

# Flash-Cosmos

## In-Flash Bulk Bitwise Operations Using Inherent Computation Capability of NAND Flash Memory

Jisung Park, Roknoddin Azizi, Geraldo F. Oliveira,  
Mohammad Sadrosadati, Rakesh Nadig, David Novo,  
Juan Gómez Luna, Myungsuk Kim, and Onur Mutlu



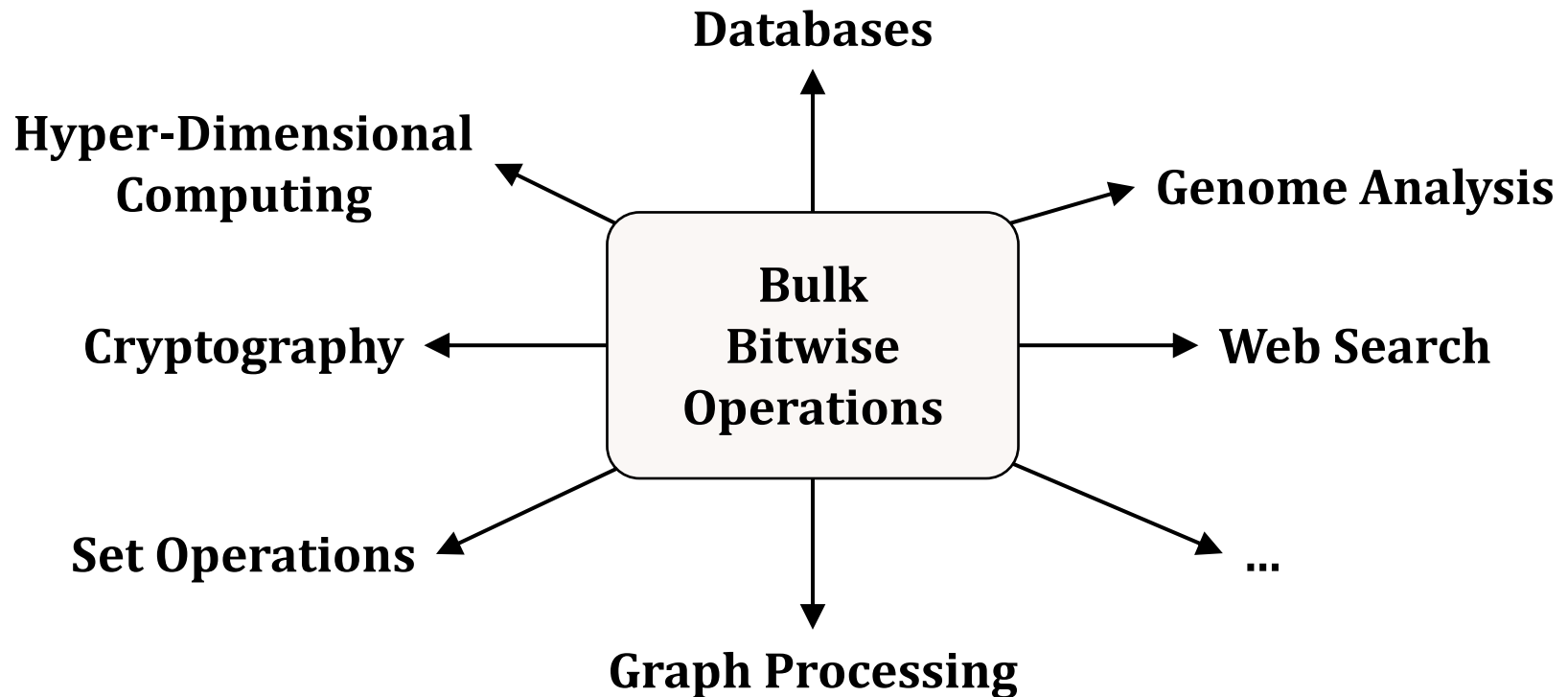
# Talk Outline

---

- Problem, Goals & Key Idea
- Background
- Flash-Cosmos: Computation with One-Shot Multi-Operand Sensing
- Evaluation
- Summary

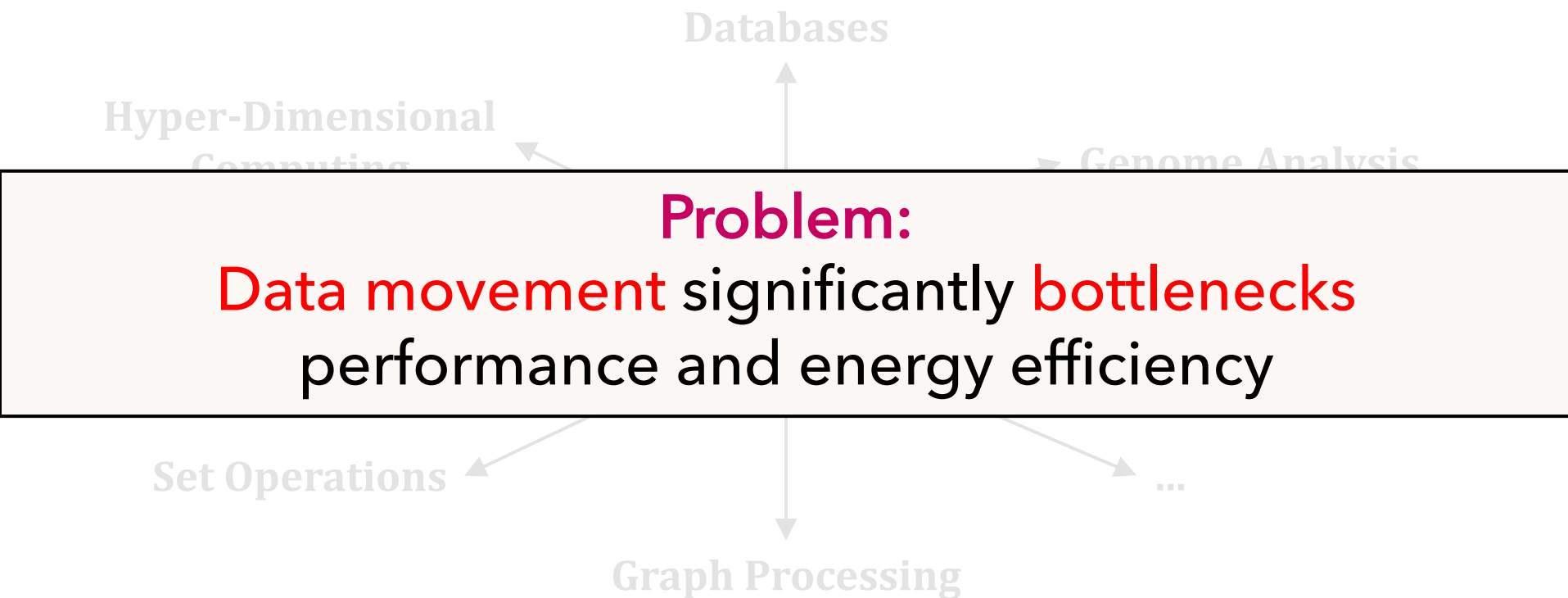
# Bulk Bitwise Operations

- Widely-used computation primitive in data-intensive applications



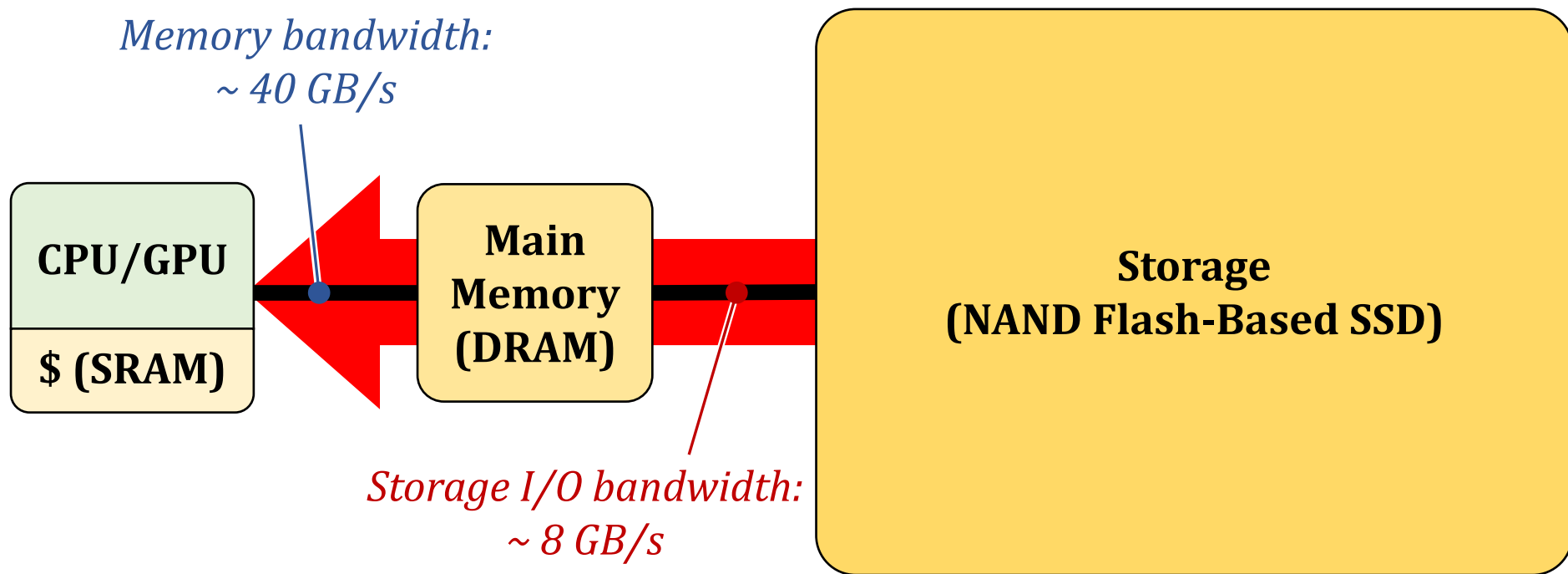
# Bulk Bitwise Operations

- Widely-used computation primitive in data-intensive applications



# Data-Movement Bottleneck

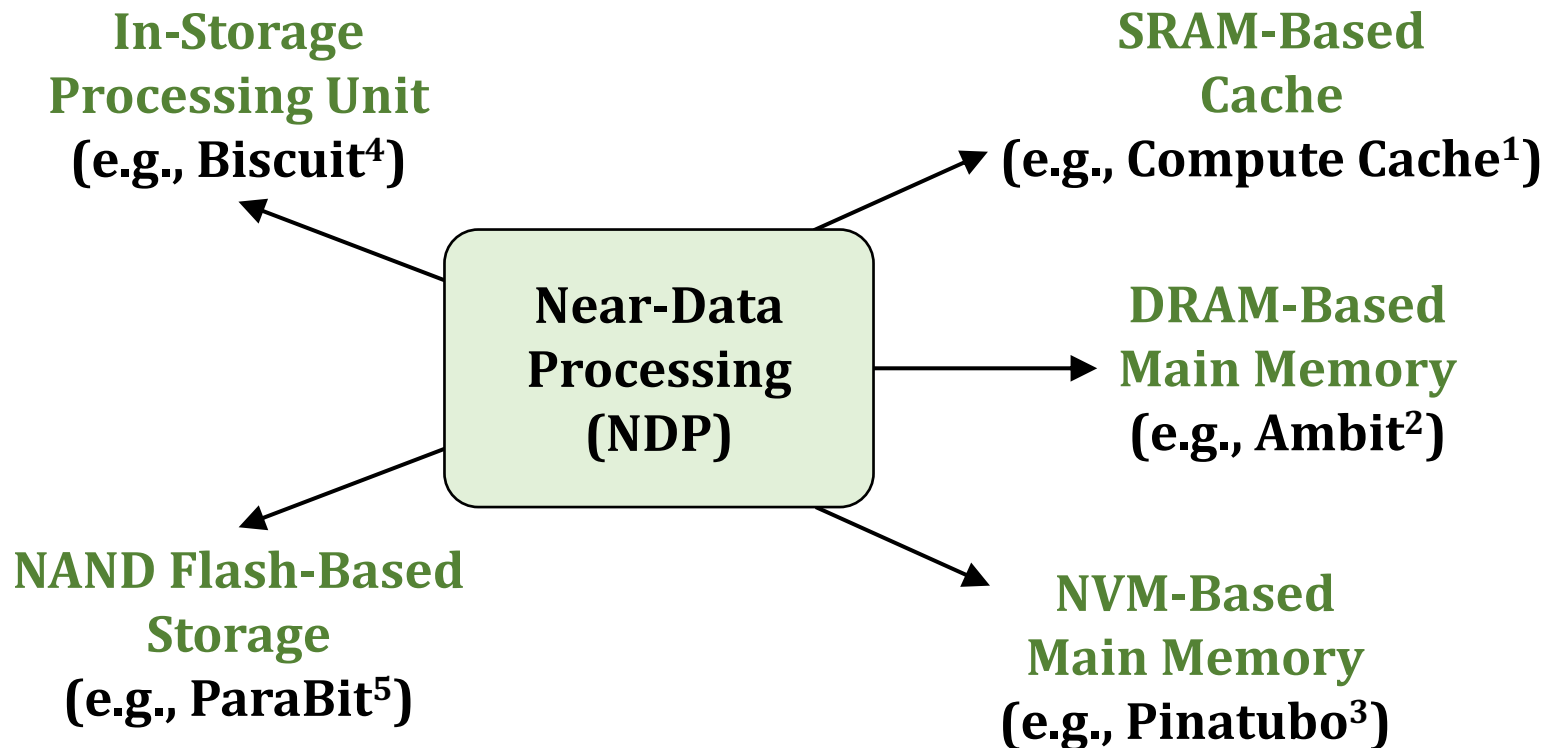
- Conventional systems: Outside-storage processing (OSP) that must move the entire data to CPUs/GPUs through the memory hierarchy



External I/O bandwidth of storage systems is the **main bottleneck** in conventional systems (OSP)

# Near-Data Processing for Bitwise Operations

- Can **effectively mitigate** data movement by performing simple bitwise operations **where the data resides**



<sup>1</sup>Aga+, "Compute Caches," HPCA, 2017

<sup>2</sup>Seshadri+, "Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology," MICRO, 2017

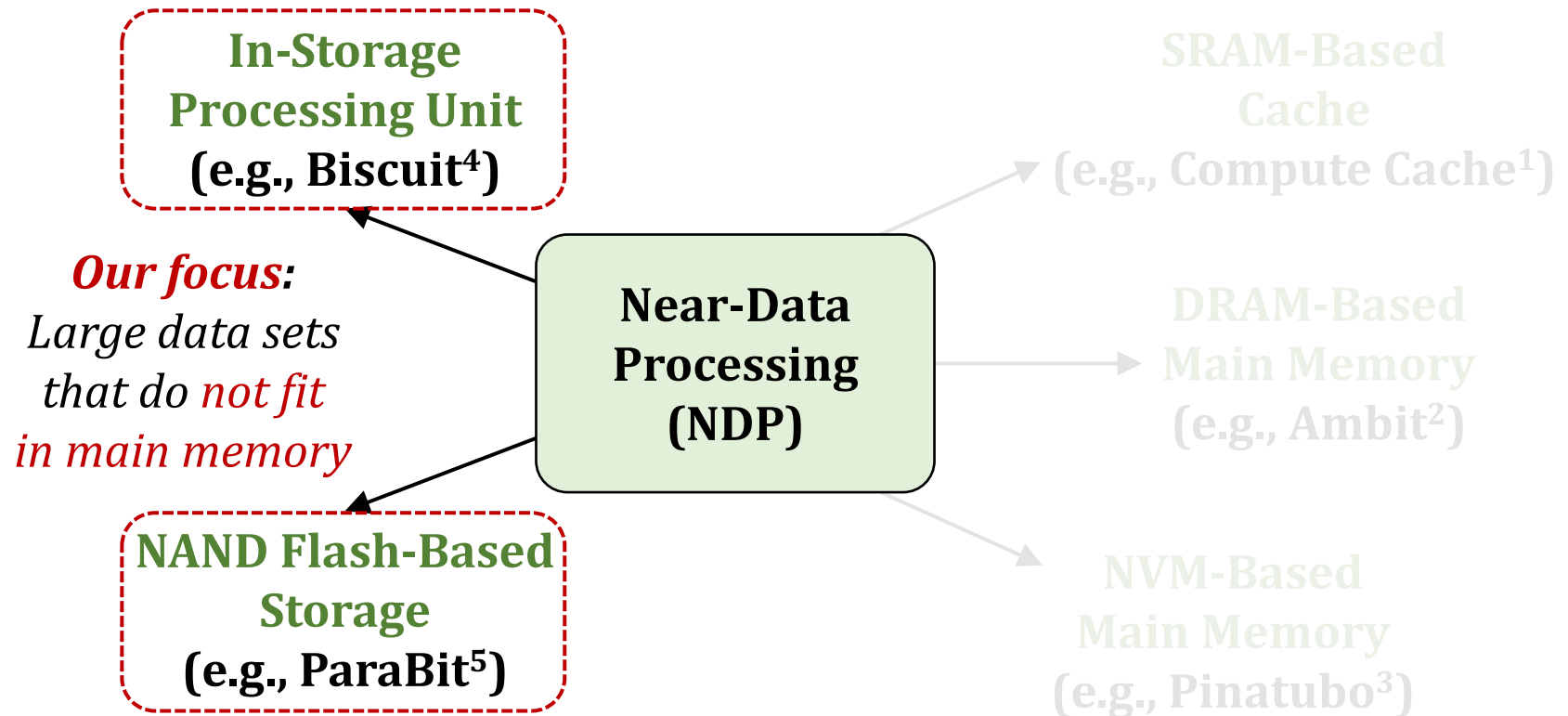
<sup>3</sup>Li+, "Pinatubo: A Processing-in-Memory Architecture for Bulk Bitwise Operations in Emerging Non-Volatile Memories," DAC, 2016

<sup>4</sup>Gu+, "Biscuit: A Framework for Near-Data Processing of Big Data Workloads," ISCA, 2016

<sup>5</sup>Gao+, "ParaBit: Processing Parallel Bitwise Operations in NAND Flash Memory Based SSDs," MICRO, 2021

# Near-Data Processing for Bitwise Operations

- Can **effectively mitigate** data movement by performing simple bitwise operations **where the data resides**



<sup>1</sup>Aga+, "Compute Caches," HPCA, 2017

<sup>2</sup>Seshadri+, "Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology," MICRO, 2017

<sup>3</sup>Li+, "Pinatubo: A Processing-in-Memory Architecture for Bulk Bitwise Operations in Emerging Non-Volatile Memories," DAC, 2016

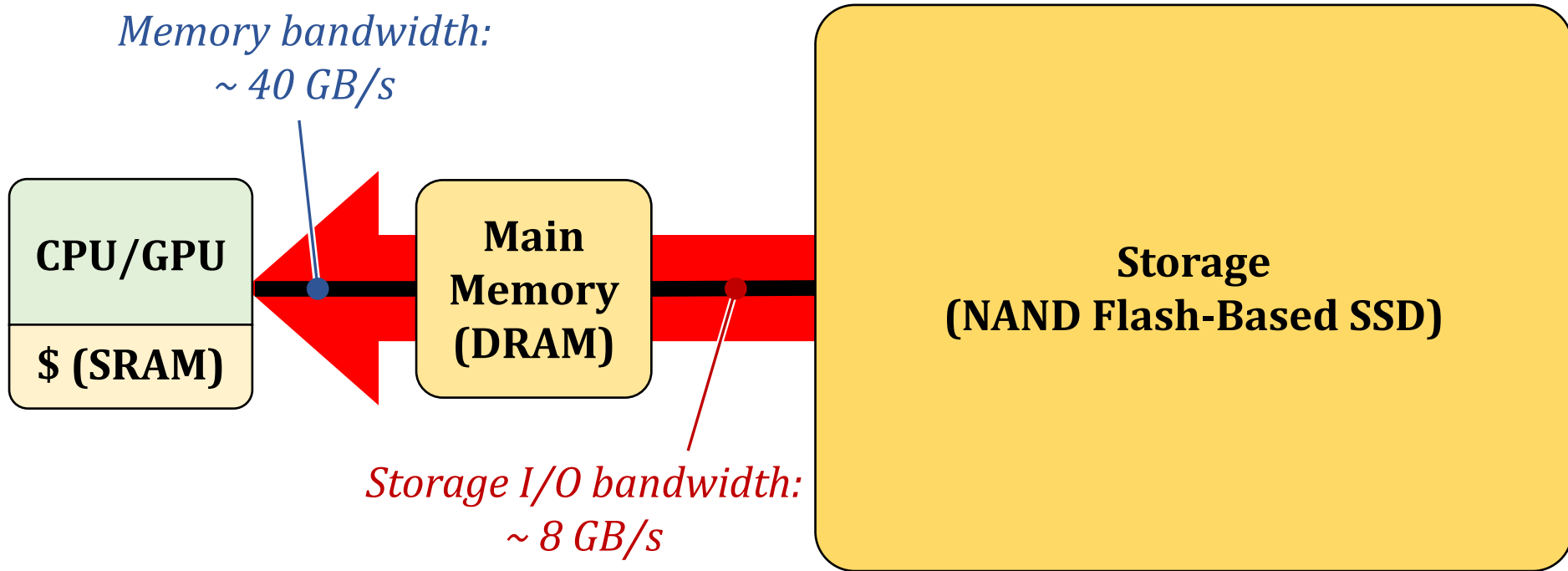
<sup>4</sup>Gu+, "Biscuit: A Framework for Near-Data Processing of Big Data Workloads," ISCA, 2016

<sup>5</sup>Gao+, "ParaBit: Processing Parallel Bitwise Operations in NAND Flash Memory Based SSDs," MICRO, 2021

# In-Storage Processing (ISP)

- Uses **in-storage compute units** (embedded cores or FPGA) to send **only the computation results**

*Memory bandwidth:  
~ 40 GB/s*

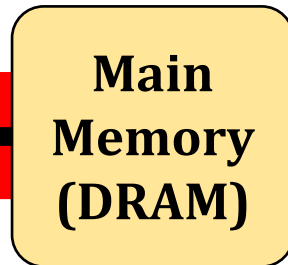
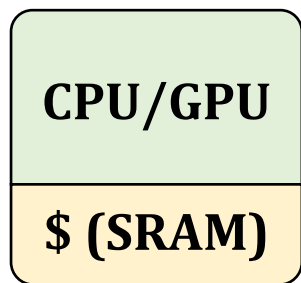




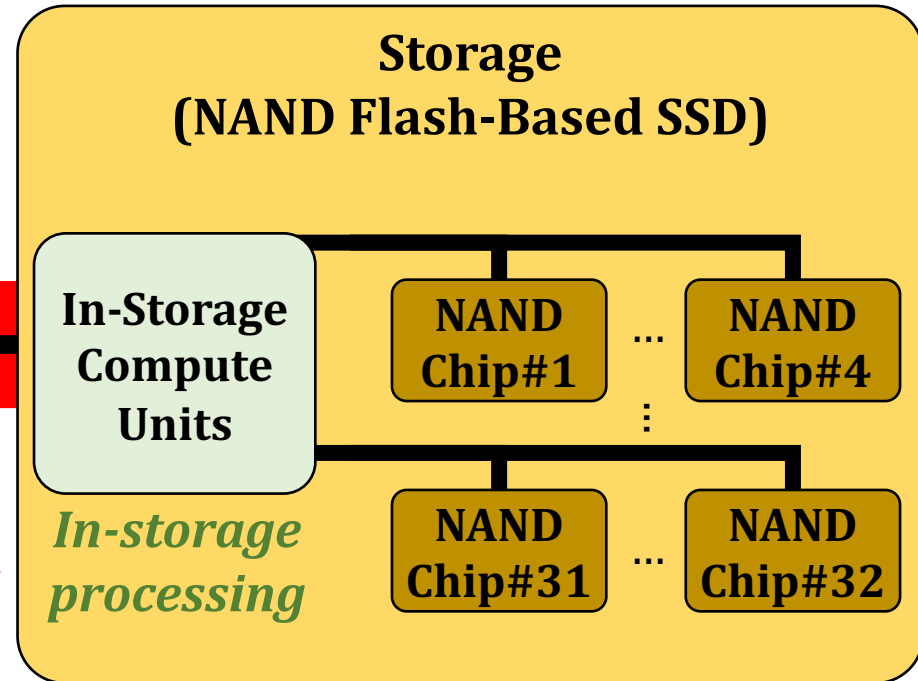
# In-Storage Processing (ISP)

- Uses **in-storage compute units** (embedded cores or FPGA) to send **only the computation results**

*Memory bandwidth:  
~ 40 GB/s*

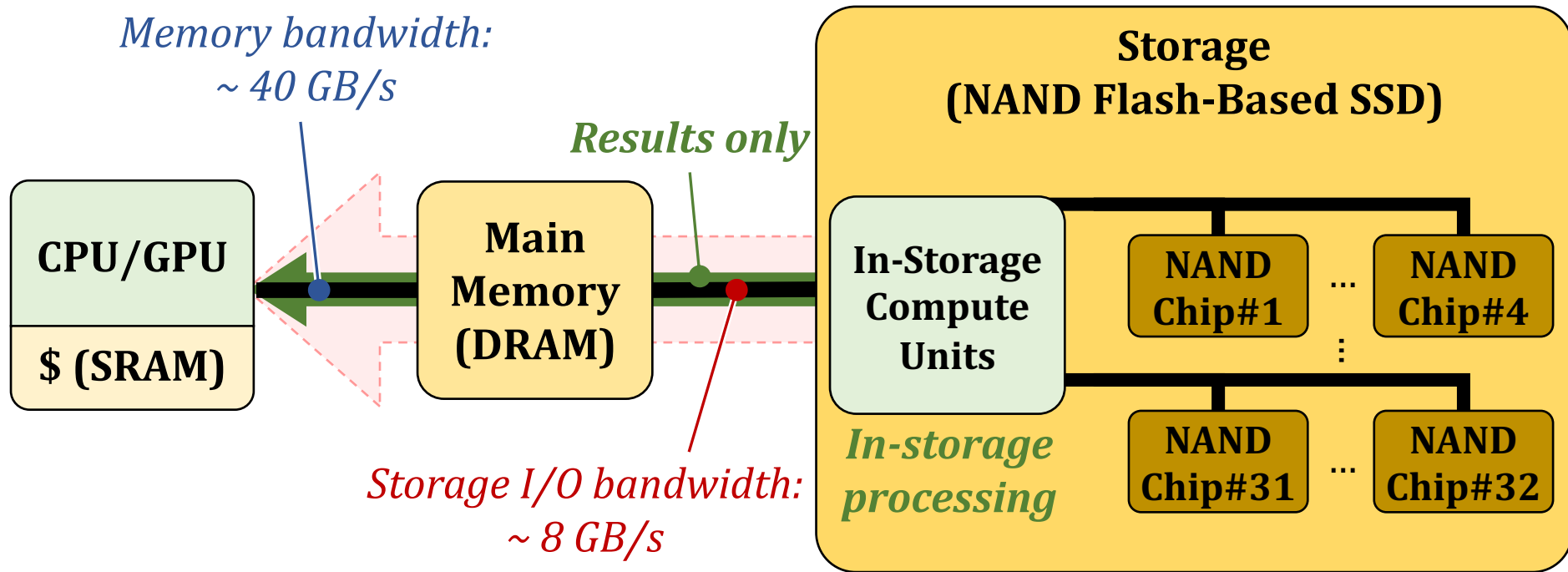


*Storage I/O bandwidth:  
~ 8 GB/s*



# In-Storage Processing (ISP)

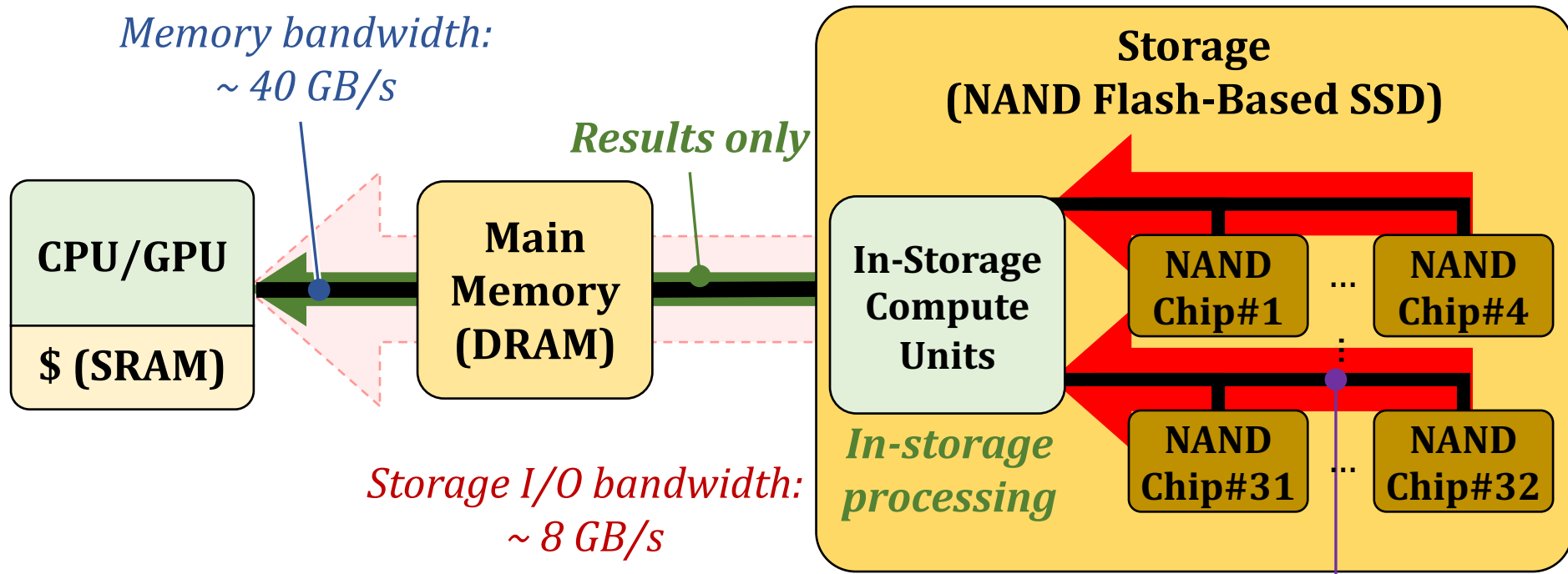
- Uses **in-storage compute units** (embedded cores or FPGA) to send **only the computation results**



ISP can mitigate data movement overhead by **reducing SSD-external data movement**

# In-Storage Processing (ISP)

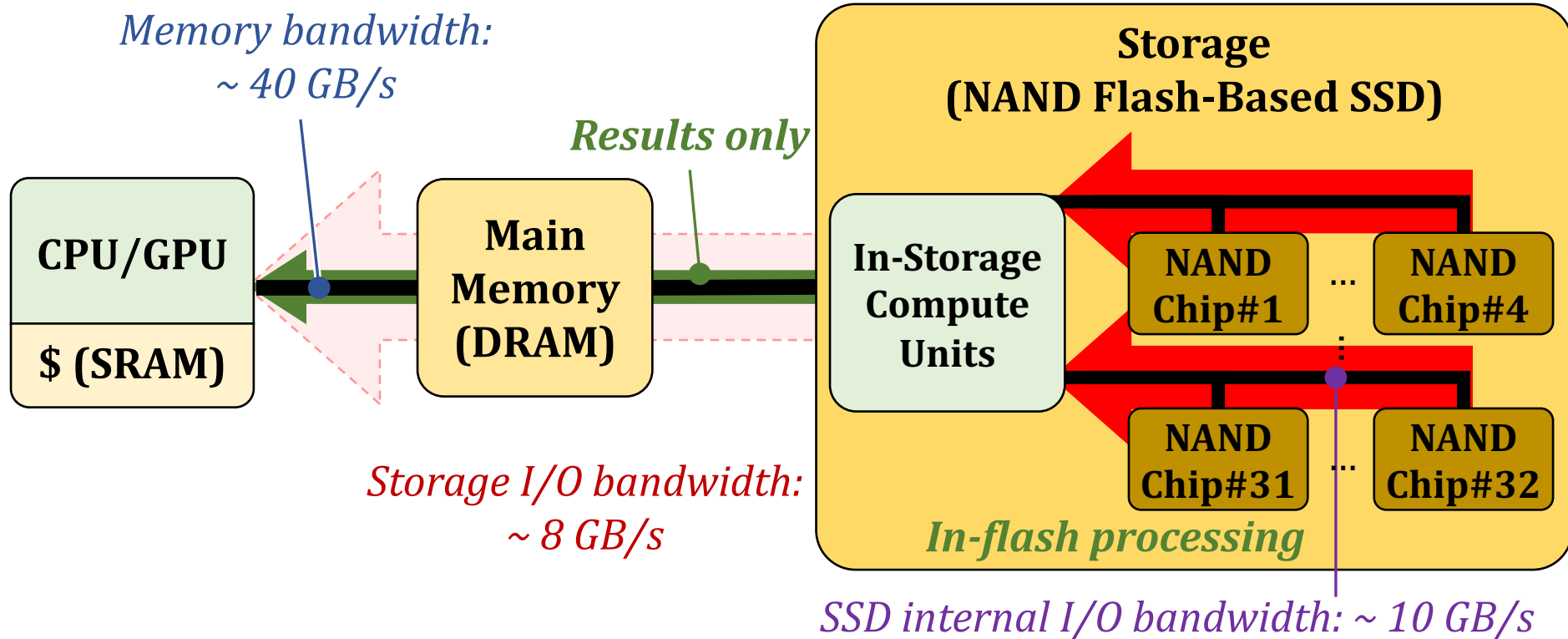
- Uses **in-storage compute units** (embedded cores or FPGA) to send **only the computation results**



SSD-internal bandwidth  
becomes the **new bottleneck** in ISP

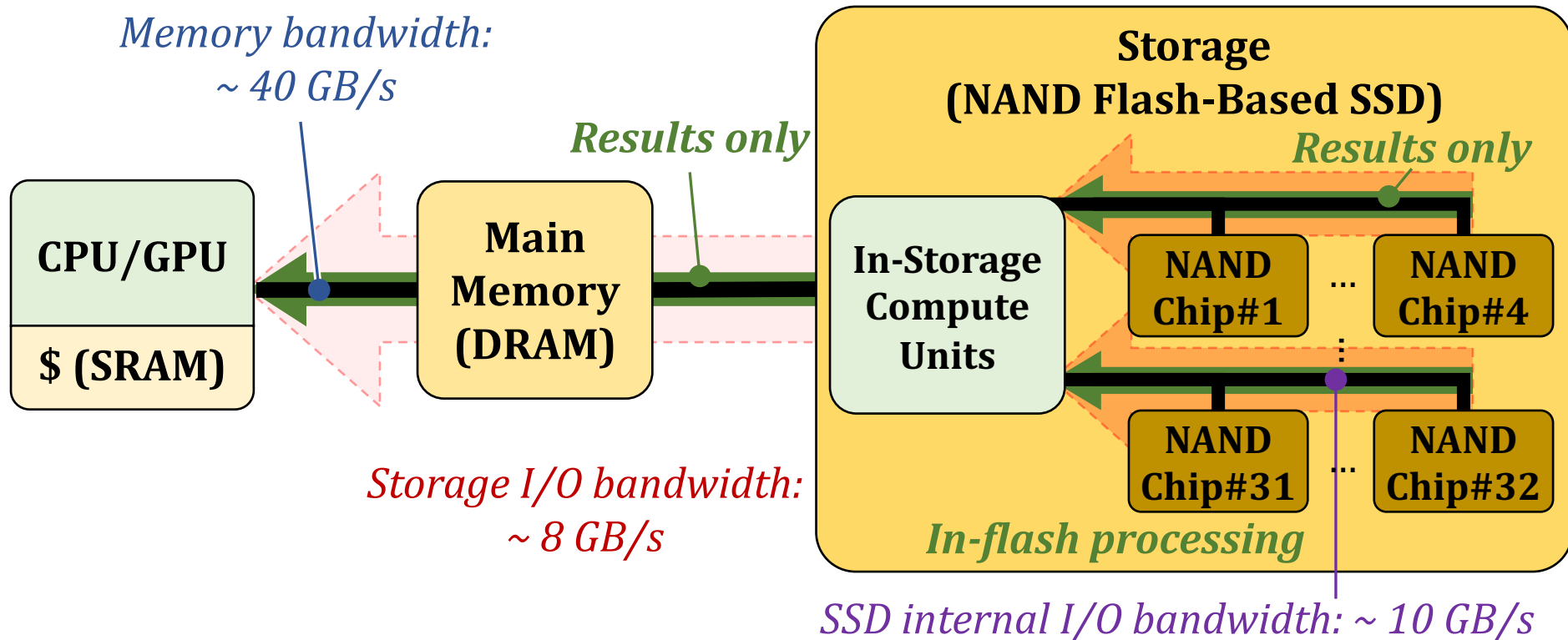
# In-Flash Processing (IFP)

- Performs computation *inside* NAND flash chips



# In-Flash Processing (IFP)

- Performs computation *inside* NAND flash chips

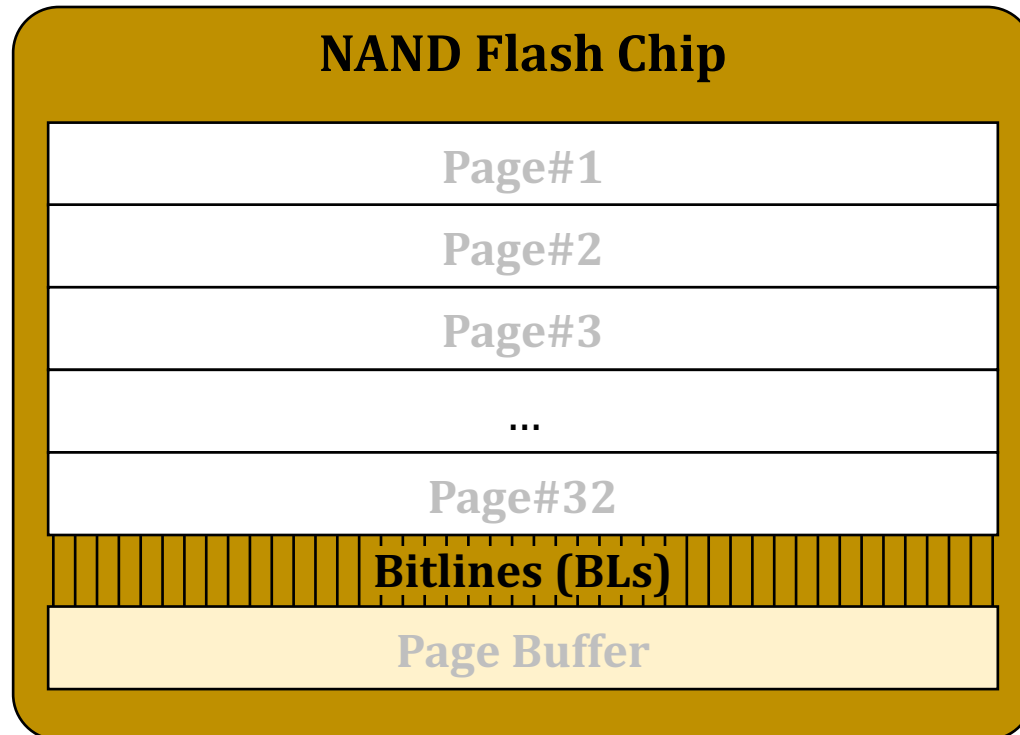


IFP fundamentally mitigates data movement

# State-of-the-Art IFP Technique (1/2)

## ▪ ParaBit [Gao+, MICRO 2021]

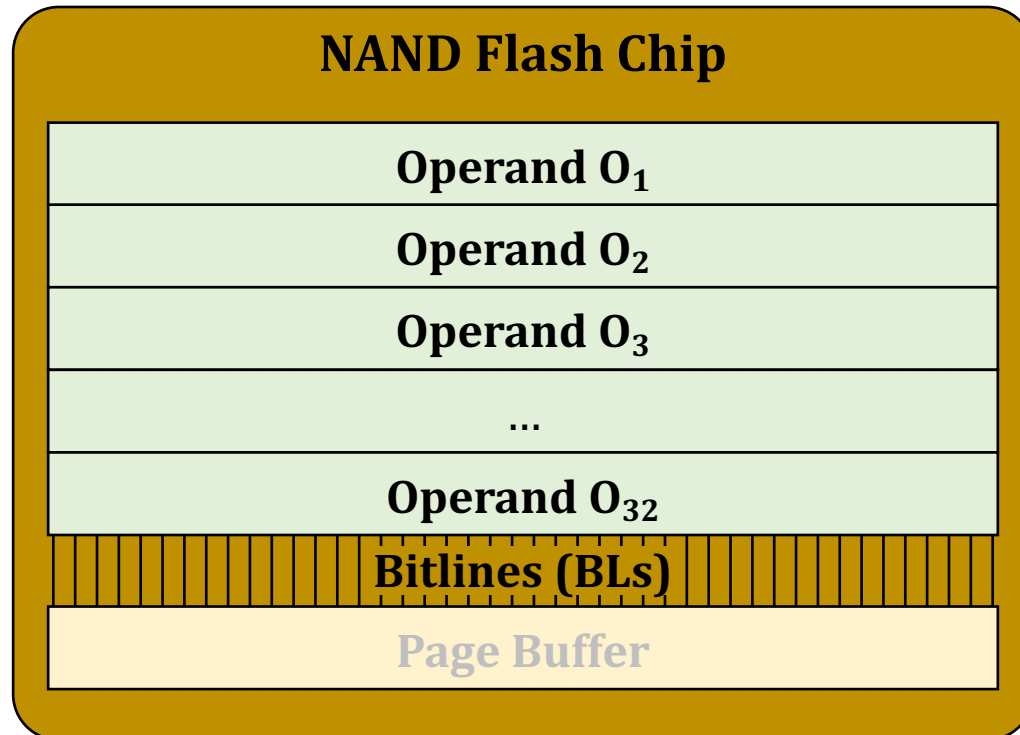
- Performs bulk bitwise operations **inside NAND flash chips** by **intelligently controlling the latching circuit of the page buffer**



# State-of-the-Art IFP Technique (1/2)

## ▪ ParaBit [Gao+, MICRO 2021]

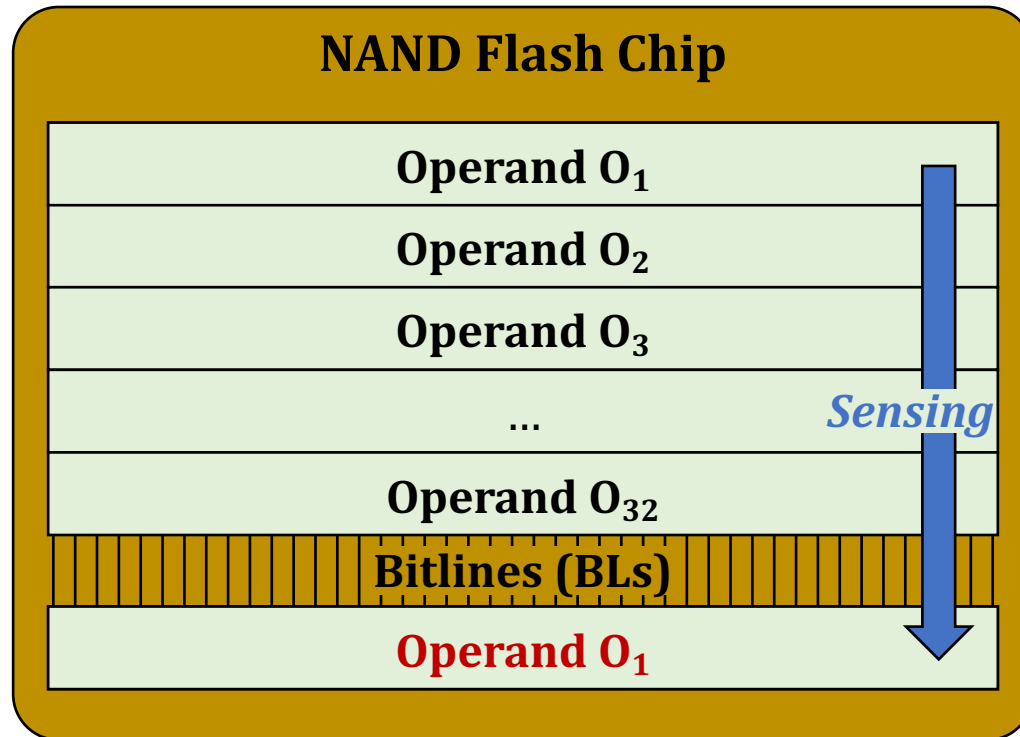
- Performs bulk bitwise operations **inside NAND flash chips** by **intelligently controlling the latching circuit of the page buffer**



# State-of-the-Art IFP Technique (1/2)

## ▪ ParaBit [Gao+, MICRO 2021]

- Performs bulk bitwise operations **inside NAND flash chips** by **intelligently controlling the latching circuit of the page buffer**

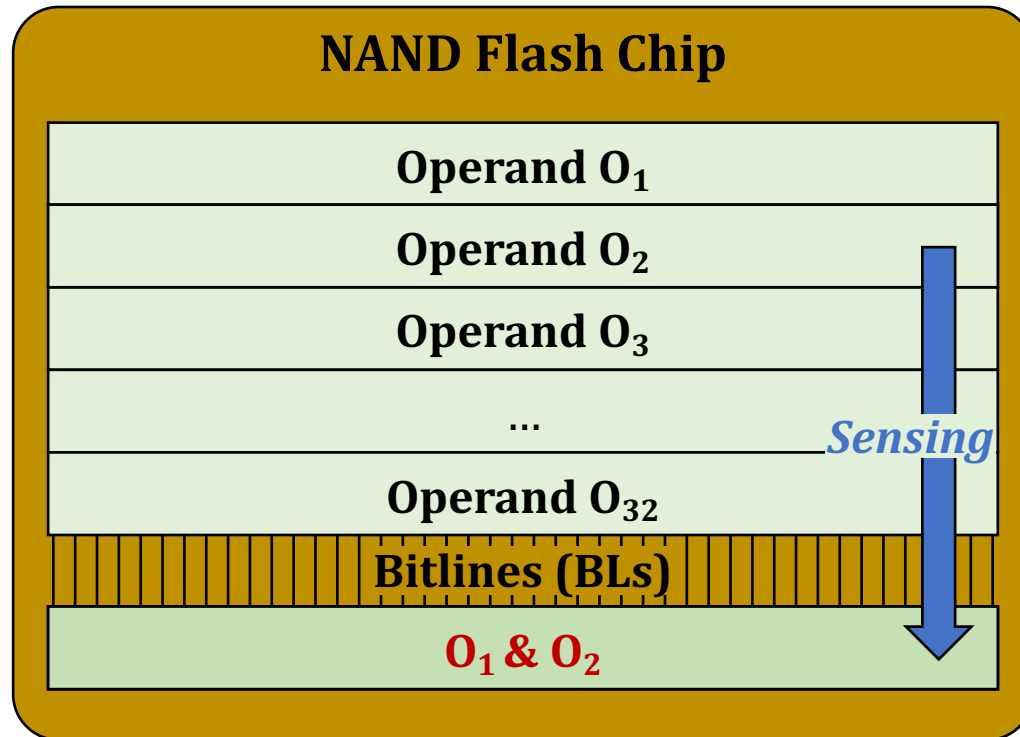




# State-of-the-Art IFP Technique (1/2)

## ▪ ParaBit [Gao+, MICRO 2021]

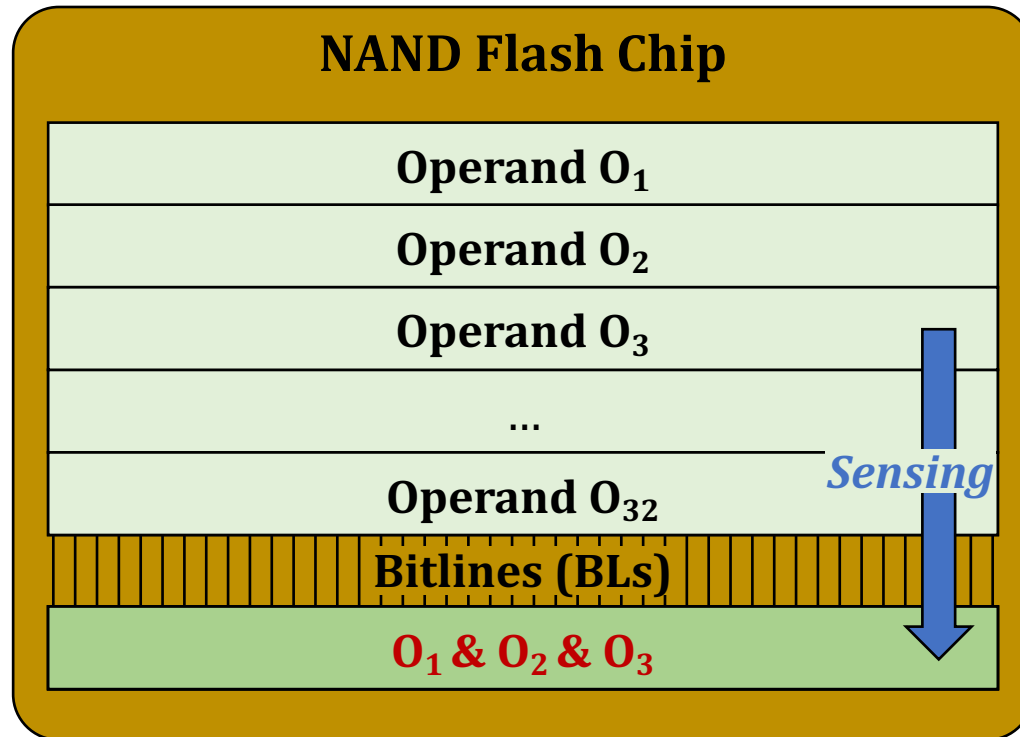
- Performs bulk bitwise operations **inside NAND flash chips** by **intelligently controlling the latching circuit of the page buffer**



# State-of-the-Art IFP Technique (1/2)

## ▪ ParaBit [Gao+, MICRO 2021]

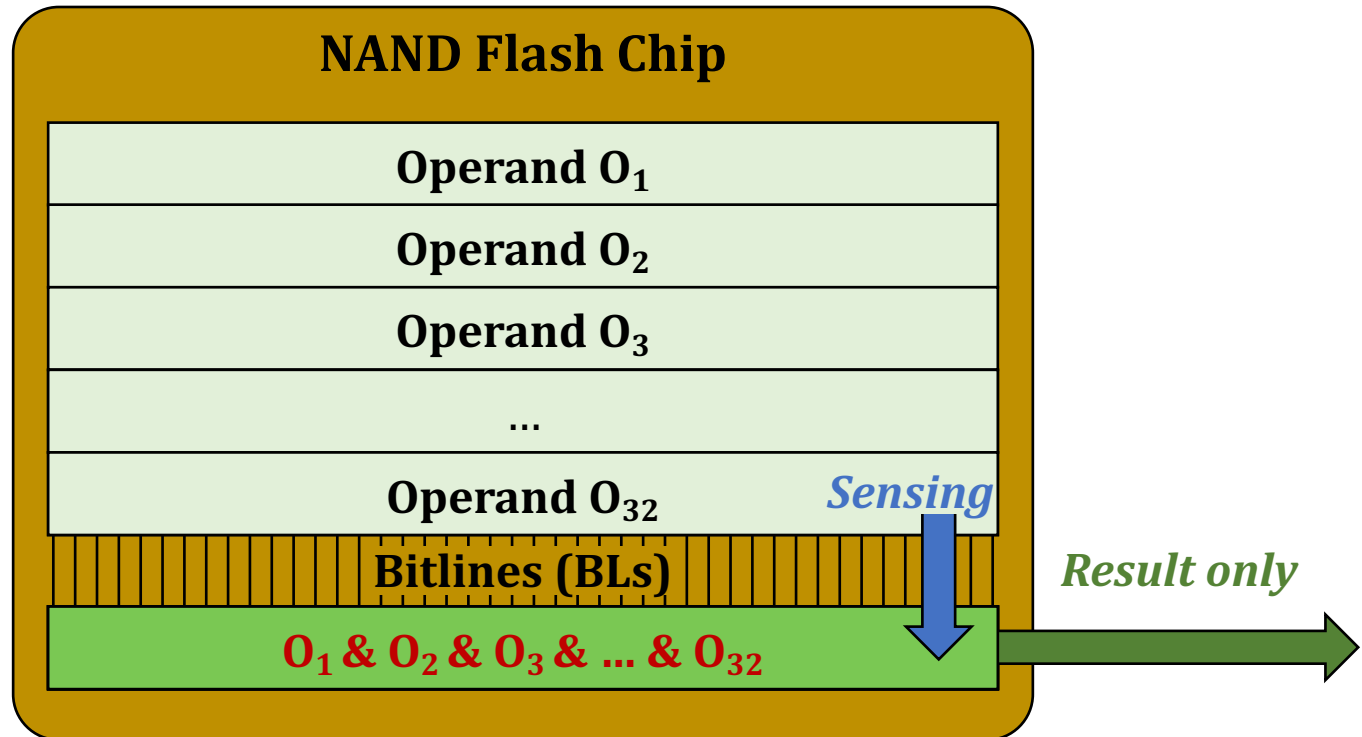
- Performs bulk bitwise operations **inside NAND flash chips** by **intelligently controlling the latching circuit of the page buffer**



# State-of-the-Art IFP Technique (1/2)

## ▪ ParaBit [Gao+, MICRO 2021]

- Performs bulk bitwise operations *inside* NAND flash chips by *intelligently* controlling the latching circuit of the page buffer



# State-of-the-Art IFP Technique (1/2)

- **ParaBit** [Gao+, MICRO 2021]

- Performs bulk bitwise operations inside NAND flash chips by intelligently controlling the latching circuit of the page buffer

NAND Flash Chip

ParaBit **significantly reduces** data movement out of NAND flash chips

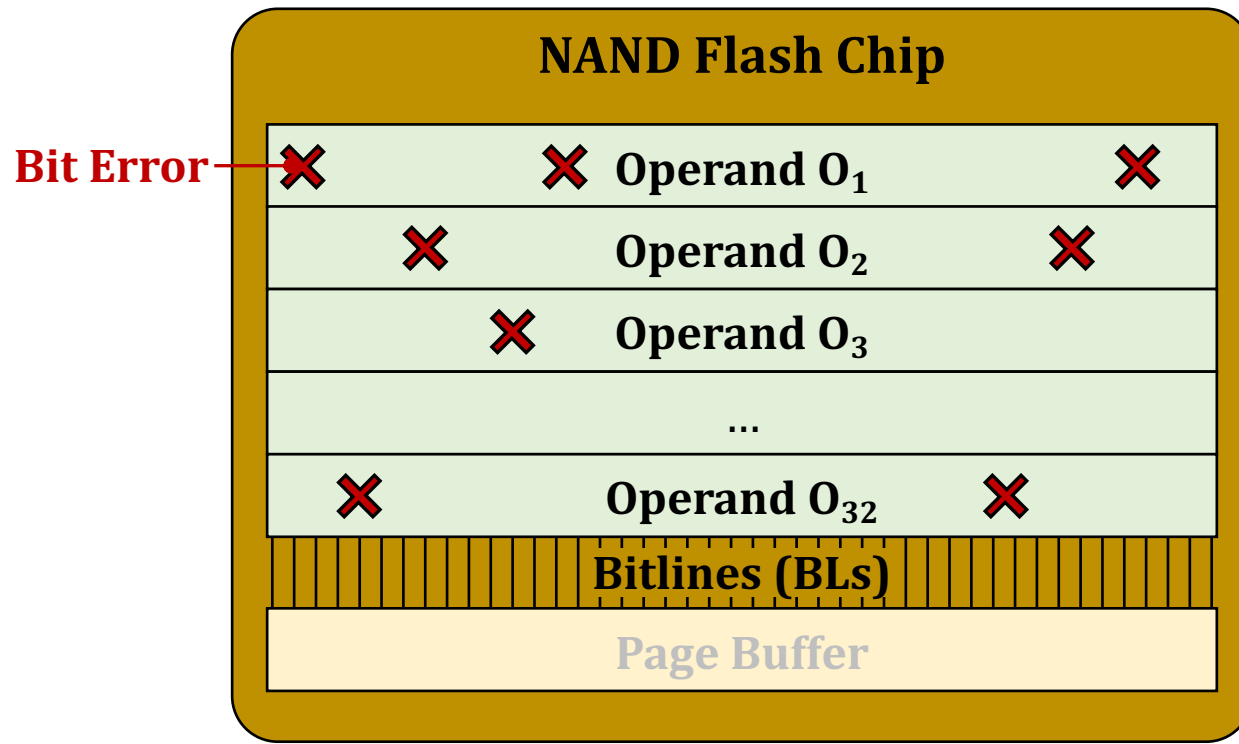
...

**Serial sensing operations**  
(e.g., 32 sensing operations for 32-operand AND)  
**bottleneck** performance and energy in ParaBit

# State-of-the-Art IFP Technique (2/2)

## ▪ Limitations of ParaBit [Gao+, MICRO 2021]

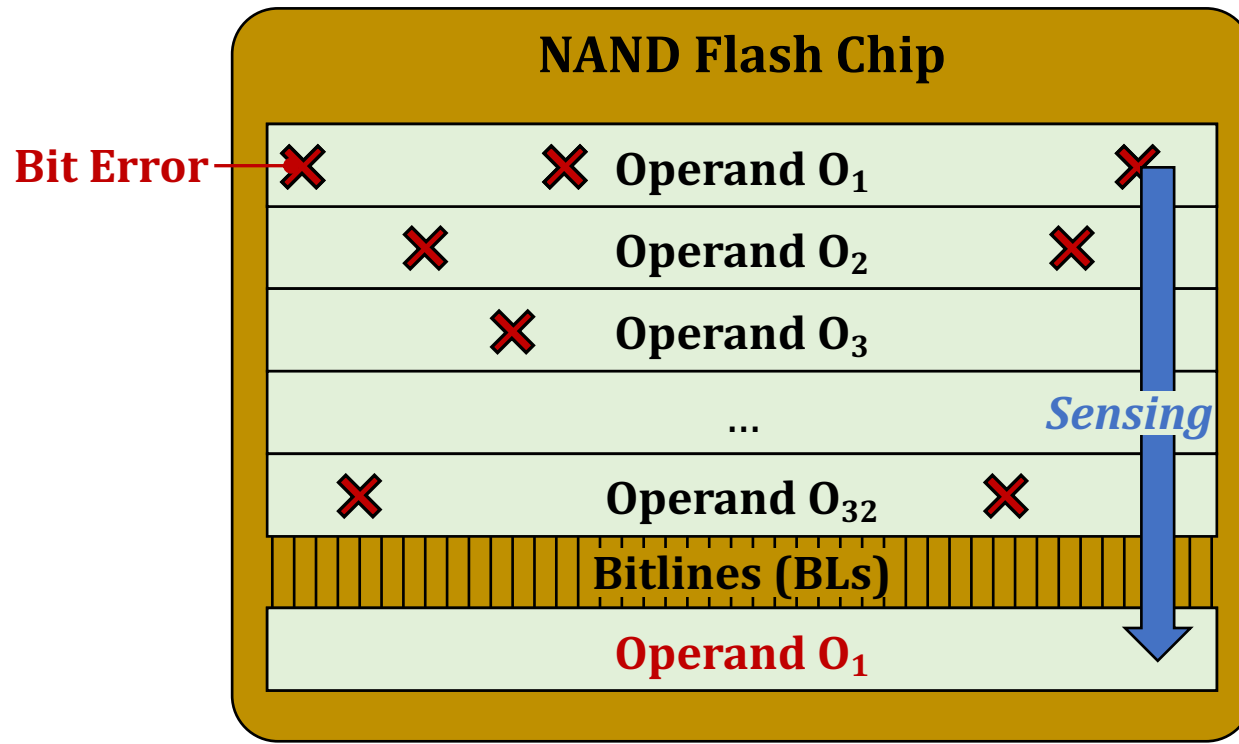
- Serial sensing operations become the new bottleneck
- Erroneous computation results due to the high raw bit-error rate of NAND flash memory



# State-of-the-Art IFP Technique (2/2)

## ▪ Limitations of ParaBit [Gao+, MICRO 2021]

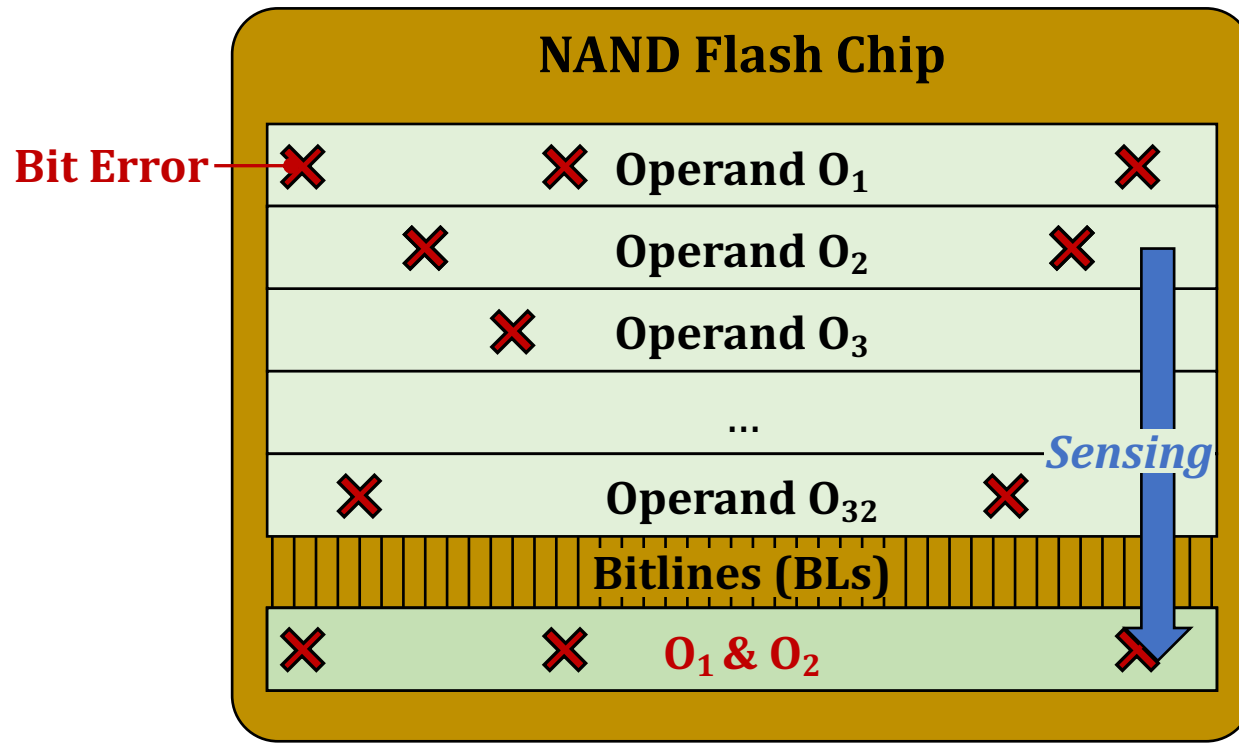
- Serial sensing operations become the new bottleneck
- Erroneous computation results due to the high raw bit-error rate of NAND flash memory



# State-of-the-Art IFP Technique (2/2)

## ▪ Limitations of ParaBit [Gao+, MICRO 2021]

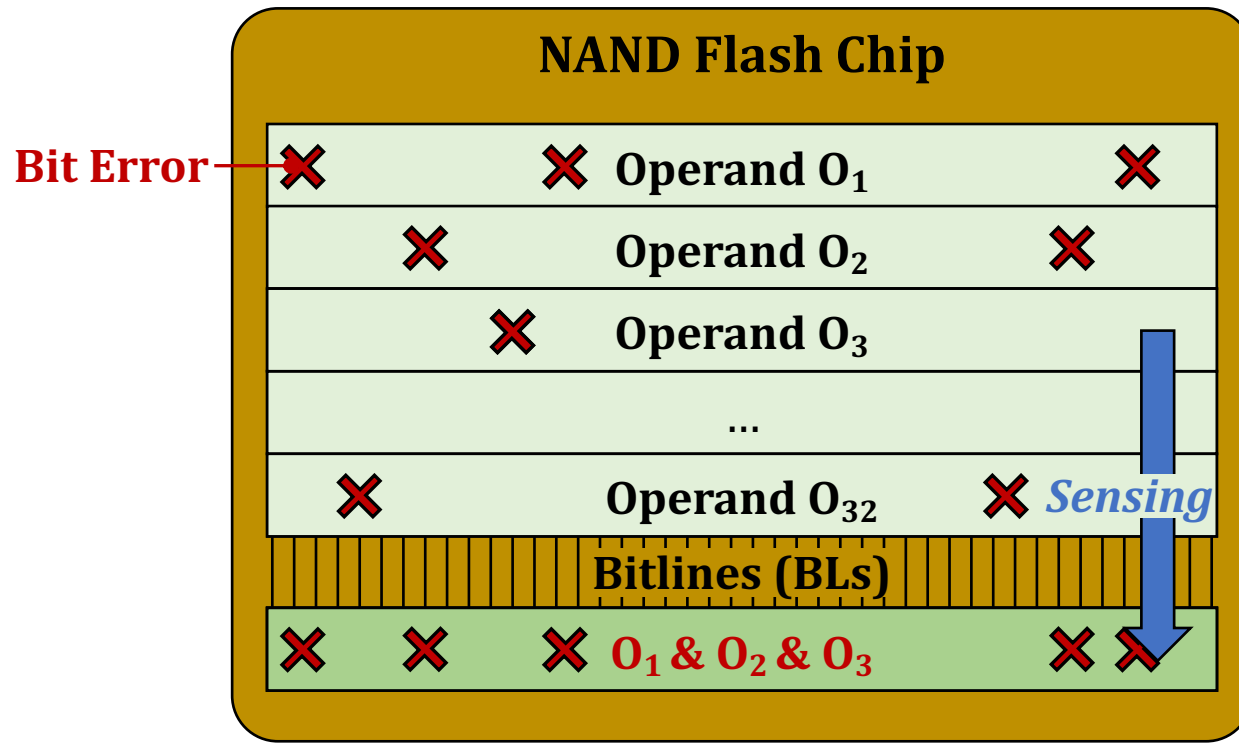
- Serial sensing operations become the new bottleneck
- Erroneous computation results due to the high raw bit-error rate of NAND flash memory



# State-of-the-Art IFP Technique (2/2)

## ▪ Limitations of ParaBit [Gao+, MICRO 2021]

- Serial sensing operations become the new bottleneck
- Erroneous computation results due to the high raw bit-error rate of NAND flash memory

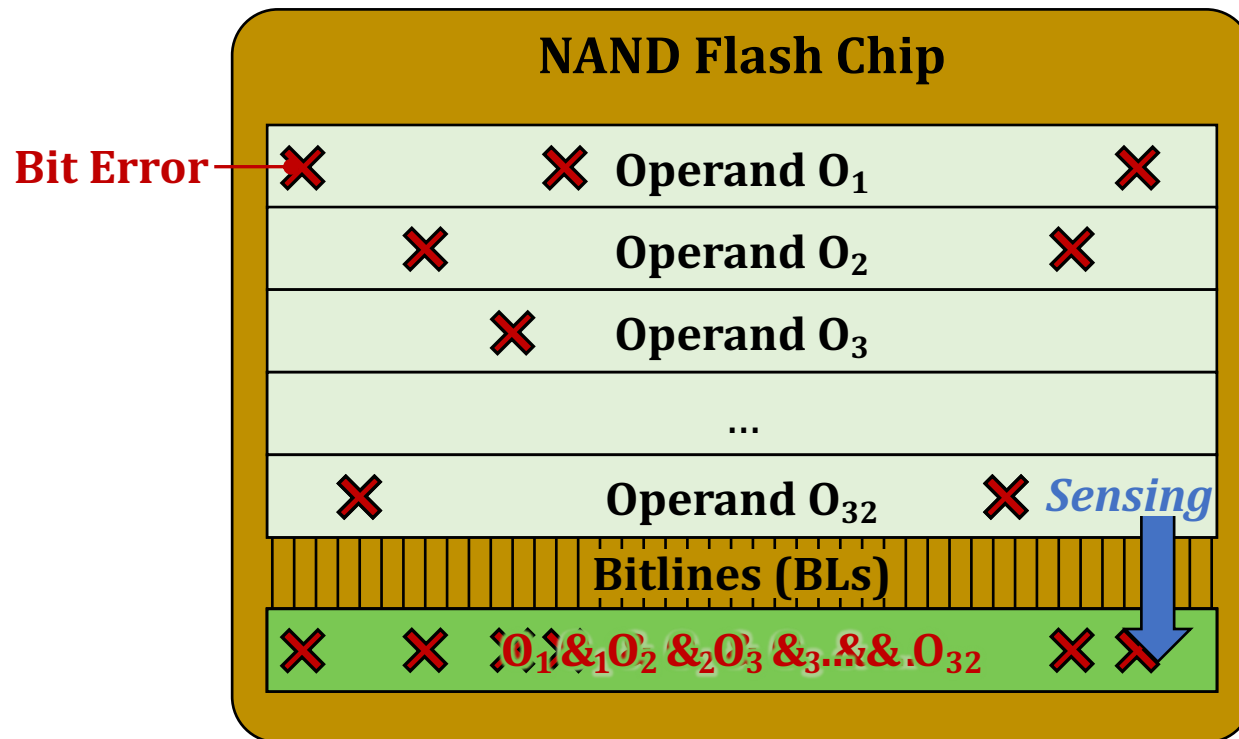




# State-of-the-Art IFP Technique (2/2)

## ▪ Limitations of ParaBit [Gao+, MICRO 2021]

- Serial sensing operations become the new bottleneck
- Erroneous computation results due to the high raw bit-error rate of NAND flash memory



# Our Goals

---

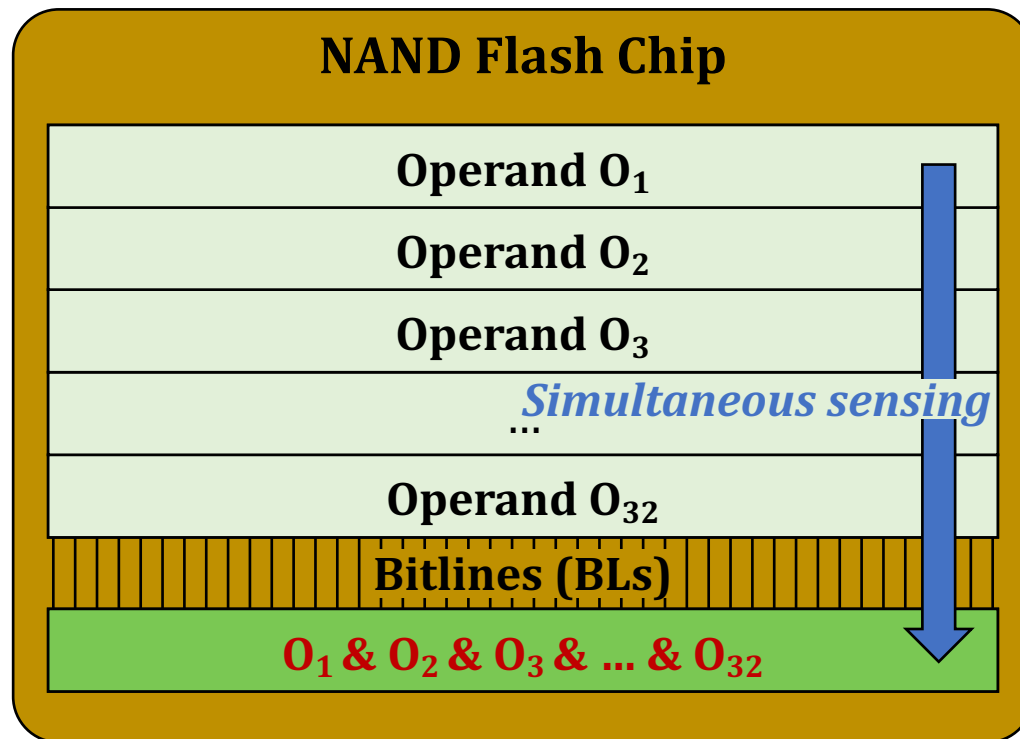
Address the new bottleneck of IFP  
(serial sensing of operands)

Make IFP reliable  
(provide accurate computation results)

# Our Proposal: Flash-Cosmos

## ▪ Flash-Cosmos enables

- Computation on multiple operands with a single sensing operation
- Accurate computation results by eliminating raw bit errors in stored data



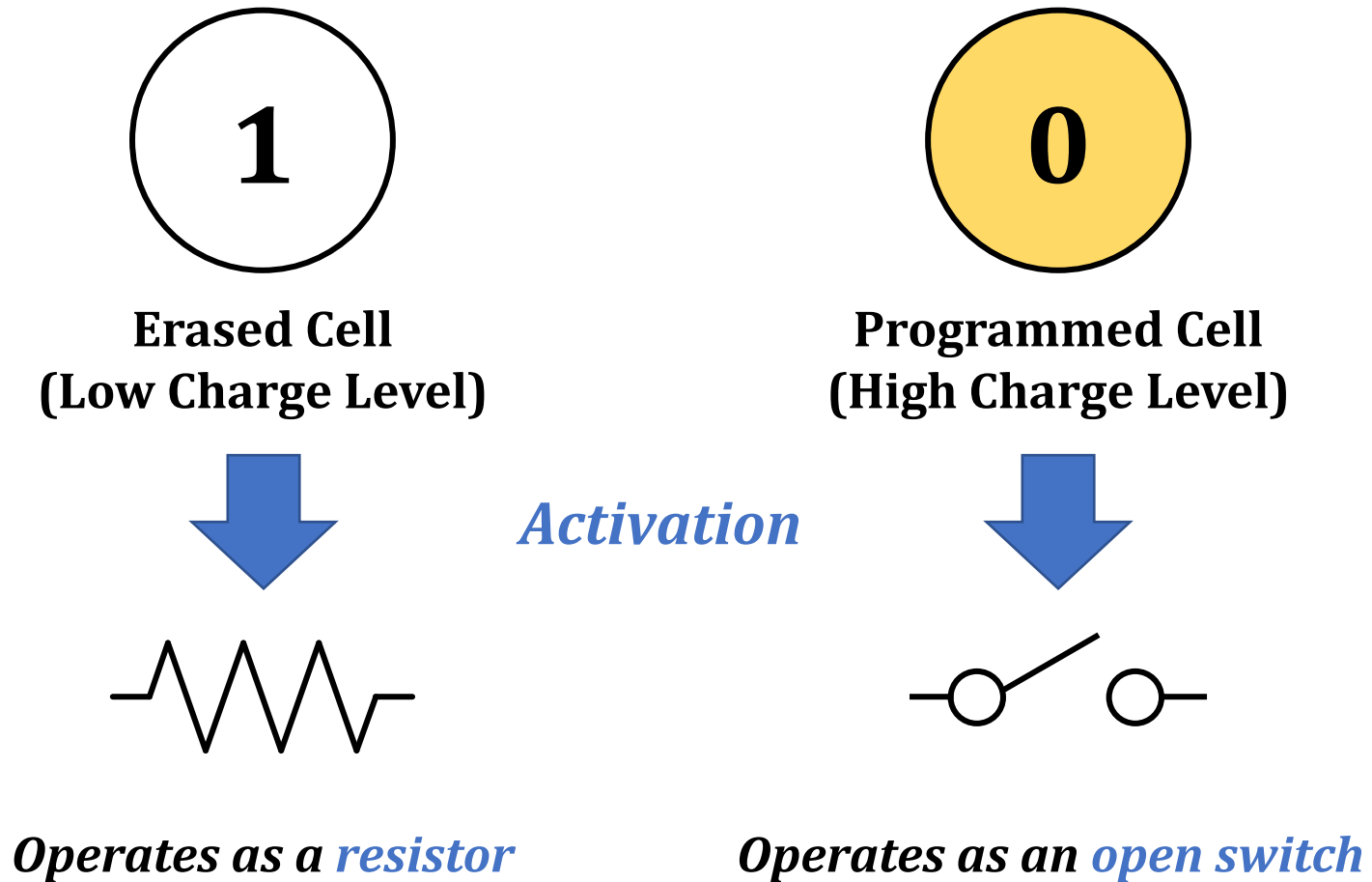
# Talk Outline

---

- Problem, Goals & Key Idea
- Background
- Flash-Cosmos: Computation with One-Shot Multi-Operand Sensing
- Evaluation
- Summary

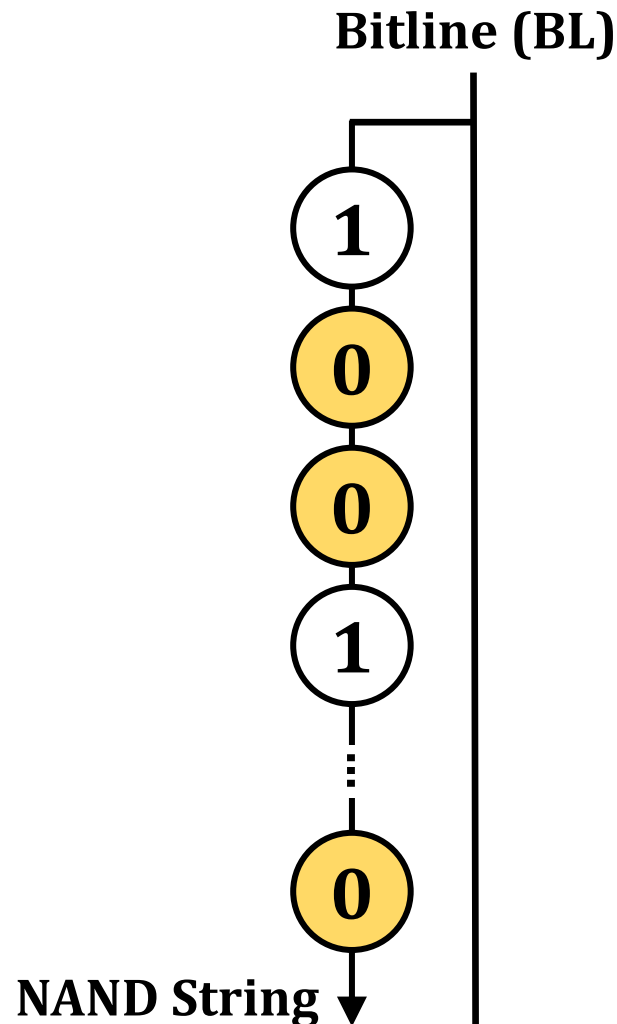
# NAND Flash Basics: A Flash Cell

- A flash cell stores data by adjusting the **amount of charge** in the cell



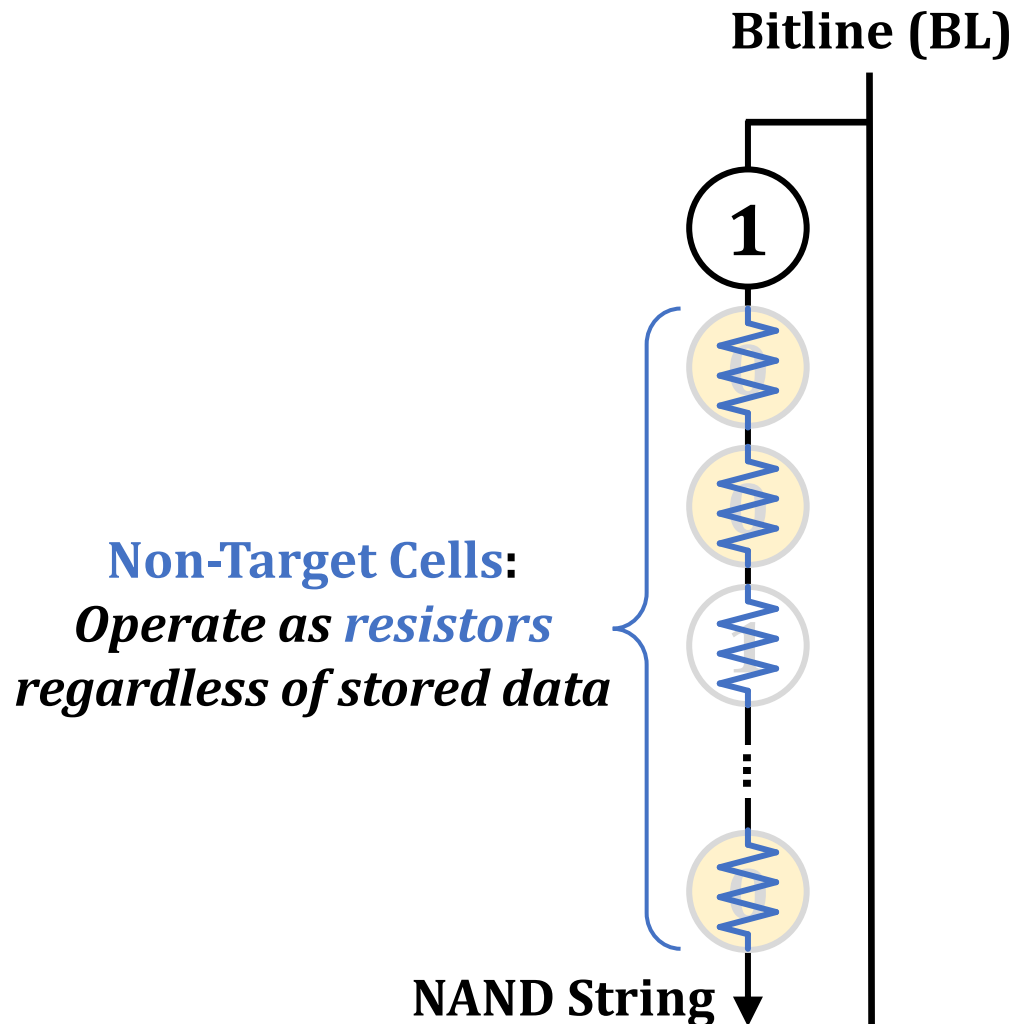
# NAND Flash Basics: A NAND String

- A set of flash cells are **serially connected**, forming a **NAND string**



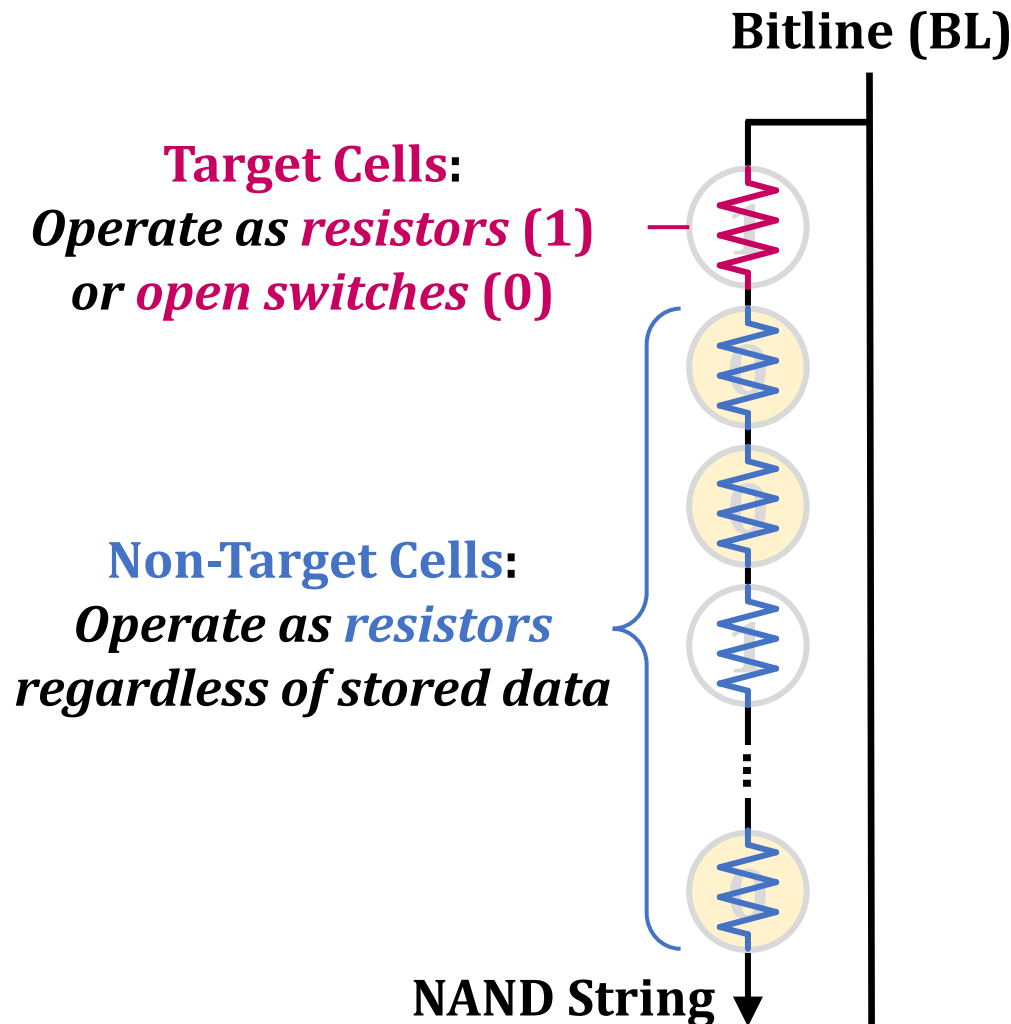
# NAND Flash Basics: Read Mechanism (1/3)

- NAND flash memory reads data by [checking the bitline current](#)



# NAND Flash Basics: Read Mechanism (2/3)

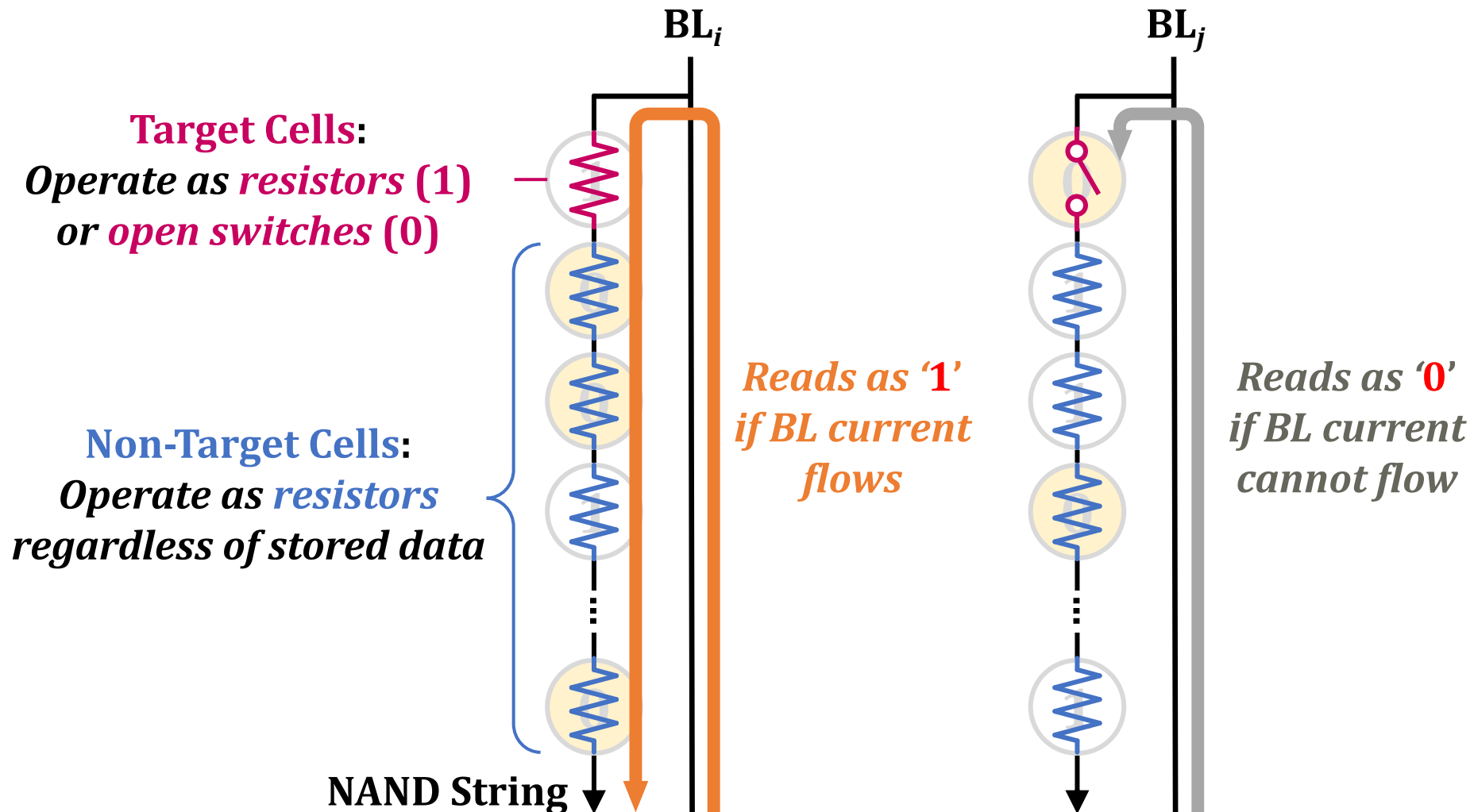
- NAND flash memory reads data by [checking the bitline current](#)





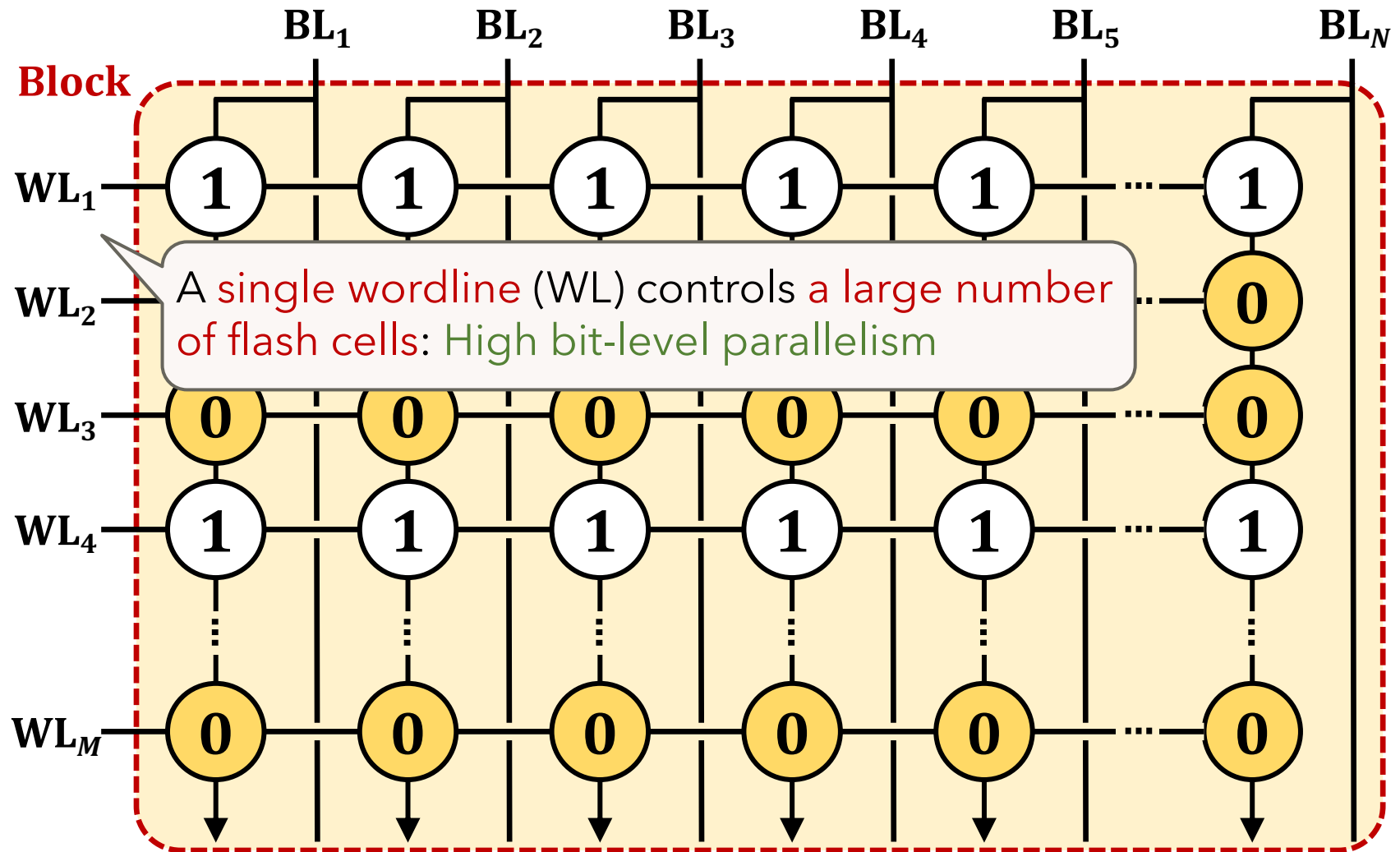
# NAND Flash Basics: Read Mechanism (3/3)

- NAND flash memory reads data by **checking the bitline current**



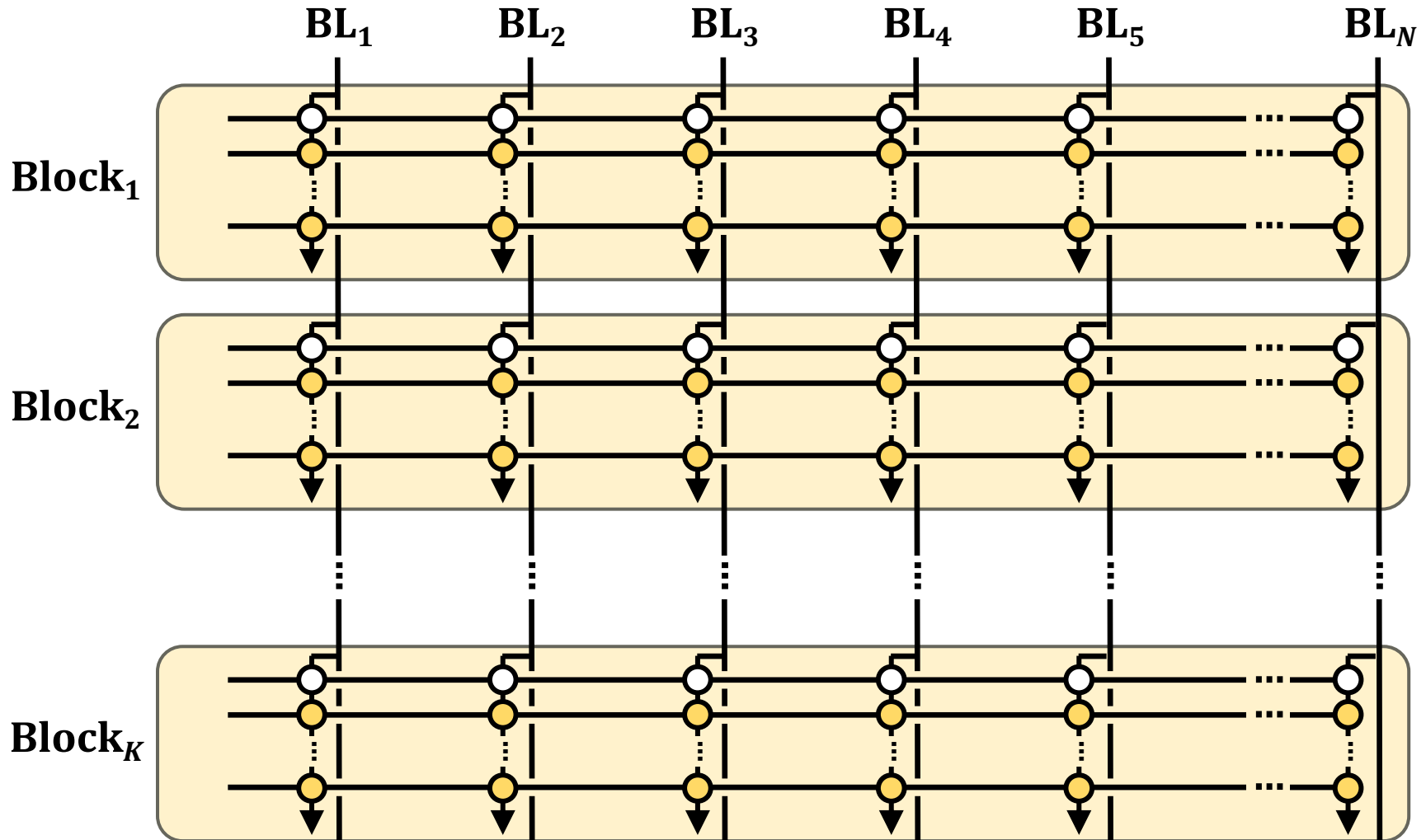
# NAND Flash Basics: A NAND Flash Block

- NAND strings connected to different bitlines comprise a **block**



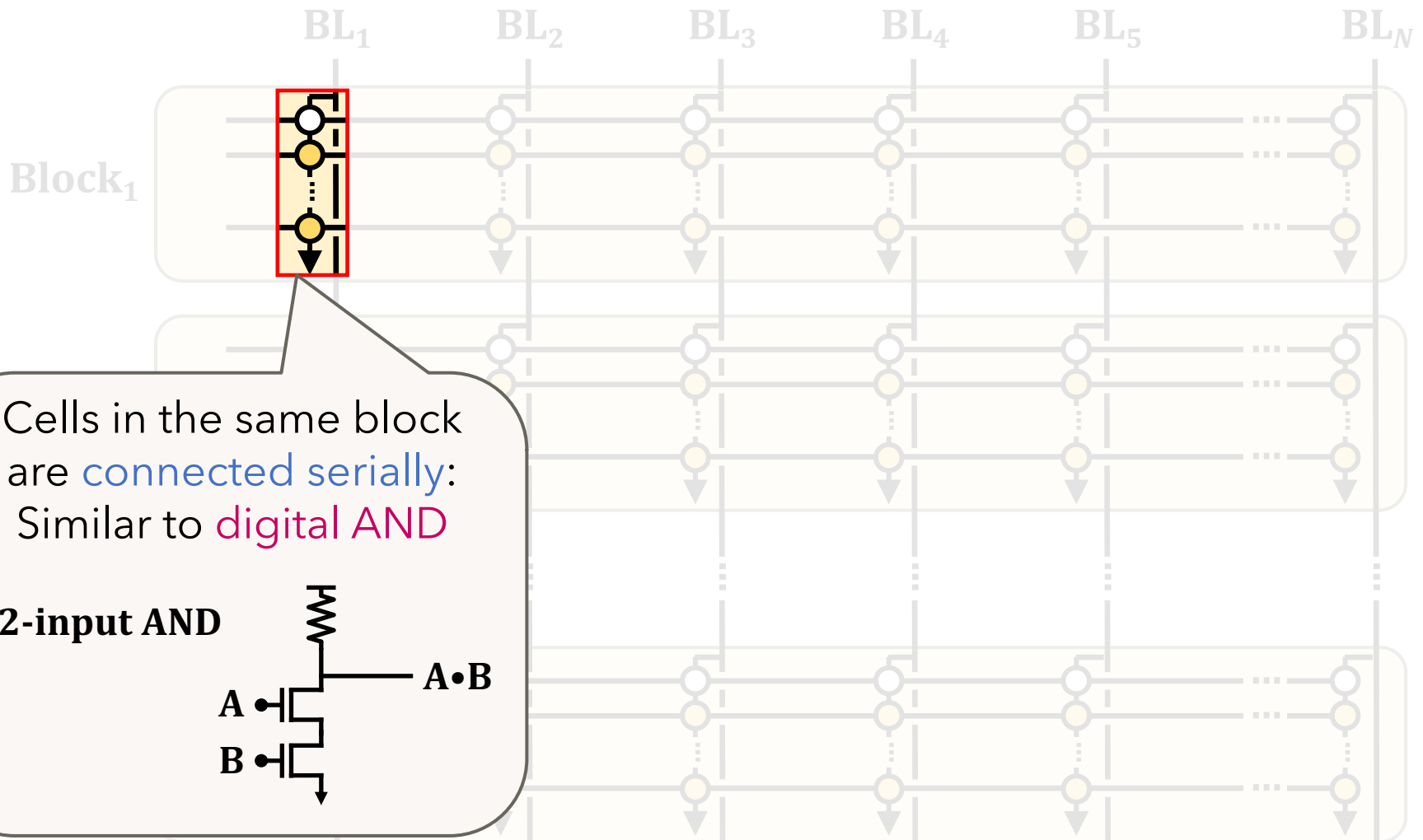
# NAND Flash Basics: Block Organization

- A large number of blocks share the same bitlines



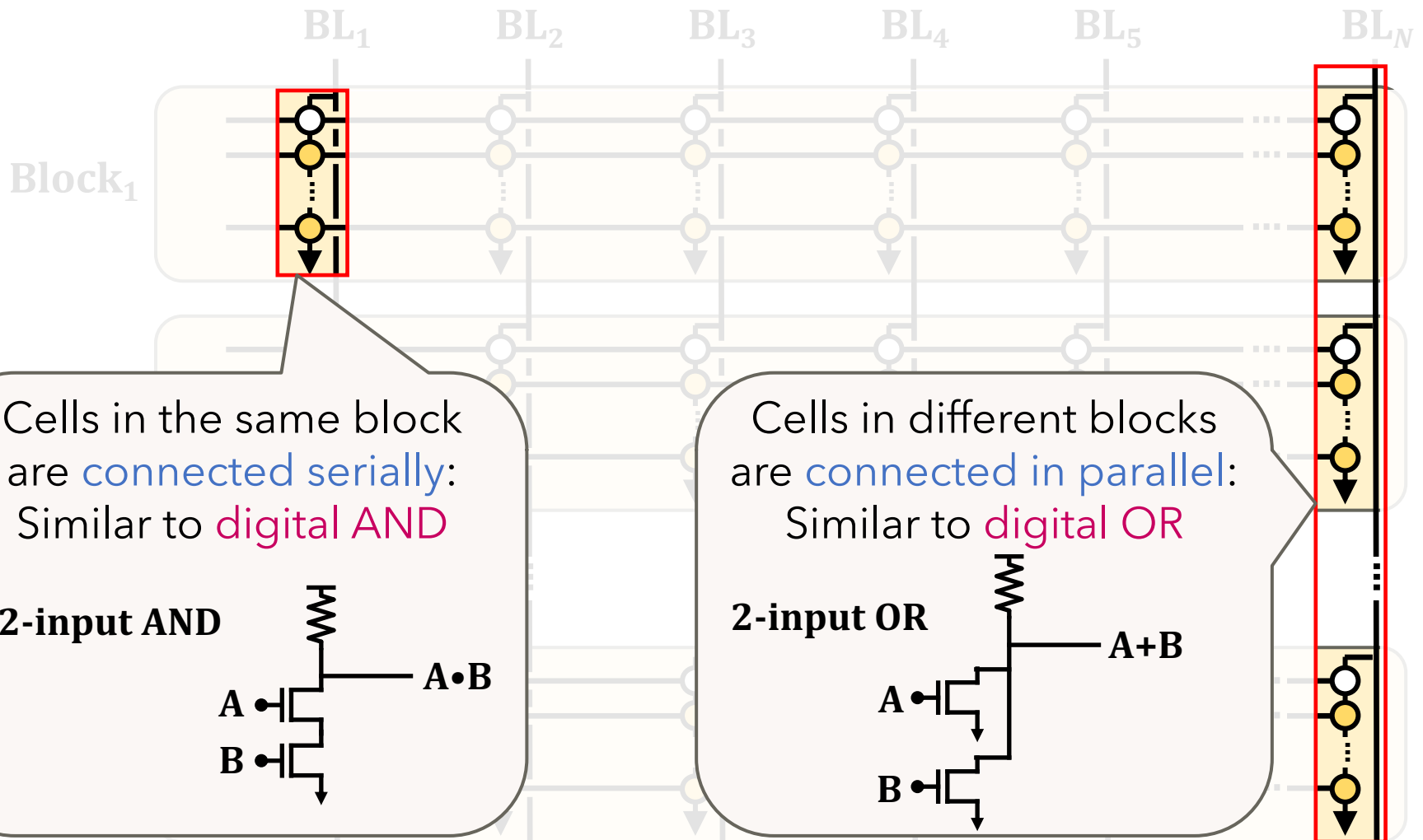
# Similarity to Digital Logic Gates

- A large number of blocks share the same bitlines



# Similarity to Digital Logic Gates

- A large number of blocks share the same bitlines.



# Talk Outline

---

- Problem, Goals & Key Idea
- Background
- Flash-Cosmos: Computation with One-Shot Multi-Operand Sensing
- Evaluation
- Summary

# Key Ideas

---



## **Multi-Wordline Sensing (MWS)**

to enable in-flash bulk bitwise operations via a single sensing operation



## **Enhanced SLC-Mode Programming (ESP)**

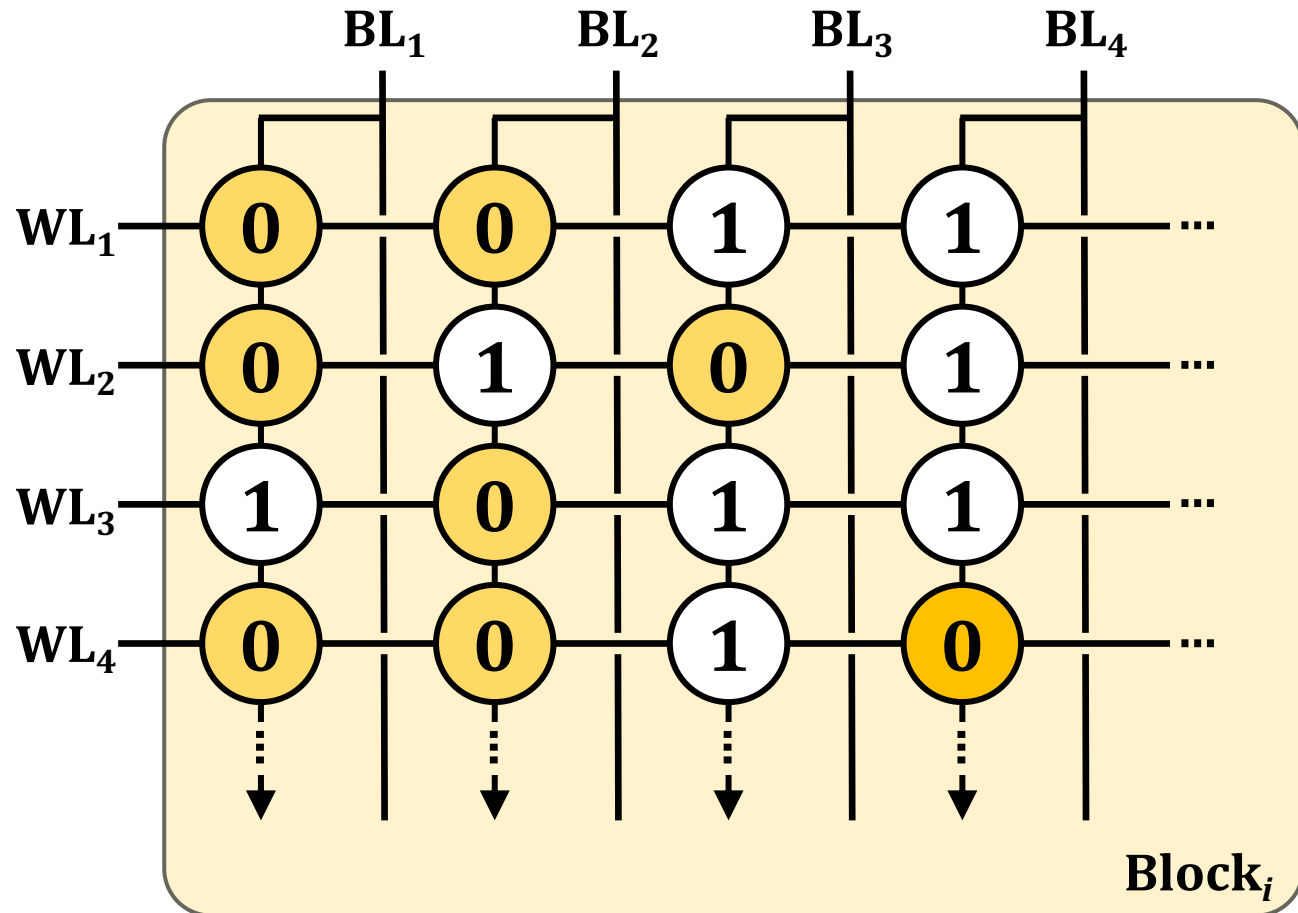
to eliminate raw bit errors in stored data (and thus in computation results)

# Multi-Wordline Sensing (MWS): Bitwise AND

- **Intra-Block MWS:**

Simultaneously activates multiple WLs in the same block

→ Bitwise AND of the stored data in the WLs



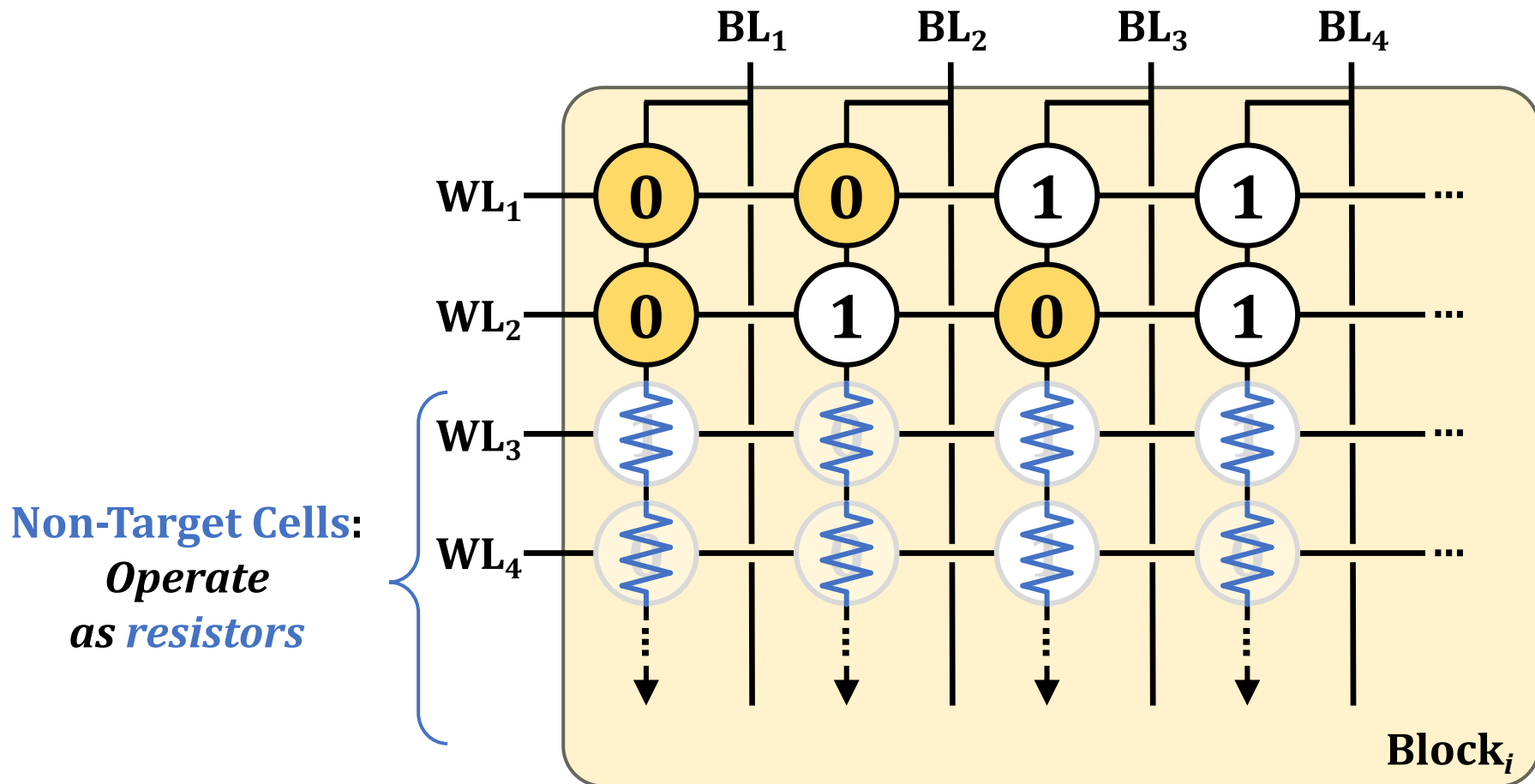


# Multi-Wordline Sensing (MWS): Bitwise AND

## ■ Intra-Block MWS:

Simultaneously activates multiple WLs in the same block

→ Bitwise AND of the stored data in the WLs

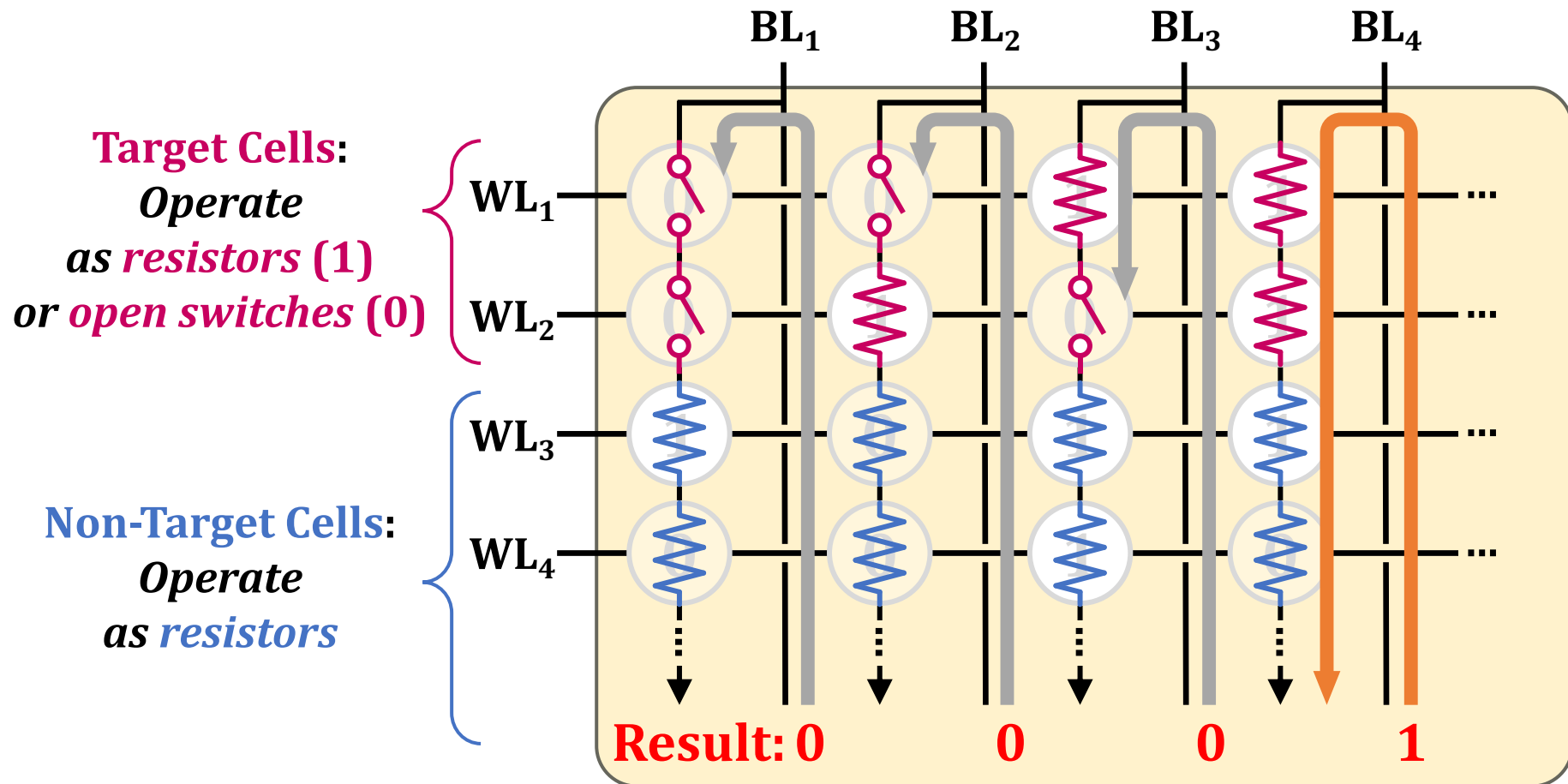


# Multi-Wordline Sensing (MWS): Bitwise AND

## ■ Intra-Block MWS:

Simultaneously activates multiple WLs in the same block

→ Bitwise AND of the stored data in the WLs



# Multi-Wordline Sensing (MWS): Bitwise AND

## ■ Intra-Block MWS:

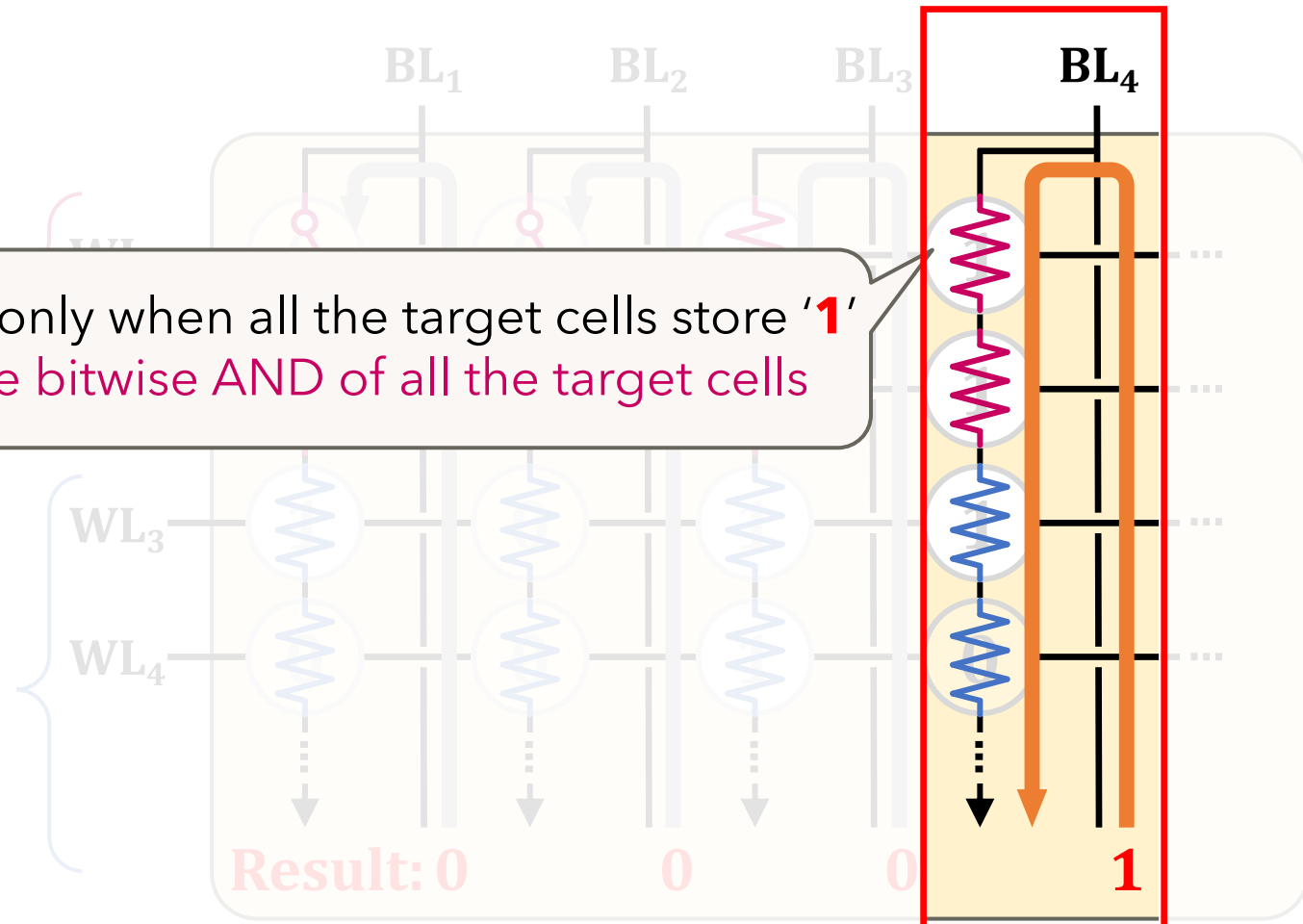
Simultaneously activates multiple WLs in the same block

→ Bitwise AND of the stored data in the WLs

Target Cell:

A bitline reads as '1' only when all the target cells store '1'  
→ Equivalent to the bitwise AND of all the target cells

Non-Target Cell:  
*Operate  
as a resistance*

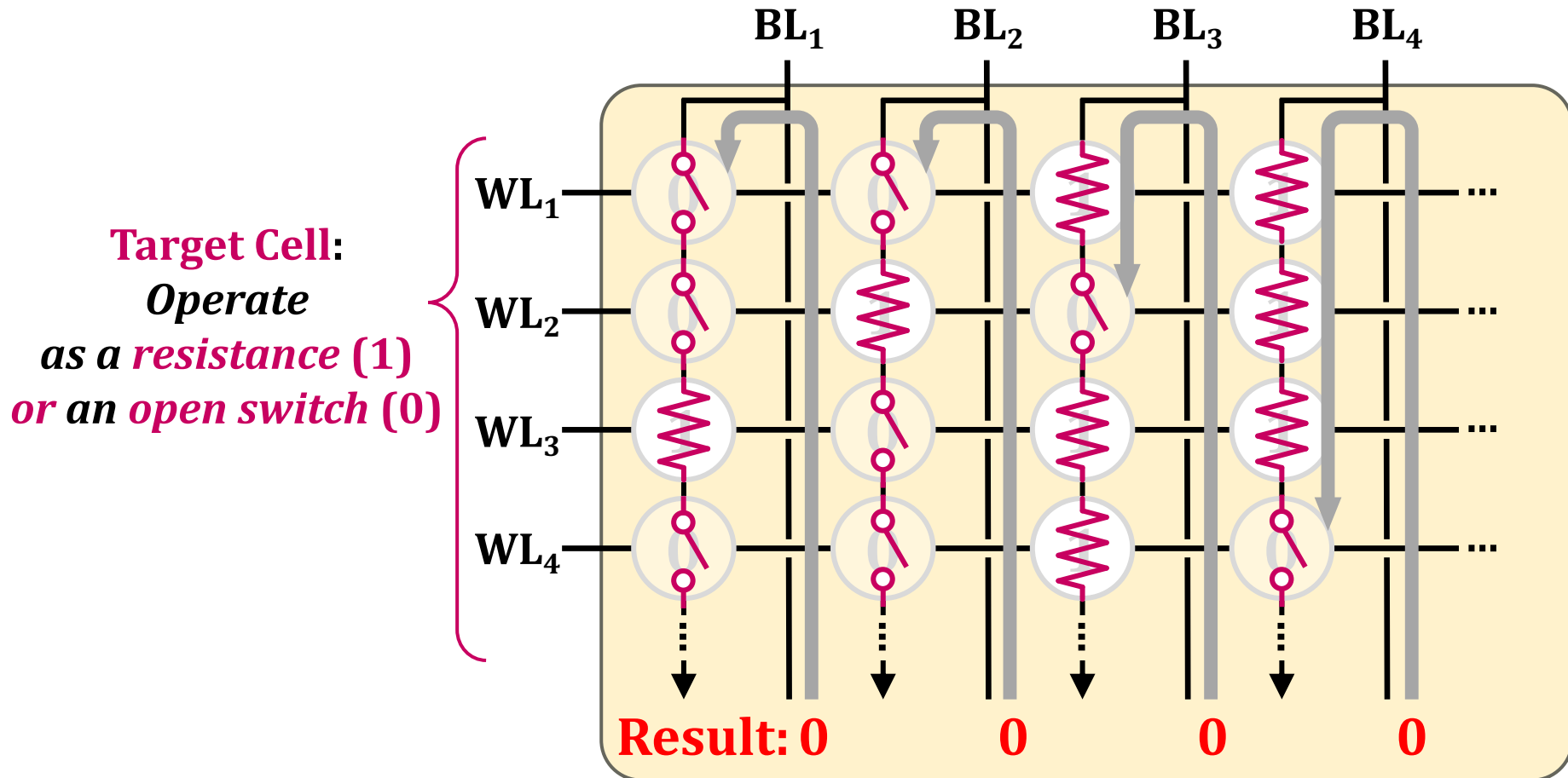


# Multi-Wordline Sensing (MWS): Bitwise AND

- **Intra-Block MWS:**

Simultaneously activates multiple WLs in the same block

→ Bitwise AND of the stored data in the WLs



# Multi-Wordline Sensing (MWS): Bitwise AND

## ■ Intra-Block MWS:

Simultaneously activates multiple WLs in the same block

→ Bitwise AND of the stored data in the WLs

A bitline reads as '**1**' only when all the target cells store '**1**'  
→ Equivalent to the bitwise AND of all the target cells

*Operate  
as a resistance (1)  
or an open switch (0)*

WL<sub>2</sub>  
WL<sub>3</sub>  
WL<sub>4</sub>

Result: 0

0

0

0

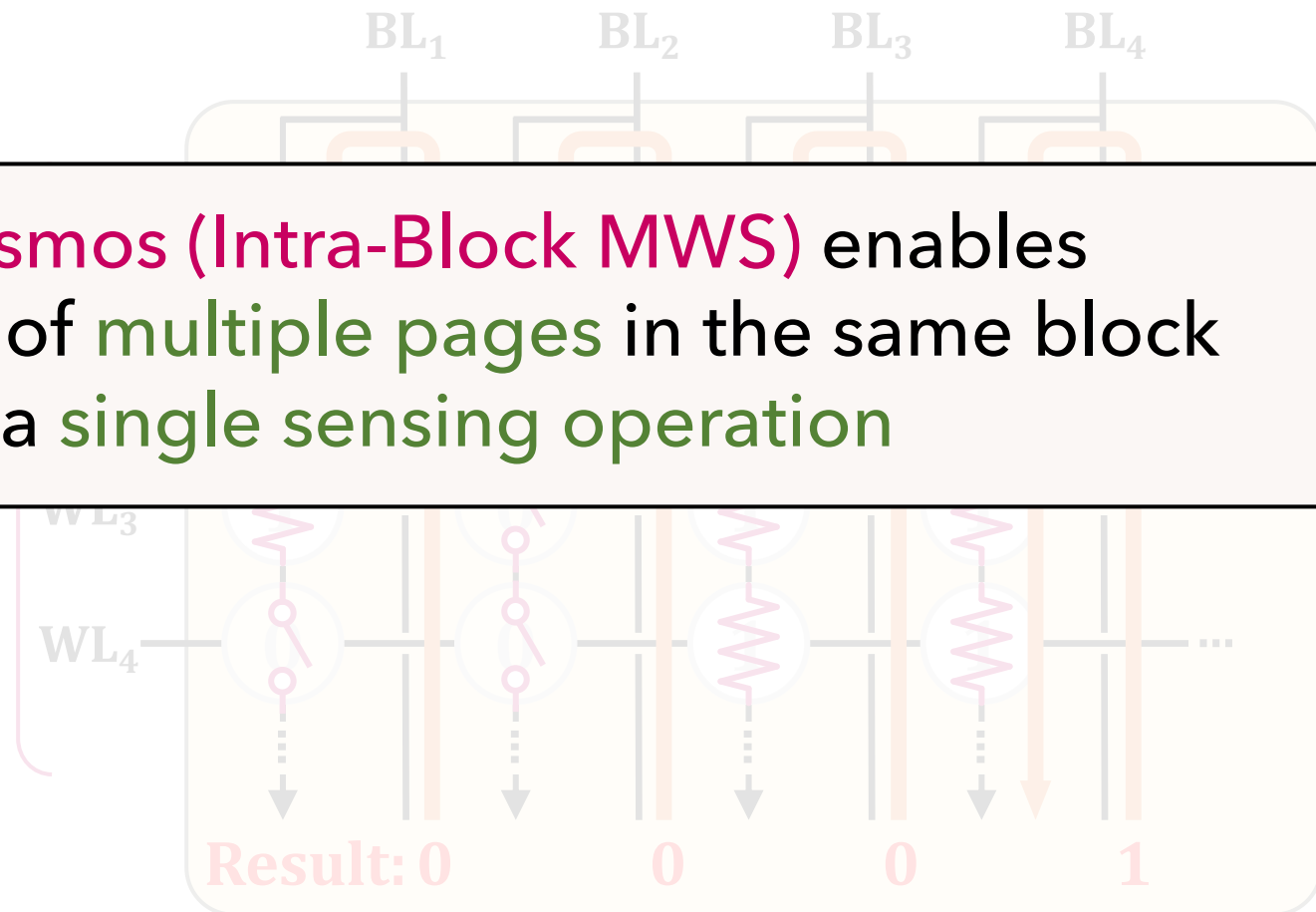
# Multi-Wordline Sensing (MWS): Bitwise AND

- Intra-Block MWS:

Simultaneously activates multiple WLs in the same block

→ Bitwise AND of the stored data in the WLs

**Flash-Cosmos (Intra-Block MWS)** enables bitwise AND of multiple pages in the same block via a single sensing operation

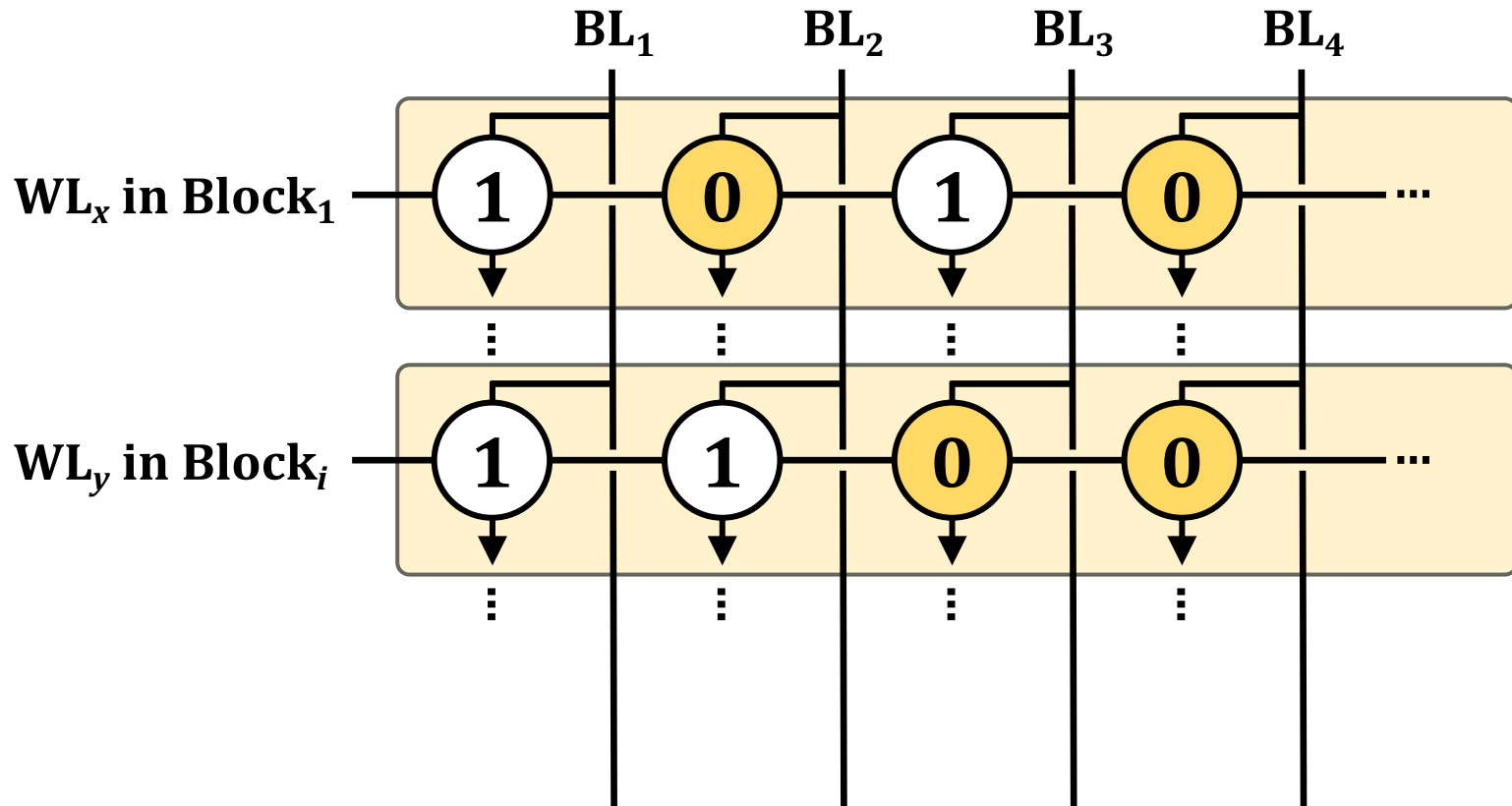


# Multi-Wordline Sensing (MWS): Bitwise OR

## ■ Inter-Block MWS:

Simultaneously activates multiple WLs in different blocks

→ Bitwise OR of the stored data in the WLs

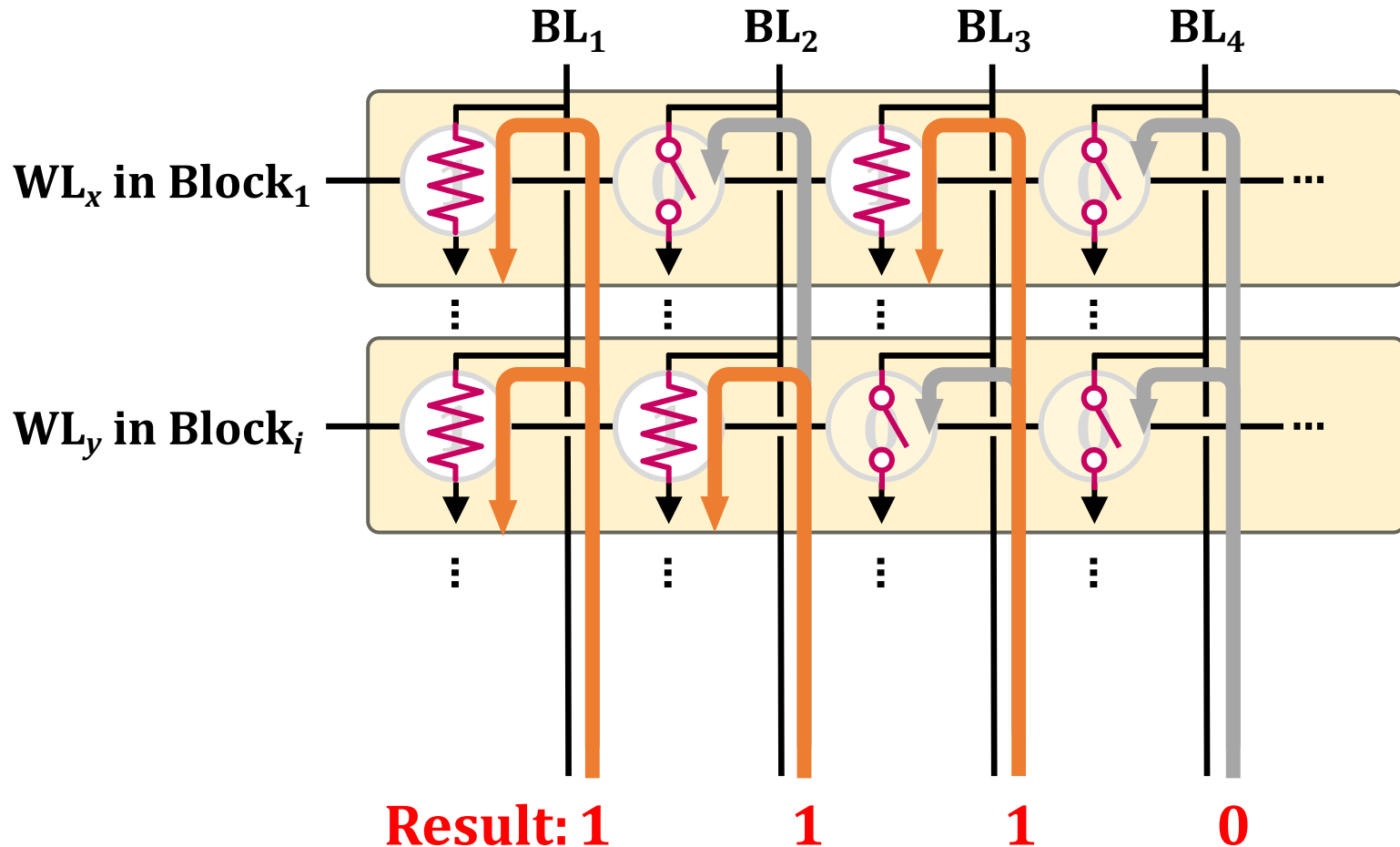


# Multi-Wordline Sensing (MWS): Bitwise OR

- **Inter-Block MWS:**

Simultaneously activates multiple WLs in different blocks

→ Bitwise OR of the stored data in the WLs



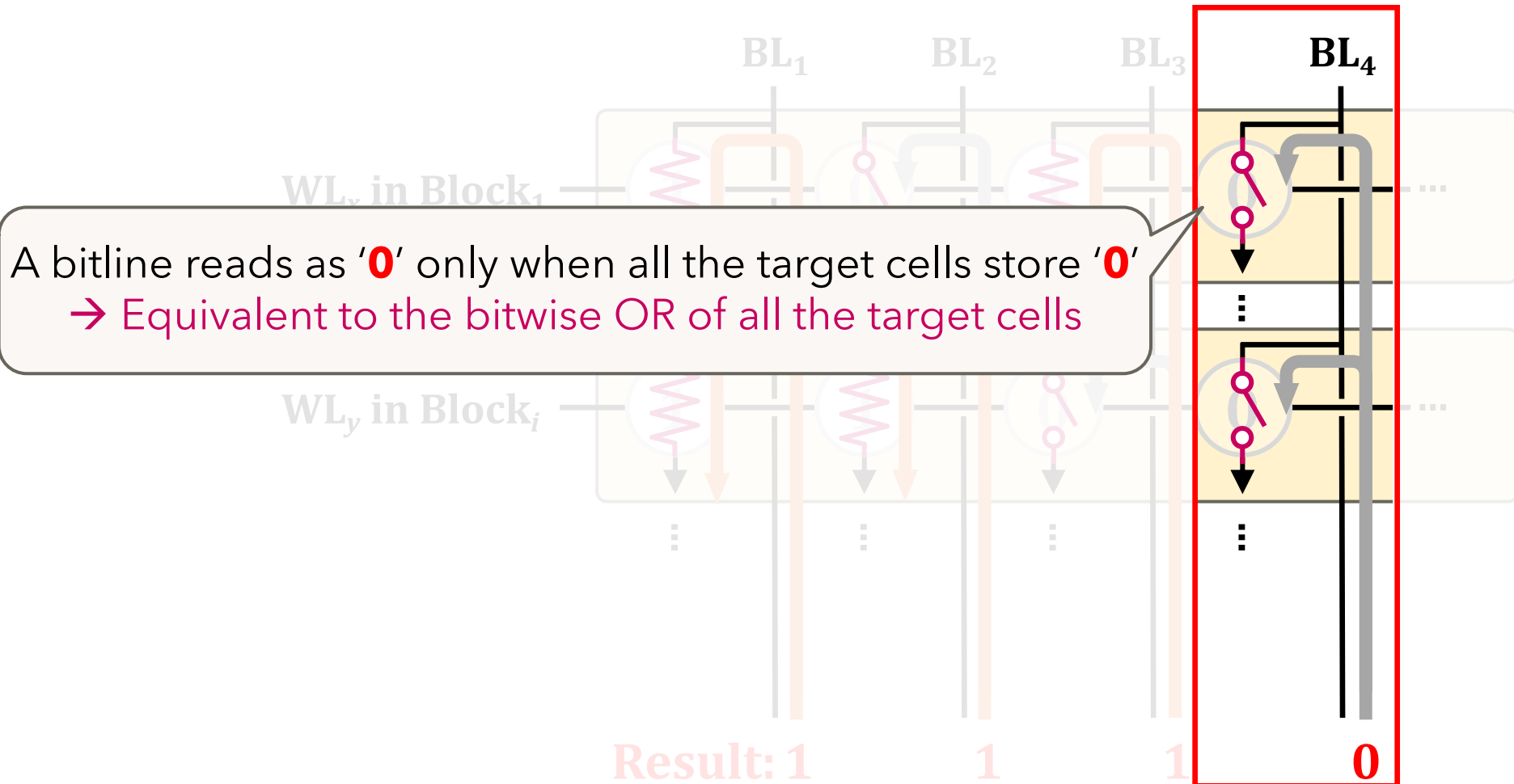


# Multi-Wordline Sensing (MWS): Bitwise OR

## ■ Inter-Block MWS:

Simultaneously activates multiple WLs in different blocks

→ Bitwise OR of the stored data in the WLs

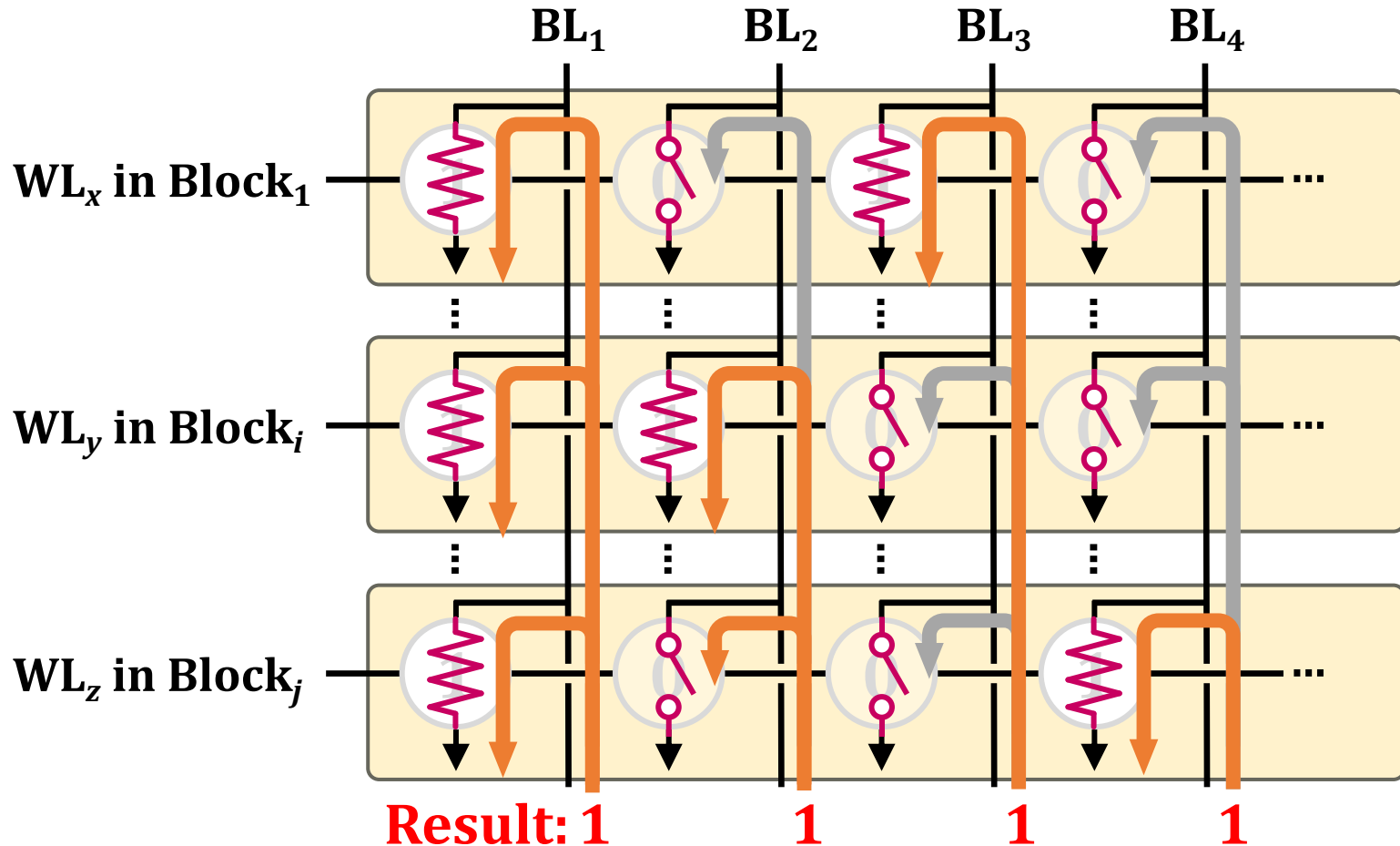


# Multi-Wordline Sensing (MWS): Bitwise OR

- **Inter-Block MWS:**

Simultaneously activates multiple WLs in different blocks

→ Bitwise OR of the stored data in the WLs

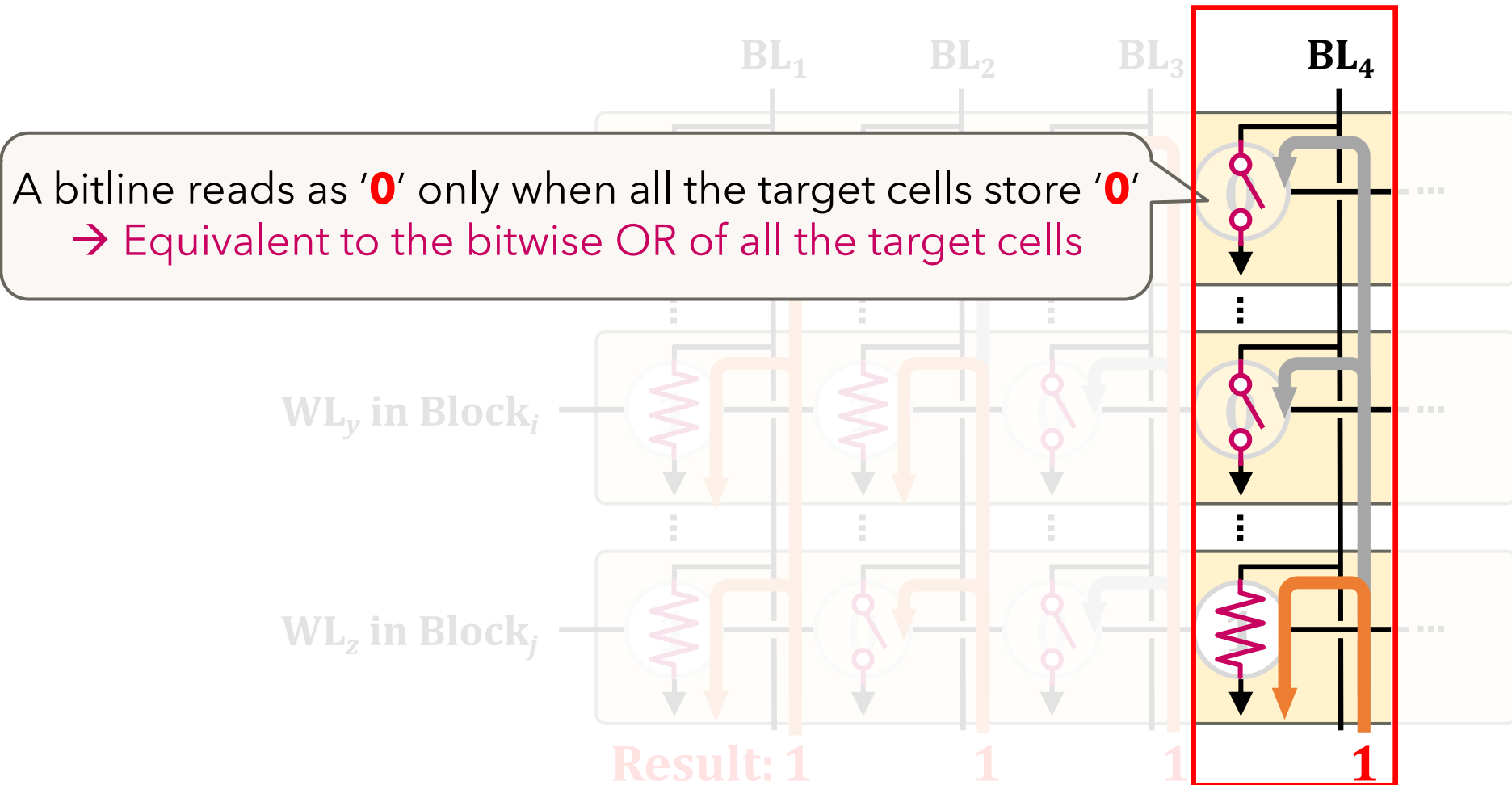


# Multi-Wordline Sensing (MWS): Bitwise OR

- Inter-Block MWS:

Simultaneously activates multiple WLs in different blocks

→ Bitwise OR of the stored data in the WLs



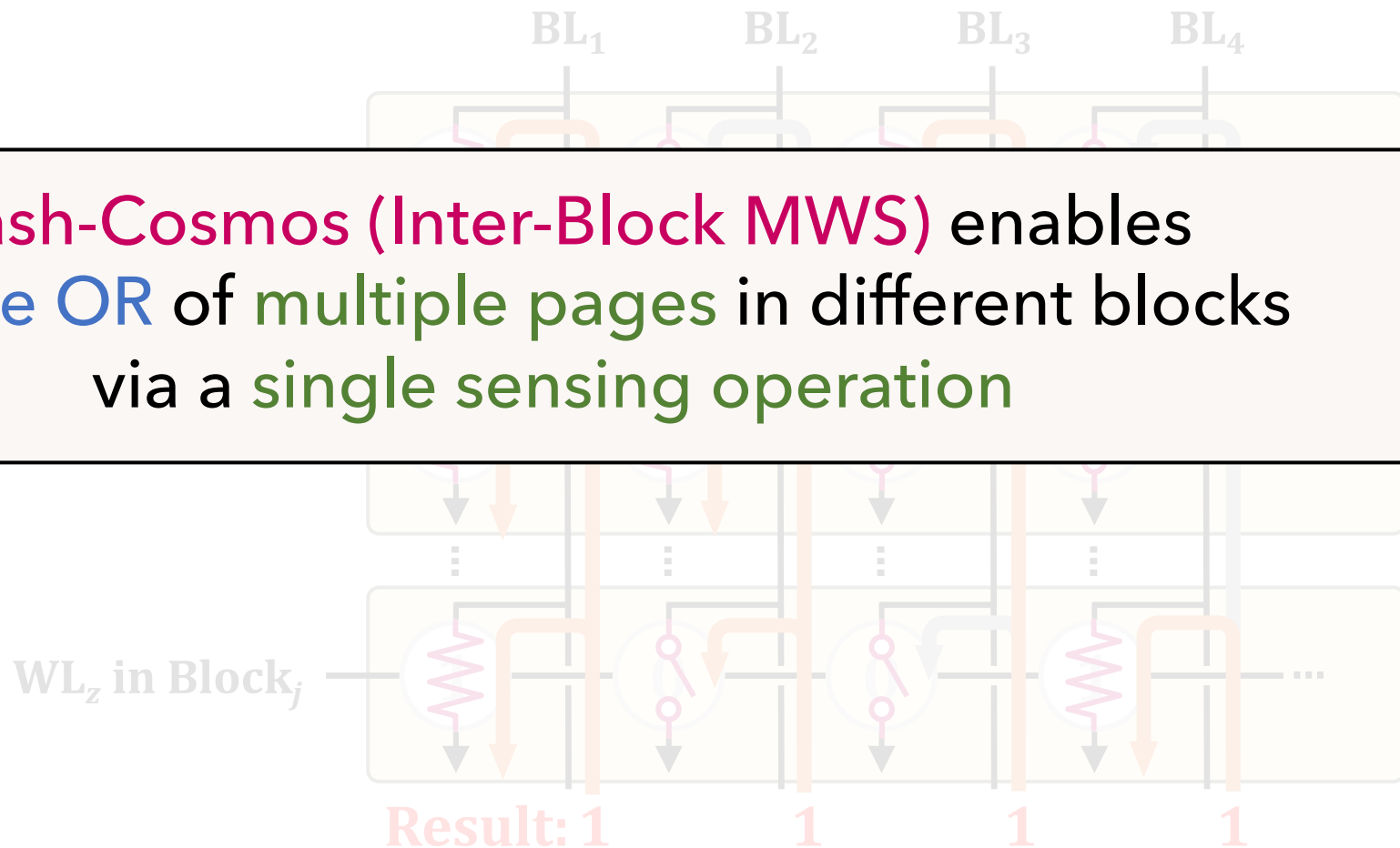
# Multi-Wordline Sensing (MWS): Bitwise OR

- **Inter-Block MWS:**

Simultaneously activates multiple WLs in different blocks

→ Bitwise OR of the stored data in the WLs

**Flash-Cosmos (Inter-Block MWS)** enables  
bitwise OR of multiple pages in different blocks  
via a single sensing operation



# Other Types of Bitwise Operations

**Flash-Cosmos** also enables  
other types of bitwise operations  
(NOT/NAND/NOR/XOR/XNOR)  
leveraging **existing features** of NAND flash memory

## Flash-Cosmos: In-Flash Bulk Bitwise Operations Using Inherent Computation Capability of NAND Flash Memory

Jisung Park<sup>§∇</sup> Roknoddin Azizi<sup>§</sup> Geraldo F. Oliveira<sup>§</sup> Mohammad Sadrosadati<sup>§</sup>  
Rakesh Nadig<sup>§</sup> David Novo<sup>†</sup> Juan Gómez-Luna<sup>§</sup> Myungsuk Kim<sup>‡</sup> Onur Mutlu<sup>§</sup>

<sup>§</sup>*ETH Zürich*    <sup>∇</sup>*POSTECH*    <sup>†</sup>*LIRMM, Univ. Montpellier, CNRS*    <sup>‡</sup>*Kyungpook National University*



<https://arxiv.org/abs/2209.05566.pdf>

# Key Ideas



## Multi-Wordline Sensing (MWS)

to enable in-flash bulk bitwise operations via a single sensing operation



## Enhanced SLC-Mode Programming (ESP)

to eliminate raw bit errors in stored data (and thus in computation results)

# Enhanced SLC-Mode Programming (ESP)

- **Goal:** eliminate raw bit errors in stored data (and computation results)
- **Key ideas**
  - Programs only a single bit per cell (SLC-mode programming)
    - Trades storage density for reliable computation
  - Performs more precise programming of the cells
    - Trades programming latency for reliable computation

Maximizes the reliability margin  
between the different states of flash cells

# Enhanced SLC-Mode Programming (ESP)

- To eliminate raw bit errors in stored data (and computation results)

Flash-Cosmos (ESP) enables  
reliable in-flash computation  
by trading storage density & programming latency

Storage & latency overheads affect  
only data used in in-flash computation



# Talk Outline

---

- Problem, Goals & Key Idea
- Background
- Flash-Cosmos: Computation with One-Shot Multi-Operand Sensing
- Evaluation
- Summary

# Evaluation Methodology

## ▪ Real-device characterization

- To validate the feasibility and reliability of Flash-Cosmos
- Using 160 48-WL-layer 3D Triple-Level Cell NAND flash chips
  - 3,686,400 tested wordlines
- Under worst-case operating conditions
  - Under a 1-year retention time at 10K P/E cycles
  - Worst-case data patterns

## ▪ System-level evaluation

- Using the state-of-the-art SSD simulator (MQSim [Tavakkol+, FAST'18])
- Three real-world applications
  - Bitmap Indices (BMI): Bitwise AND of up to ~1,000 operands
  - Image Segmentation (IMS): Bitwise AND of 3 operands
  - K-clique Star Listing (KCS): Bitwise OR of up to 32 operands
- Baselines
  - Outside-Storage Processing (OSP): A multi-core CPU (Intel i7-11700K)
  - In-Storage Processing (ISP): An in-storage hardware accelerator
  - ParaBit [Gao+, MICRO'21]: State-of-the-art in-flash processing mechanism

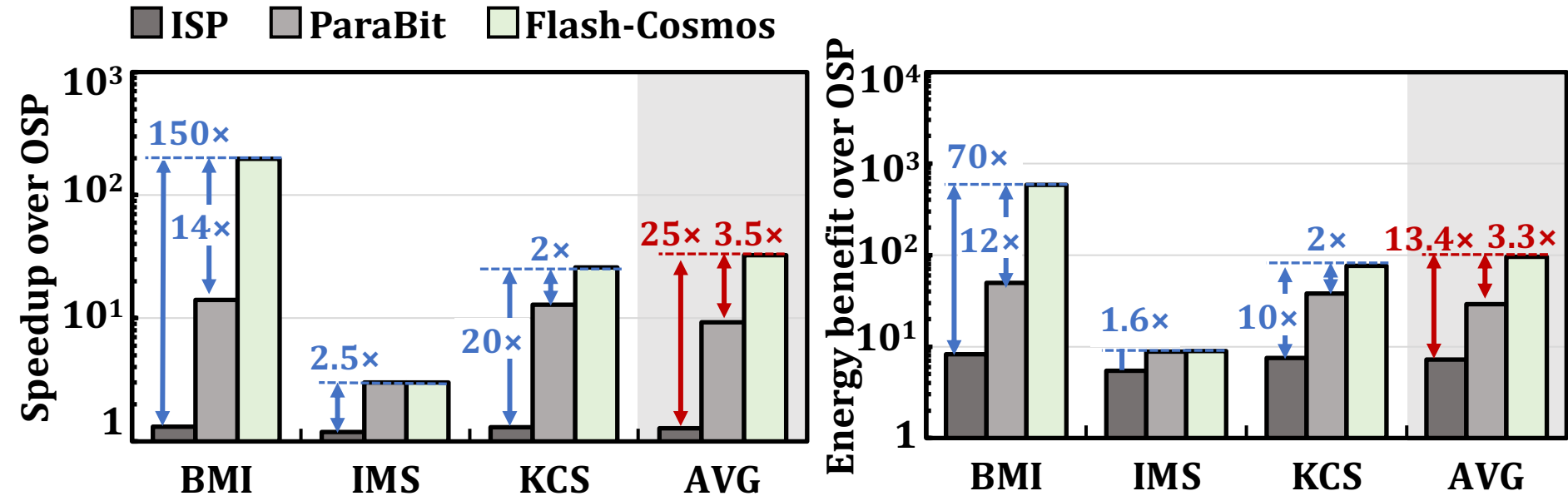
# Results: Real-Device Characterization

Both intra- and inter-block MWS operations require no changes to the cell array of commodity NAND flash chips

Both MWS operations can activate multiple WLs (intra: up to 48, inter: up to 4) at the same time with small increase in sensing latency ( $< 10\%$ )

ESP significantly improves the reliability of computation results (no observed bit error in the tested flash cells)

# Results: Performance & Energy



Flash-Cosmos provides **significant performance & energy benefits** over all the baselines

The larger the number of operands,  
the higher the performance & energy benefits

# More in the Paper

---

- Support for other types of bitwise operations
  - NOT/NAND/NOR/XOR/XNOR
- More detailed real-device characterization results
- Optimizations to improve bitwise operation performance
- Flash-Cosmos command interface
- System support
- Overhead analysis

## Flash-Cosmos: In-Flash Bulk Bitwise Operations Using Inherent Computation Capability of NAND Flash Memory

Jisung Park<sup>§∇</sup> Roknoddin Azizi<sup>§</sup> Geraldo F. Oliveira<sup>§</sup> Mohammad Sadrosadati<sup>§</sup>  
Rakesh Nadig<sup>§</sup> David Novo<sup>†</sup> Juan Gómez-Luna<sup>§</sup> Myungsuk Kim<sup>‡</sup> Onur Mutlu<sup>§</sup>

<sup>§</sup>*ETH Zürich*   <sup>∇</sup>*POSTECH*   <sup>†</sup>*LIRMM, Univ. Montpellier, CNRS*   <sup>‡</sup>*Kyungpook National University*



<https://arxiv.org/abs/2209.05566.pdf>

# Summary: Flash-Cosmos



The first work that enables  
in-flash multi-operand bulk bitwise operations  
with a single sensing operation and high reliability



Improves performance  
by 32x/25x/3.5x over OSP/ISP/ParaBit



Improves energy efficiency  
by 95x/13.4x/3.3x over OSP/ISP/ParaBit



Low-cost & requires no changes to flash cell arrays

# Flash-Cosmos

## In-Flash Bulk Bitwise Operations Using Inherent Computation Capability of NAND Flash Memory

Jisung Park, Roknoddin Azizi, Geraldo F. Oliveira,  
Mohammad Sadrosadati, Rakesh Nadig, David Novo,  
Juan Gómez Luna, Myungsuk Kim, and Onur Mutlu

