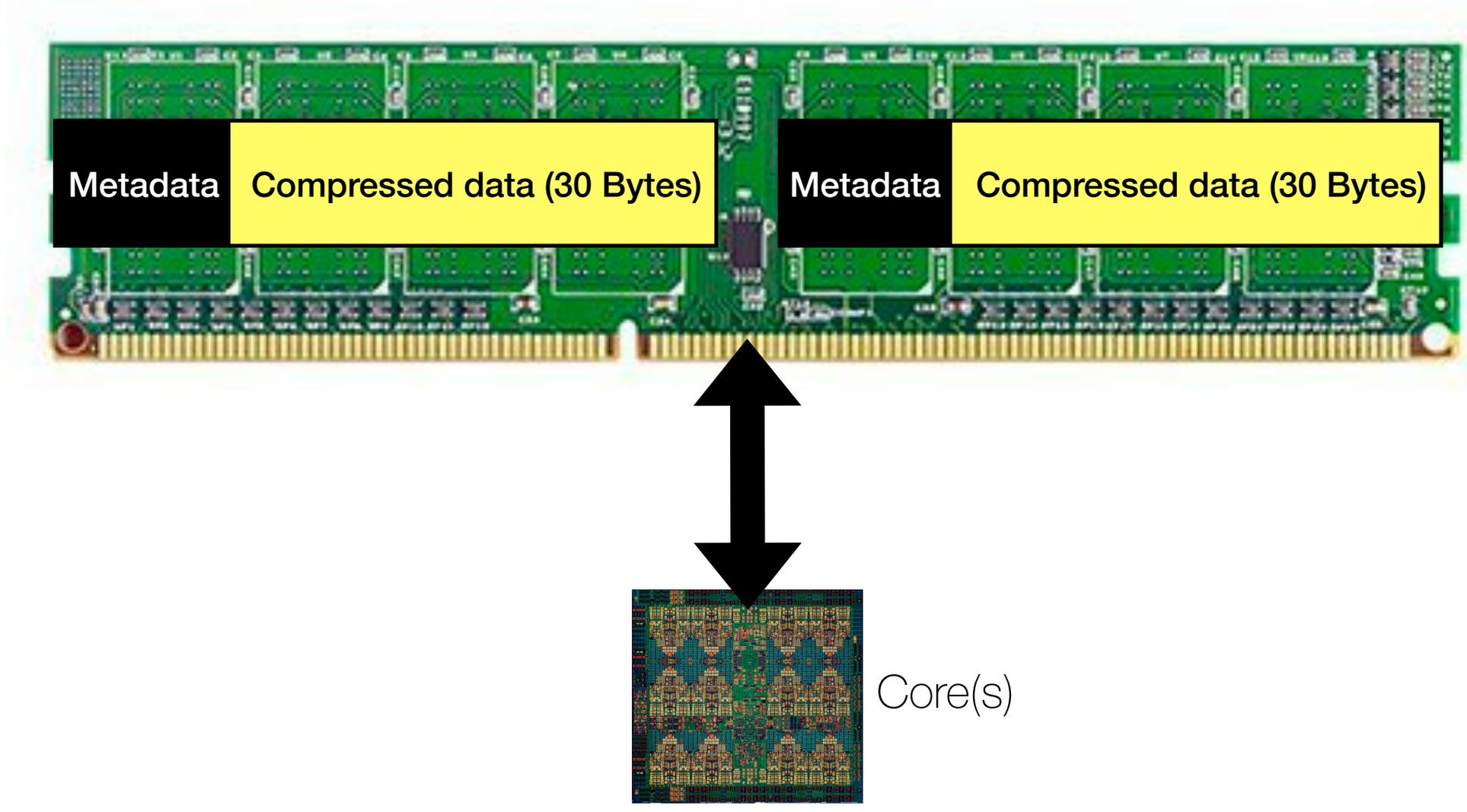
Attaché

Blended Metadata: Place metadata ahead of compressed data



Attaché

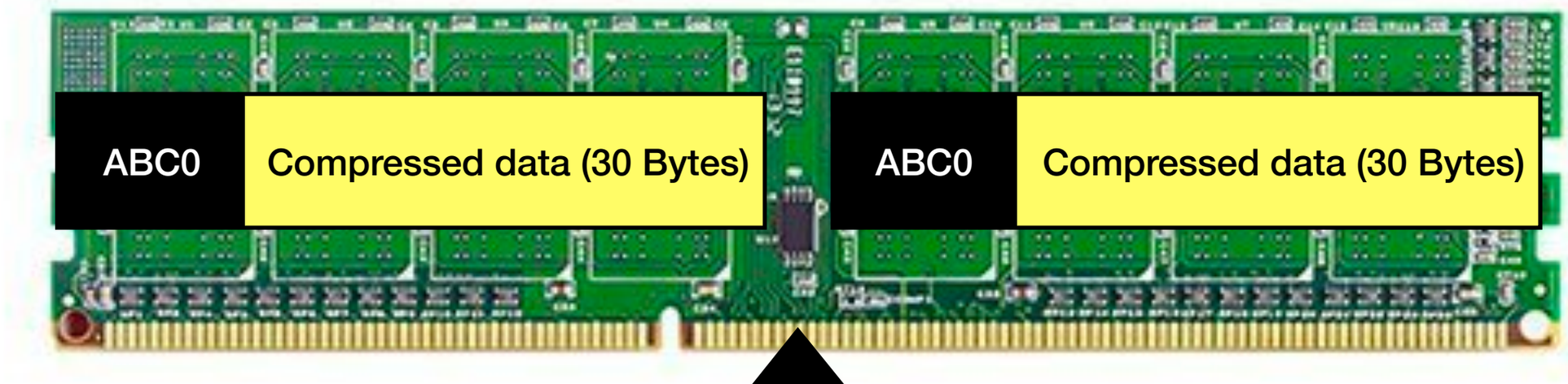
Blended Metadata: Place metadata ahead of compressed data



Metadata can be accessed with data: No additional bandwidth overheads

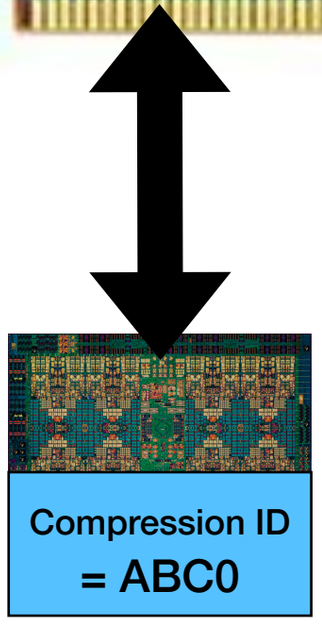
Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$



Main Memory

Metadata matches?



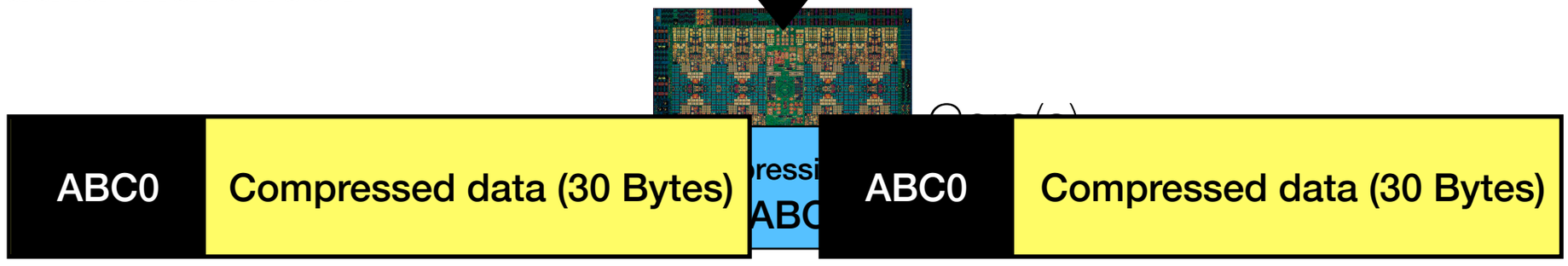
Core(s)

Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$

Main Memory

Metadata matches?

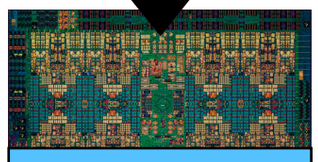


Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$

Main Memory

Metadata matches?



Core(s)

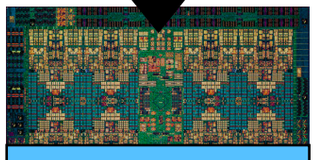
Compression ID
= ABC0

Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$

Main Memory

Metadata matches? ✓



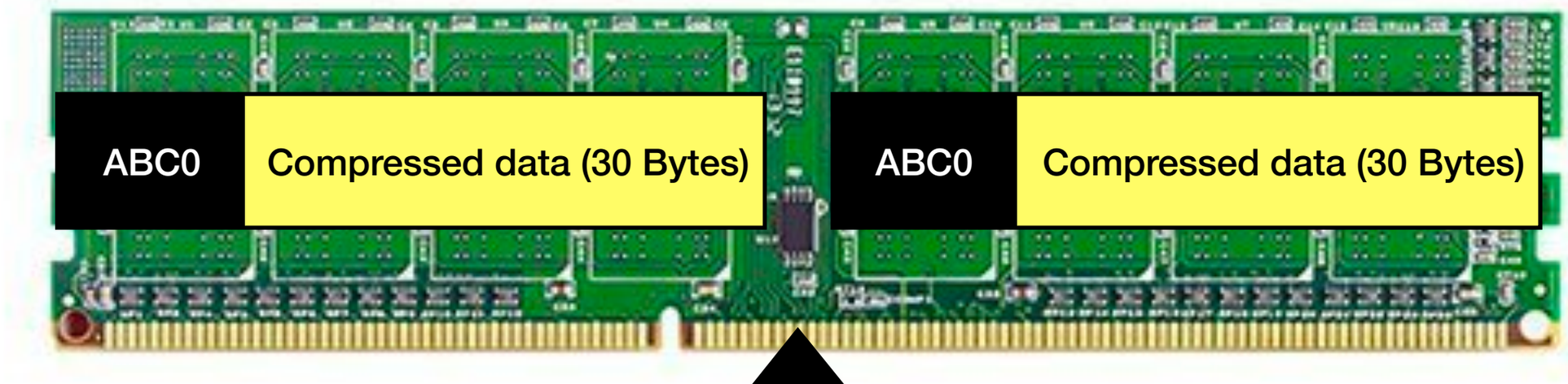
Core(s)

Compression ID
= ABC0

If the Compression ID matches at the memory controller = Compressed Line

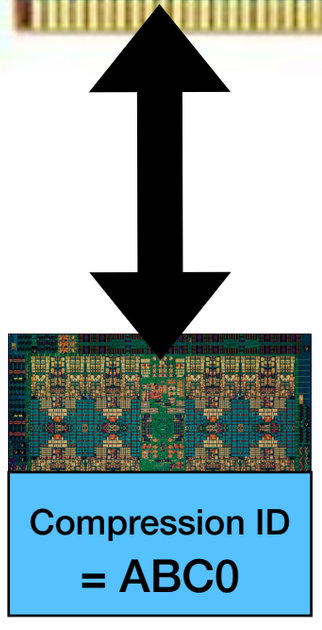
Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$



Main Memory

Metadata matches?



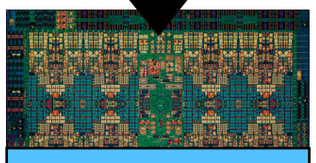
Core(s)

Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$

Main Memory

Metadata matches? ✓

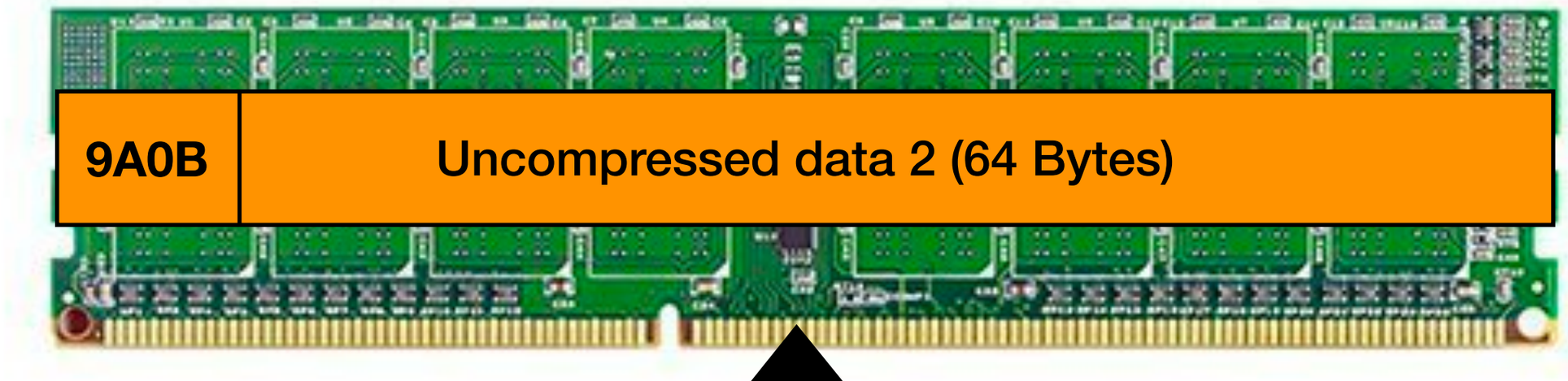


Core(s)

Compression ID = ABC0

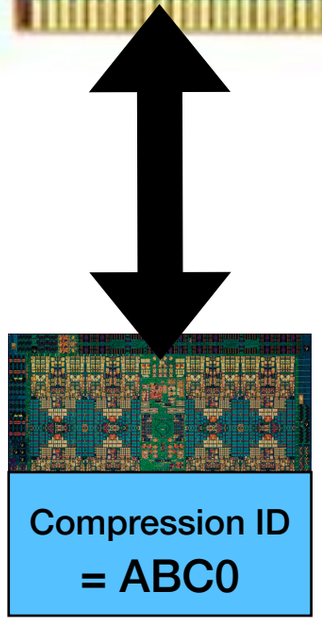
Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$



Main Memory

Metadata matches? ✗



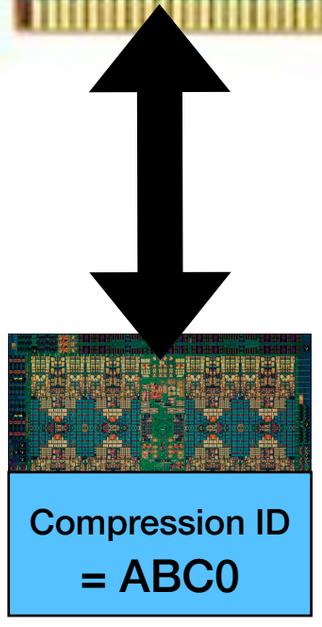
Core(s)

Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$

Main Memory

Metadata matches? ✗

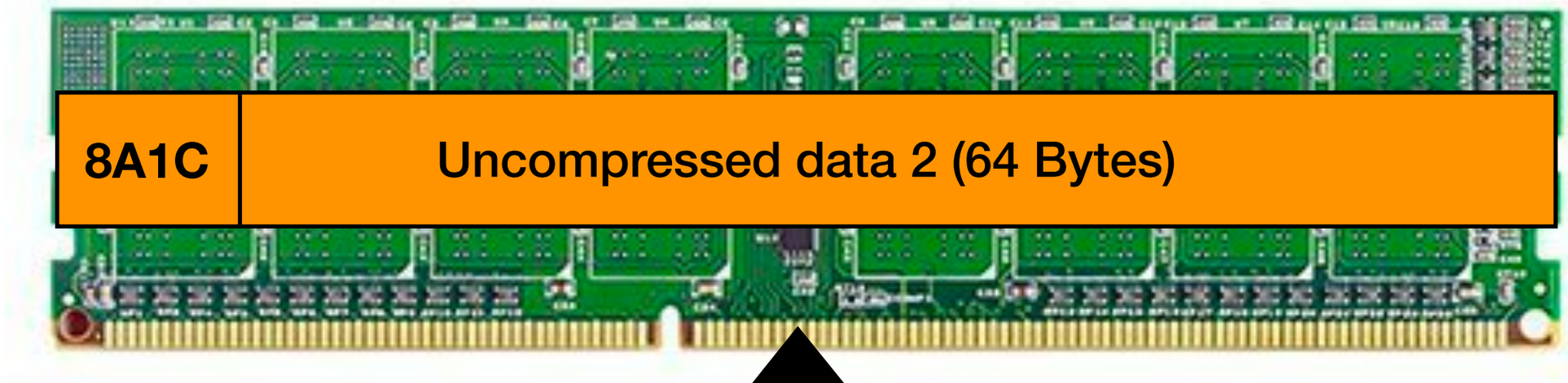


Core(s)

If the Compression ID does not match at the memory controller = Uncompressed Line

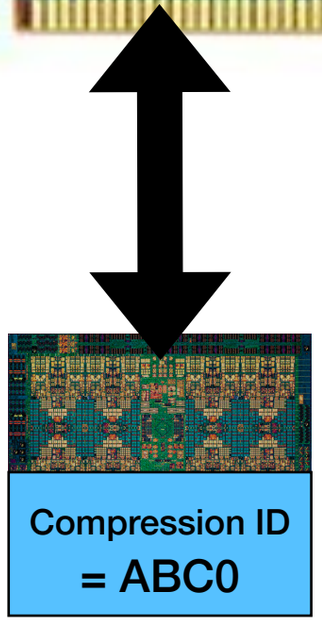
Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$



Main Memory

Metadata matches? ✗



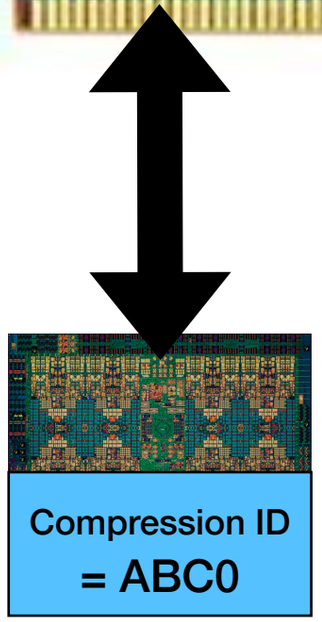
Core(s)

Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$

Main Memory

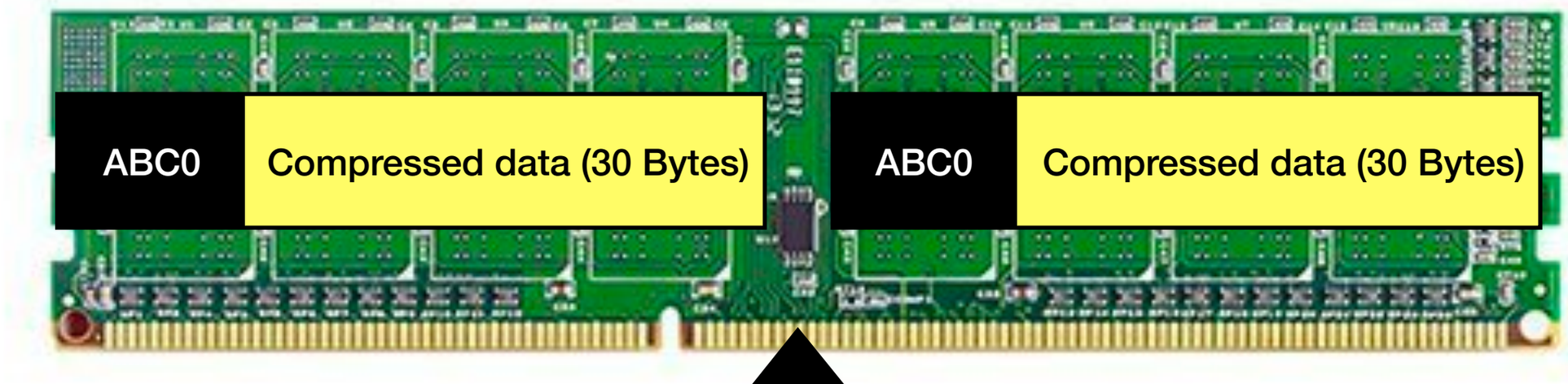
Metadata matches? ✗



Core(s)

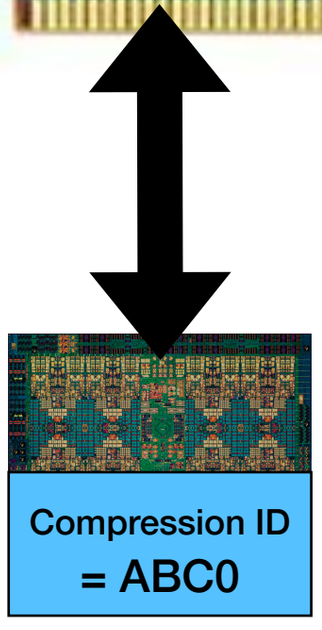
Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$



Main Memory

Metadata matches?



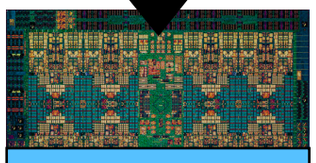
Core(s)

Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$

Main Memory

Metadata matches? ✓

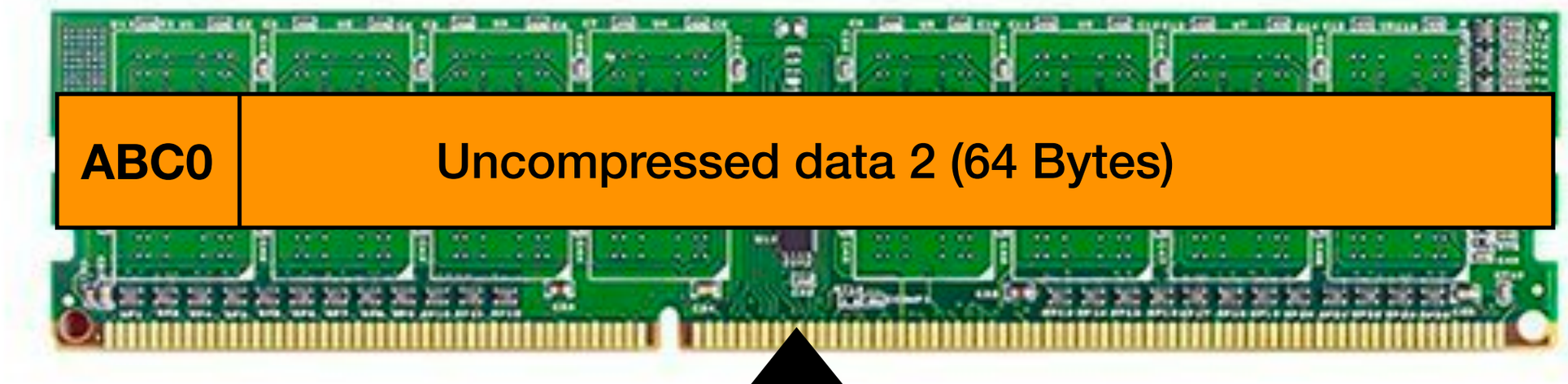


Core(s)

Compression ID = ABC0

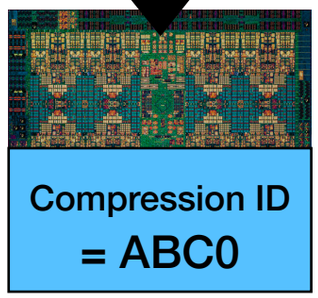
Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$



Main Memory

Metadata matches?



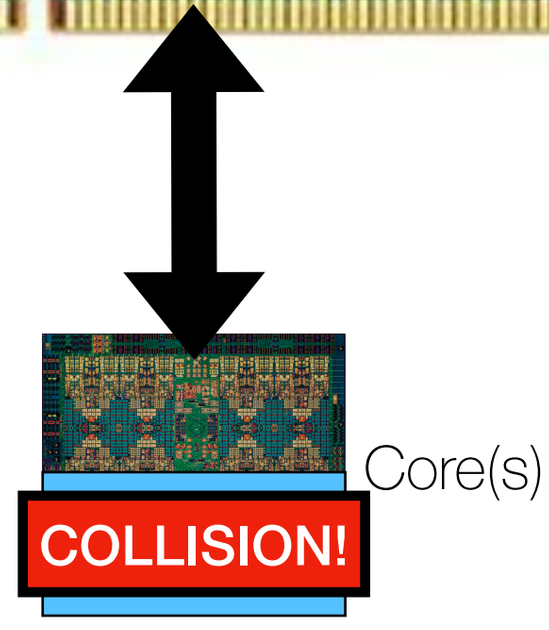
Core(s)

Attaché: Example

Metadata = Compressed ID = $[ABC0]_H$

Main Memory

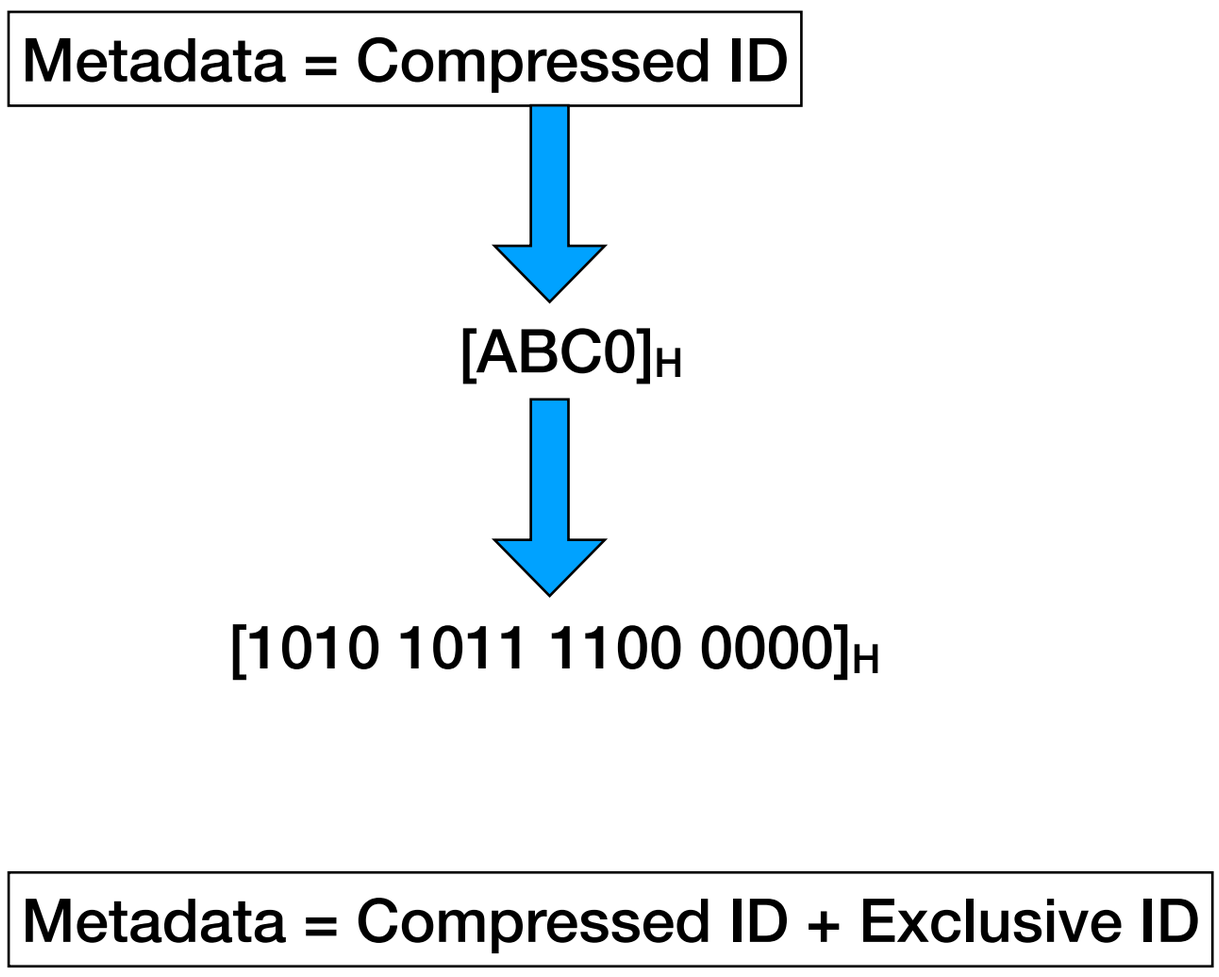
Metadata matches? ✓



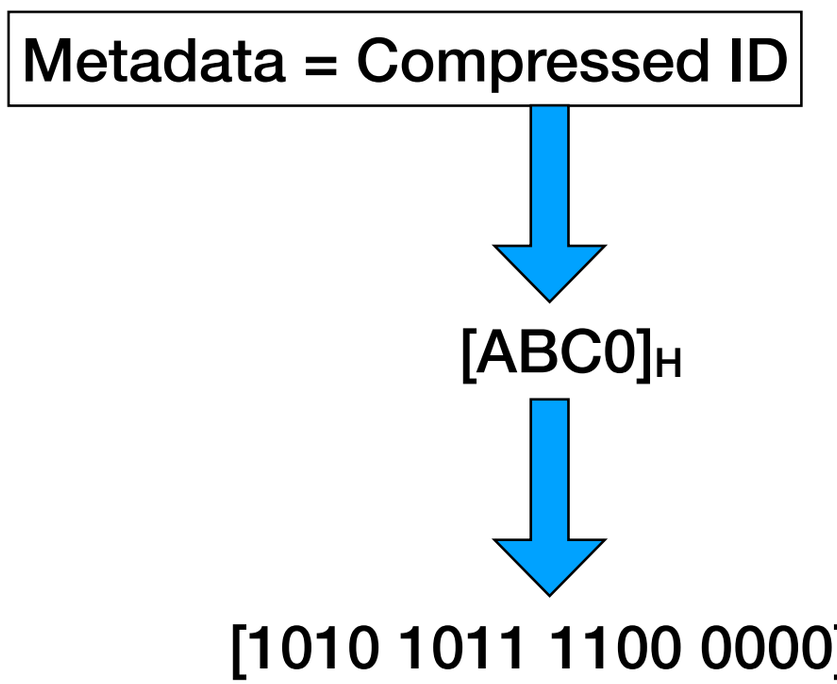
Core(s)

It is possible for the Compressed ID to collide for uncompressed lines

Attaché: Detect Collisions



Attaché: Detect Collisions

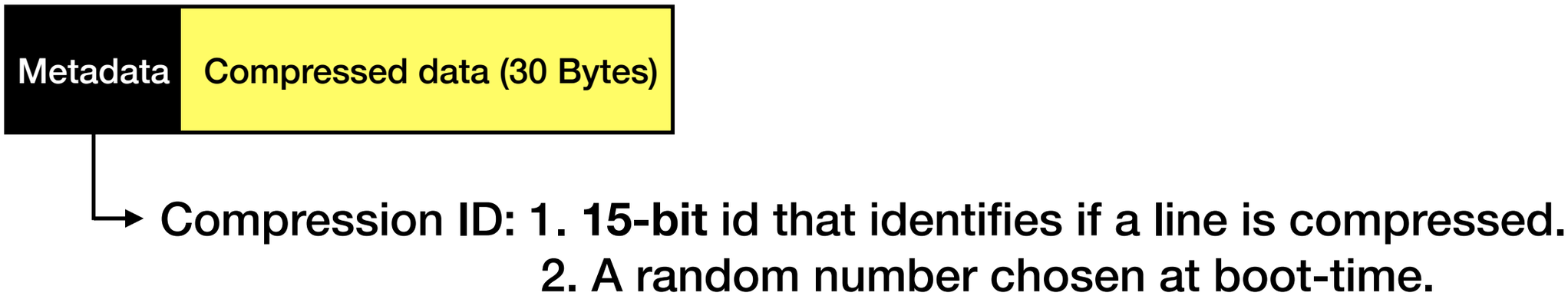


Metadata = Compressed ID + Exclusive ID

It is possible for the Compressed ID to collide for uncompressed lines

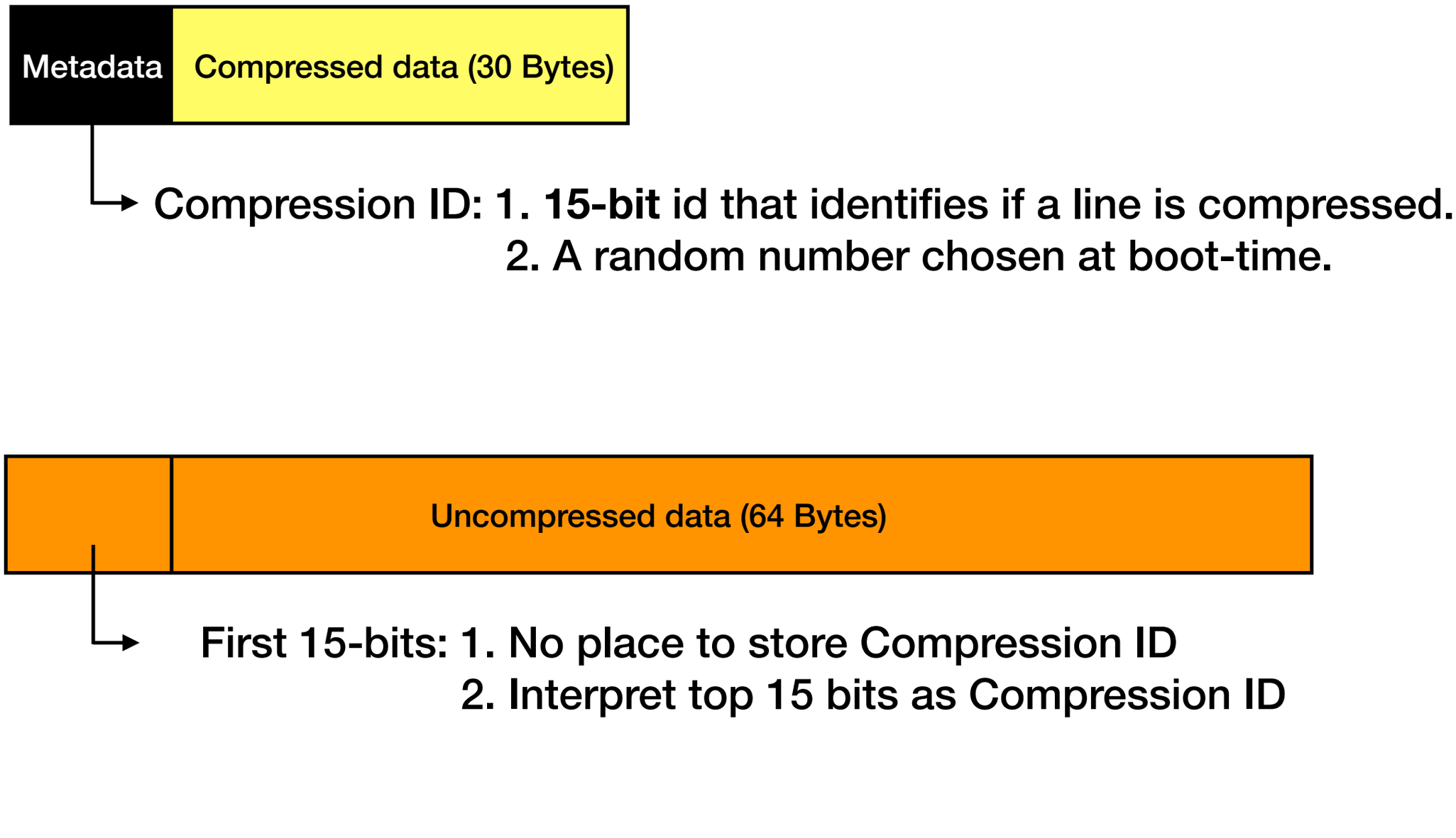
Attaché

Blended Metadata: Place metadata ahead of compressed data



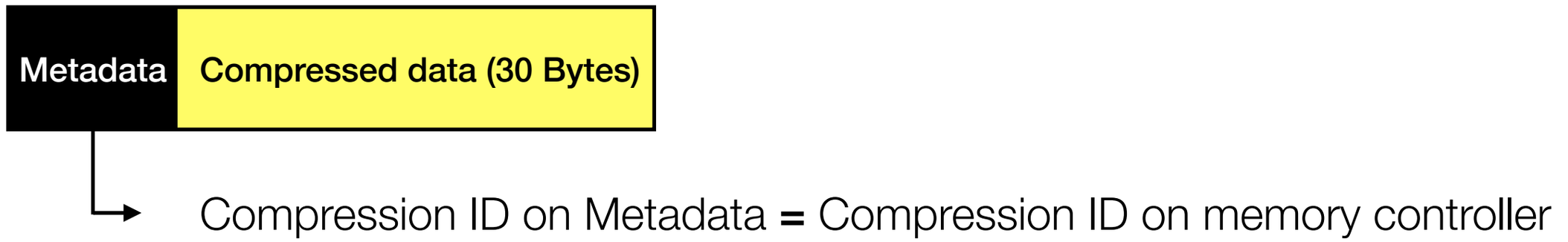
Attaché

Blended Metadata: Place metadata ahead of compressed data



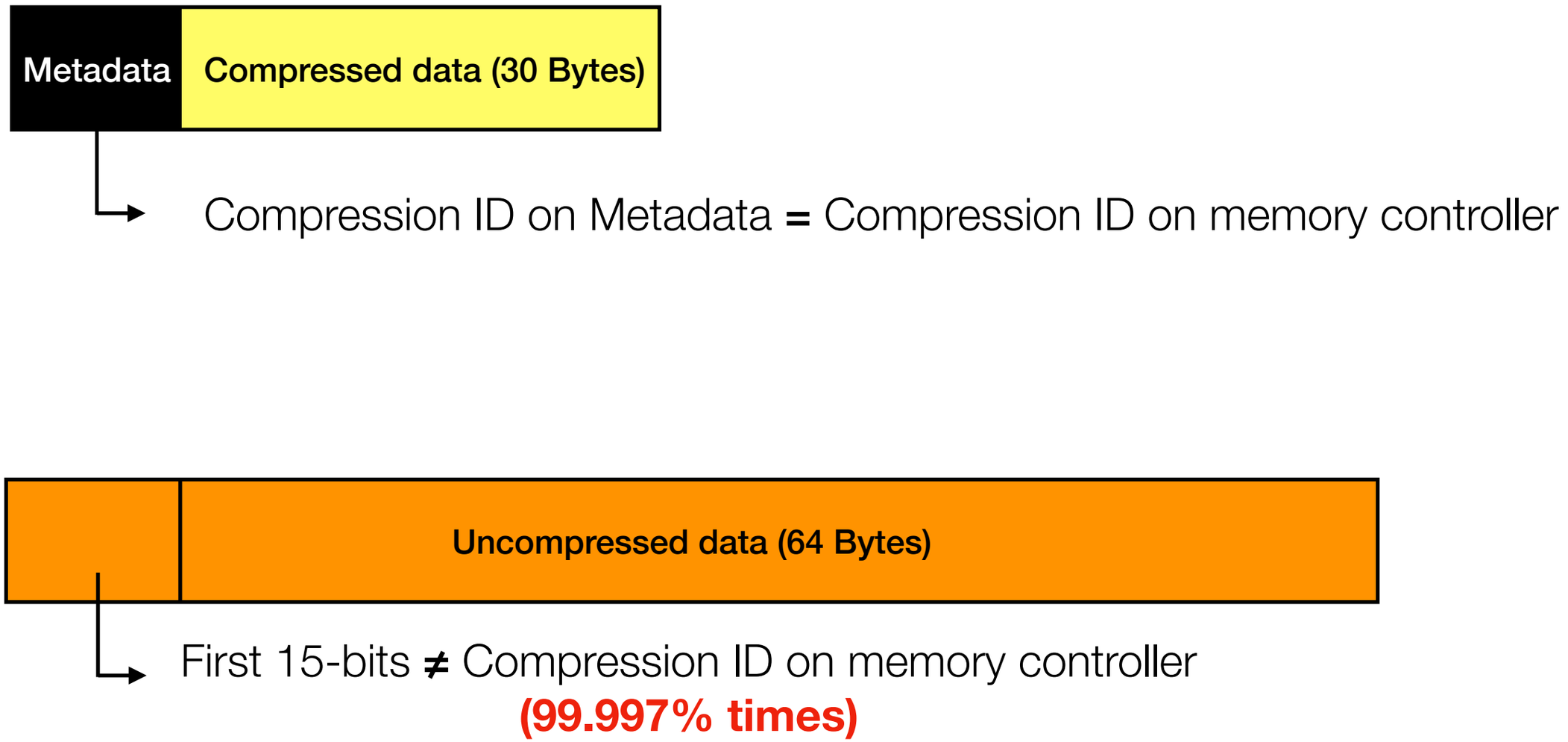
Attaché: Concerns

Blended Metadata: *Collision*



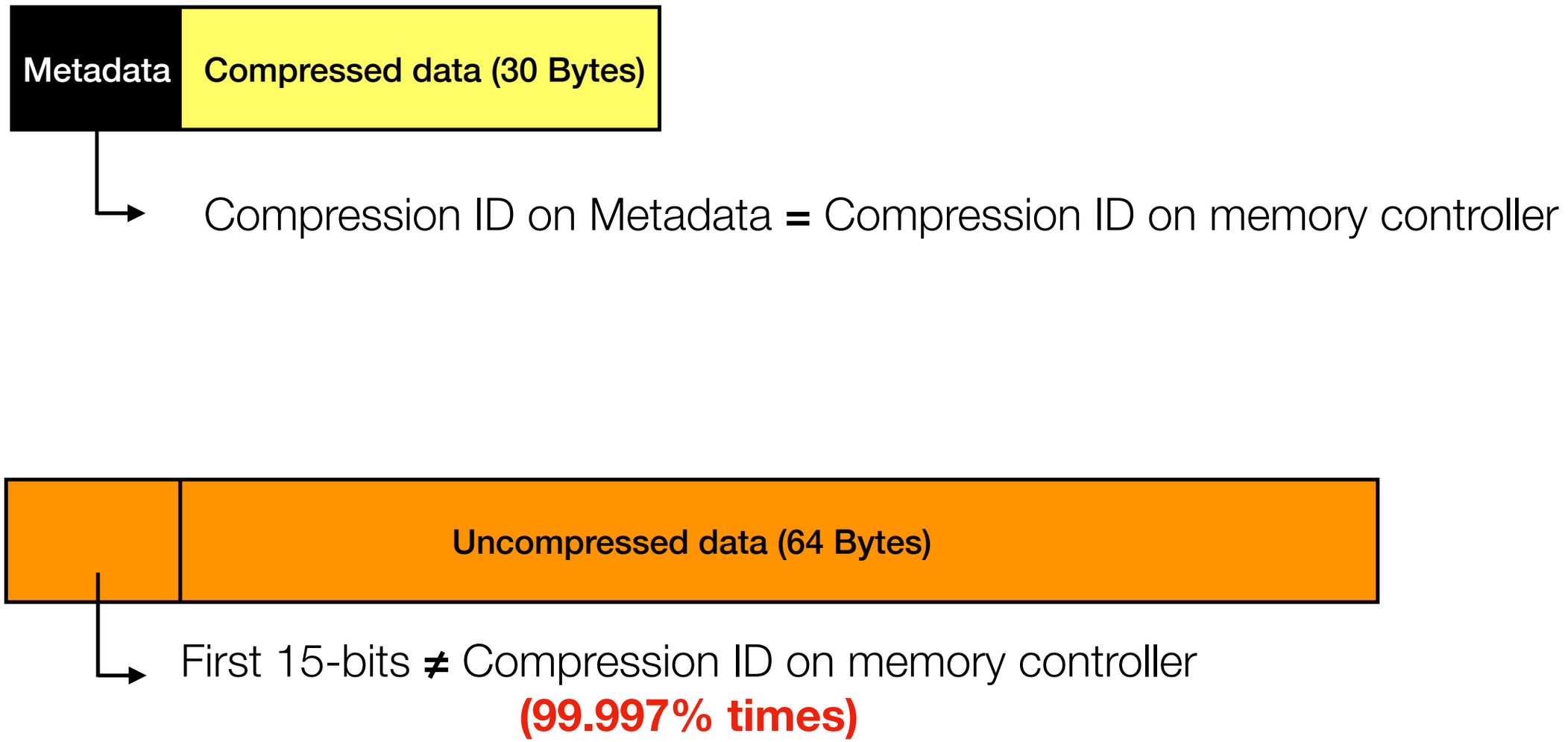
Attaché: Concerns

Blended Metadata: *Collision*



Attaché: Concerns

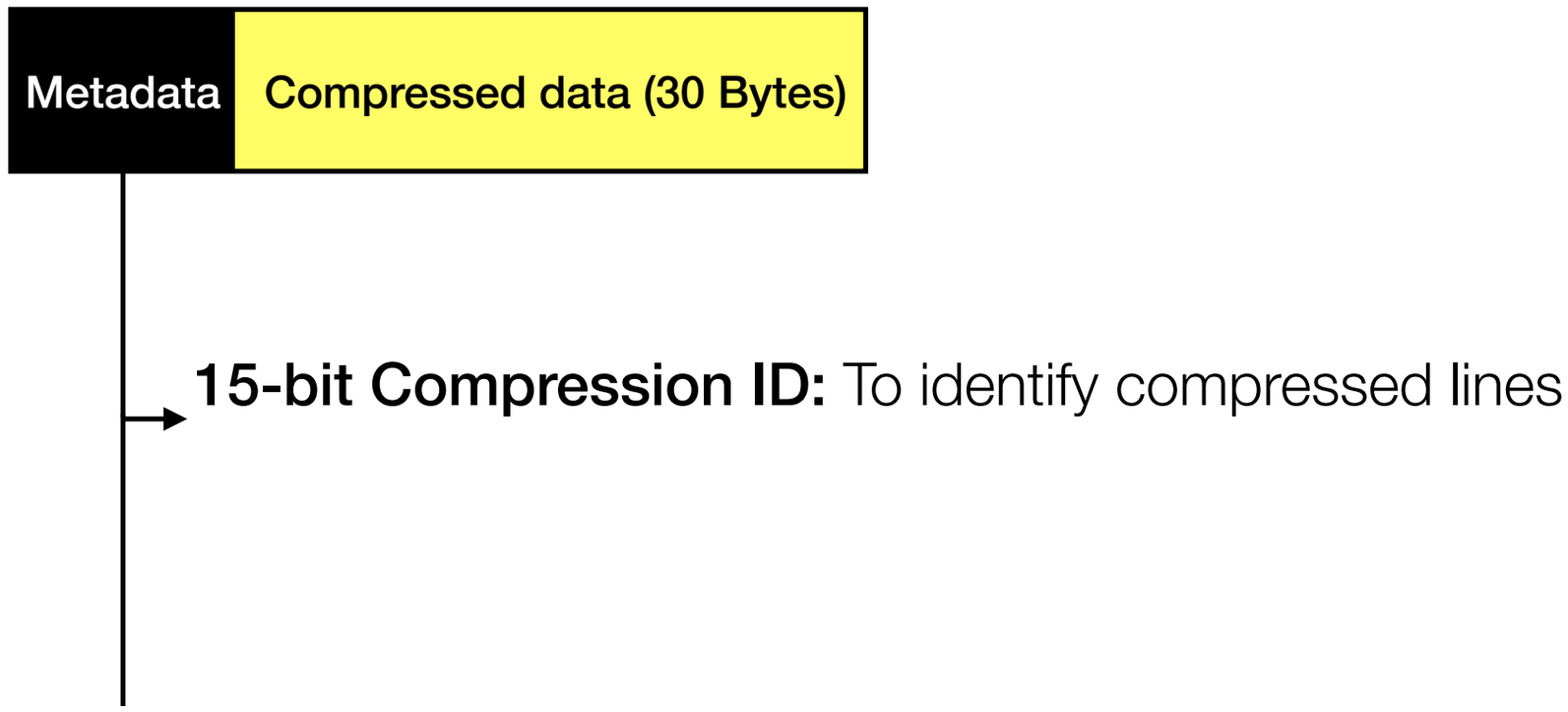
Blended Metadata: *Collision*



There is a chance (0.03%) that Uncompressed data is misinterpreted as compressed data

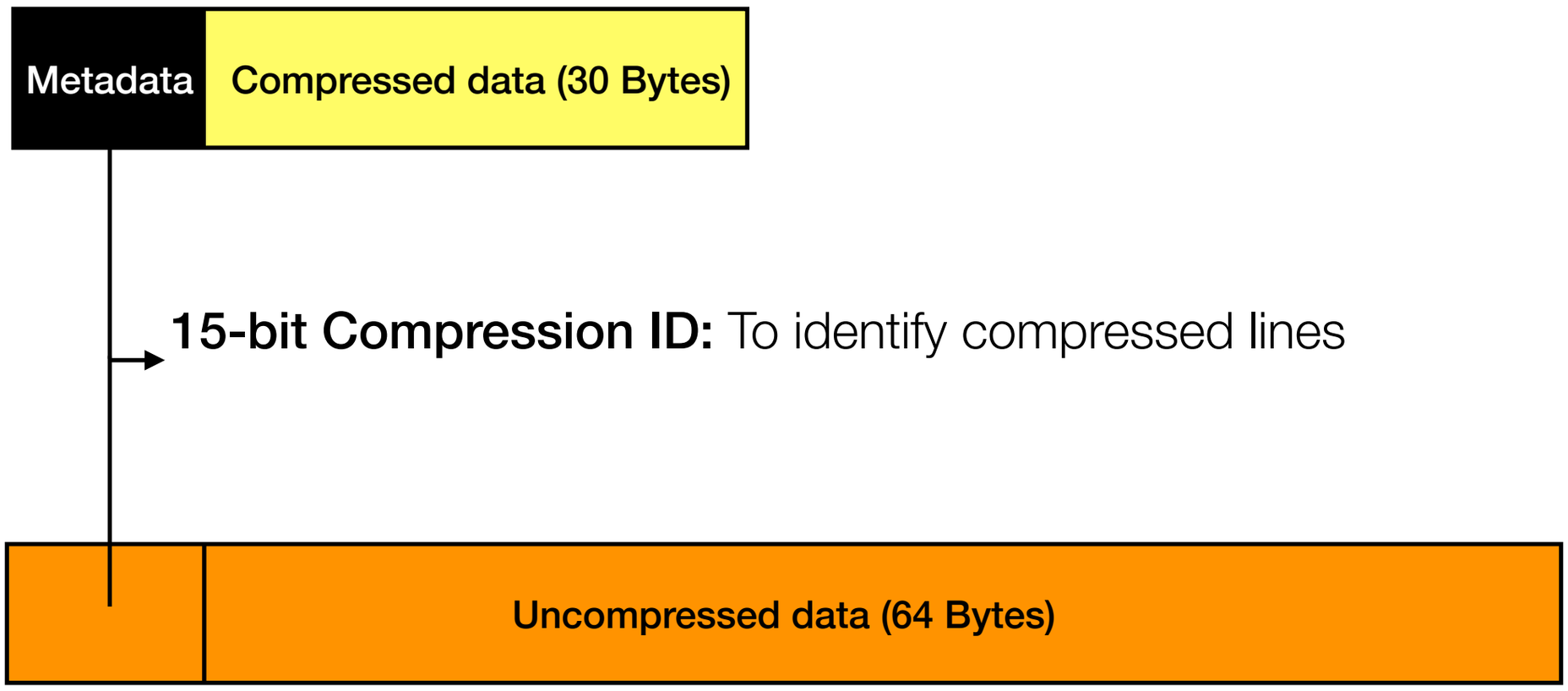
Attaché: Solution

Blended Metadata: *Mitigating Collision*



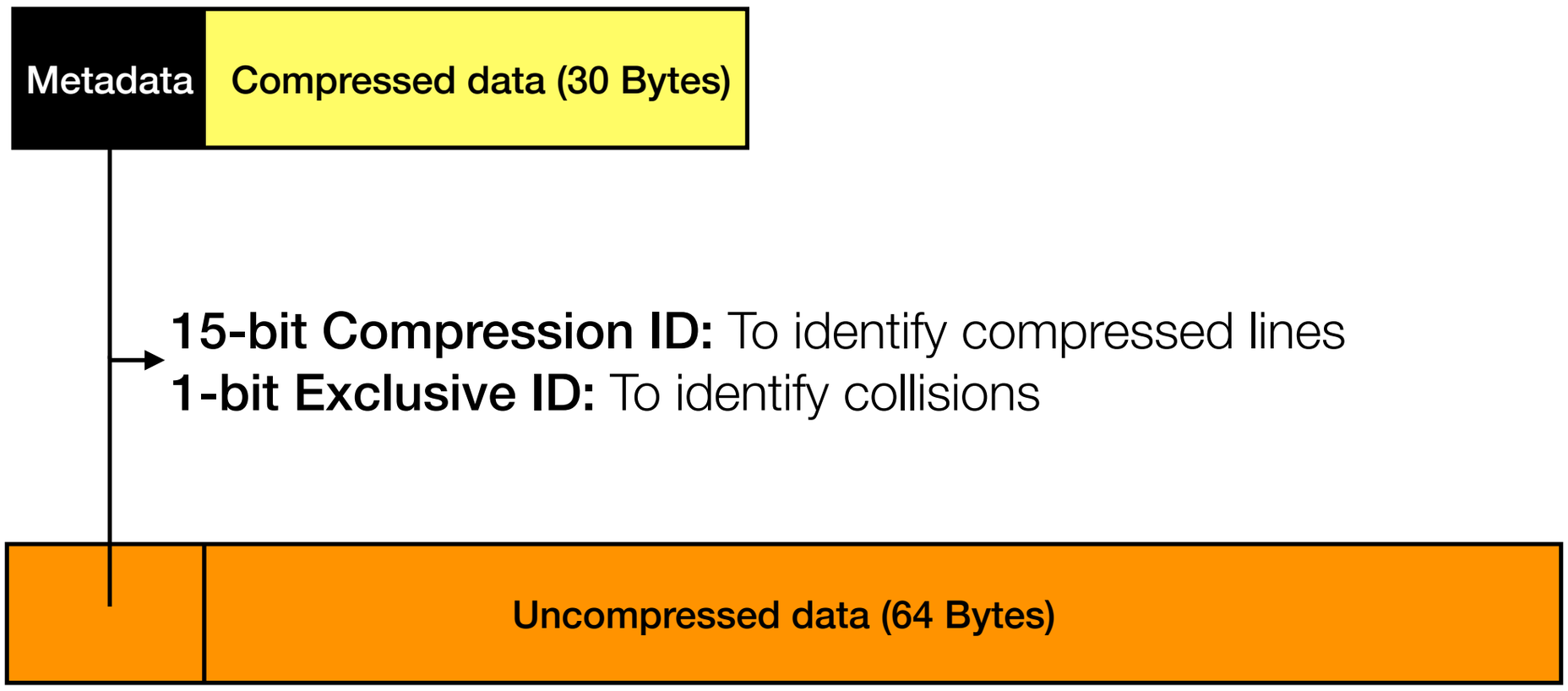
Attaché: Solution

Blended Metadata: *Mitigating Collision*



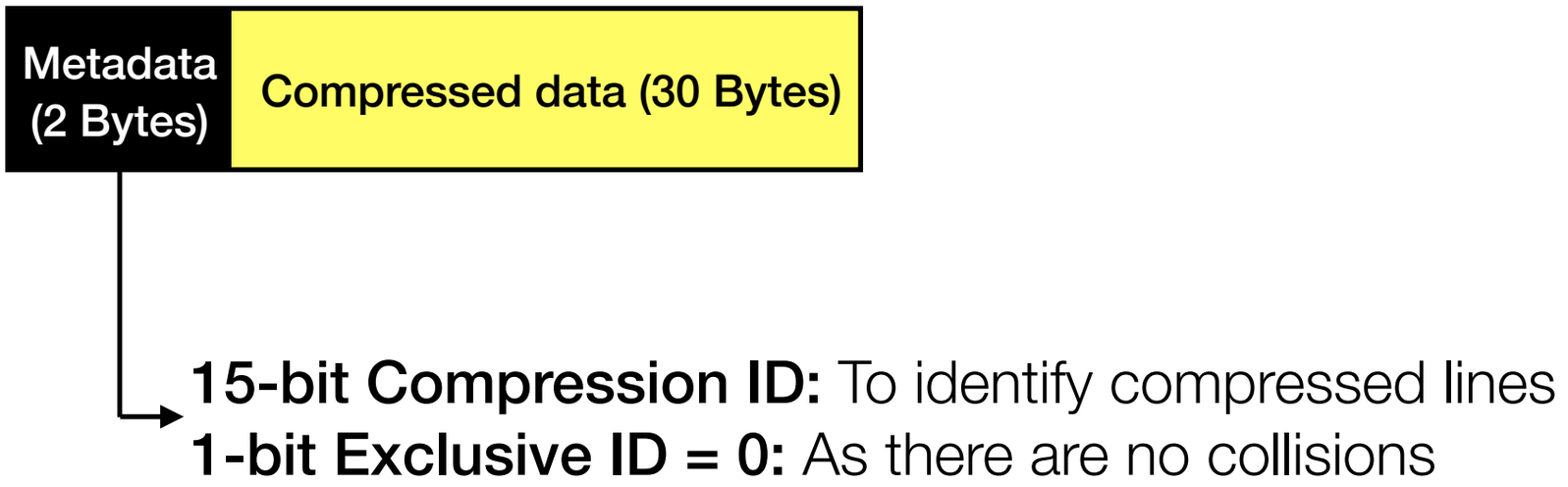
Attaché: Solution

Blended Metadata: *Mitigating Collision*



Attaché: Solution

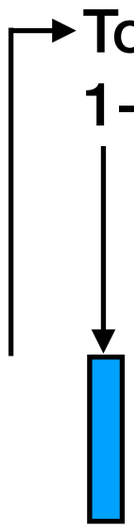
Blended Metadata: *Mitigating Collision*



15-bit Compression ID: To identify compressed lines
1-bit Exclusive ID = 0: As there are no collisions

Attaché: Solution

Blended Metadata: *Mitigating Collision*



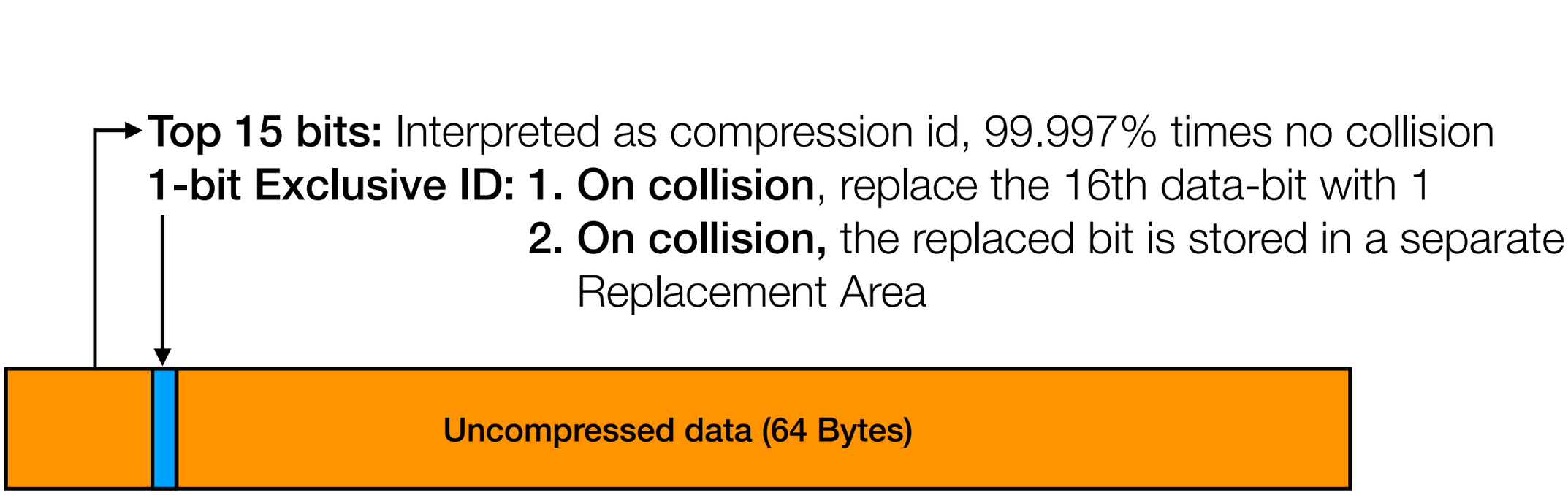
Top 15 bits: Interpreted as compression id, 99.997% times no collision

1-bit Exclusive ID: 1. **On collision**, replace the 16th data-bit with 1

2. **On collision,** the replaced bit is stored in a separate Replacement Area

Attaché: Solution

Blended Metadata: *Mitigating Collision*



All collisions can be detected 100% of the times: No issue of correctness

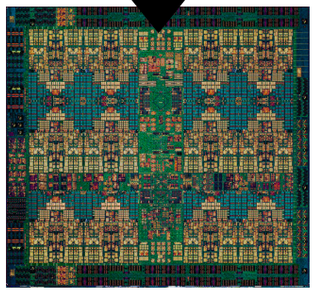
Attaché

Blended Metadata: Overheads

Main Memory

0.2% additional storage-overhead for Replacement Area

0.03% additional storage-overhead for Replacement Area



Core(s)

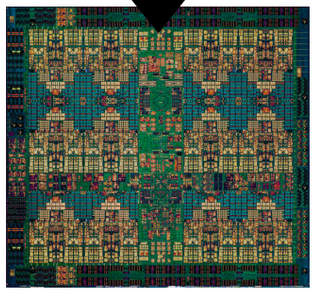
Attaché

Blended Metadata: Overheads

Main Memory

0.2% additional storage-overhead for Replacement Area

0.03% additional storage-overhead for Replacement Area

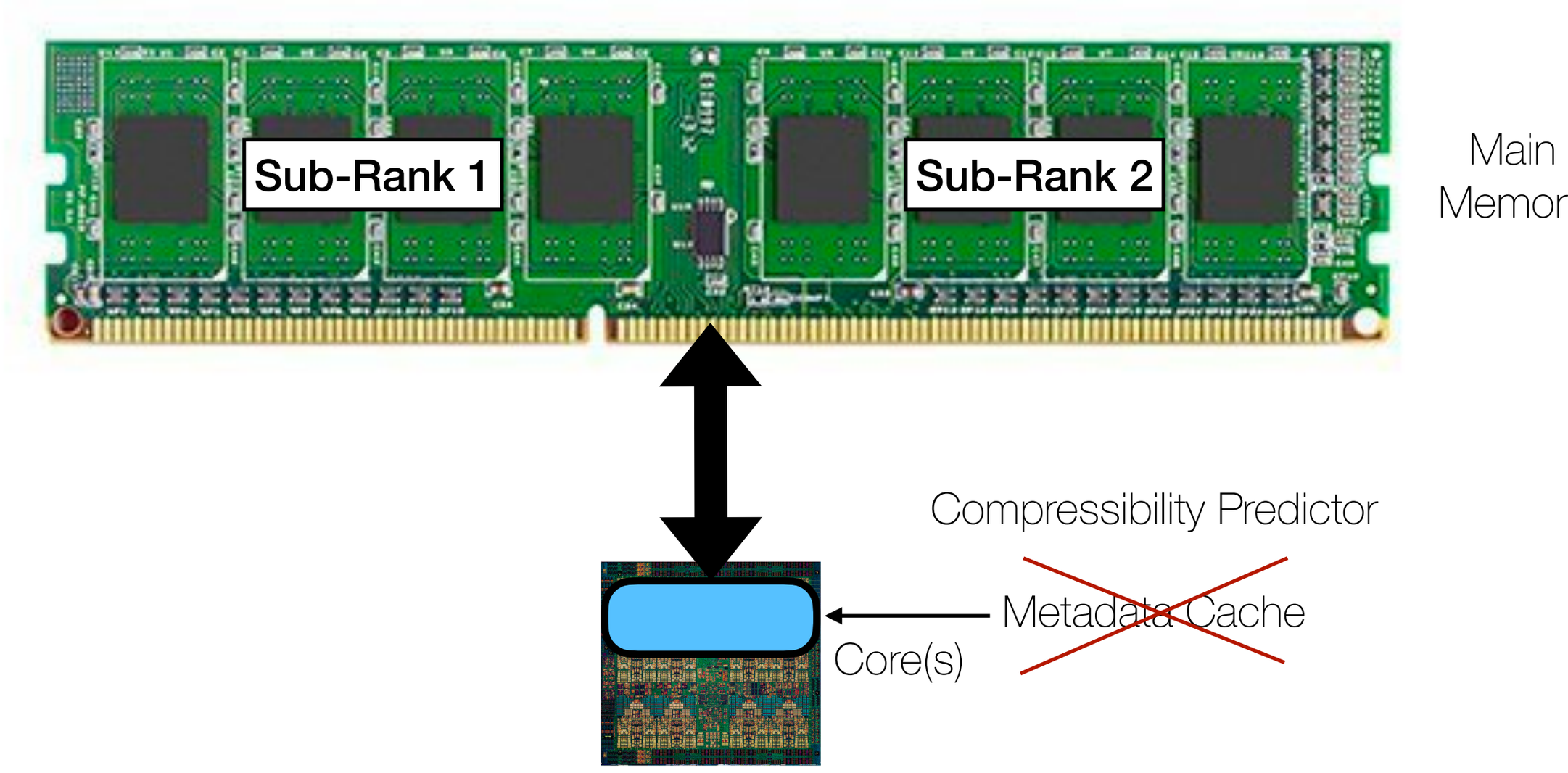


Core(s)

Blended Metadata has negligible overheads enabling near-ideal performance

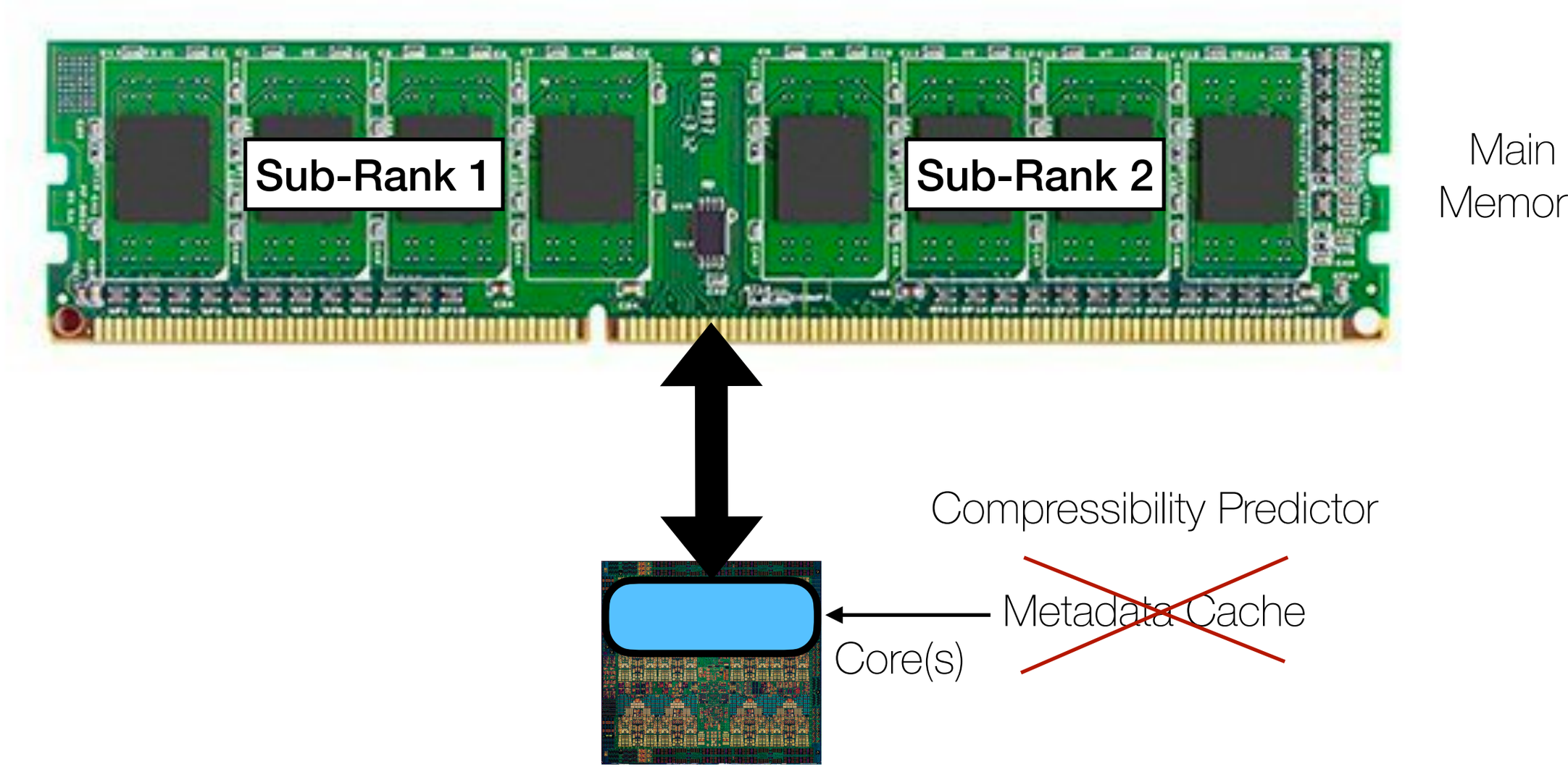
Attaché

Compressibility Predictor: Try to predict if the line is compressed or not



Attaché

Compressibility Predictor: Try to predict if the line is compressed or not



Only the appropriate Sub-Rank is enabled on a correct prediction

Thank You