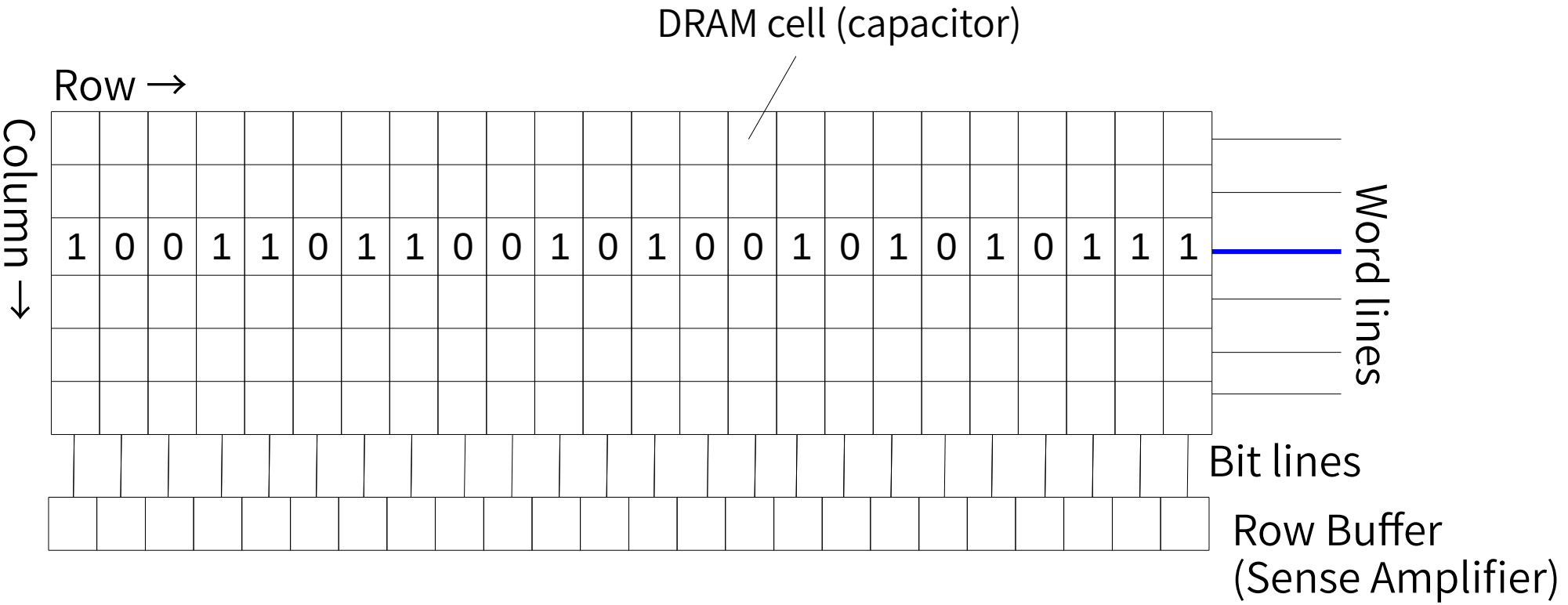
Partial Row Activation for Low-Power DRAM System (HPCA'17)

AI Research Center, AIST
Soramichi Akiyama

Background: DRAM Energy Consumption

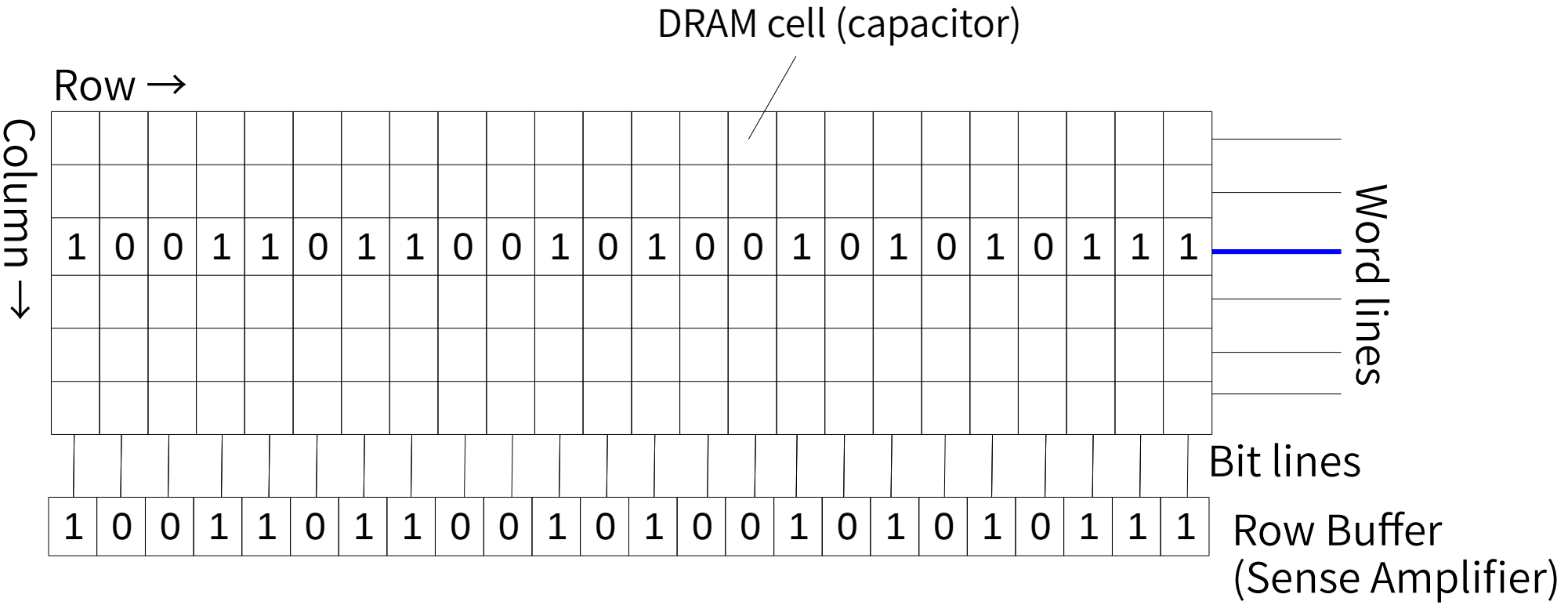
- Energy consumption is the key issue for data centers
- Demand for high-speed and large-capacity memory (e.g. big-data, simulations, deep learning)
- **25% – 57% of a system's power is used for memory subsystems**
- → This paper addresses this issue by reducing power consumption of DRAM modules

DRAM Preliminary Preliminary



(1) The memory controller “activates” a row (corresponding word line is selected)

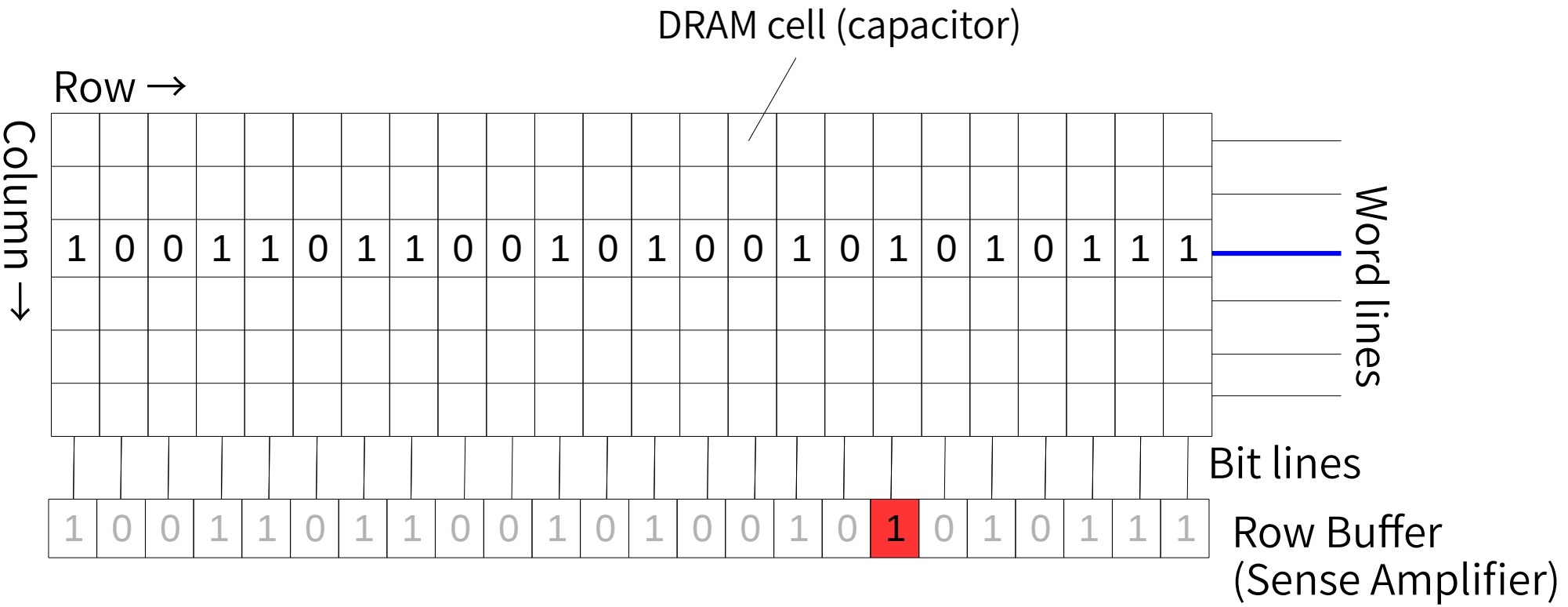
DRAM Preliminary Preliminary



(1) The memory controller “activates” a row (corresponding word line is selected)

(2) Data in the selected row is “sensed” by the row buffer via the bit lines

DRAM Preliminary Preliminary



(1) The memory controller “activates” a row (corresponding word line is selected)

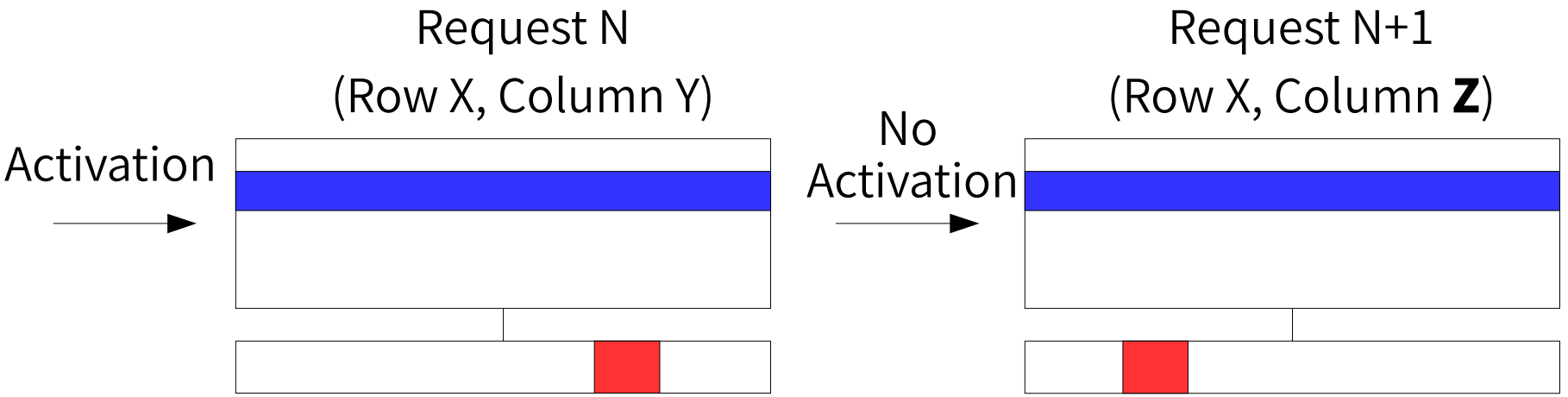
(2) Data in the selected row is “sensed” by the row buffer via the bit lines

(3) A column is selected and the data is transferred to CPU

(why not sending the whole row? → for parallelism, explained later)

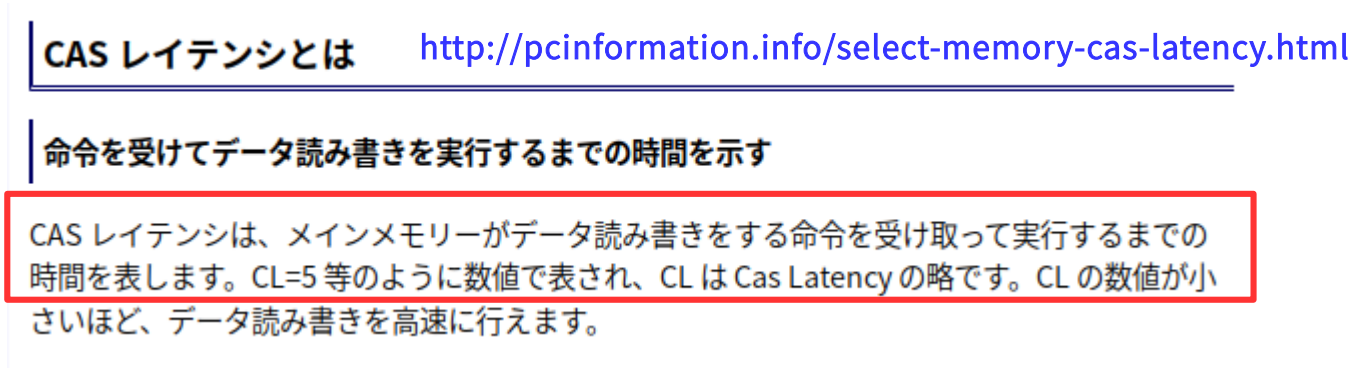
Row Buffer Hit

- Two sequential requests access the same row → The 2nd access does not need a row activation
- → Reduced energy consumption, reduced access latency



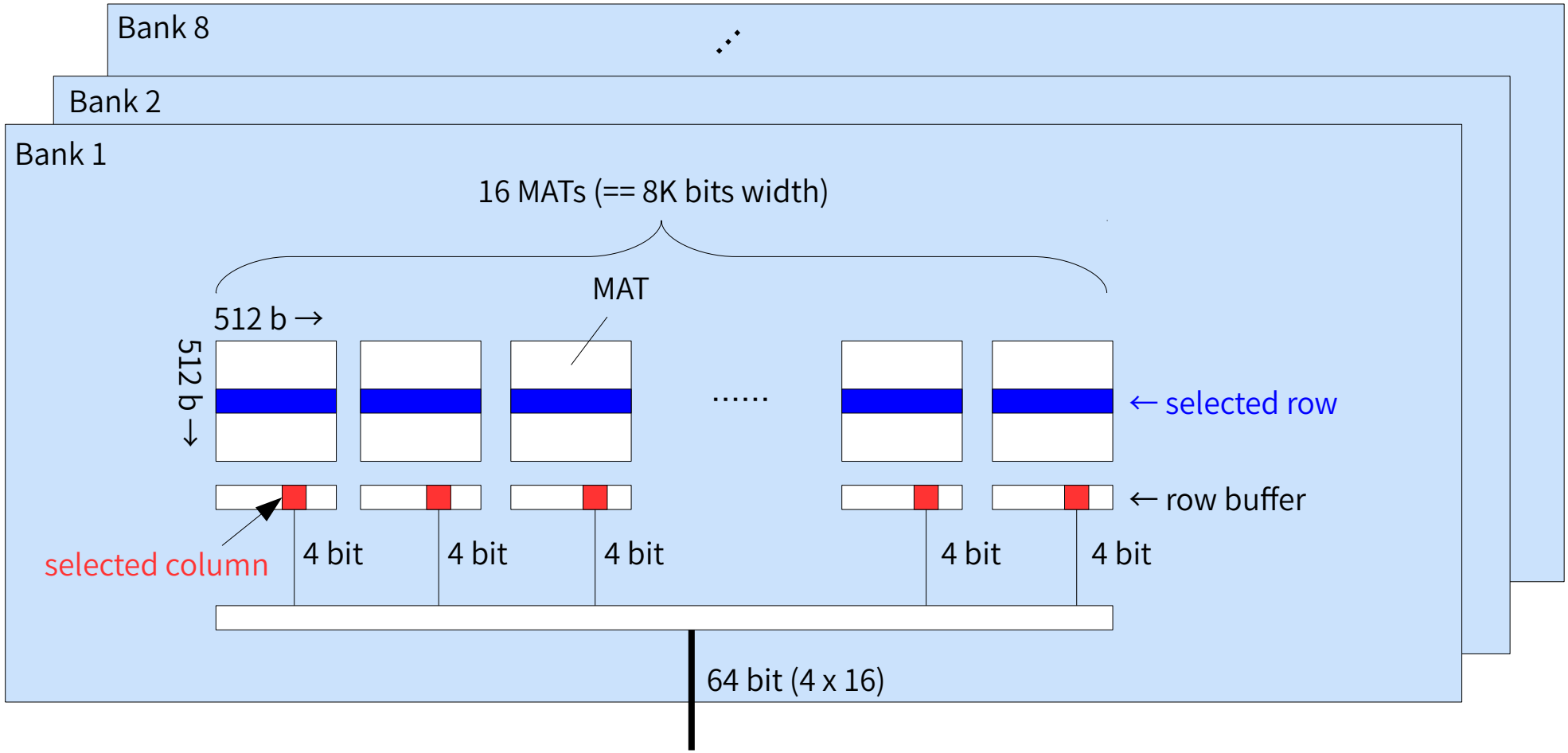
FYI (Not in the Paper): “Memory Latency”

- A widely believed myth:
  - Memory latency == CAS latency



- The truth:
  - CAS latency == DRAM module latency when row buffer hits
  - Memory latency == row activation latecy + CAS latecy + CPU-side latency (for cache misses) + …

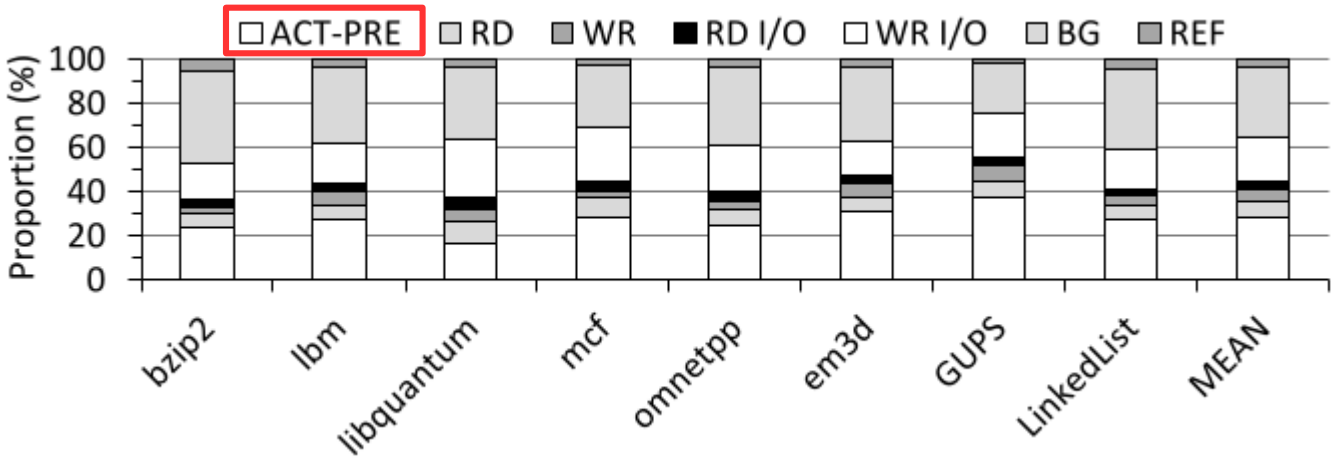
DRAM Preliminary



- Bank is divided into small MATs as building a large MAT reduces reliability
- Only 64 bits out of 8K bits “sensed” to the row buffer is transferred

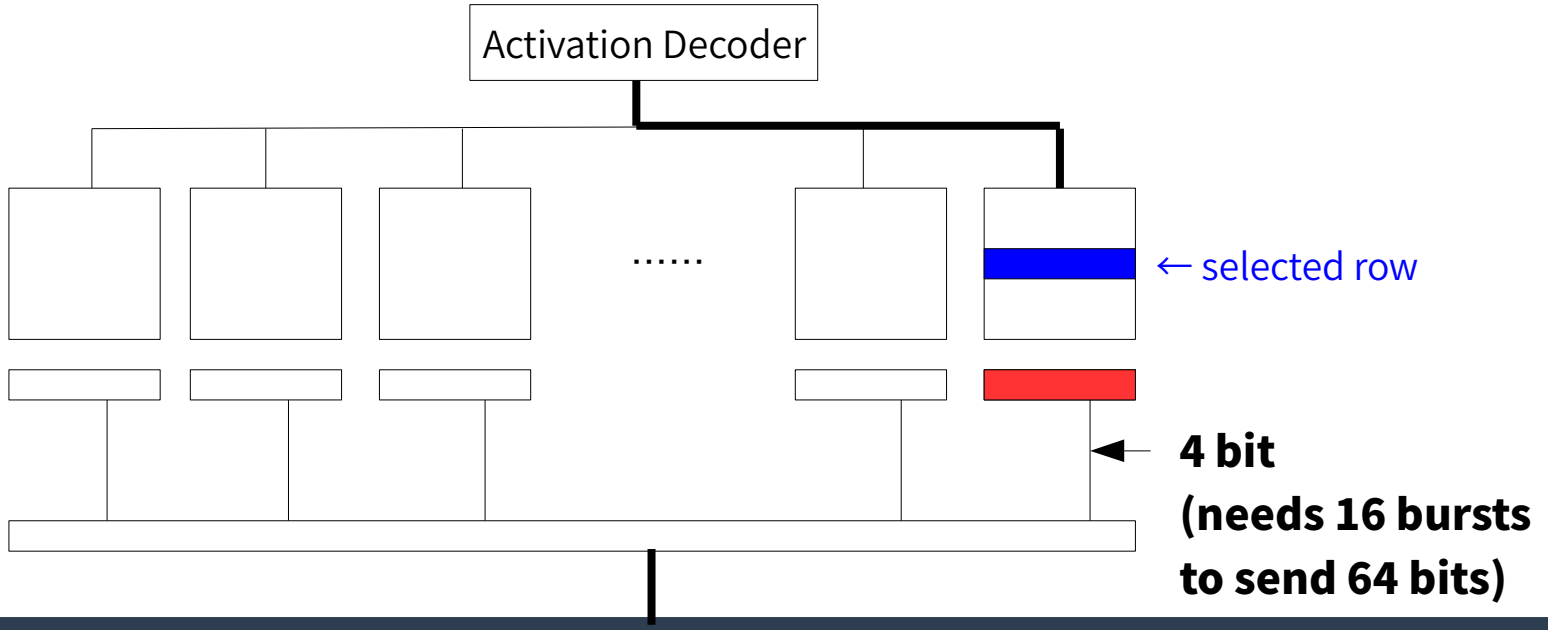
Row Overfetching Problem

- A row (8KB) is opened to access a cache line (64B)
- If row buffer hit ratio is high → next access does not require activating the same row
- Extra energy required to activate unused columns in the same row



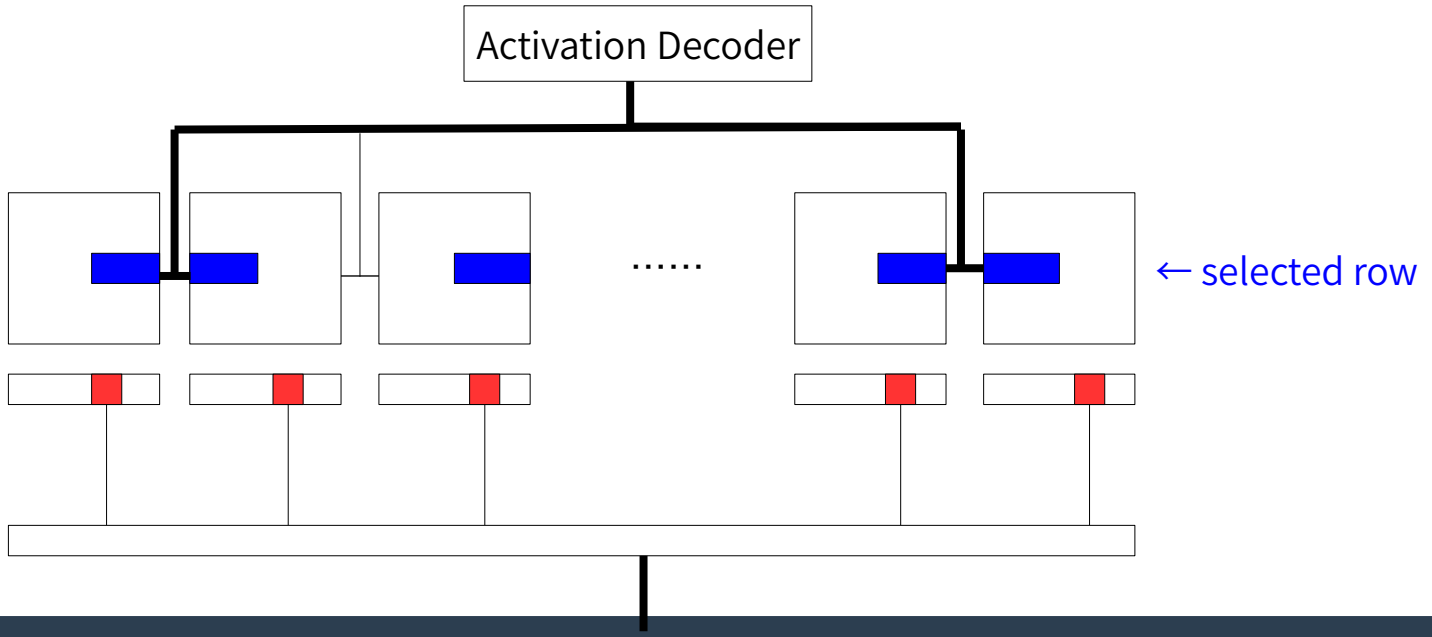
Related Work (1/2)

- Fine Grained Activation (FGA): Activate only a single MAT
  - Reduces activation energy
  - Reduces parallelism as well → performance degradation (comparison in the experiment sections)



Related Work (2/2)

- Half-DRAM: Activate Only a half (or less) of a MAT
  - Reduces activation energy w/o compromising the performance
  - Area overhead becomes large to make it finer-grained (e.g. 1/8 DRAM uses 24% of DRAM die area)



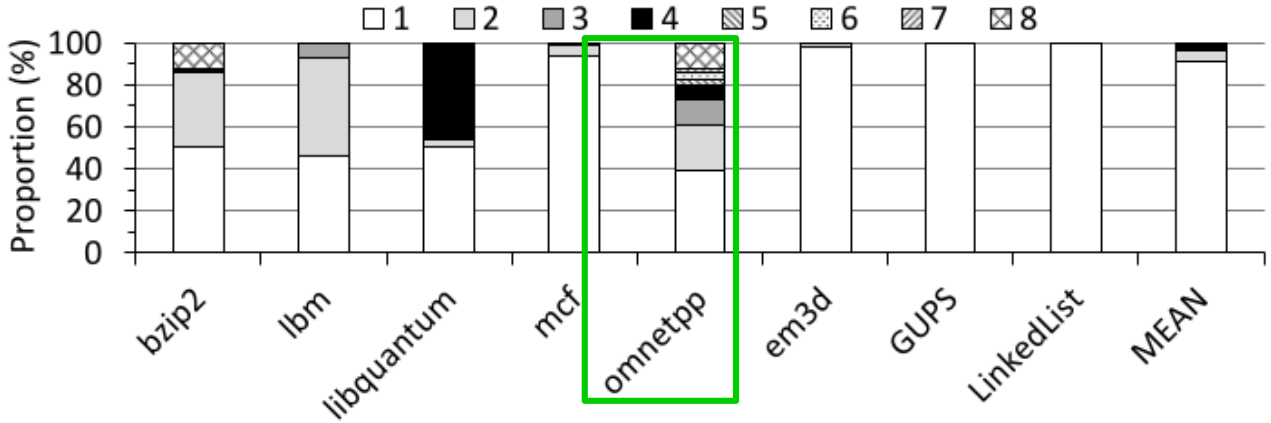
Locality Asymmetry of DRAM Read/Write

- Read and Write row buffer hit rates differ a lot
  - Read requests are critical to performance, thus memory controllers prioritize them
- → **DRAM writes waste huge energy for row activations**

Benchmark	Row buffer hit rate (%)		Memory traffic (%)		Row activation (%)	
	Read	Write	Read	Write	Read	Write
bzip2	32	1	69	31	60	40
lbm	29	18	57	43	54	46
libquantum	73	48	66	34	50	50
mcf	18	1	79	21	76	24
omnetpp	47	2	71	29	57	43
em3d	5	1	51	49	50	50
GUPS	3	1	53	47	52	48
LinkedList	4	1	65	35	64	36
average	26	9	64	36	58	42

Partial Row Activation (PRA)

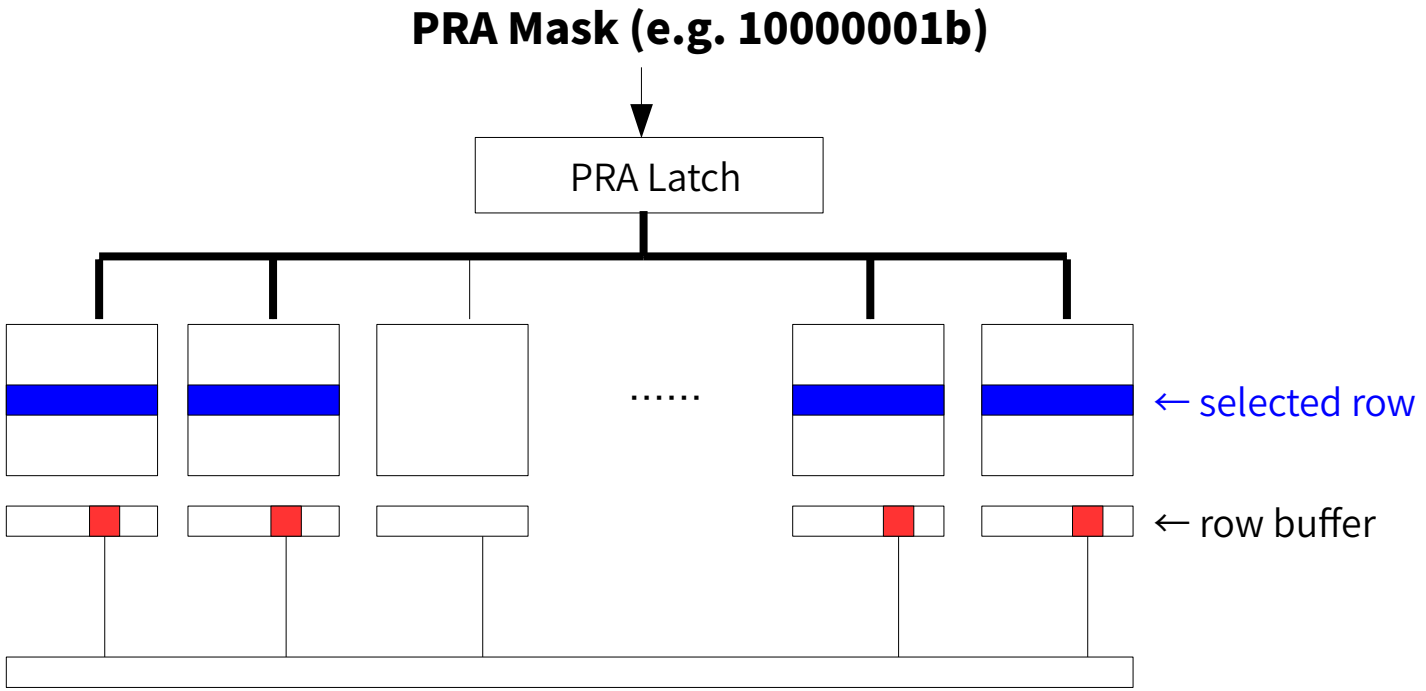
- **Idea: Distinguish reads and writes**
  - **Write accesses do not impact the performance much**
  - **Write accesses do not hit row buffer (previous slide)**
  - **The number of updated words in a cache line is small**
- → Small performance penalty, small area overhead



The number of updated words in a cache line when the cache line is written to DRAM

Hardware Design of PRA

- Write: Activate MATs corresponding to updated words, Send only updated data to the DRAM
- Read: Activate all MATs to leverage the parallelism



* There are 16 MATs each corresponding to 4 bits, thus 1 bit in the PRA mask activates 2 MATs

Overhead Analysis

- An extra pin required for PRA command
  - Commodity DRAM chips have unused pins → Not a problem
- Fine-grained dirty bit (FGD)
  - CPU caches must have expanded 8-bit dirty bites (instead of 1bit normal dirty bit) for each to tell which word is updated
  - L1 cache suffers 0.31% more area, 0.12% more energy, and 1.26% more leakage power → Not a problem as accessing DRAM uses 250x energy than accessing cache (cache overhead invisible)

* I don't really know the real hardware internals (pins, chips, area, ...), thus the contents of this slide might have mistakes

Experiment: Methodology (1/2)

- Energy Model
  - CACTI-3DD, Memory system power calculator (Micron)
  - Power (Portion of a row activated: power)
    - Full: 22.2 mW, 7/8: 18.6 mW, 6/8: 16.9 mW, 5/8: 14.3 mW, 4/8: 11.6 mW, 3/8: 9.1 mW, 2/8: 6.4 mW, 1/8: 3.7 mW
- Workload

MIX1	bzip2, lbm, libquantum, omnetpp
MIX2	mcf, em3d, GUPS, LinkedList
MIX3	bzip2, mcf, lbm, em3d
MIX4	libquantum, GUPS, omnetpp, LinkedList
MIX5	bzip2, LinkedList, lbm, GUPS
MIX6	libquantum, em3d, omnetpp, mcf

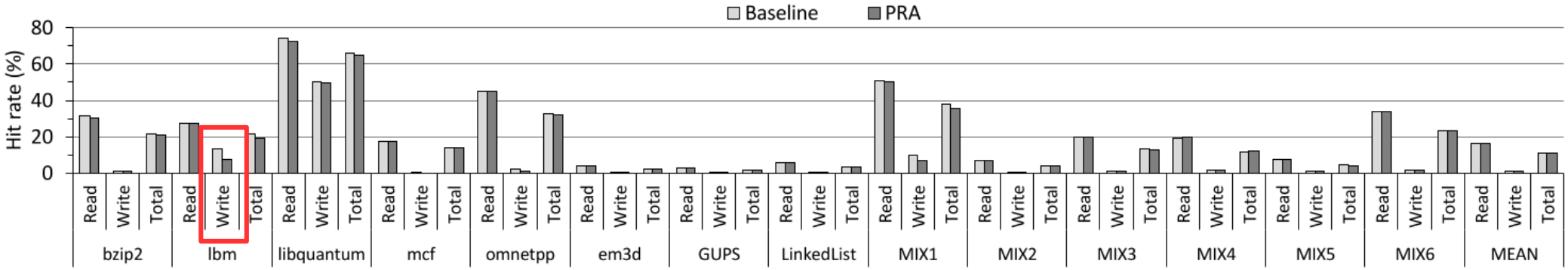
Executed on a cycle accurate simulator (gem5 + DRAMSim2)

Experiment: Methodology (2/2)

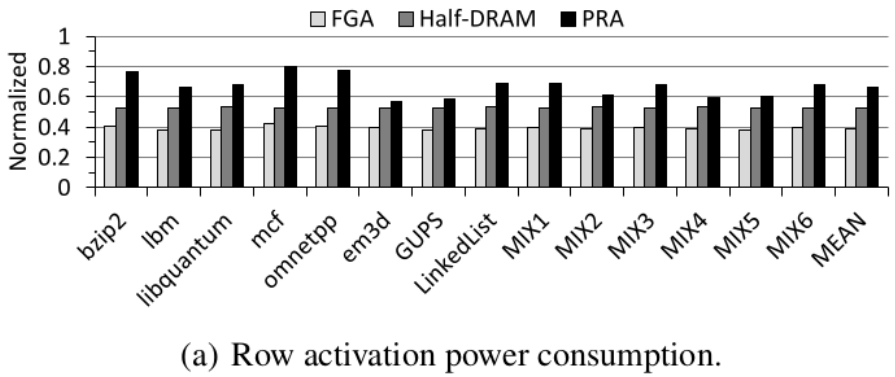
- Enemies (Existing methods)
  - FGA: Activate only one MAT at a time (reduced energy, reduced memory parallelism → workload performance penalty)
    - FGA with 4-MATs granularity (4 MATs are activated) is used, as FGA with one-MAT granularity is too slow
  - Half-DRAM: Divide MATs, activate the halves (reduced energy w/o performance loss, large area overhead)
- Baseline (for normalization)
  - Relaxed close-policy: an activated row is closed when there is no requests that hit the row buffer in the read/write queues

False Row Buffer Hit

- PRA reduces row buffer hit rate due to false row buffer hit
  - A write opens 1st MAT → Next read opens 2nd MAT of the same row
  - A write opens 1st MAT → Next write opens 2nd MAT of the same row
  - These cases hit the row buffer in normal DRAM, but miss it in PRA
  - A false row buffer hit delays the request / increases energy consumption due to an extra activation

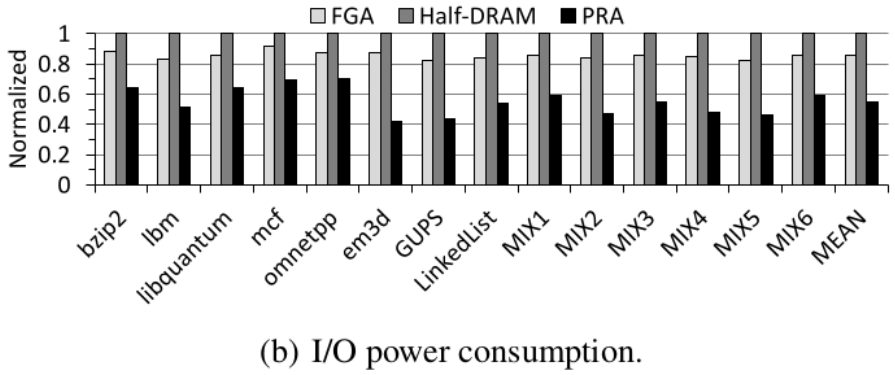


Energy Consumption



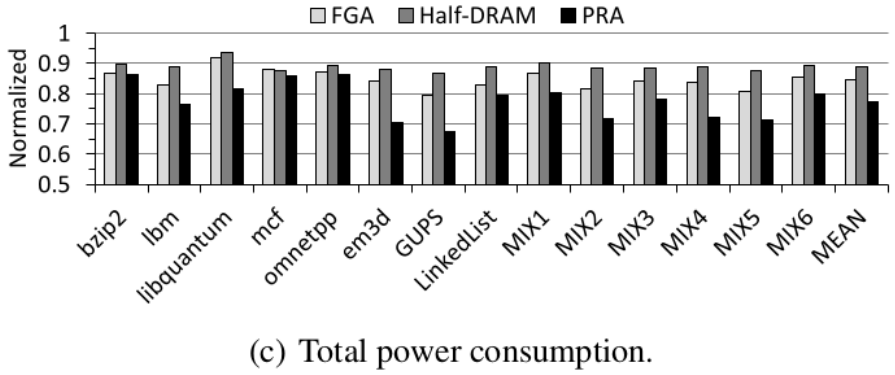
(a) Row activation power consumption.

- PRA consumes more row activation energy than existing methods
- More energy than FGA is obvious, why more energy than half-DRAM??



(b) I/O power consumption.

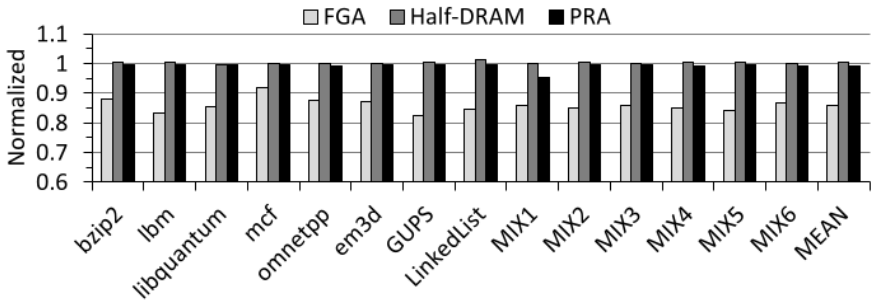
- PRA consumes less IO power consumption than existing methods
- Existing ones send/receive a full row to/from the DRAM even if a part of it is opened



(c) Total power consumption.

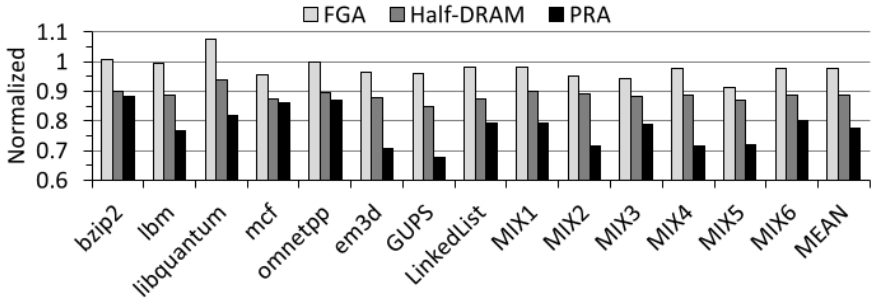
- PRA consumes less energy in total than existing methods
- The sum of the effects by (a) and (b))

Total Performance



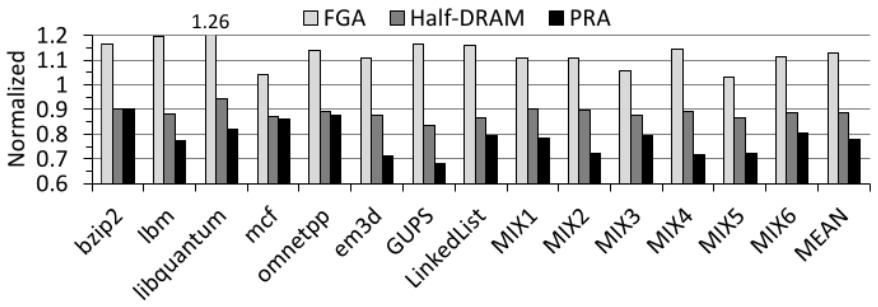
(a) Normalized performance.

- PRA degrades the workload performance due to false row buffer hit, but the effect is small (ave: 0.8%, max: 4.8%)
- FGA degrades the performance a lot



(b) DRAM energy consumption.

- PRA reduces the power consumption for the DRAM modules (the difference btw the figure in the previous slide??)



(c) Energy-delay product (EDP).

- PRA achieves the best EDP compared to the existing methods → Total performance is the best in PRA

* EDP := Power x Perf degradation (joule)

Conclusion

- Increased demand for high-speed, large-capacity memory → Increased energy consumption for memory subsystem
- A memory access activates a whole row (8K bytes) to access a cache line (64 bytes) for parallelism
  - Write accesses enjoy row buffer hit ratio → activating a whole row is a huge waste of energy
- Partial Row Activation: activates only a portion of a row for write accesses to reduce energy while not sacrificing the performance