Ambit

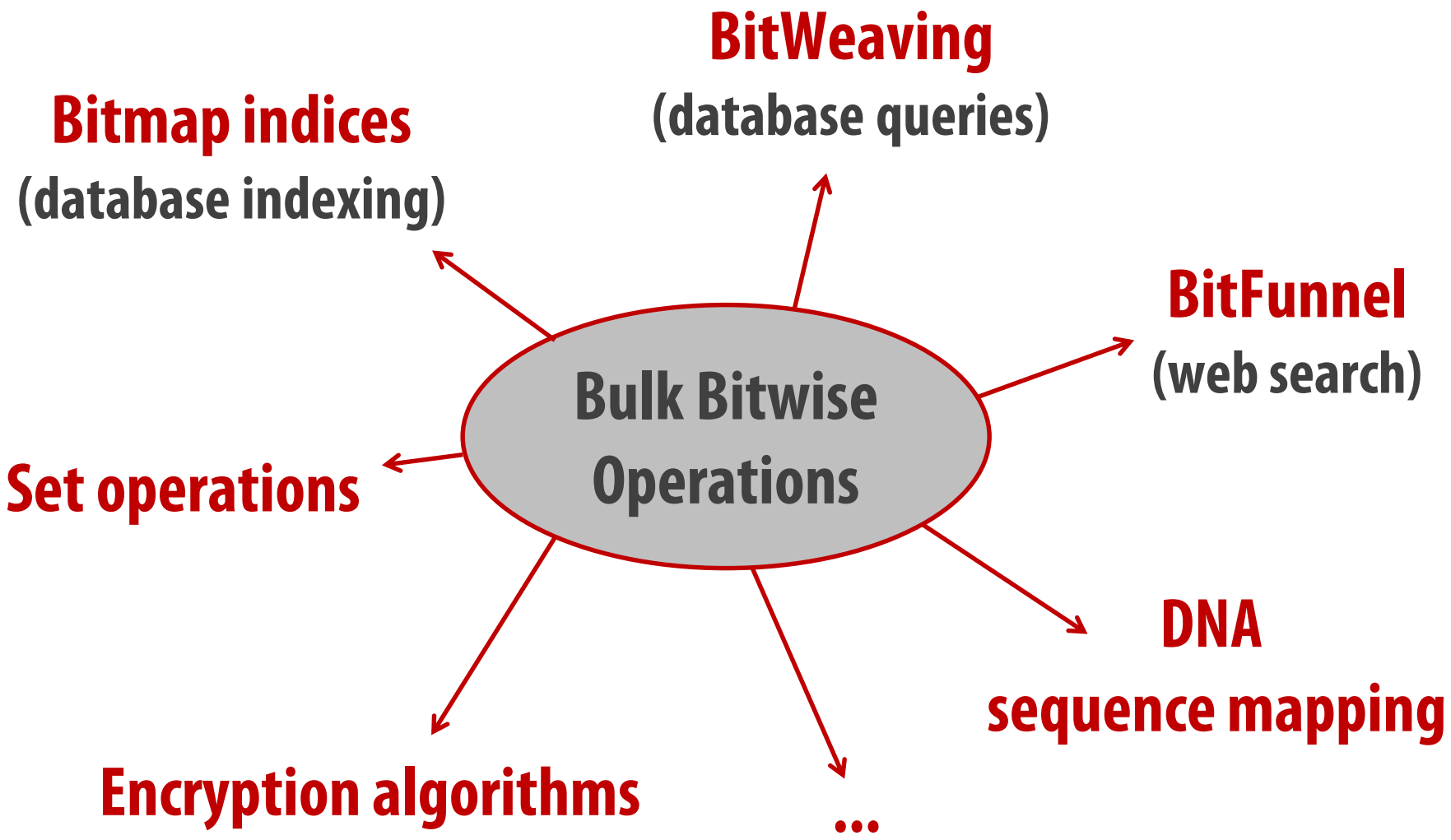
In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology

Vivek Seshadri

Donghyuk Lee, Thomas Mullins, Hasan Hassan, Amirali Boroumand, Jeremie Kim, Michael A. Kozuch, Onur Mutlu, Phillip B. Gibbons, Todd C. Mowry

Executive Summary

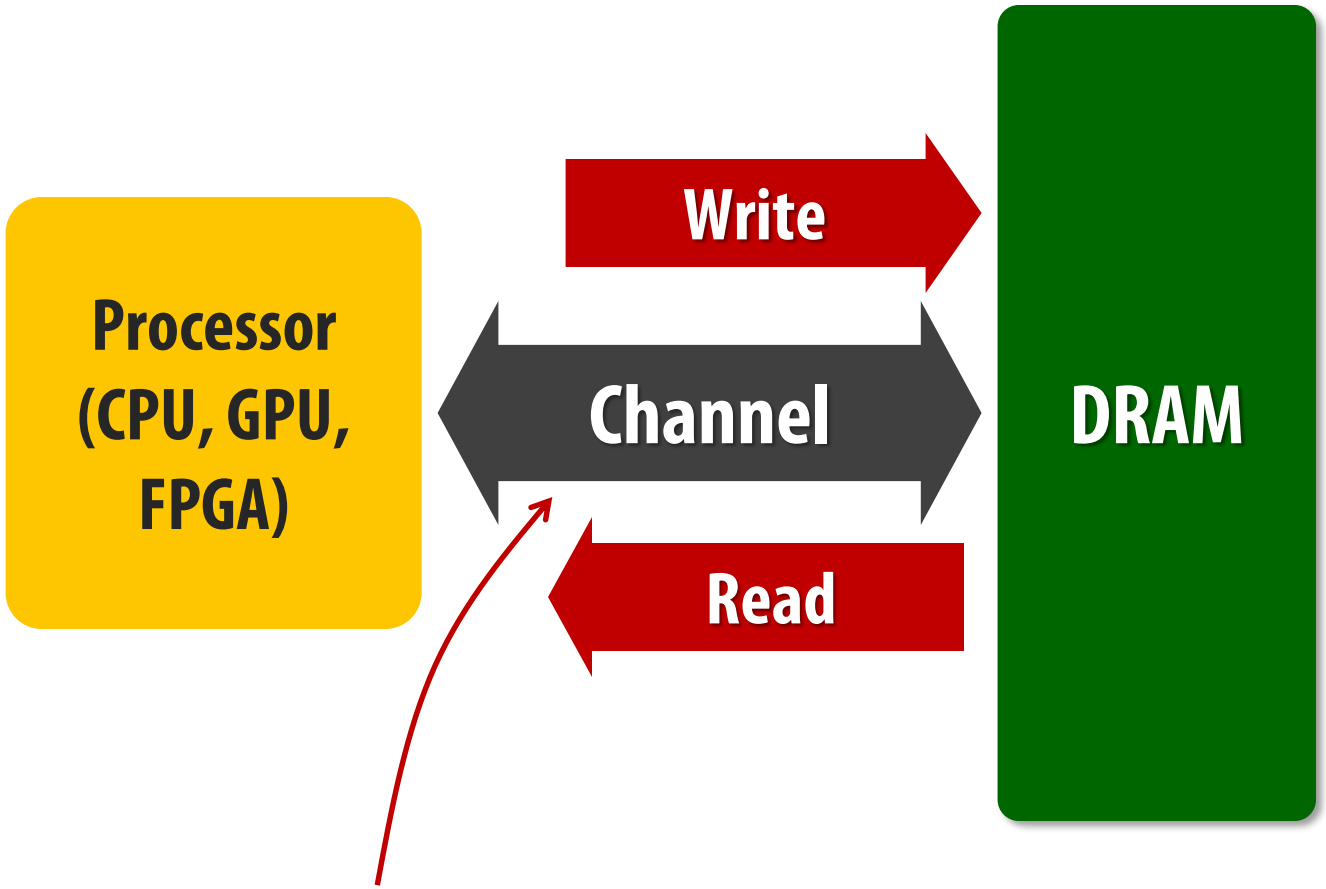
- **Problem: Bulk bitwise operations**
  - present in many applications, e.g., databases, search filters
  - existing systems are memory bandwidth limited
- **Our Proposal: Ambit**
  - perform bulk bitwise operations **completely inside DRAM**
  - **bulk bitwise AND/OR**: simultaneous activation of three rows
  - **bulk bitwise NOT**: inverters already in sense amplifiers
  - less than 1% area overhead over existing DRAM chips
- **Results compared to state-of-the-art baseline**
  - average across seven bulk bitwise operations
    - 32X performance improvement, 35X energy reduction
  - 3X-7X performance for real-world data-intensive applications



[1] Li and Patel, BitWeaving, SIGMOD 2013

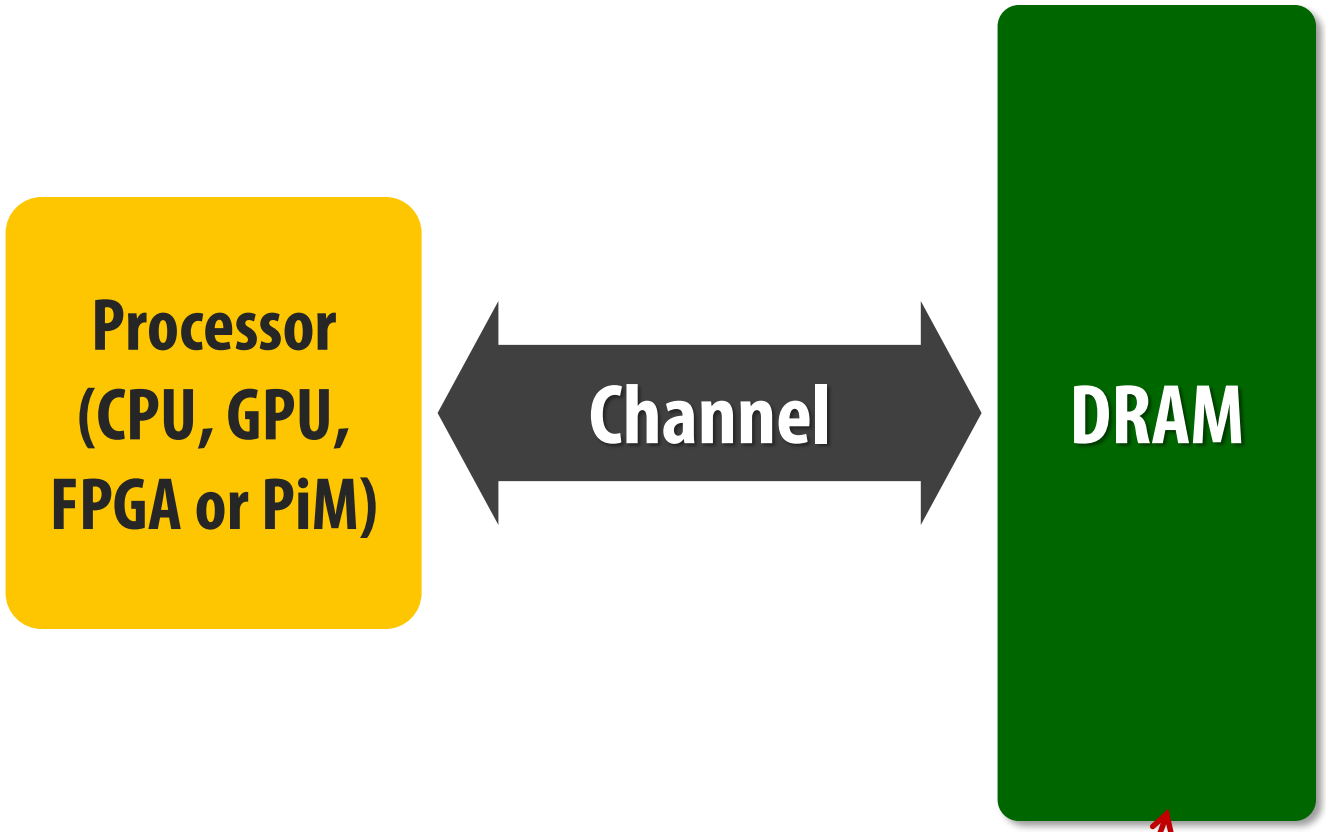
[2] Goodwin+, BitFunnel, SIGIR 2017

Today, DRAM is just a storage device!



Throughput of bulk bitwise operations limited by available memory bandwidth

Our Approach

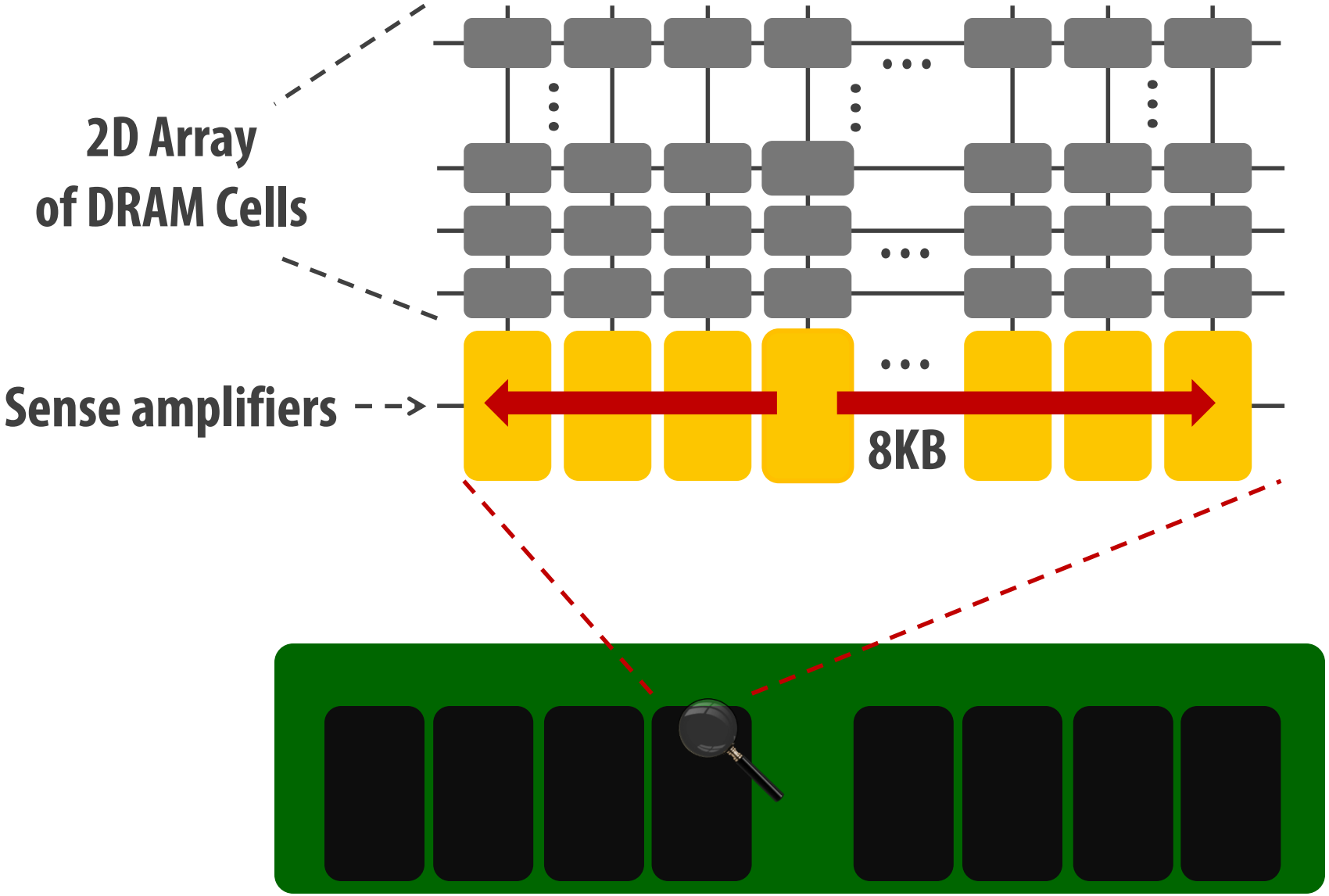


Use analog operation of DRAM to perform bitwise operations completely inside memory!

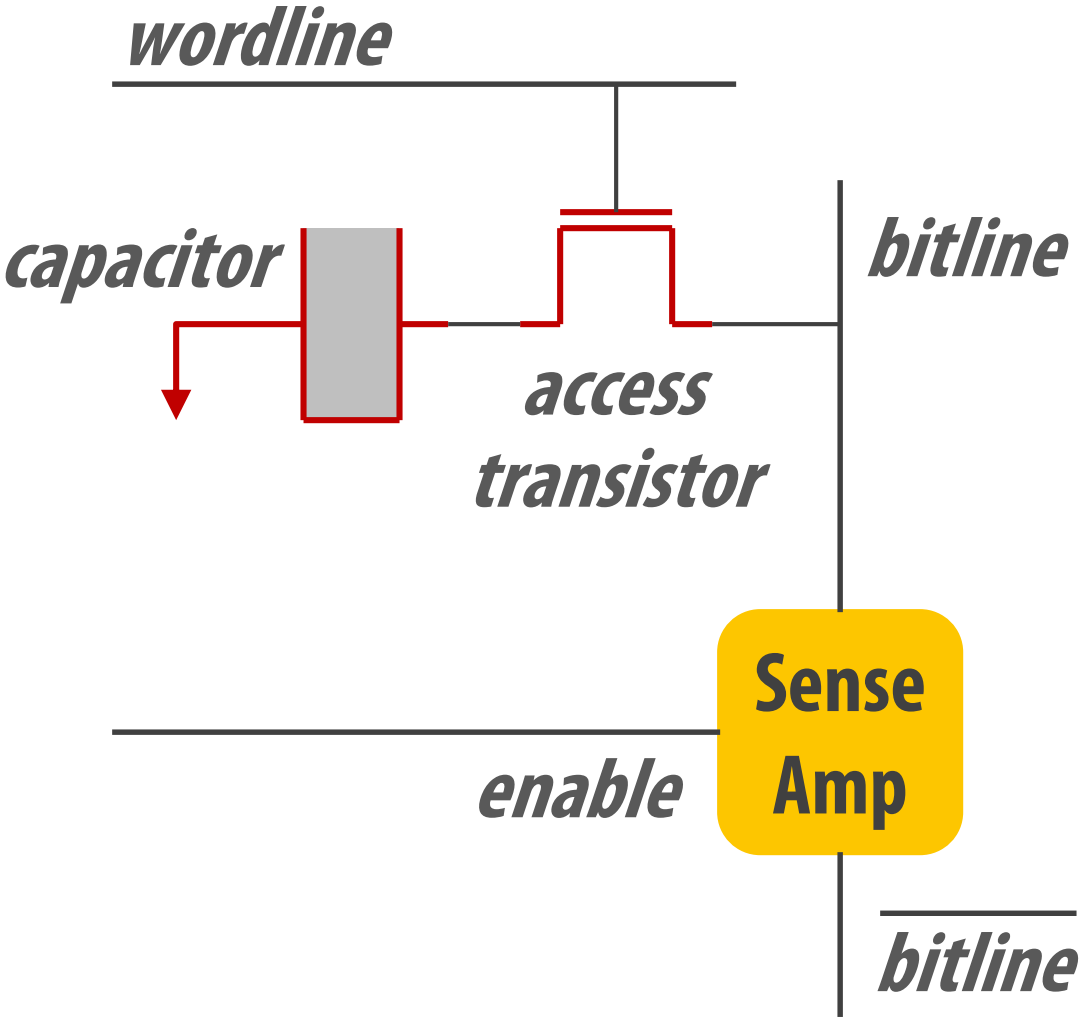
Outline of the talk

1. DRAM Background
2. Ambit-AND-OR: Bitwise AND/OR in DRAM
3. Ambit-NOT: Bitwise NOT in DRAM
4. Ambit Implementation
5. Applications and Evaluation

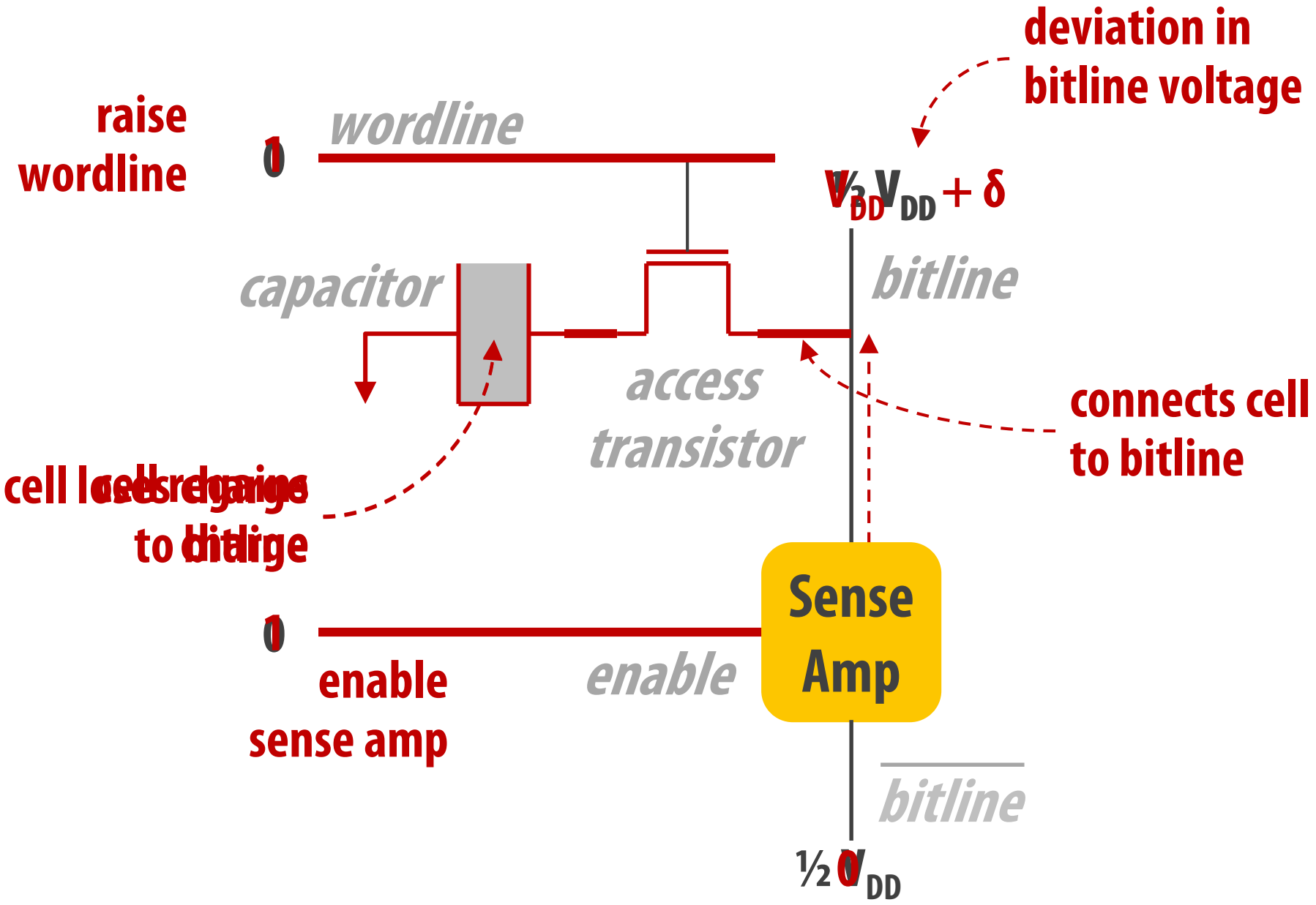
Inside a DRAM Chip



DRAM Cell Operation



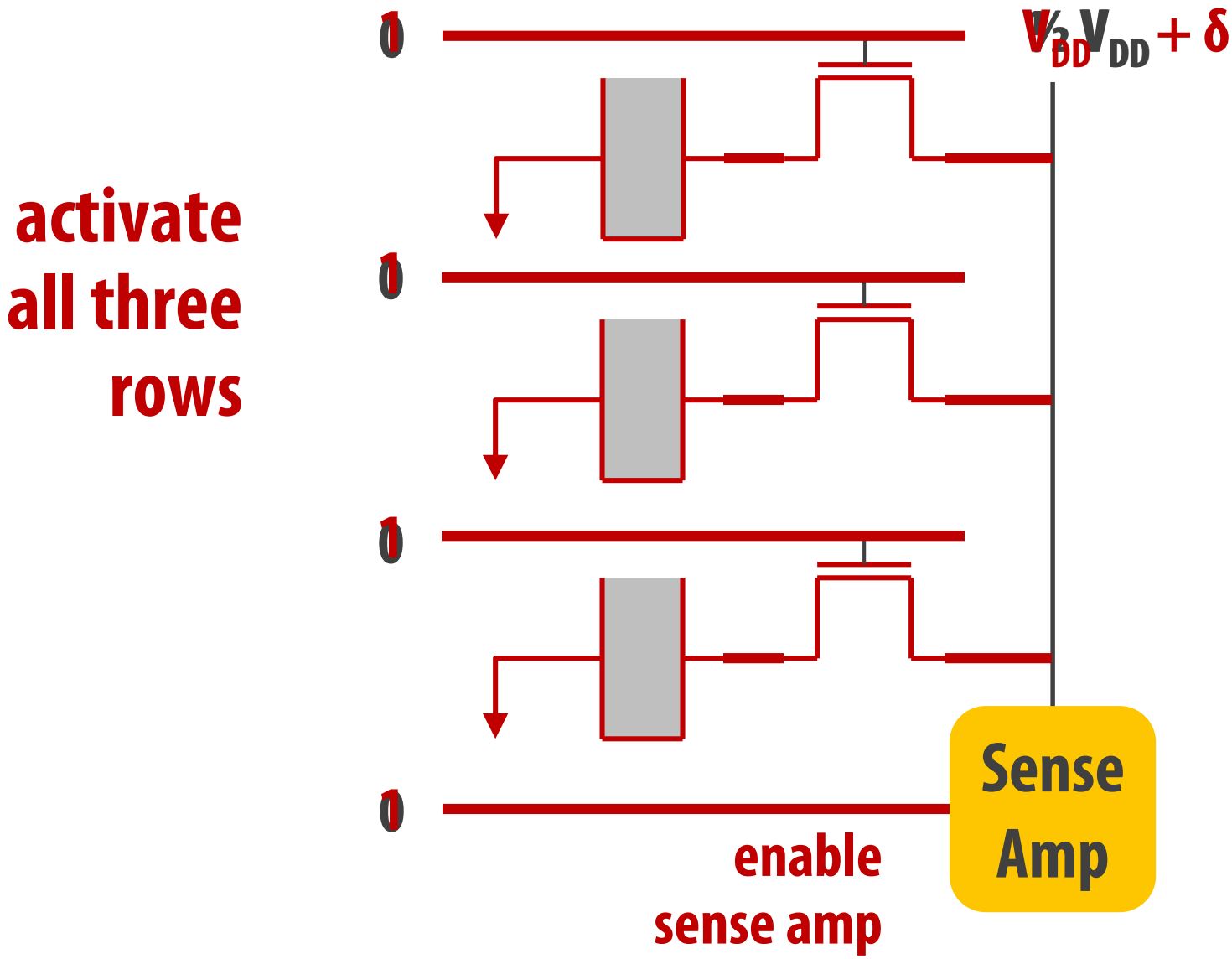
DRAM Cell Operation



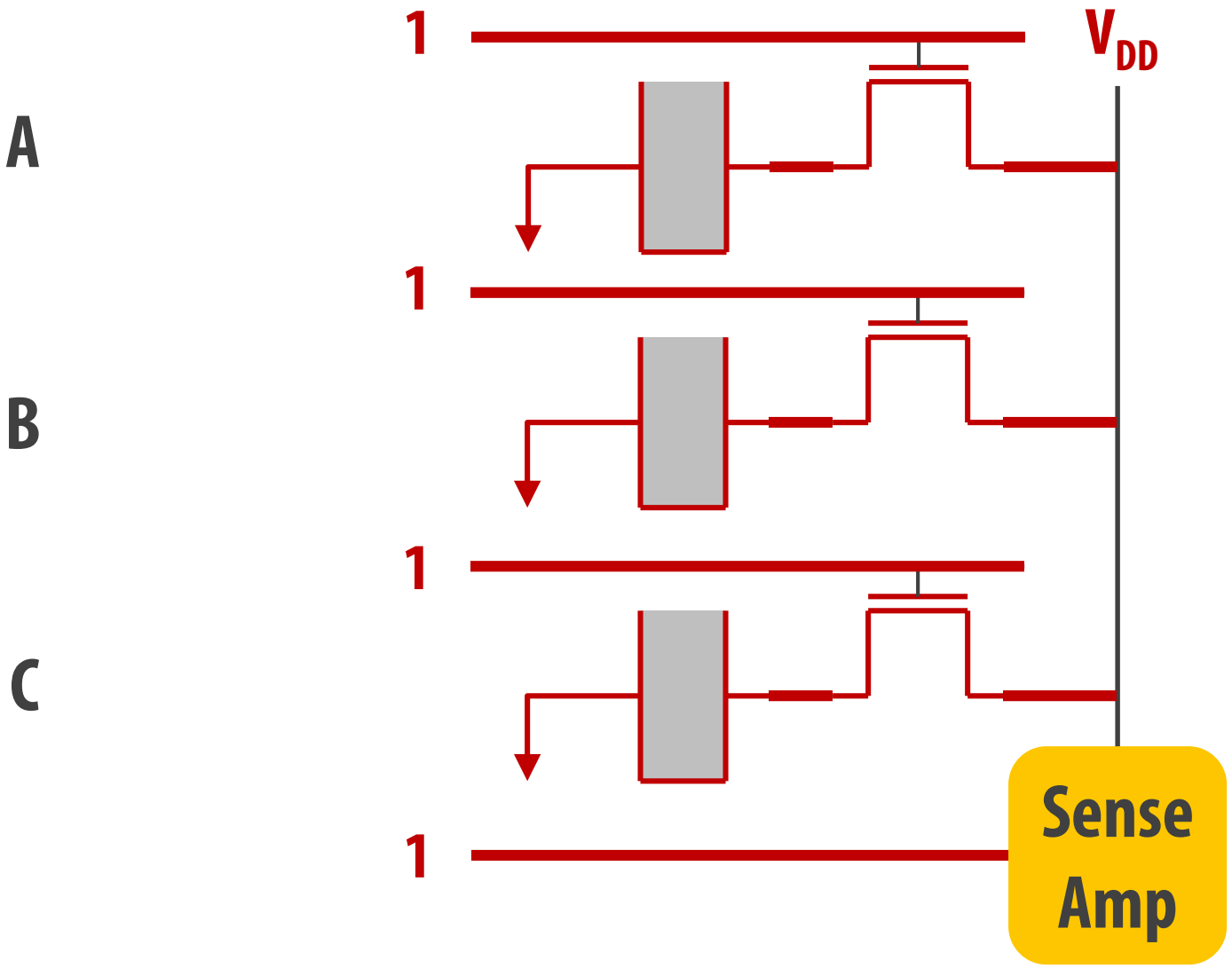
Outline of the talk

1. DRAM Background
2. Bitwise AND/OR in DRAM
3. Bitwise NOT in DRAM
4. Ambit Implementation
5. Applications and Evaluation

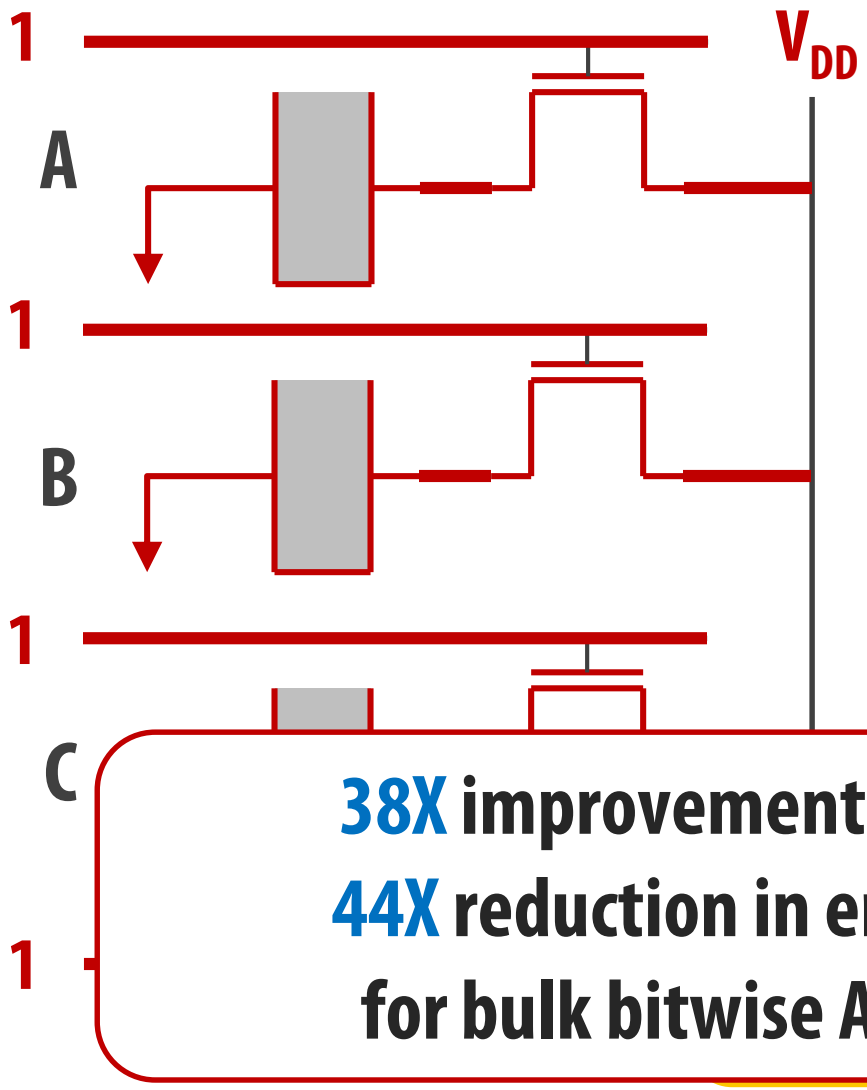
Triple-Row Activation: Majority Function



Bitwise AND/OR Using Triple-Row Activation



Bitwise AND/OR Using Triple-Row Activation



$$\text{Output} = AB + BC + CA$$
$$= C\,(A \textbf{ OR } B) +$$
$$\sim C\,(A \textbf{ AND } B)$$

Control the value of C to perform bitwise OR or B

38X improvement in raw throughput
44X reduction in energy consumption
for bulk bitwise AND/OR operations

Potential Concerns with Triple-Row Activation

1. With three cells, bitline deviation may not be enough
2. Process variation: all cells are not equal

Spice simulations put these concerns to rest.

(Section 6 in paper)

3. Cells leak charge
4. Memory controller may have to send three addresses
5. Source data gets destroyed

Address these challenges through implementation

(next slide)

Bulk Bitwise AND/OR in DRAM

Statically reserve three designated rows t1, t2, and t3

Result = row A AND/OR row B

1. Copy data of row A to row t1
2. Copy data of row B to row t2
3.
4.
5.

MICRO 2013

RowClone: Fast and Energy-Efficient In-DRAM Bulk Data Copy and Initialization

Vivek Seshadri vseshadr@cs.cmu.edu
Yoongu Kim yoongukim@cmu.edu
Chris Fallin* cfallin@c1f.net
Donghyuk Lee donghyuk1@cmu.edu
Rachata Ausavarungnirun rachata@cmu.edu
Gennady Pekhimenko gpekhime@cs.cmu.edu
Yixin Luo yixinluo@andrew.cmu.edu
Onur Mutlu onur@cmu.edu
Phillip B. Gibbons† phillip.b.gibbons@intel.com
Michael A. Kozuch† michael.a.kozuch@intel.com
Todd C. Mowry tcm@cs.cmu.edu
Carnegie Mellon University †Intel Pittsburgh

Bulk Bitwise AND/OR in DRAM

Statically reserve three designated rows t1, t2, and t3

Result = row A AND/OR row B

1. Copy data of row A to row t1 (RowClone)
2. Copy data of row B to row t2 (RowClone)
3. Initialize data of row t3 to 0/1 (RowClone)
4. Activate rows t1/t2/t3 simultaneously
5. Copy data of row t1/t2/t3 to Result row (RowClone)

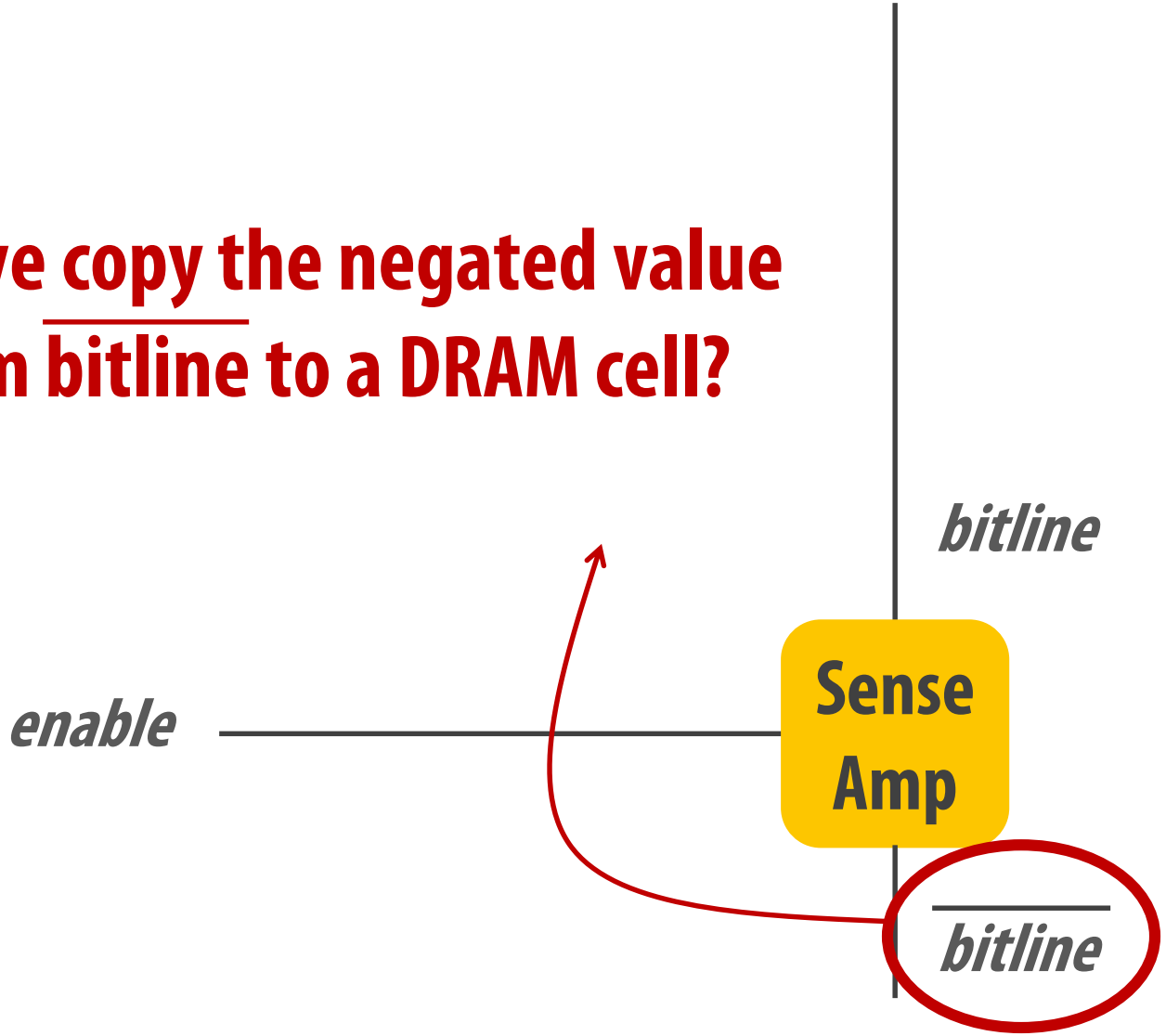
Use RowClone to perform copy and initialization operations completely in DRAM!

Outline of the talk

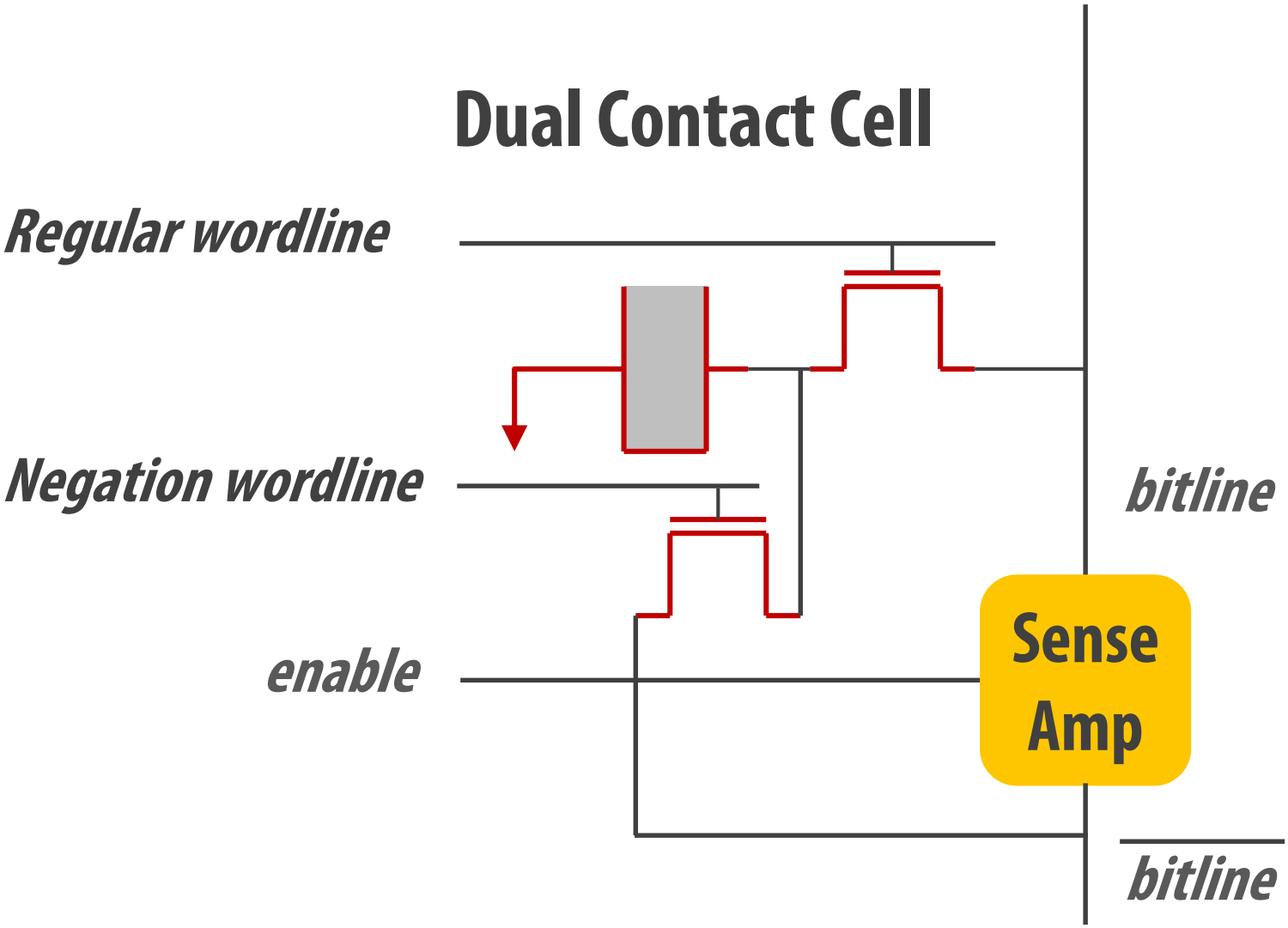
1. DRAM Background
2. Bitwise AND/OR in DRAM
3. Bitwise NOT in DRAM
4. Ambit Implementation
5. Applications and Evaluation

Negation Using the Sense Amplifier

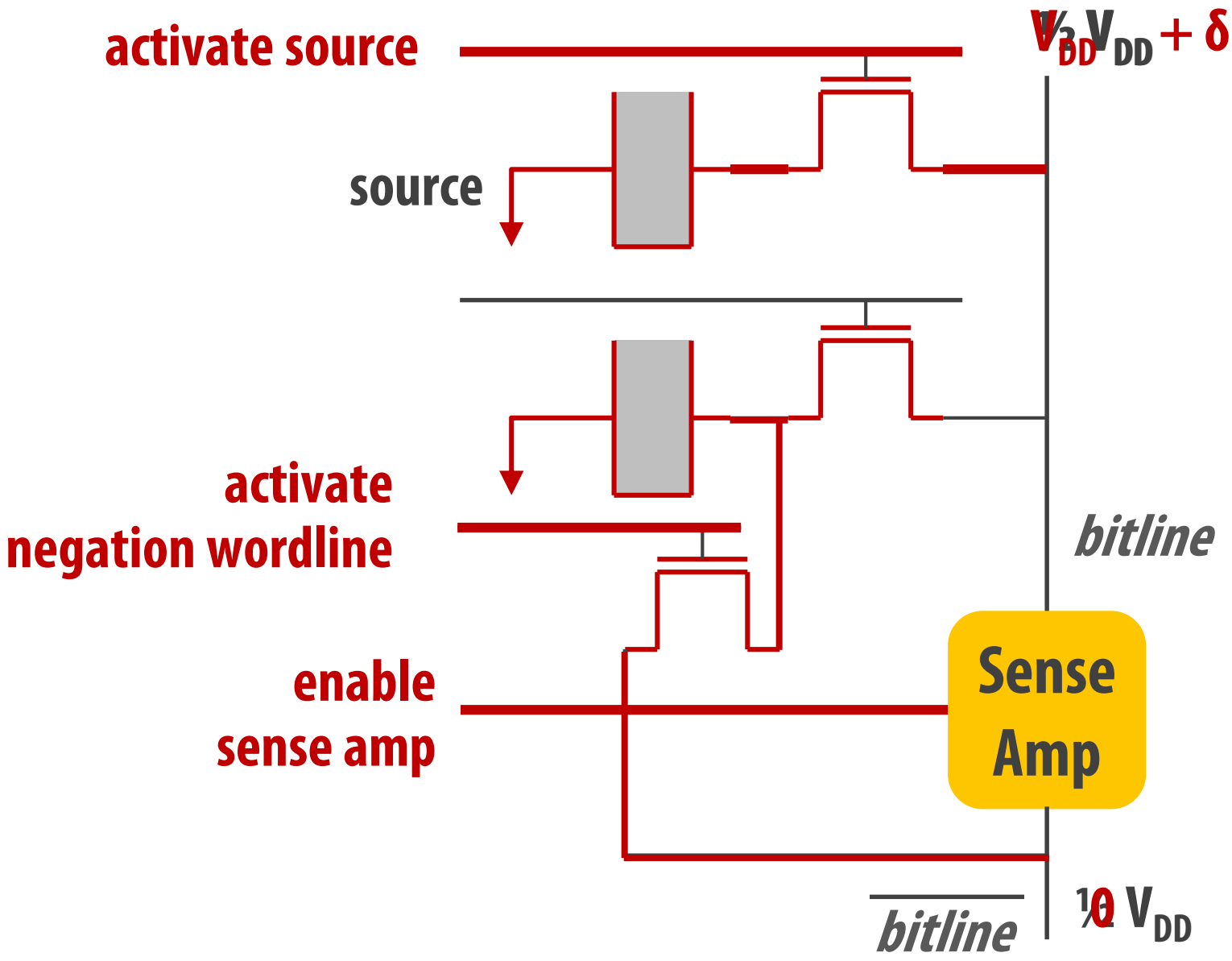
Can we copy the negated value from $\overline{\text{bitline}}$ to a DRAM cell?



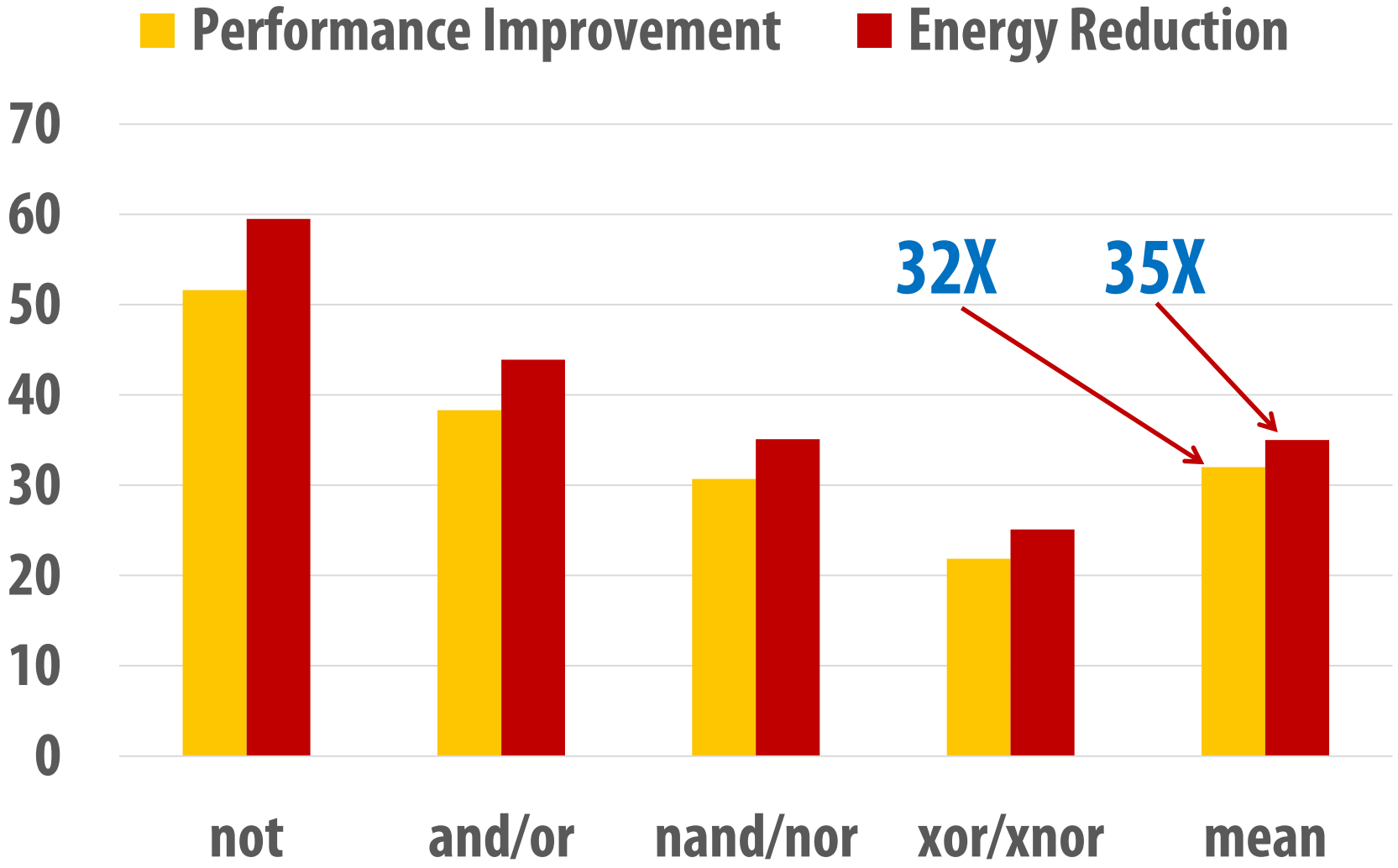
Negation Using the Sense Amplifier



Negation Using the Sense Amplifier



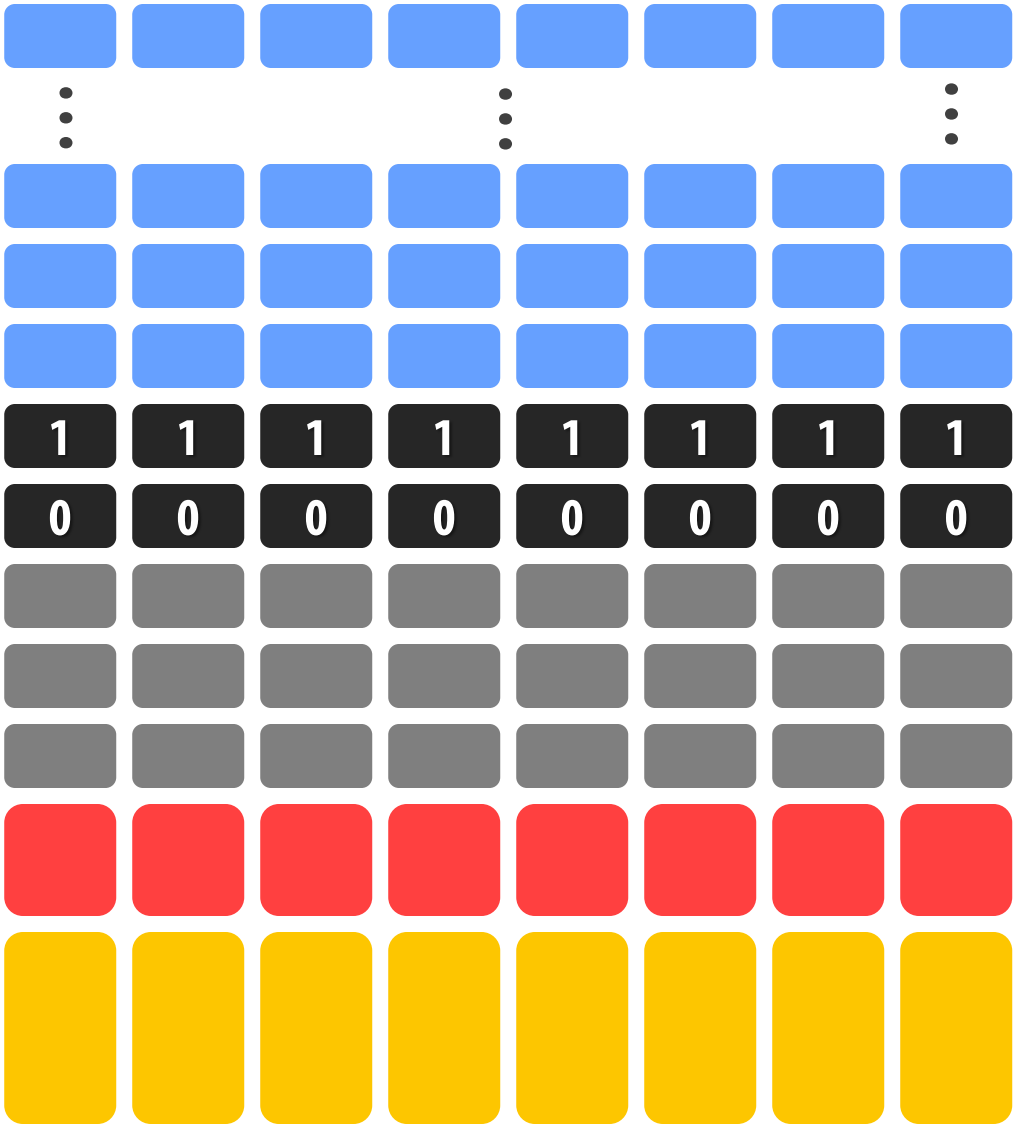
Ambit vs. DDR3: Performance and Energy



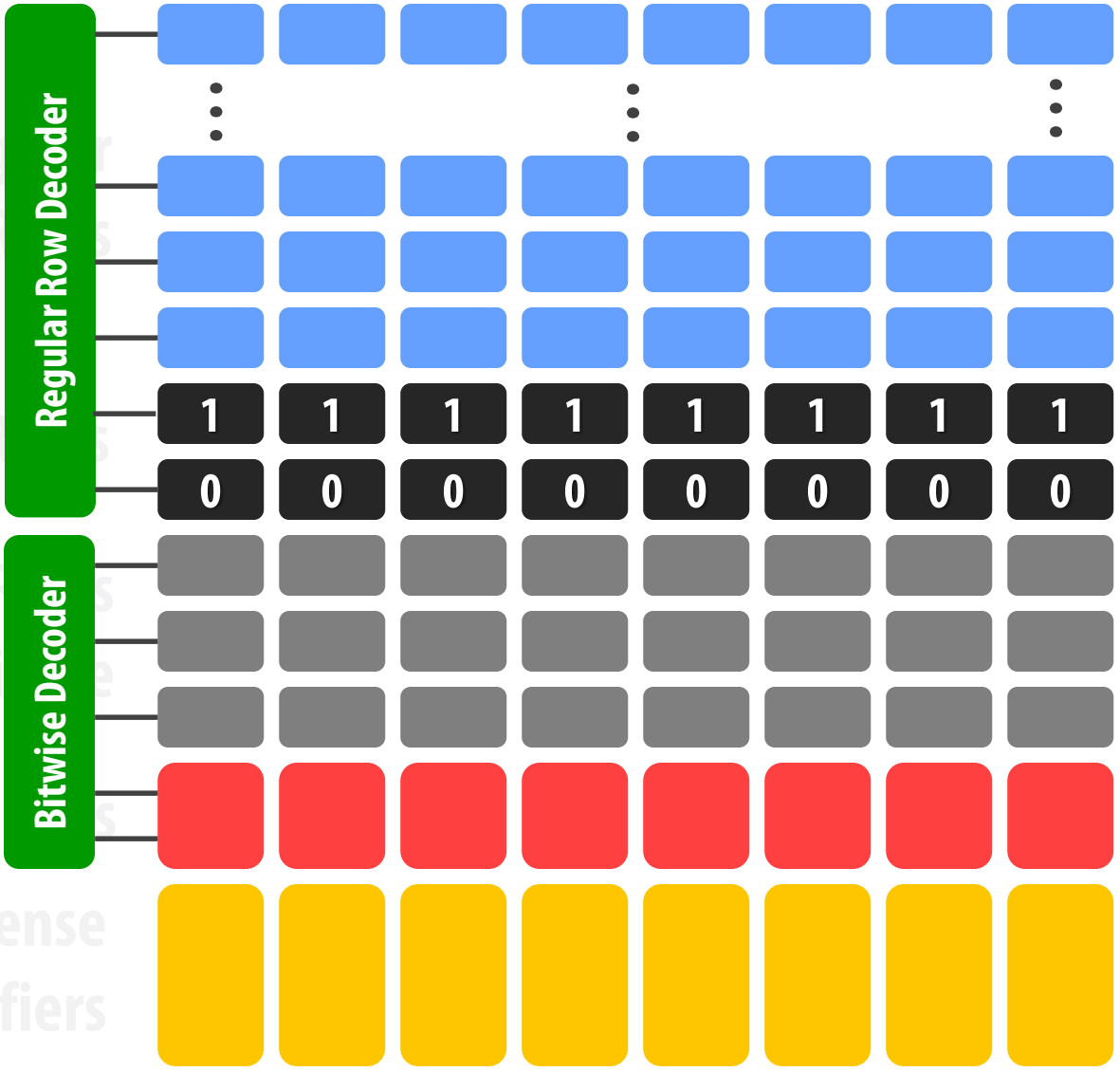
Outline of the talk

1. DRAM Background
2. Bitwise AND/OR in DRAM
3. Bitwise NOT in DRAM
4. Ambit Implementation
5. Applications and Evaluation

Ambit – Implementation



Ambit – Implementation



Integrating Ambit with the System

1. **PCIe device**
   - Similar to other accelerators (e.g., GPU)

2. **System memory bus**
   - Ambit uses the same DRAM command/address interface

Pros and cons discussed in paper (Section 5.4)

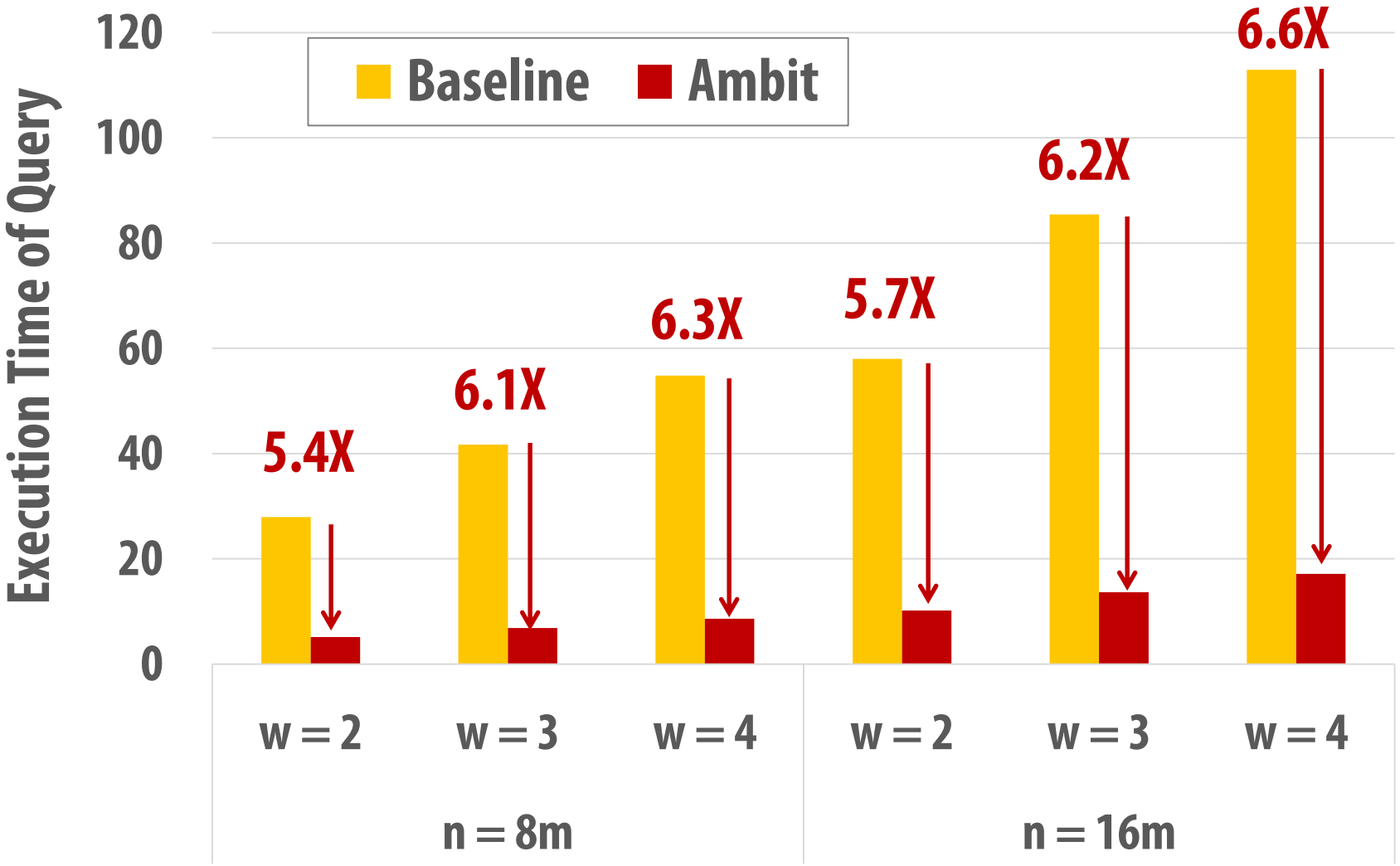
Outline of the talk

1. DRAM Background
2. Bitwise AND/OR in DRAM
3. Bitwise NOT in DRAM
4. Ambit Implementation
5. Applications and Evaluation

Real-world Applications

- **Methodology** (Gem5 simulator)
  - Processor: x86, 4 GHz, out-of-order, 64-entry instruction queue
  - L1 cache: 32 KB D-cache and 32 KB I-cache, LRU policy
  - L2 cache: 2 MB, LRU policy
  - Memory controller: FR-FCFS, 8 KB row size
  - Main memory: DDR4-2400, 1 channel, 1 rank, 8 bank
- **Workloads**
  - **Database bitmap indices**
  - **BitWeaving** –column scans using bulk bitwise operations
  - **Set operations** – comparing bitvectors with red-black trees

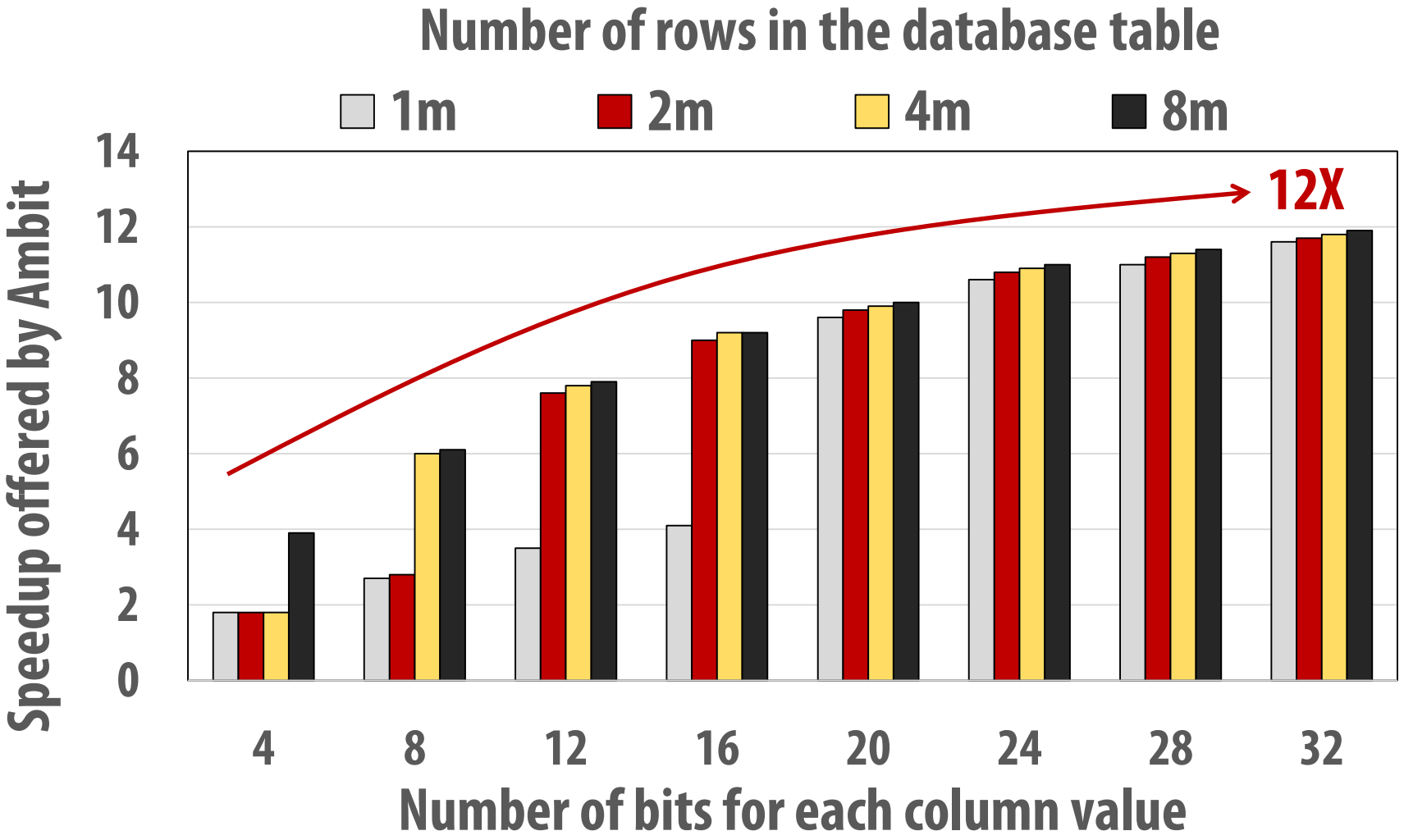
Bitmap Indices: Performance



Consistent reduction in execution time. 6X on average

Speedup offered by Ambit for BitWeaving

```
select count(*) where c1 < field < c2
```



Other Details and Results in Paper

- **Detailed implementation of Ambit**
  - Changes to DRAM chips
  - Optimizations to improve performance
  - Error correction codes (open problem)
- **Detailed SPICE simulation analysis**
- **Comparison to 3D-stacked DRAM**
- **Other applications**
  - Set operations
  - BitFunnel: Web search document filtering
  - Masked initialization
  - Cryptography
  - DNA sequence mapping

Conclusion

- **Problem: Bulk bitwise operations**
  - present in many applications, e.g., databases, search filters
  - existing systems are memory bandwidth limited
- **Our Proposal: Ambit**
  - perform bulk bitwise operations **completely inside DRAM**
  - **bulk bitwise AND/OR**: simultaneous activation of three rows
  - **bulk bitwise NOT**: inverters already in sense amplifiers
  - less than 1% area overhead over existing DRAM chips
- **Results compared to state-of-the-art baseline**
  - average across seven bulk bitwise operations
    - 32X performance improvement, 35X energy reduction
  - 3X-7X performance for real-world data-intensive applications

Ambit

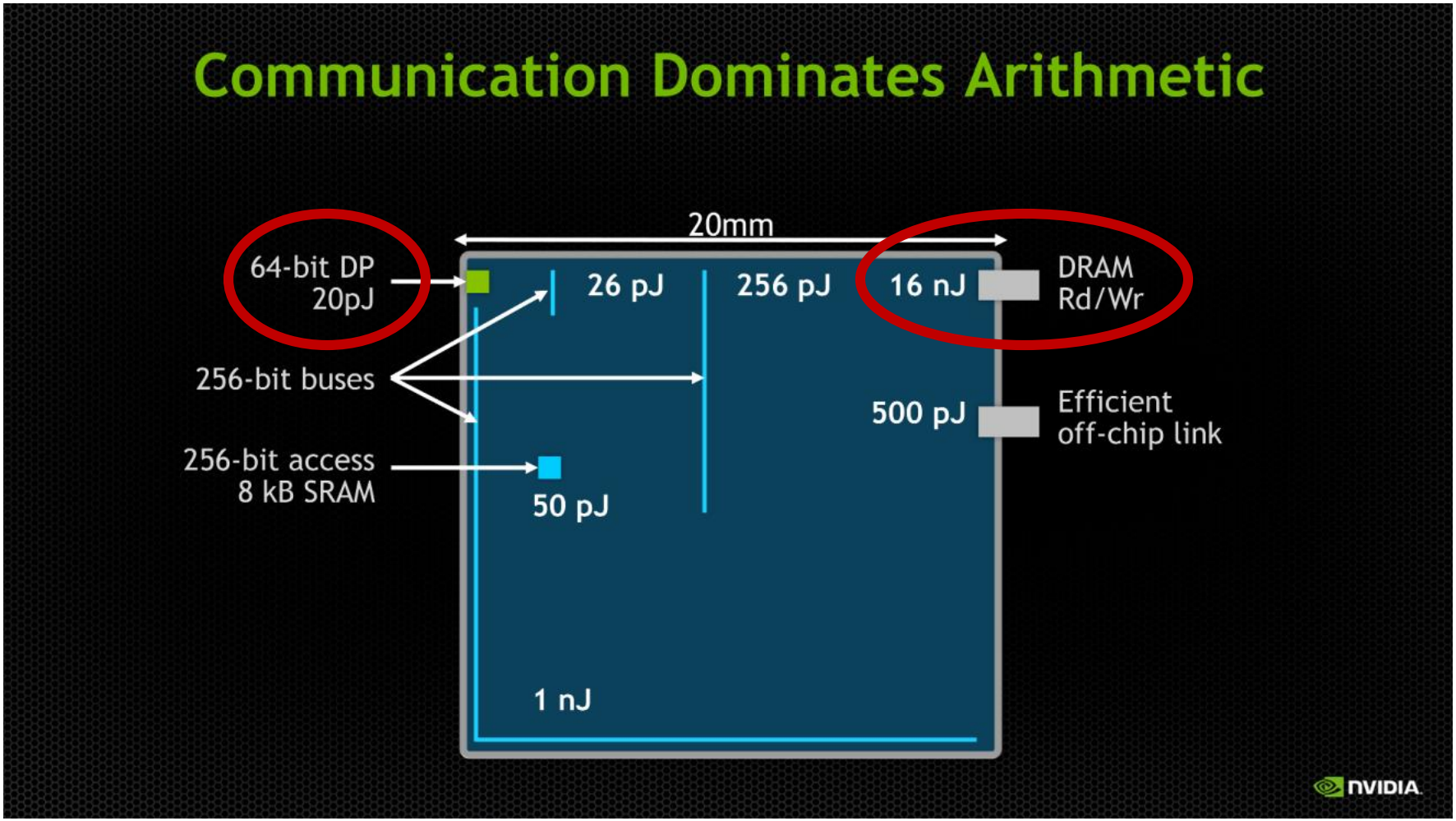
In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology

Vivek Seshadri

Donghyuk Lee, Thomas Mullins, Hasan Hassan, Amirali Boroumand, Jeremie Kim, Michael A. Kozuch, Onur Mutlu, Phillip B. Gibbons, Todd C. Mowry

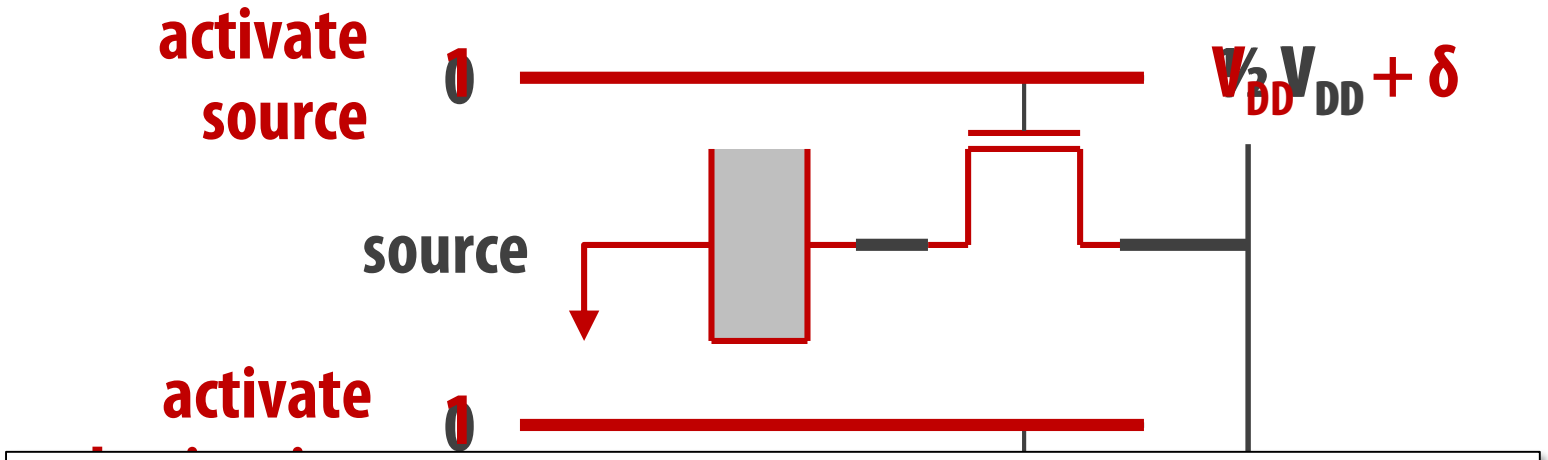
Backup Slides

Data movement consumes high energy



src: Bill Dally Keynote, "Challenge for Future Computer Systems," HIPEAC 2015.

RowClone: In-DRAM Bulk Data Copy (MICRO 2013)

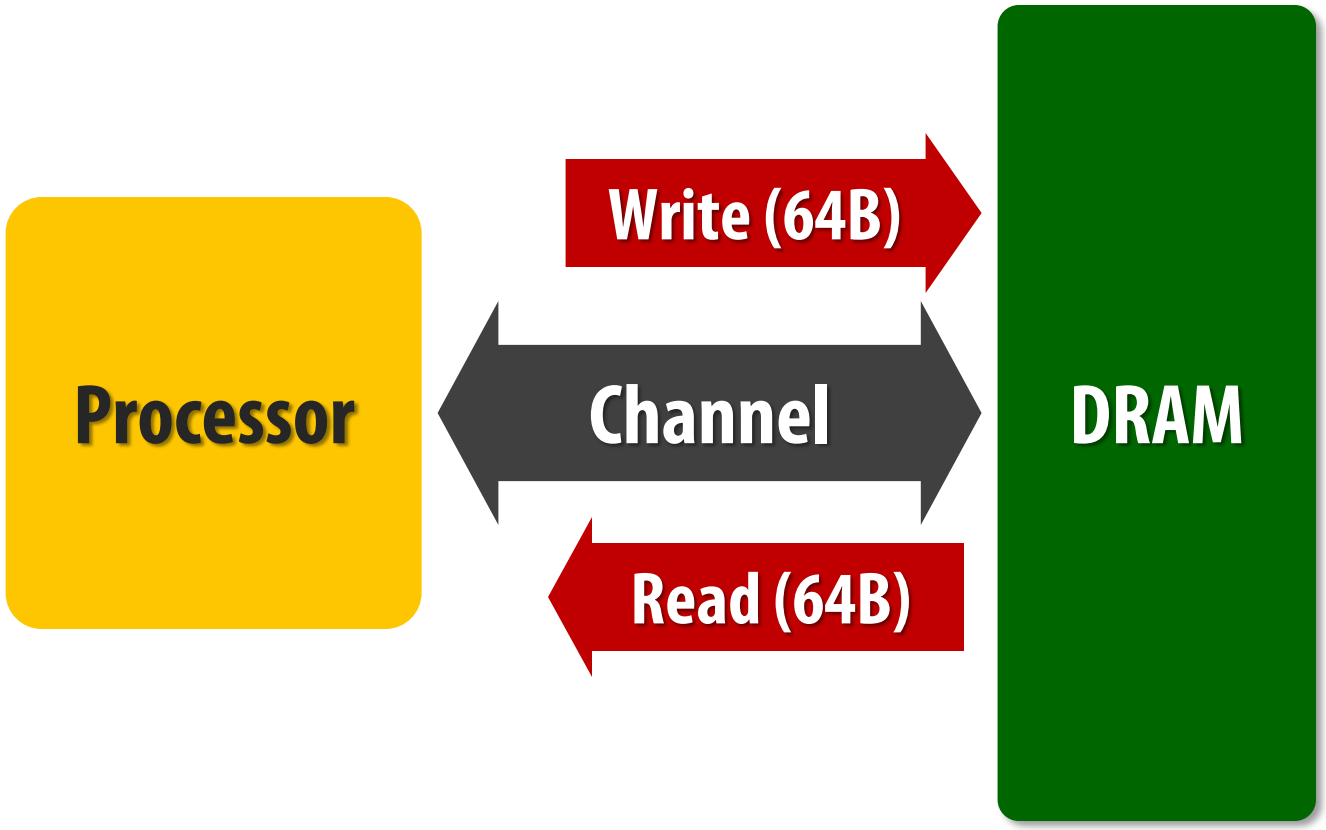


MICRO 2013

RowClone: Fast and Energy-Efficient In-DRAM Bulk Data Copy and Initialization

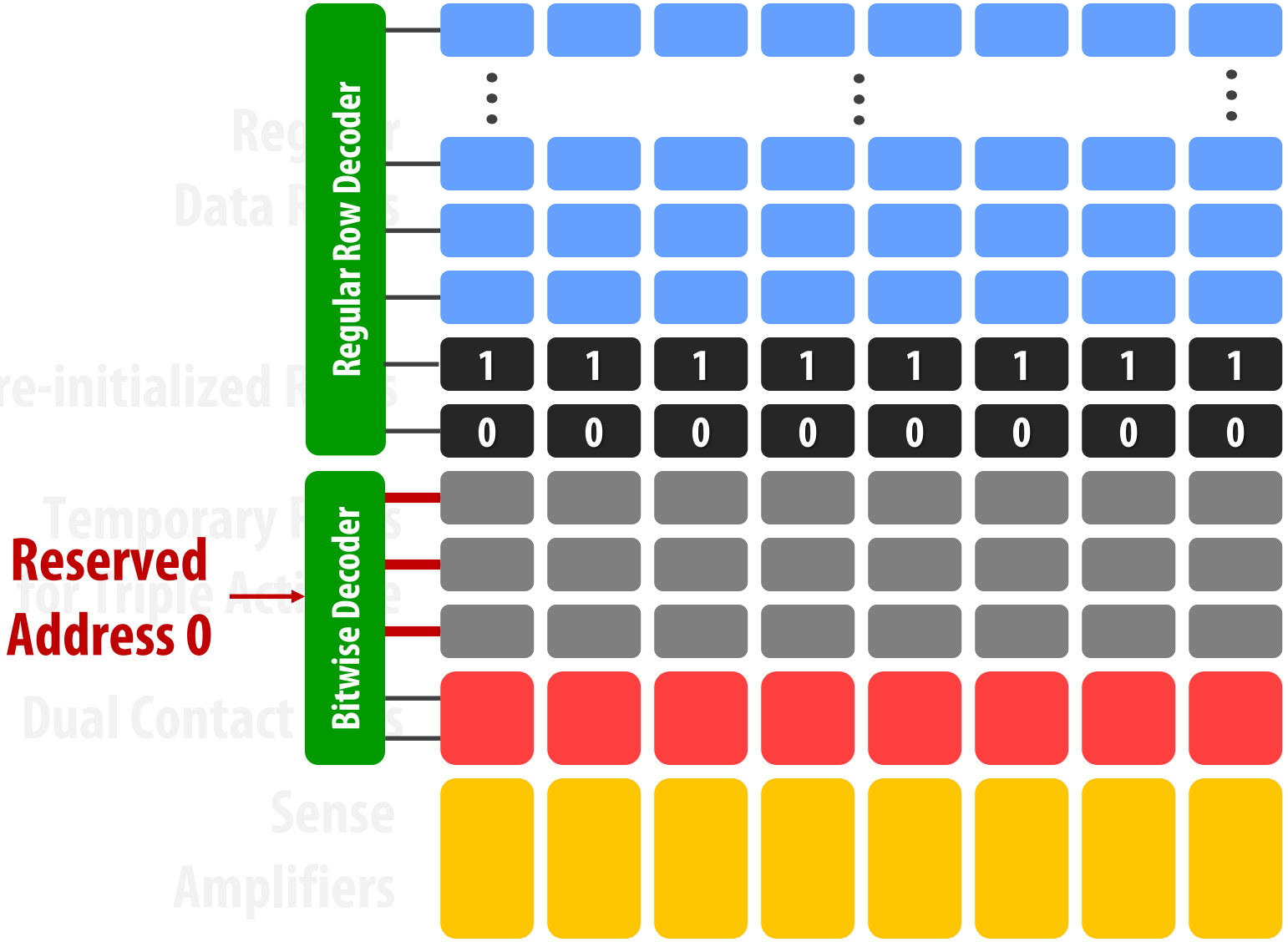
Vivek Seshadri vseshadr@cs.cmu.edu
Yoongu Kim yoongukim@cmu.edu
Chris Fallin* cfallin@c1f.net
Donghyuk Lee donghyuk1@cmu.edu
Rachata Ausavarungnirun rachata@cmu.edu
Gennady Pekhimenko gpekhime@cs.cmu.edu
Yixin Luo yixinluo@andrew.cmu.edu
Onur Mutlu onur@cmu.edu
Phillip B. Gibbons† phillip.b.gibbons@intel.com
Michael A. Kozuch† michael.a.kozuch@intel.com
Todd C. Mowry tcm@cs.cmu.edu
Carnegie Mellon University †Intel Pittsburgh

Today, DRAM is just a storage device!



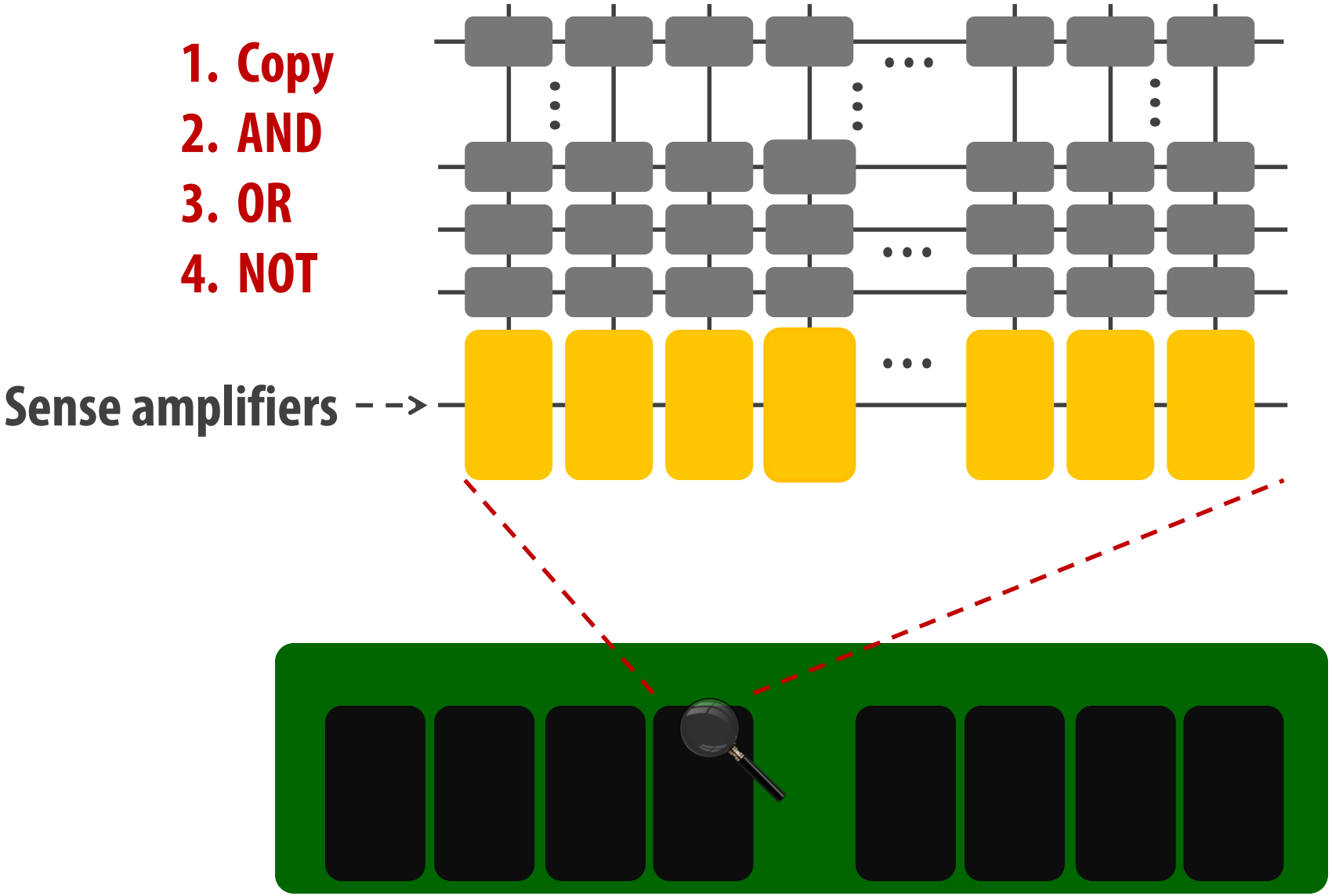
Can we do more with DRAM?

Ambit – Implementation

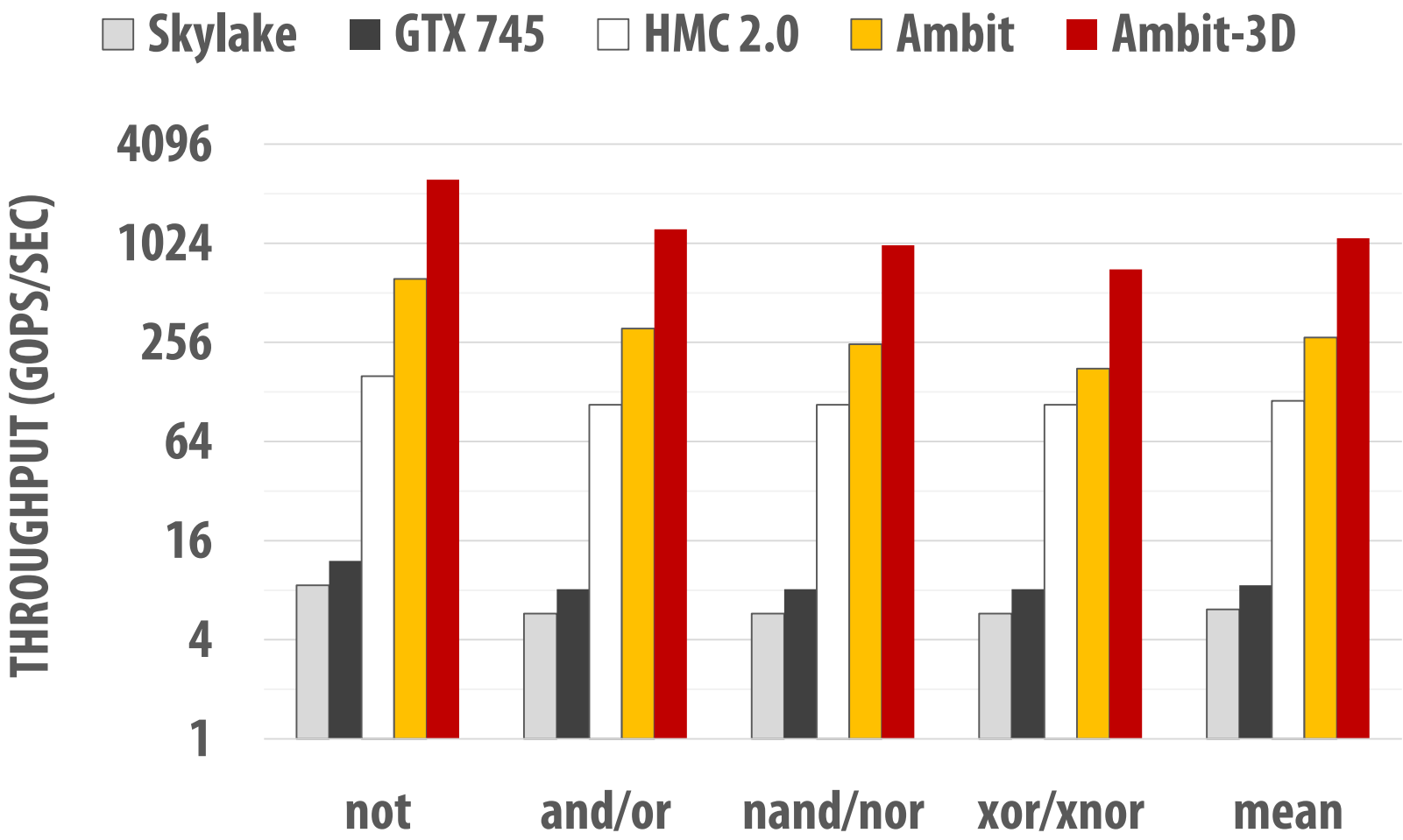


Summary of operations

1. Copy
2. AND
3. OR
4. NOT



Ambit Throughput



Error Correction Code

- Need ECC that is homomorphic over bitwise operations
  - ECC(A **and** B) = ECC(A) **and** ECC(B)
  - ECC(A **or** B) = ECC(A) **or** ECC(B)
  - ECC(**not** A) = **not** ECC(A)

- **Triple Modular Redundancy**
  - trivially satisfies the above condition
  - 2X capacity overhead
  - Better performance and energy efficiency
  - Lower overall cost