DRAM Tutorial

18-447 Lecture

Vivek Seshadri
DRAM Module and Chip
Goals

• Cost
• Latency
• Bandwidth
• Parallelism
• Power
• Energy
DRAM Chip
Sense Amplifier

Inverter

top

bottom

enable
Sense Amplifier – Two Stable States

Logical “1”

Logical “0”

$V_{DD}$
Sense Amplifier Operation

\[ V_T > V_B \]
DRAM Cell – Capacitor

- **Empty State**
  - Logical “0”
  - Small – Cannot drive circuits
  - Reading destroys the state

- **Fully Charged State**
  - Logical “1”
Capacitor to Sense Amplifier

Diagram showing the conversion process from a capacitor to a sense amplifier, with voltage levels and symbols indicating the flow of information.
DRAM Cell Operation

\[ \frac{1}{2} V_{DD} + \delta \]

\[ \frac{1}{2} V_{DD} \]
DRAM Subarray – Building Block for DRAM Chip

Row Decoder

Cell Array

Array of Sense Amplifiers (Row Buffer) 8Kb

Cell Array
DRAM Bank

Row Decoder

Cell Array

Array of Sense Amplifiers (8Kb)

Cell Array

Cell Array

Array of Sense Amplifiers

Cell Array

Bank I/O (64b)

Address

Data
DRAM Chip

Shared internal bus

Memory channel - 8bits
DRAM Operation

1. **ACTIVATE Row**
2. **READ/WRITE Column**
3. **PRECHARGE**

Diagram:
- Row Address
- Column Address
- Bank I/O
- Cell Array
- Array of Sense Amplifiers
- Data
RowClone

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

Vivek Seshadri

Memory Channel – Bottleneck

Limited Bandwidth

Core

Cache

MC

Channel

Memory

High Energy
Goal: Reduce Memory Bandwidth Demand

Reduce unnecessary data movement
Bulk Data Copy and Initialization

Bulk Data Copy

Bulk Data Initialization

src \[\rightarrow\] dst

val \[\rightarrow\] dst
Bulk Data Copy and Initialization

The Impact of Architectural Trends on Operating System Performance
Mendel Rosenblum, Edouard Bugnion, Stephen Alan Herrod,
Emmett Witchel, and Anoop Gupta

Hardware Support for Bulk Data Movement in Server Platforms
Li Zhao†, Ravi Iyer‡ Srihari Makineni‡, Laxmi Bhuyan‡ and Don Newell‡
†Department of Computer Science and Engineering, University of California, Riverside, CA 92521
Email: {zhao, bhuyan}@cs.ucr.edu
‡Communications Technology Lab, Intel Corp.

Architecture Support for Improving Bulk Memory Copying and Initialization Performance
Xiaowei Jiang, Yan Solihin
Dept. of Electrical and Computer Engineering
North Carolina State University
Raleigh, USA

Li Zhao, Ravishankar Iyer
Intel Labs
Intel Corporation
Hillsboro, USA
Bulk Copy and Initialization – Applications

- Forking
- Zero initialization (e.g., security)
- Checkpointing
- VM Cloning
- Deduplication
- Page Migration
- Many more
Shortcomings of Existing Approach

- High latency (1046 ns to copy 4 KB)
- High energy (3600 nJ to copy 4 KB)
- Interference
Our Approach: In-DRAM Copy with Low Cost

- Core
- Cache
- MC
- Channel
- High Energy
- High latency
- Interference

Vivek Seshadri – Thesis Proposal
RowClone: In-DRAM Copy
Two Key Observations

1. Any operation on one sense amplifier can be easily performed in bulk.

2. Many DRAM cells share the same sense amplifier.
Data gets copied

Bulk Copy in DRAM – RowClone

$\frac{1}{2}V_{dd} + \delta$

$\frac{1}{2}V_{dd}$
Fast Parallel Mode – Benefits

Bulk Data Copy (4KB across a module)

- **Latency**: 1046ns to 90ns (11X)
- **Energy**: 3600nJ to 40nJ (74X)

- No bandwidth consumption
- Very little changes to the DRAM chip
Fast Parallel Mode – Constraints

- Location constraint
  - Source and destination in same subarray
- Size constraint
  - Entire row gets copied (no partial copy)

1. Can still accelerate many existing primitives
   (*copy-on-write, bulk zeroing*)

2. Alternate mechanism to copy data across banks
   (*pipelined serial mode – lower benefits than Fast Parallel*)
End-to-end System Design

• Software interface
  – `memcpy` and `meminit` instructions

• Managing cache coherence
  – Use existing DMA support!

• Maximizing use of Fast Parallel Mode
  – Smart OS page allocation
Applications Summary

The diagram illustrates the fraction of memory traffic for different applications. The applications shown are:

- bootup
- compile
- forkbench
- mcached
- mysql
- shell

The diagram uses four categories to represent the fraction of memory traffic:

- Zero
- Copy
- Write
- Read

The chart visually shows the proportion of each category for each application.
Results Summary

- IPC Improvement
- Memory Energy Reduction

Compared to Baseline:

- bootup
- compile
- forkbench
- mcached
- mysql
- shell