ETH zürich

Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology, MICRO 2017

Vivek Seshadri[1,5], Donghyuk Lee[2,5], Thomas Mullins[3,5], Hasan Hassan[4], Amirali Boroumand[5], Jeremie Kim[4,5], Michael A. Kozuch[3], Onur Mutlu[4,5], Phillip B. Gibbons[5], Todd C. Mowry[5]

[1]Microsoft Research India , [2]NVIDIA Research, [3]Intel, [4]ETH Zürich, [5]Carnegie Mellon University

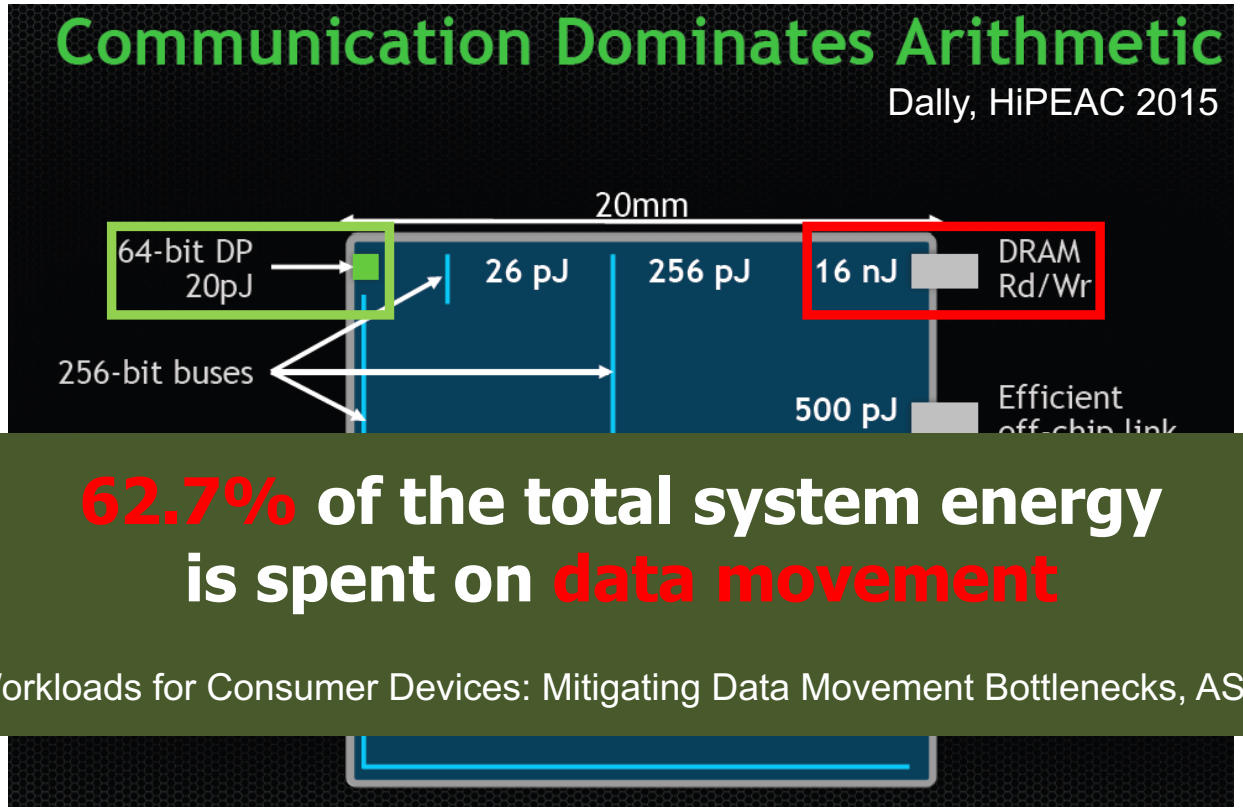
Outline

- Executive Summary
- Prerequisites
- Ambit AND-OR
- Ambit NOT
- Putting It All Together
- Evaluation & Testing
- Conclusion

- Strengths/Weaknesses
- Related Work
- Discussion

Executive Summary

- Problem: Data Movement Bottleneck
  - **Throughput limits** performance
  - Data movement is very **expensive energy-wise** (~1000x compared to arithmetic)



62.7% of the total system energy is spent on data movement

Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks, ASPLOS '18

Executive Summary

- Problem: Data Movement Bottleneck
  - **Throughput limits** performance
  - Data movement is very **expensive energy-wise** (~1000x compared to arithmetic)

- Goal
  - **Reduce data movement**
  - Instead, **compute in memory**
  - In this paper, performing **bulk bitwise operations** completely **inside DRAM**
    - Throughput limited by memory bandwidth
    - Utilized by many applications, e.g., databases, sets, encryption

Executive Summary

- Key Ideas
  - Use **existing analog structures** to perform bulk bitwise AND-OR
  - Utilize **already present inverters** to perform bulk bitwise NOT
  - Together, this set of operations is **logically complete**

- Key Mechanisms
  - **Triple Row Activation** to get a majority function
  - **Dual Contact Cells** to store negated data

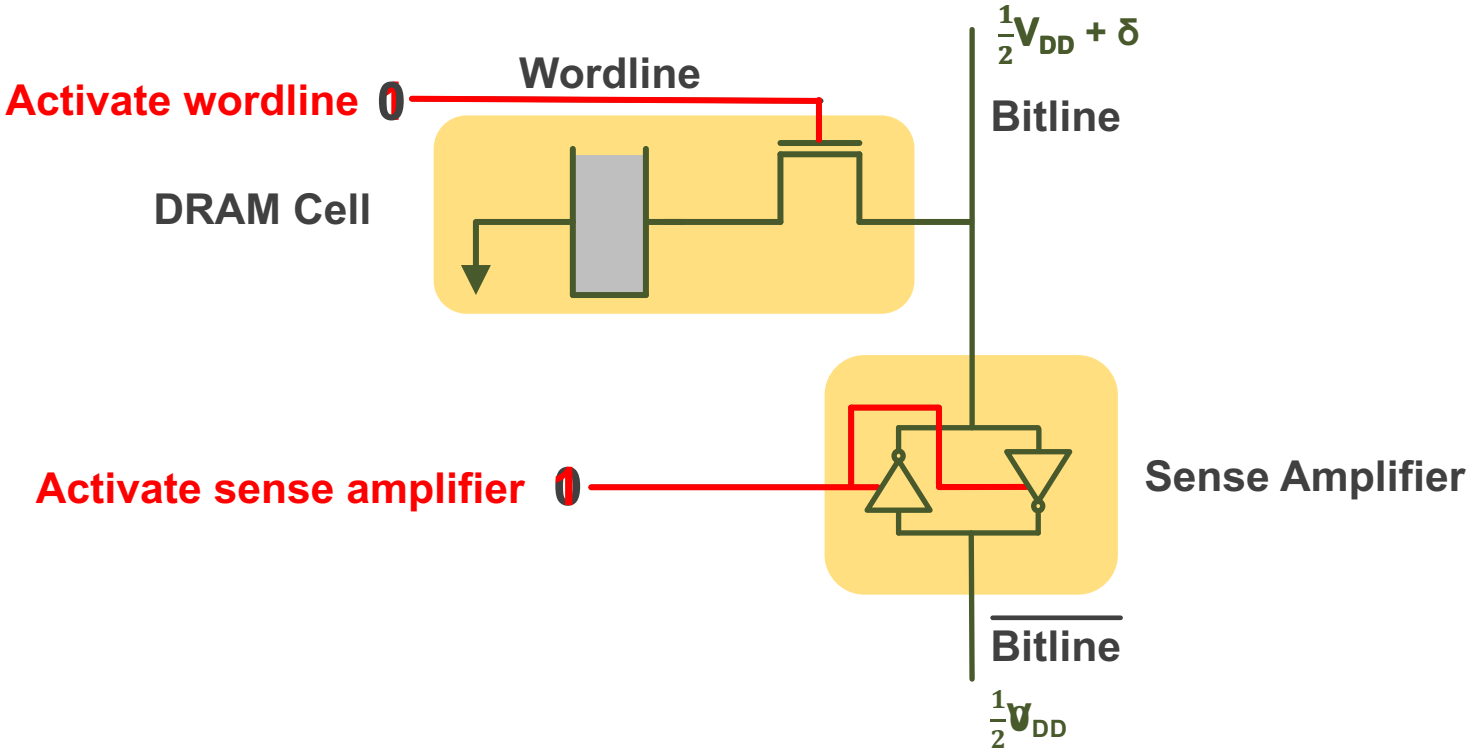
- Results
  - Up to **32x performance improvement** & **35x energy reduction** across 7 bulk bitwise operations
  - **3x-7x performance increase** for selected data-intensive workloads
  - **≤ 1% area overhead** over existing DRAM chips

Outline

- Executive Summary
- **Prerequisites**
- Ambit AND-OR
- Ambit NOT
- Putting It All Together
- Evaluation & Testing
- Conclusion

- Strengths/Weaknesses
- Related Work
- Discussion

Prerequisites - DRAM



Outline

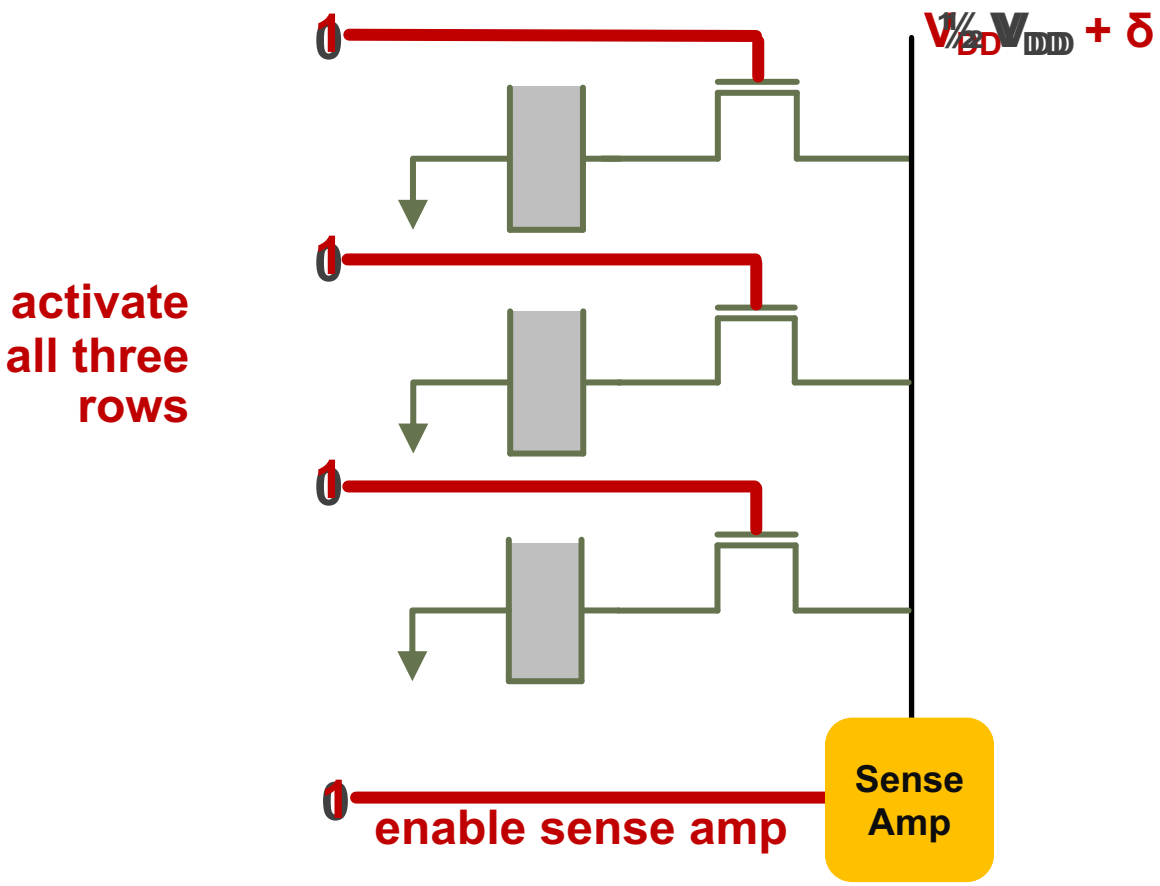
- Executive Summary
- Prerequisites
- **Ambit AND-OR**
- Ambit NOT
- Putting It All Together
- Evaluation & Testing
- Conclusion

- Strengths/Weaknesses
- Related Work
- Discussion

Ambit AND-OR

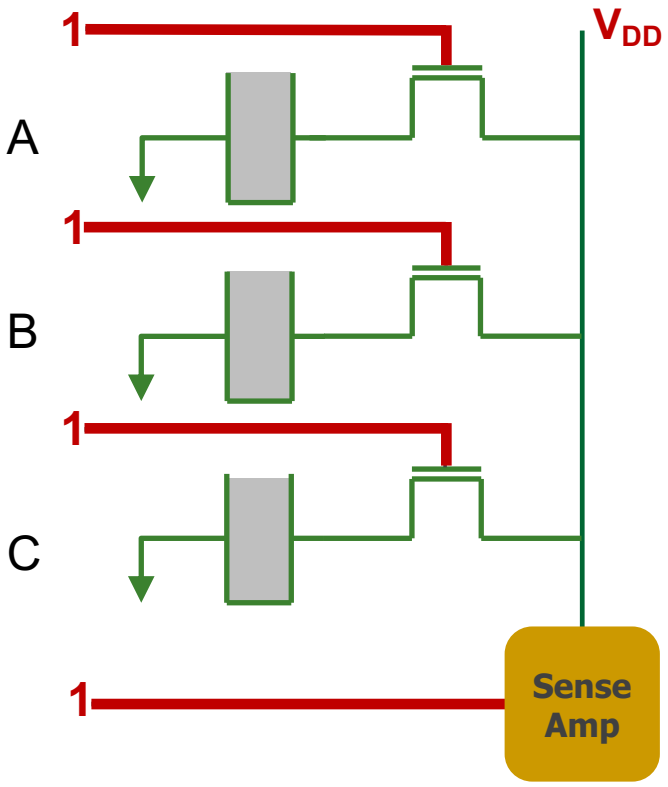
- Ambit AND-OR relies on **analog charge sharing**
- **Triple Row Activation:** Activating three rows together will average their voltage deviations
- Results in a bitwise **majority function**
- Enables selectively bulk bitwise AND or OR operation of two rows

Ambit AND-OR – Triple Row Activation (TRA)



Source Animation: https://www.archive.ece.cmu.edu/~safari/pubs/ambit-bulk-bitwise-dram_micro17-talk.pptx

Ambit AND-OR – Triple Row Activation (TRA)



Result:
AB + BC + AC =
C(A + B) + ~C(AB)

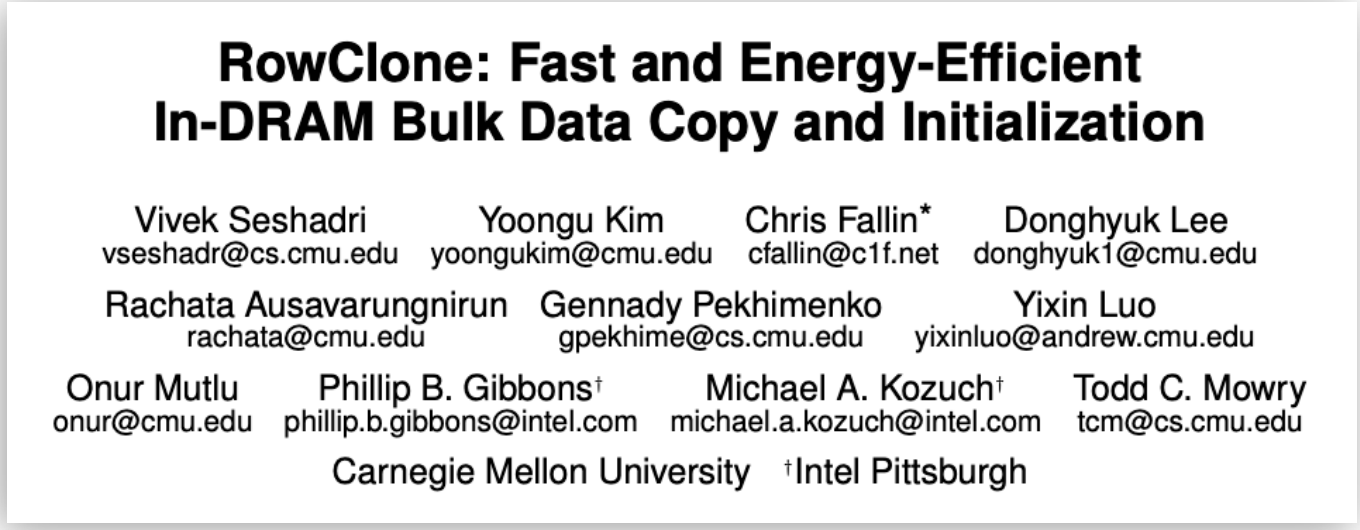
C can be used to control what operation should be performed!

Ambit AND-OR - Challenges

- Source data in all cells gets destroyed

- Solution:

  Don’t operate on the source directly, copy data into other rows first.



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

Ambit AND-OR - Challenges

- Naïve implementation would require memory controller to send three addresses

- Solution:
  Dedicated rows for Triple Row Activation.
  One address maps to TRA on the dedicated rows.

Ambit AND-OR - Challenges

- We assume that cells are fully charged or discharged

- Solution:
  “RowCloning” the data refreshes the cells.

Ambit AND-OR - Challenges

- Source data in all cells gets destroyed
- Naïve implementation would require memory controller to send three addresses
- We assume that cells are fully charged or discharged

-> Solved by the implementation

- Cells and wires are not equal (process variation)
- Bitline deviation may not be sufficient to trigger amplifier

-> Will be discussed in the testing section

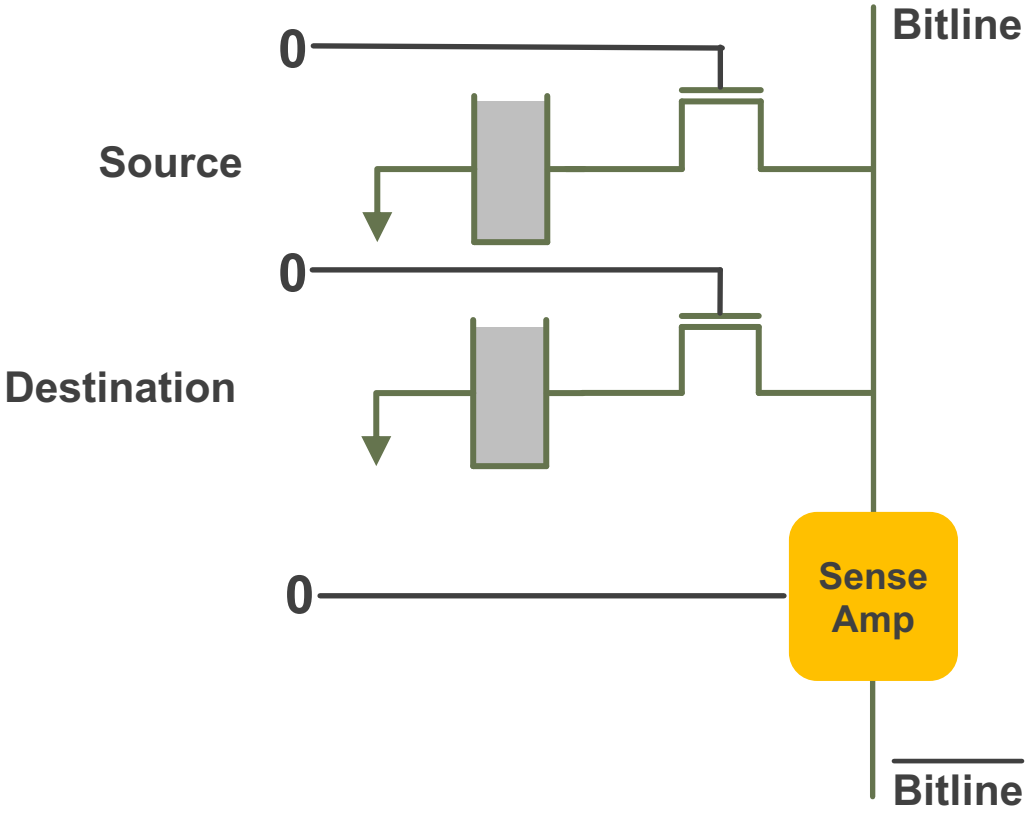
Outline

- Executive Summary
- Prerequisites
- Ambit AND-OR
- **Ambit NOT**
- Putting It All Together
- Evaluation & Testing
- Conclusion

- Strengths/Weaknesses
- Related Work
- Discussion

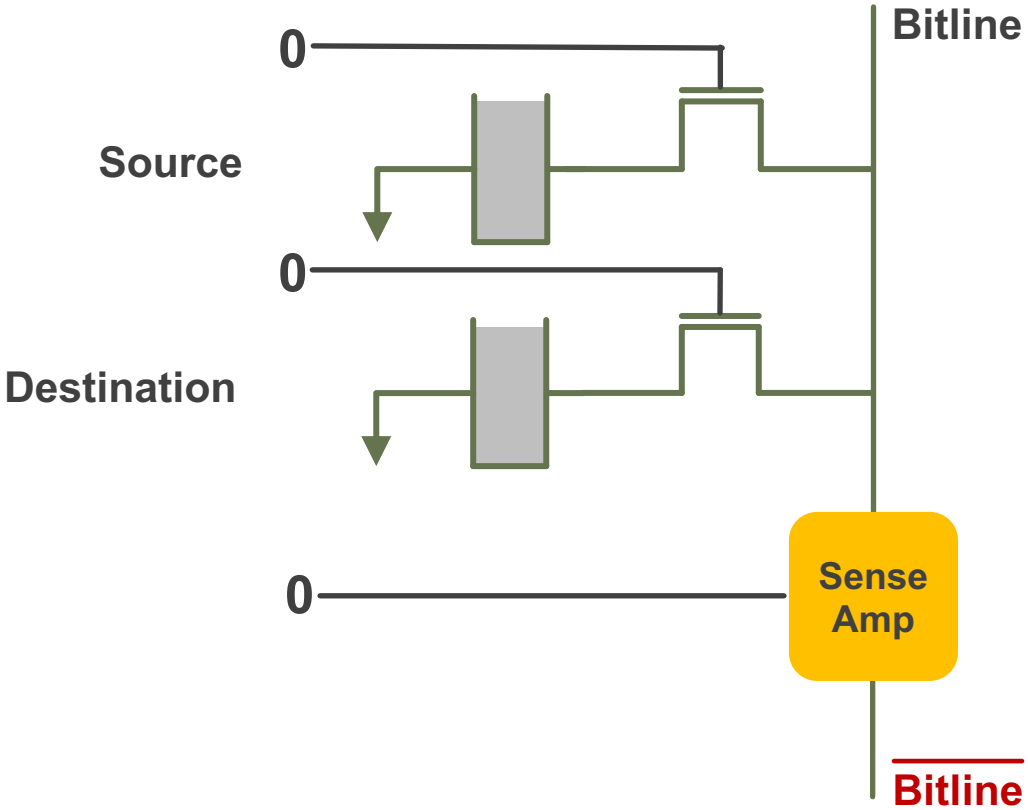
Ambit NOT

- Use the inverters in the amplifiers to negate rows



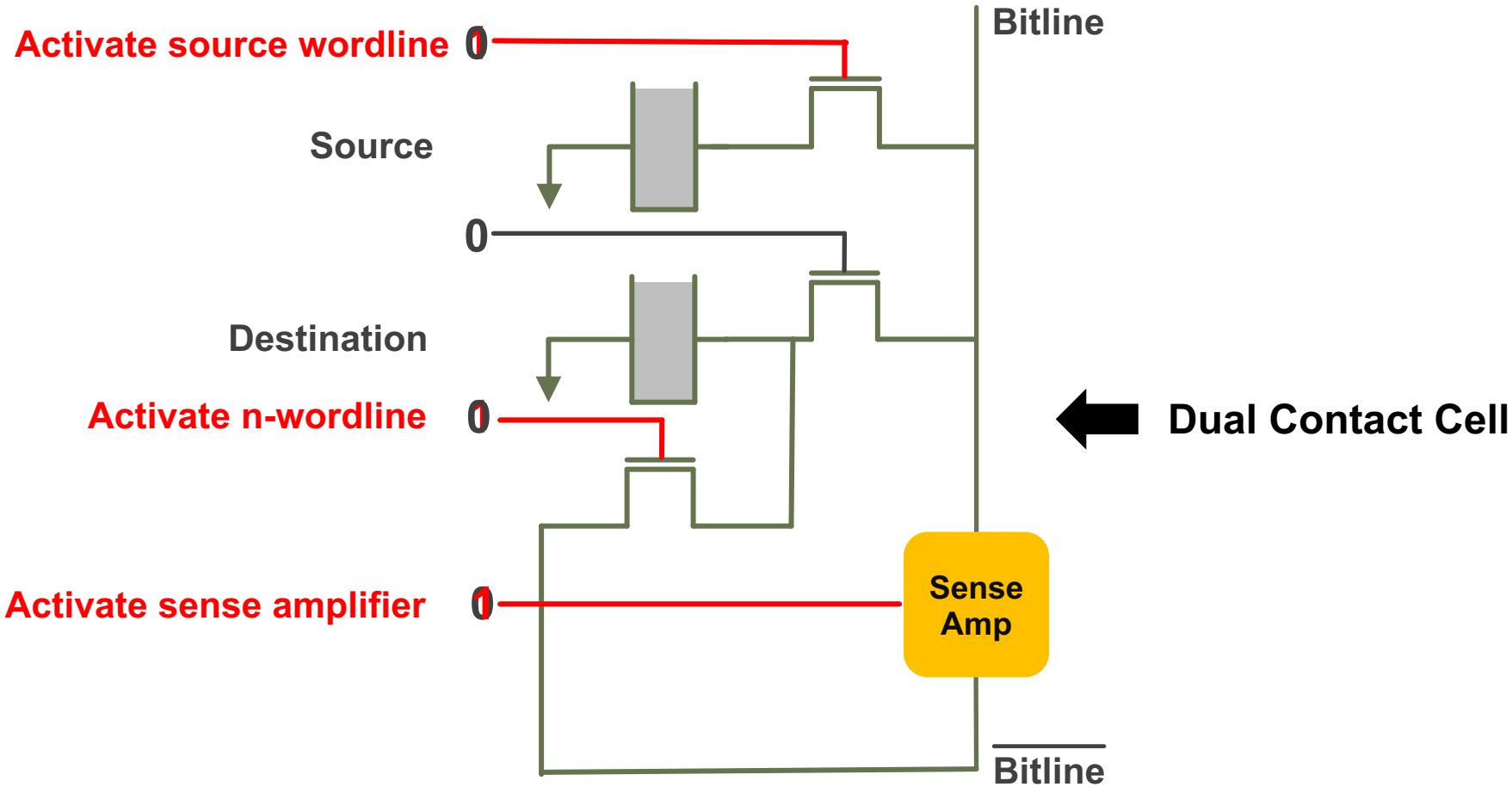
Ambit NOT

- Use the inverters in the amplifiers to negate rows



Ambit NOT

- Use the inverters in the amplifiers to negate rows



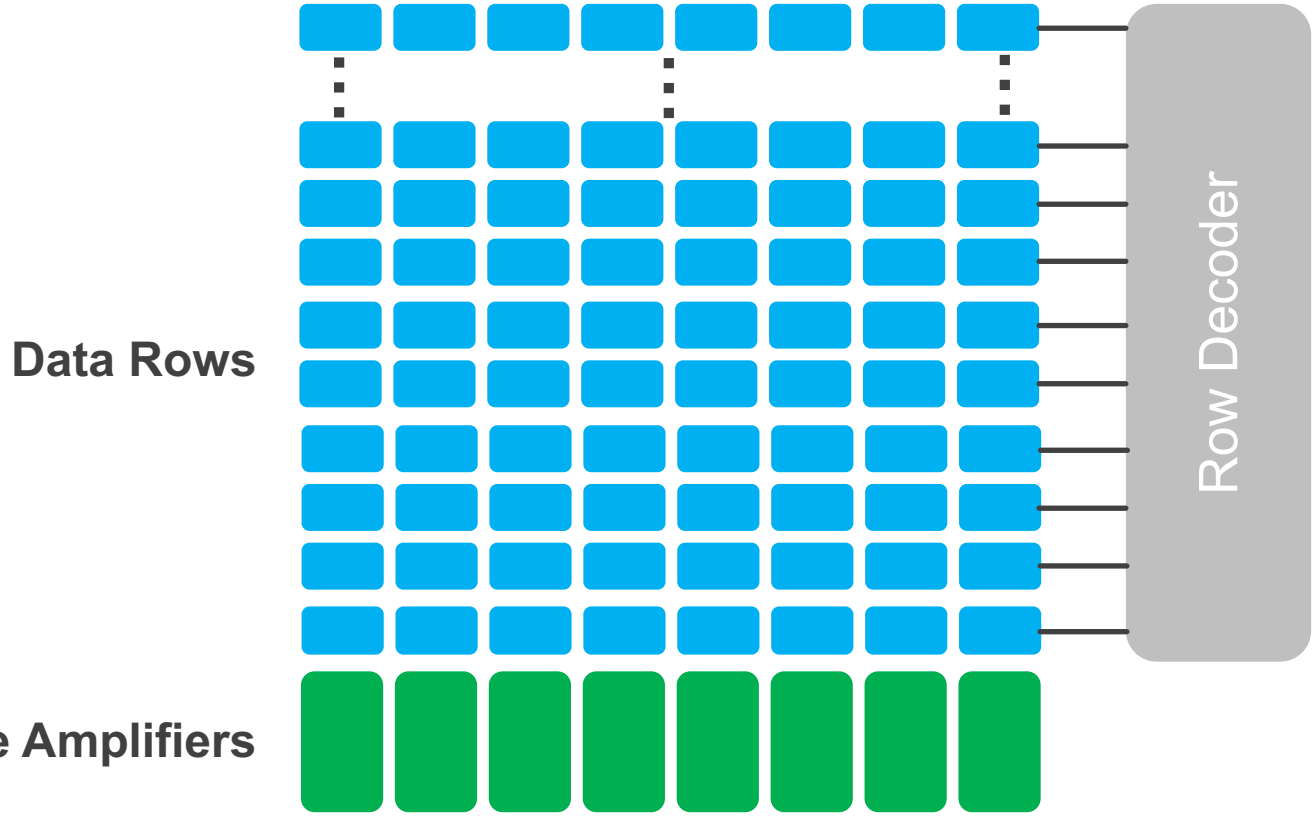
Outline

- Executive Summary
- Prerequisites
- Ambit AND-OR
- Ambit NOT
- **Putting It All Together**
- Evaluation & Testing
- Conclusion

- Strengths/Weaknesses
- Related Work
- Discussion

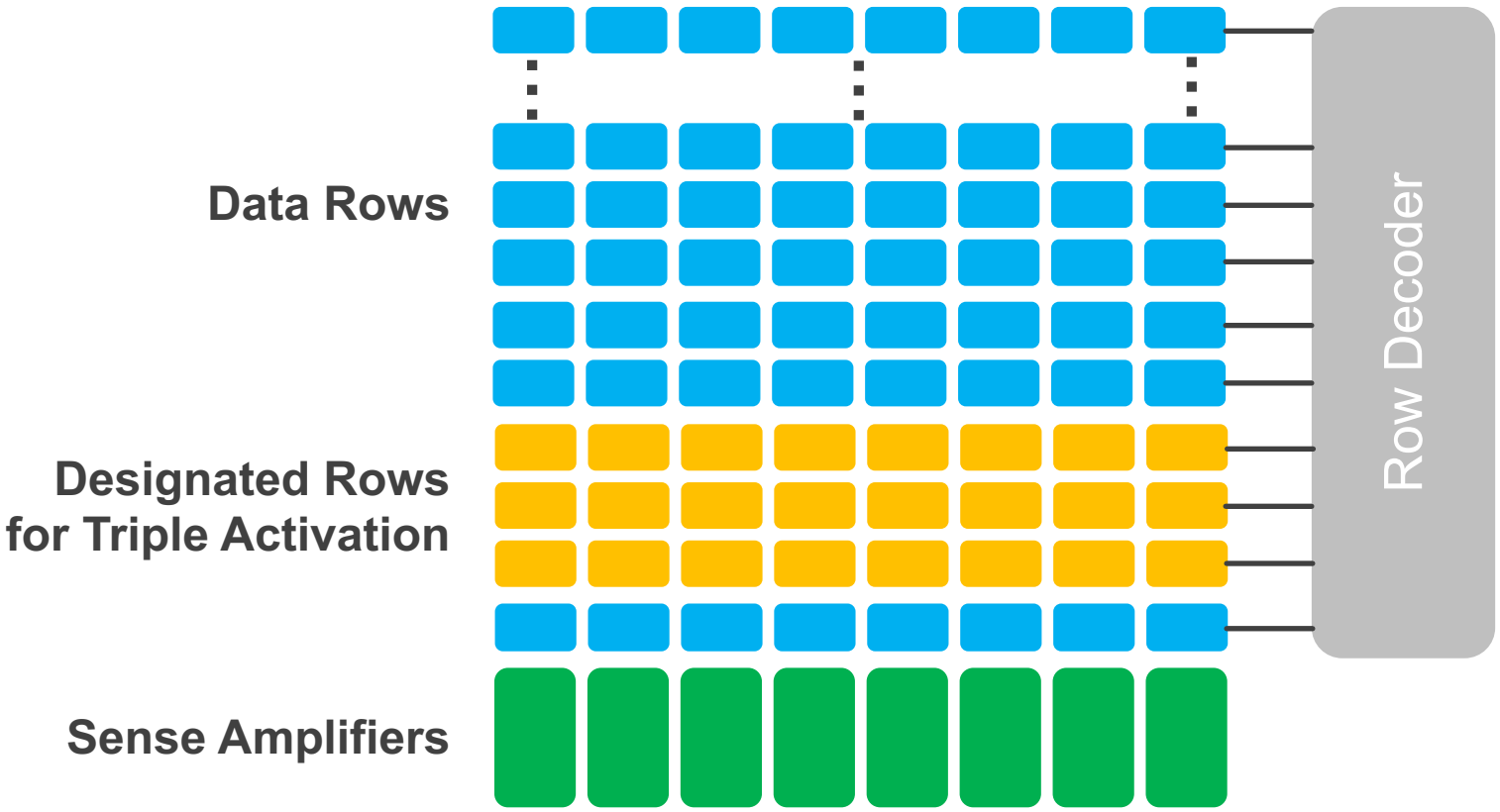
Putting It All Together

- Let's start with a normal DRAM subarray and add Ambit



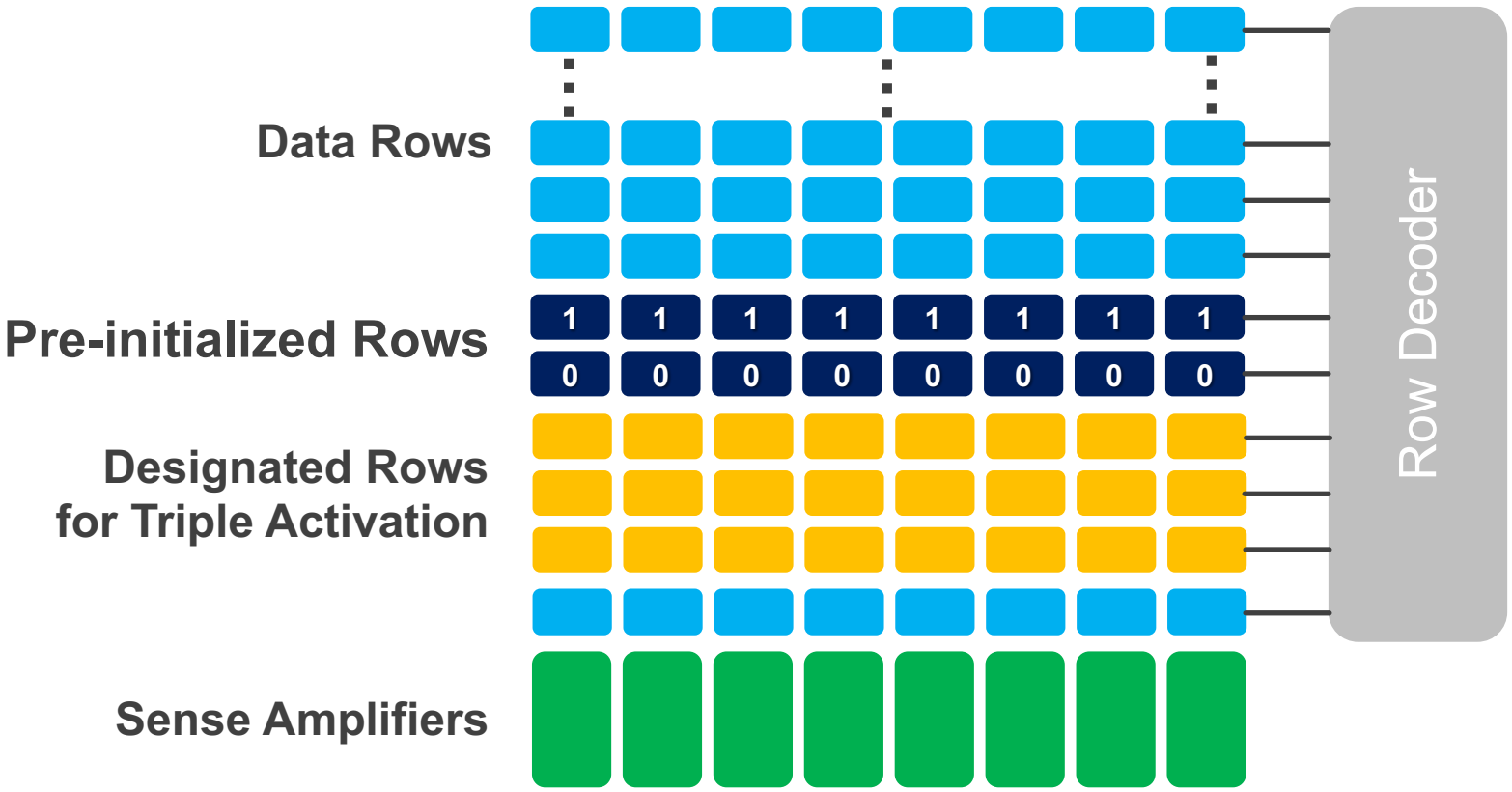
Putting It All Together

- Let's start with a normal DRAM subarray and add Ambit



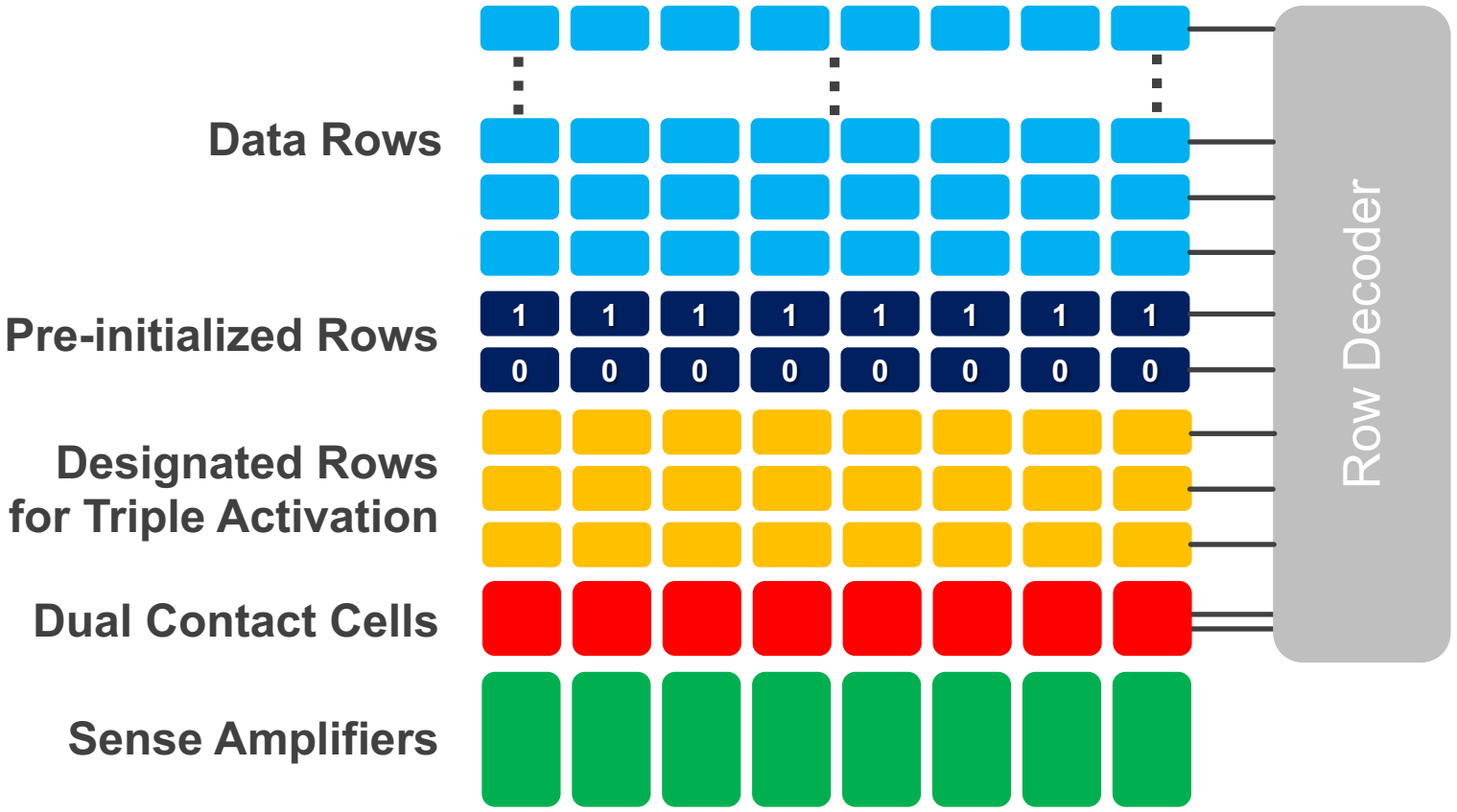
Putting It All Together

- Let's start with a normal DRAM subarray and add Ambit



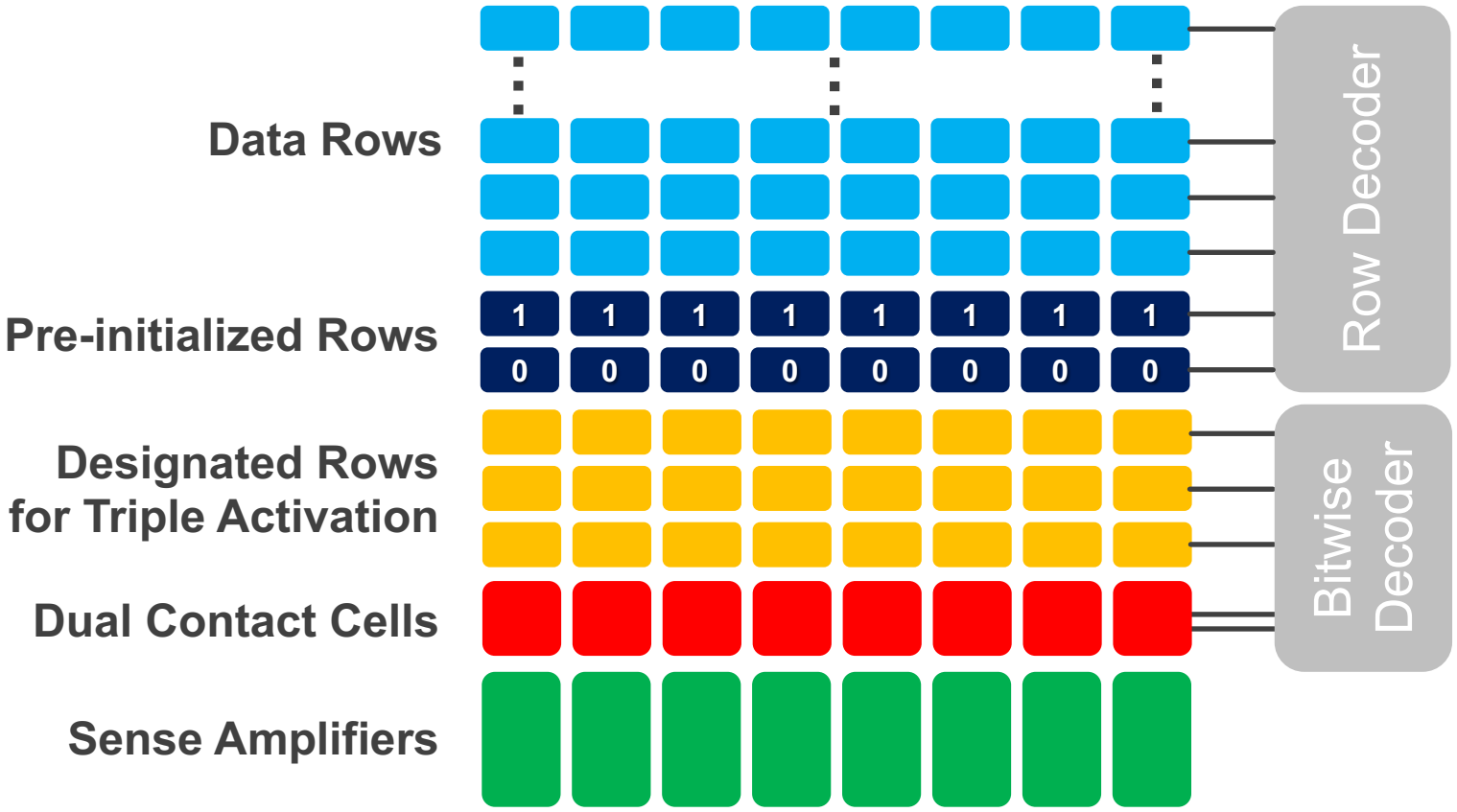
Putting It All Together

- Let's start with a normal DRAM subarray and add Ambit



Putting It All Together

- Let's start with a normal DRAM subarray and add Ambit



Putting It All Together

- Ambit AND-OR
  - At least **three rows for triple row activation**
  - 1 and 0 pre-initialized rows for operation selection
- Ambit NOT
  - At least one row of **dual-contact cells**
- Row Decoder aware of Ambit organization
  - Continuous view of normal data rows to software
  - Split to reduce complexity

Putting It All Together

- How can we integrate Ambit into a system?
  - I/O Device (PCIe)
    + Simple
    -- **Overhead** (must prepare device and retrieve data after computation)
  - Memory Bus
    + Applications can **directly trigger Ambit** operations
    + Data stays in the same memory
    + Existing **cache coherence** protocols can keep Ambit memory and on-chip cache coherent
    -- **Additions** to the rest of the **system stack**
      - ISA support
      - Ambit API/Driver

Putting It All Together

- ISA Support
  - **Machine instruction** to perform a bulk bitwise operation

    `bbop dst src1 [src2] size`
    - Size must be a multiple of the row size
    - Source(es) and destination must be row-aligned
    - If these constraints are violated, the operation is performed in the CPU

- Ambit API/Driver
  - Rows must be in the **same subarray** to use RowClone Fast Parallel Mode
  - Applications need to specify which parts of the memory are likely to be involved in bulk bitwise operations

- Cache Coherence
  - Ambit and CPU **both change memory directly**
  - Existing **DMA techniques** can be used
  - Or, `bbop` instruction could manage caches

Putting It All Together

- Combination with other accelerators
  - E.g. Hybrid Memory Cube, 3D stacked memory with a logic layer
  - We will some results of Ambit + 3D stacked memory in the next section

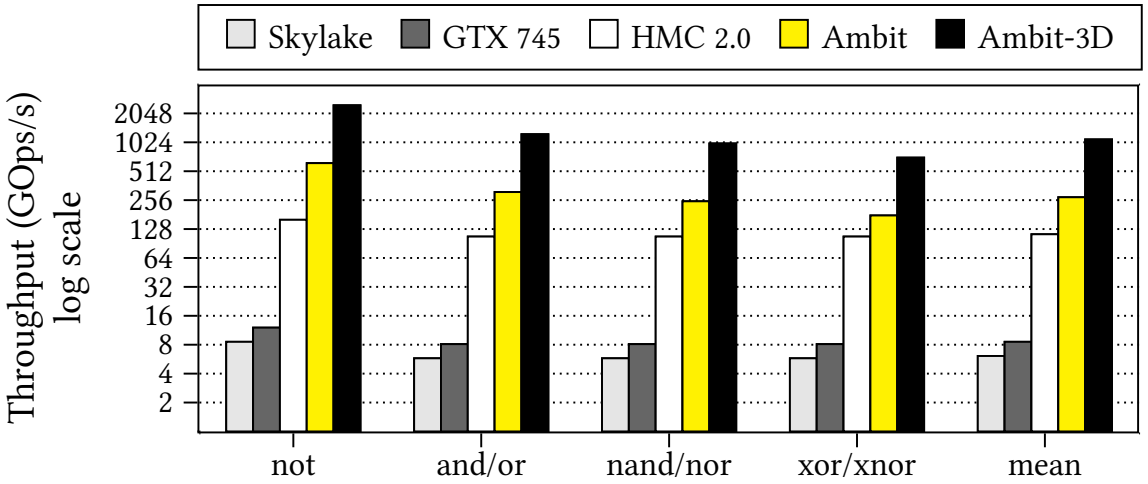
Outline

- Executive Summary
- Prerequisites
- Ambit AND-OR
- Ambit NOT
- Putting It All Together
- **Evaluation & Testing**
- Conclusion

- Strengths/Weaknesses
- Related Work
- Discussion

Evaluation & Testing

- Throughput of bulk bitwise operations



- Energy consumed by DRAM and memory channel:

  Estimated for DDR3-1333

	Design	not	and/or	nand/nor	xor/xnor
DRAM & Channel Energy (nJ/KB)	**DDR3**	93.7	137.9	137.9	137.9
	Ambit	1.6	3.2	4.0	5.5
	(↓)	59.5X	43.9X	35.1X	25.1X

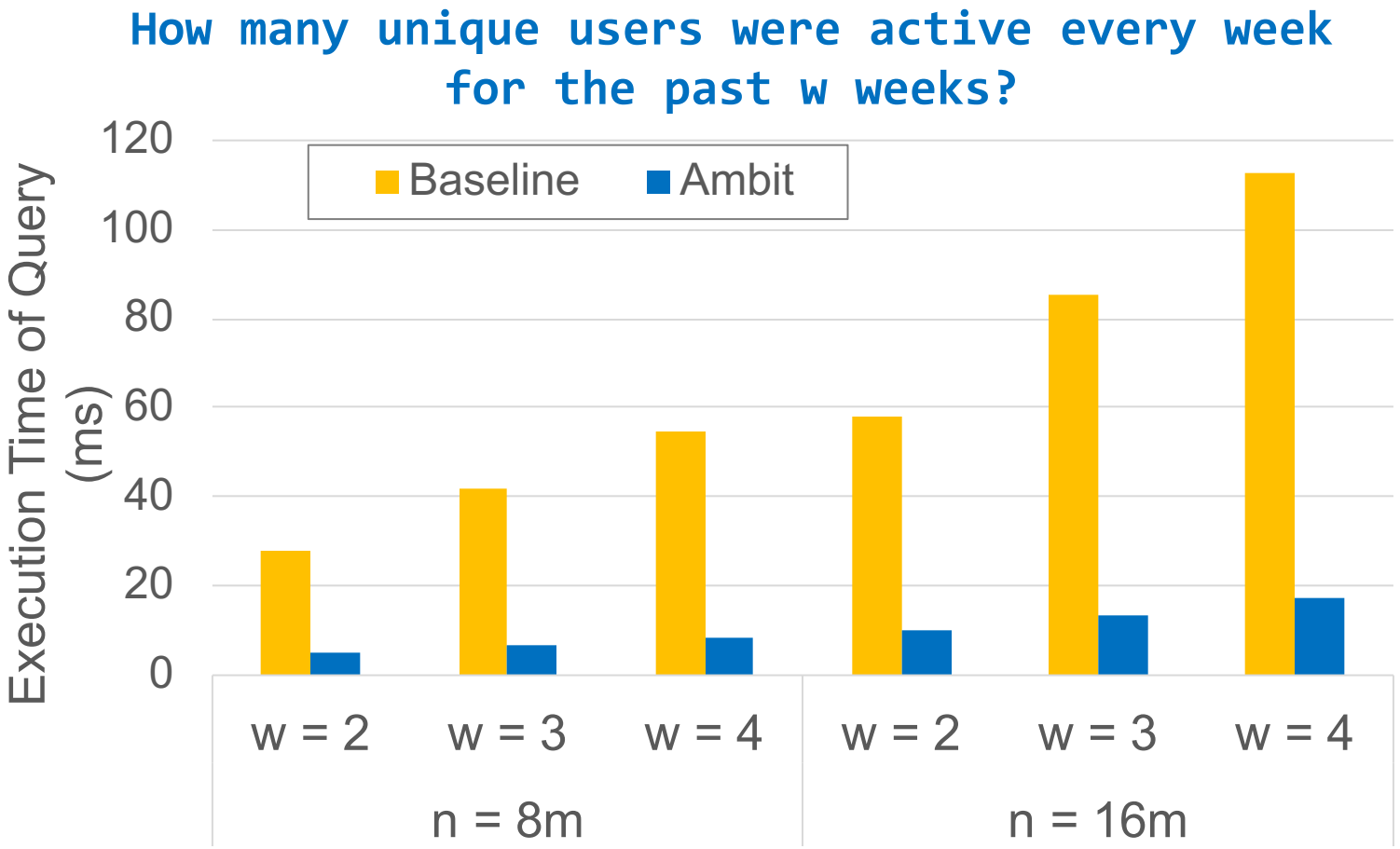
Evaluation & Testing

- Major simulation parameters

Processor:	x86, 8-wide, out-of-order, 4 Ghz
	64-entry instruction queue
L1 Cache:	32 KB D-cache, 32 KB I-cache, LRU policy
L2 Cache:	2 MB, LRU policy, 64 B cache line size
Memory Controller:	8 KB row size, FR-FCFS scheduling
Main Memory:	DDR4-2400, 1-channel, 1-rank, 16 banks

Evaluation & Testing

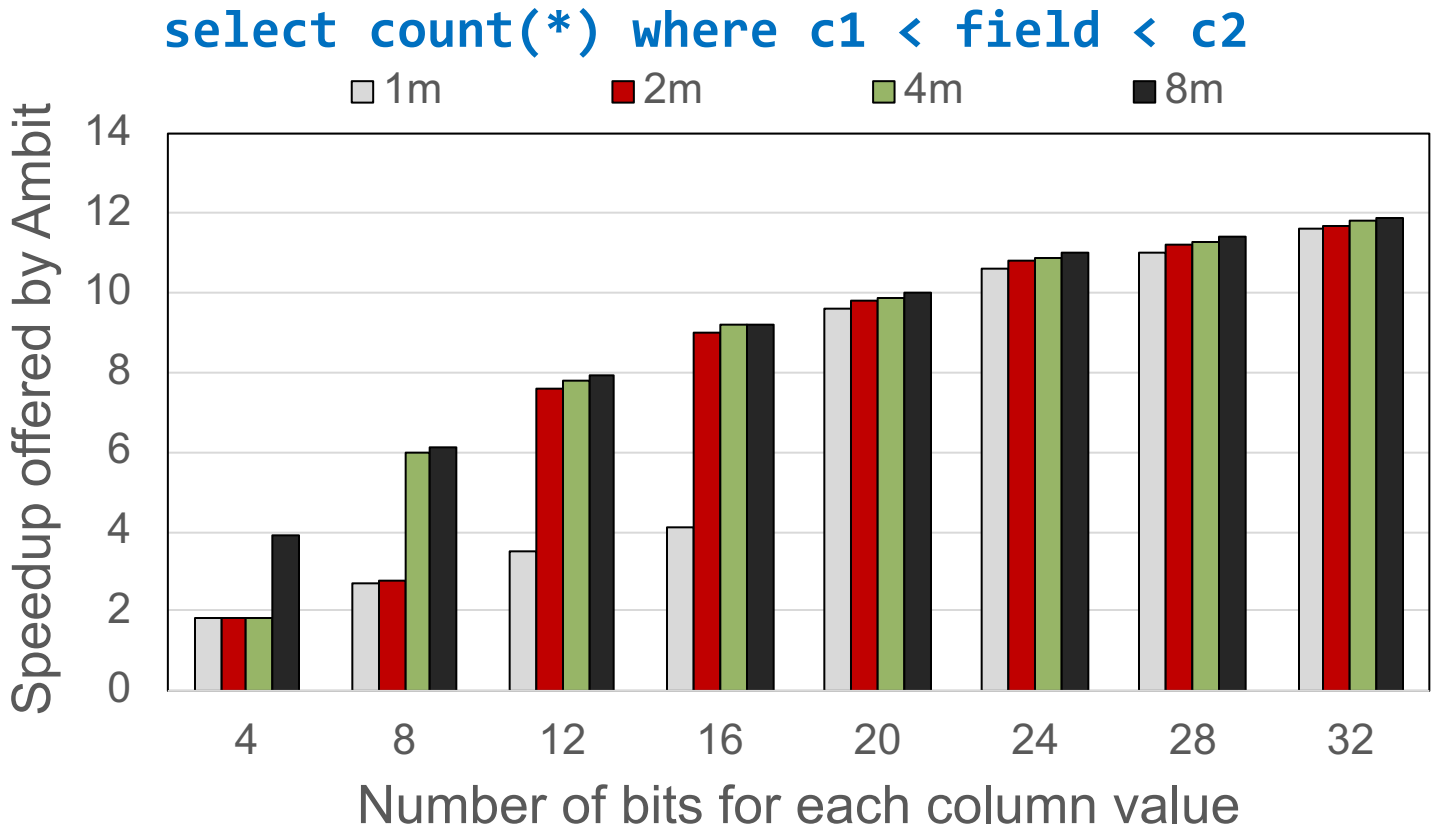
- Bitmap Indices



Source Table: https://www.archive.ece.cmu.edu/~safari/pubs/ambit-bulk-bitwise-dram_micro17-talk.pptx

Evaluation & Testing

- BitWeaving



Source Table: https://www.archive.ece.cmu.edu/~safari/pubs/ambit-bulk-bitwise-dram_micro17-talk.pptx
BitWeaving: http://pages.cs.wisc.edu/~jignesh/publ/BitWeaving.pdf

Evaluation & Testing

- All testing performed in **simulation**
- Potential Issues with Triple Row Activation
  - Cells and wires are not equal (process variation)
  - Bitline deviation may not be sufficient to trigger amplifier
- Ambit is **reliable** even in the presence of high process variation

Effect of Process Variation on TRA (n=100'000)

Variation	±0%	±5%	±10%	±15%	±20%	±25%
% Failures	0.00%	0.00%	0.29%	6.01%	16.36%	26.19%

Evaluation & Testing

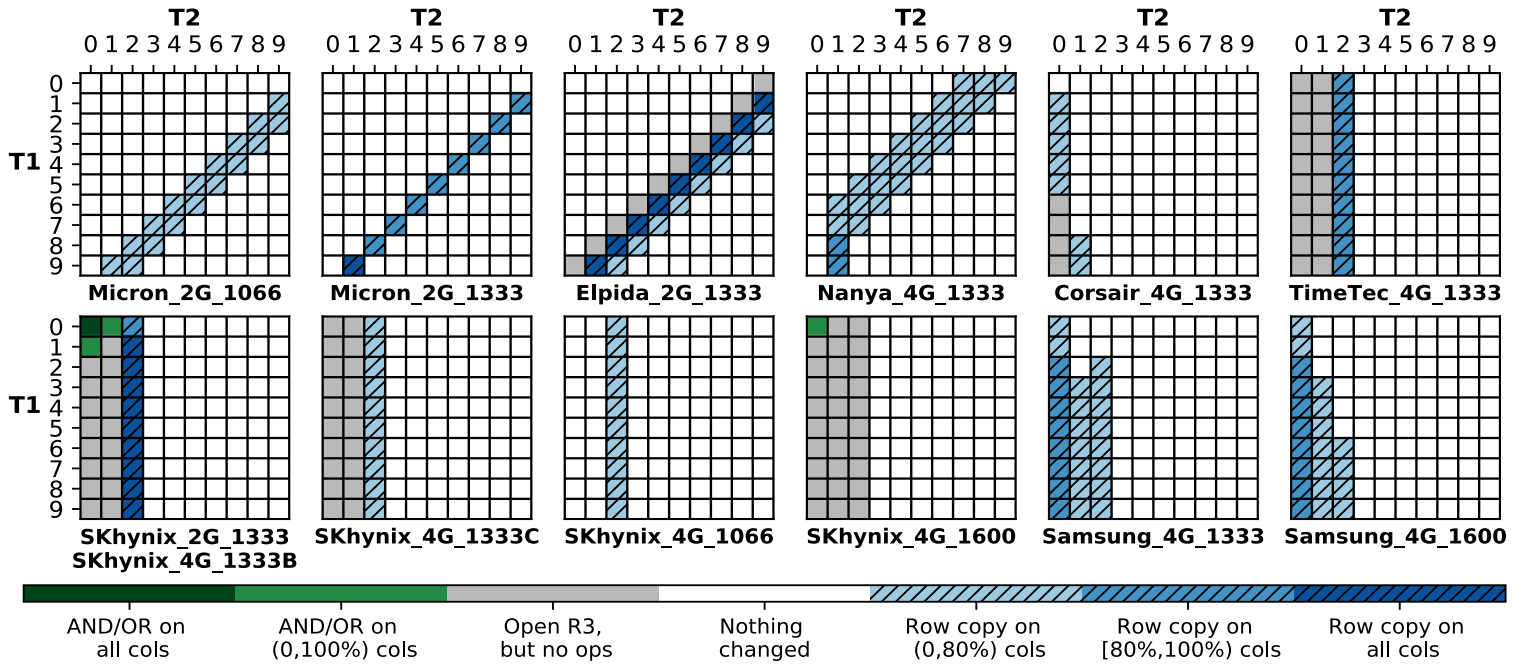
ComputeDRAM: In-Memory Compute Using Off-the-Shelf DRAMs

Fei Gao
feig@princeton.edu
Department of Electrical Engineering
Princeton University

Georgios Tziantzioulis
georgios.tziantzioulis@princeton.edu
Department of Electrical Engineering
Princeton University

David Wentzlaff
wentzlaf@princeton.edu
Department of Electrical Engineering
Princeton University

MICRO 2019



Outline

- Executive Summary
- Prerequisites
- Ambit AND-OR
- Ambit NOT
- Putting It All Together
- Evaluation & Testing
- **Conclusion**

- Strengths/Weaknesses
- Related Work
- Discussion

Conclusion

- New accelerator that can perform **any bulk bitwise operation in memory**
- Performs AND/OR operations with **Triple Row Activation**
- Uses **Dual-Contact Cells** for NOT
- **32x throughput improvement** and **35x energy reduction**
  - Translates into significant improvement for real-world data-intensive applications
- Minimal changes to hardware (<1% area cost)
- Moves **computation closer to memory** instead of memory closer to computation

Outline

- Executive Summary
- Prerequisites
- Ambit AND-OR
- Ambit NOT
- Putting It All Together
- Evaluation & Testing
- Conclusion

- **Strengths/Weaknesses**
- Related Work
- Discussion

Strengths

- Simple, novel, and effective solution
- Minimal changes to existing DRAM chips
- Can easily be integrated into systems and combined with other accelerators
- Inspired a lot of promising follow up work
- Well structured paper, Prerequisites explained

Weaknesses

- Very limited applications
- Doesn’t work with ECC memory or data scrambling mechanisms
- Up to 0.29-6.01% failures for 10-15% percent process variation
- Requires proper subarray mapping to be utilized
- Only tested in simulation

Outline

- Executive Summary
- Prerequisites
- Ambit AND-OR
- Ambit NOT
- Putting It All Together
- Evaluation & Testing
- Conclusion

- Strengths/Weaknesses
- **Related Work**
- Discussion

Related Work

ComputeDRAM: In-Memory Compute Using Off-the-Shelf DRAMs

Fei Gao
feig@princeton.edu
Department of Electrical Engineering
Princeton University

Georgios Tziantzioulis
georgios.tziantzioulis@princeton.edu
Department of Electrical Engineering
Princeton University

David Wentzlaff
wentzlaf@princeton.edu
Department of Electrical Engineering
Princeton University

MICRO 2019

Related Work

Accelerating Bulk Bit-Wise X(N)OR Operation in Processing-in-DRAM Platform

Shaahin Angizi and Deliang Fan
Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL 32816
angizi@knights.ucf.edu,dfan@ucf.edu

GraphiDe: A Graph Processing Accelerator leveraging In-DRAM-Computing

Shaahin Angizi and Deliang Fan
Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL 32816
angizi@knights.ucf.edu,dfan@ucf.edu

GLSVLSI 2019

Related Work

DRISA: A DRAM-based Reconfigurable In-Situ Accelerator

Shuangchen Li[1] Dimin Niu[2] Krishna T. Malladi[2] Hongzhong Zheng[2]
Bob Brennan[2] Yuan Xie[1]
[1]University of California, Santa Barbara [2]Samsung Semiconductor Inc.

MICRO 2017

DrAcc: a DRAM based Accelerator for Accurate CNN Inference

Quan Deng
College of Computer
National University of Defense Technology
dengquan12@nudt.edu.cn

Lei Jiang
Intelligent Systems Engineering
School of Informatics and Computing
Indiana University Bloomington
jiang60@ie.edu

Youtao Zhang
Computer Science Department
University of Pittsburgh
zhangyt@cs.pitt.edu

Minxuan Zhang
College of Computer
National University of Defense Technology
mxzhang@nudt.edu.cn

Jun Yang
Electrical and Computer Engineering Department
University of Pittsburgh
juy9@pitt.edu

DAC 2018

Related Work

Compute Caches

Shaizeen Aga, Supreet Jeloka, Arun Subramaniyan, Satish Narayanasamy, David Blaauw, and Reetuparna Das
University of Michigan, Ann Arbor
{*shaizeen, sjeloka, arunsub, nsatish, blaauw, reetudas*}*@umich.edu*

HPCA 2017

Neural Cache: Bit-Serial In-Cache Acceleration of Deep Neural Networks

Charles Eckert, Xiaowei Wang, Jingcheng Wang, Arun Subramaniyan,
Ravi Iyer†, Dennis Sylvester, David Blaauw, and Reetuparna Das
University of Michigan †Intel Corporation
{eckertch, xiaoweiw, jiwang, arunsub, dmcs, blaauw, reetudas}@umich.edu, ravishankar.iyer@intel.com

ISCA 2018

Duality Cache for Data Parallel Acceleration

Daichi Fujiki
dfujiki@umich.edu
University of Michigan

Scott Mahlke
mahlke@umich.edu
University of Michigan

Reetuparna Das
reetudas@umich.edu
University of Michigan

ISCA 2019

Outline

- Executive Summary
- Prerequisites
- Ambit AND-OR
- Ambit NOT
- Putting It All Together
- Evaluation & Testing
- Conclusion

- Strengths/Weaknesses
- Related Work
- **Discussion**

Discussion

- Any questions?
- How could the presented implementation be improved?
- Do you see any issues for Ambit as technology scales in the future?
- Is there a way to make more workloads work with Ambit?
- Is there a way to have Ambit enabled DRAM provide properties similar to ECC?
- Does Ambit create security risks?

Discussion

- Moodle:

https://moodle-app2.let.ethz.ch/mod/forum/discuss.php?d=40549