ADVANCED MEMORY SYSTEMS

Mahdi Nazm Bojnordi

Assistant Professor

School of Computing

University of Utah

THE UNIVERSITY OF UTAH

CS/ECE 6810: Computer Architecture

Programmable Controller

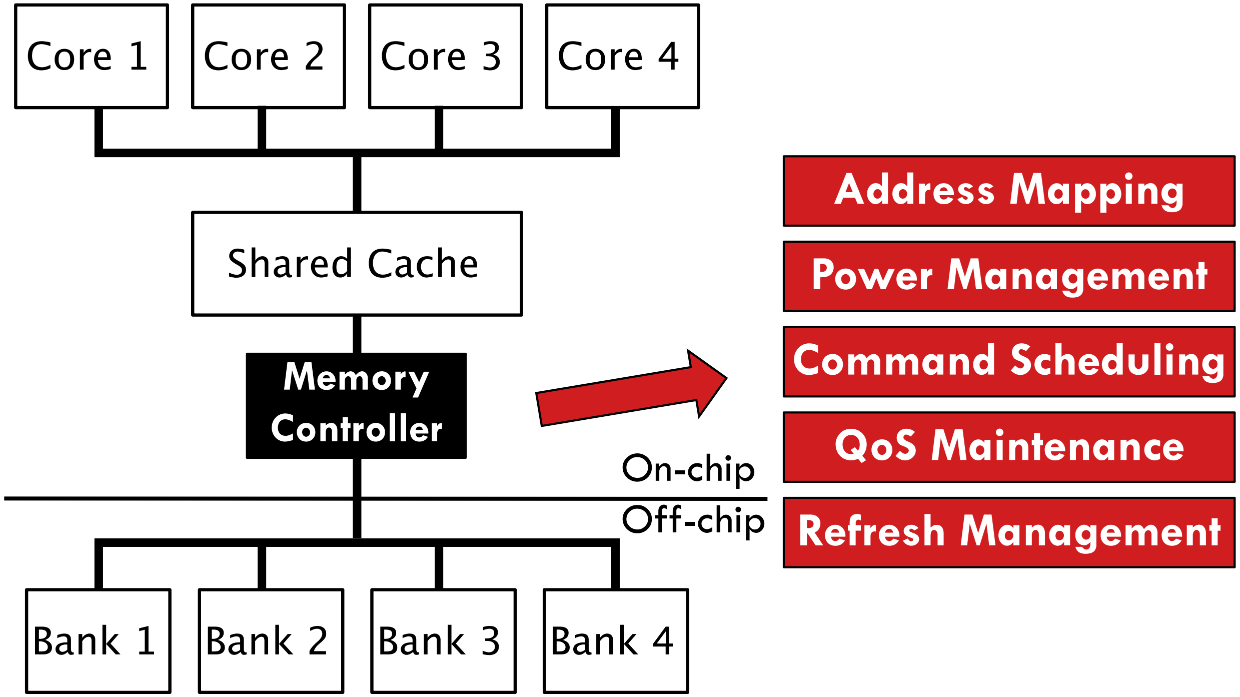
Limitations to Existing Memory Controllers

- Modern memory controllers are performance-critical and complex

Multiple performance objectives ✗

Application-specific optimizations ✗

Patches and in-field updates ✗



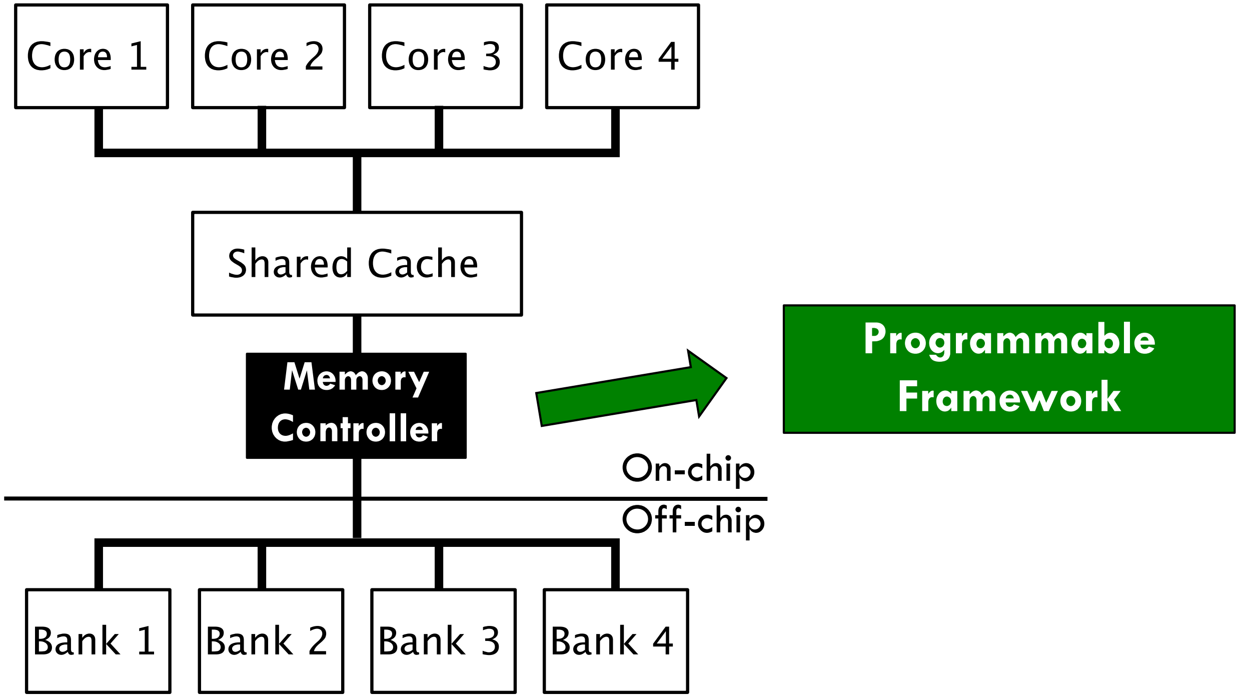
Programmable Memory Controllers

- Programmability can make a memory controller higher-performance and more flexible

Multiple performance objectives ✓

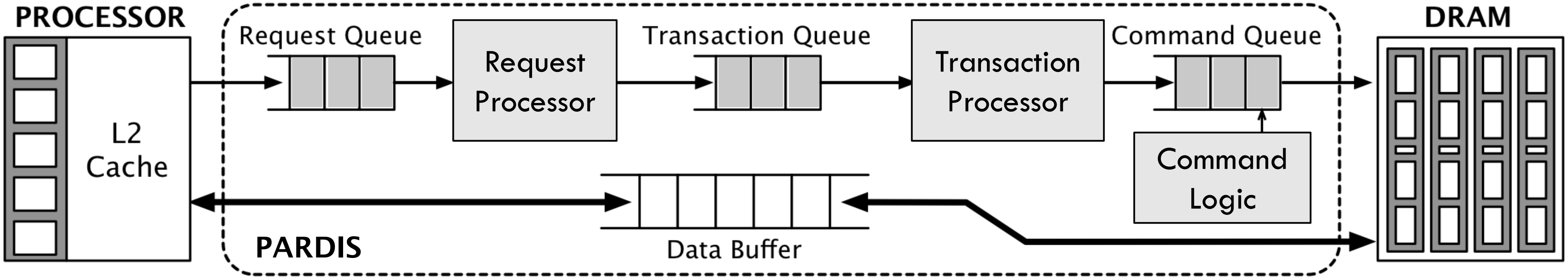
Application-specific optimizations ✓

Patches and in-field updates ✓



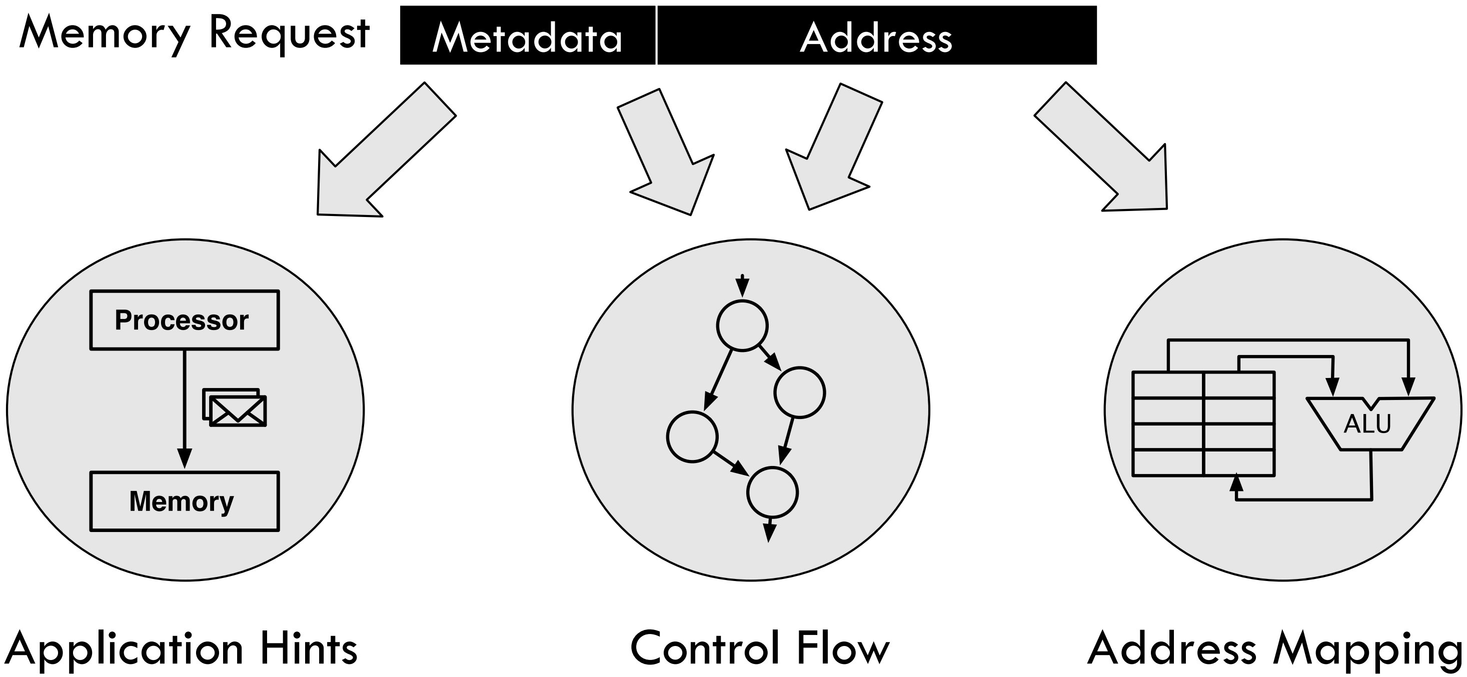
Design Overview

- Key idea: Judicious division of labor between specialized hardware and firmware
  - Request and transaction processing in firmware
  - Configurable timing validation in hardware



Request Processing

- A RISC ISA for operating on memory requests



Memory Request

Metadata

Address

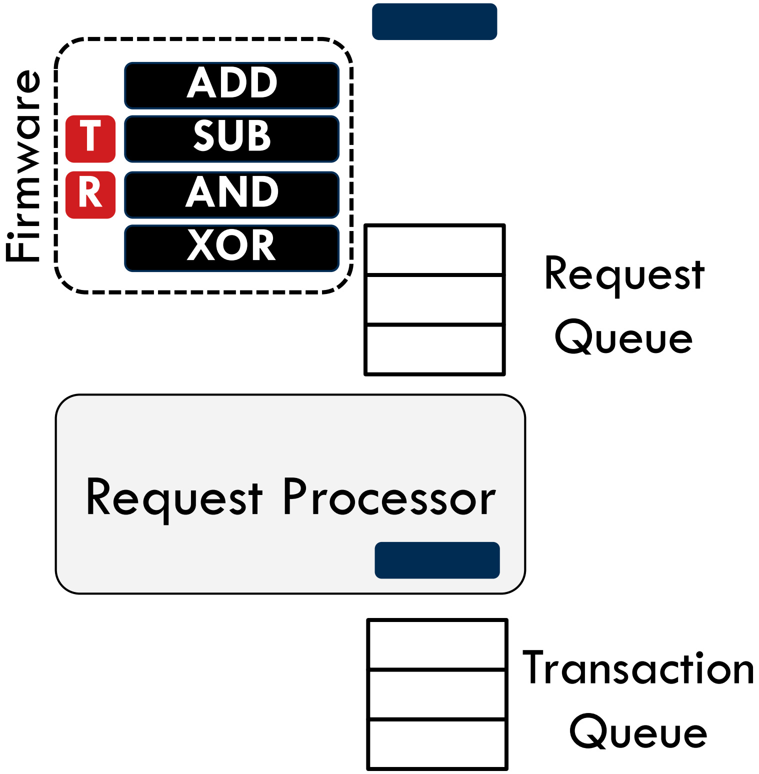
Application Hints

Control Flow

Address Mapping

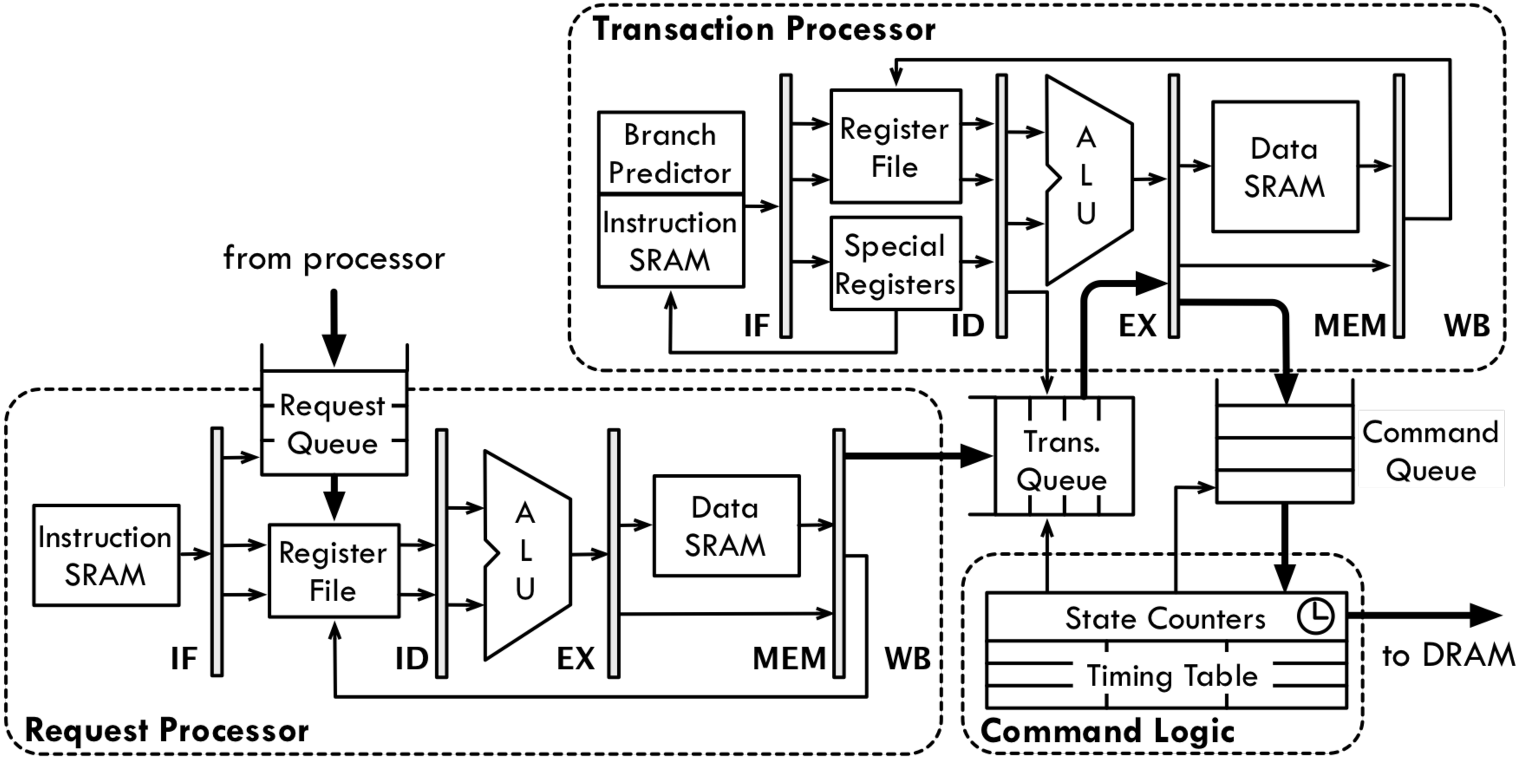
Request Processing

- Queue management with instruction flags
  - R flag enqueues a request
  - T flag dequeues a transaction
- An instruction can be annotated with both R and T flags if needed



Implementation

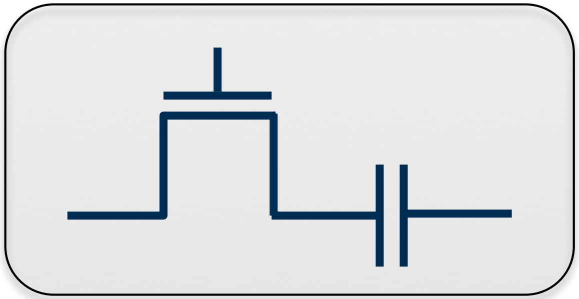
- Two five-stage pipelines and one configurable timing validation circuit



Emerging Technologies

DRAM Cell Structure

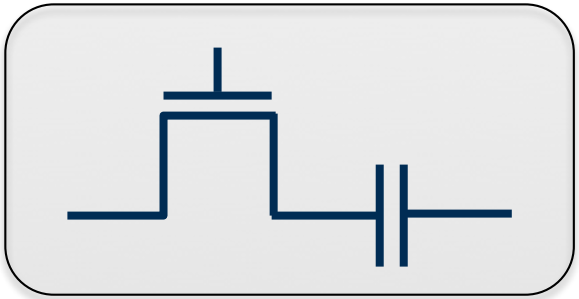
- One-transistor, one-capacitor
  - Realizing the capacitor is challenging



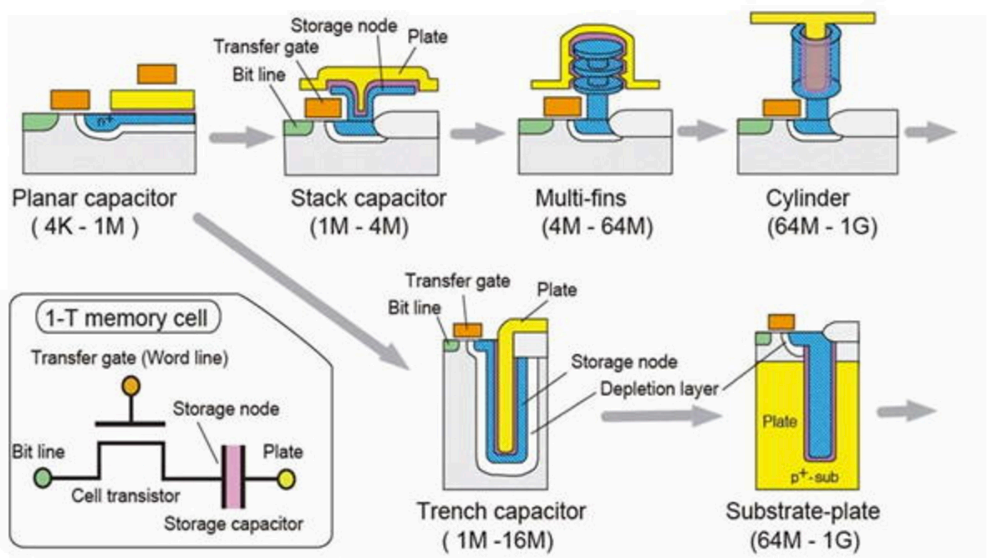
- 1T-1C DRAM
- Charge based sensing
- Volatile

DRAM Cell Structure

- One-transistor, one-capacitor
  - Realizing the capacitor is challenging



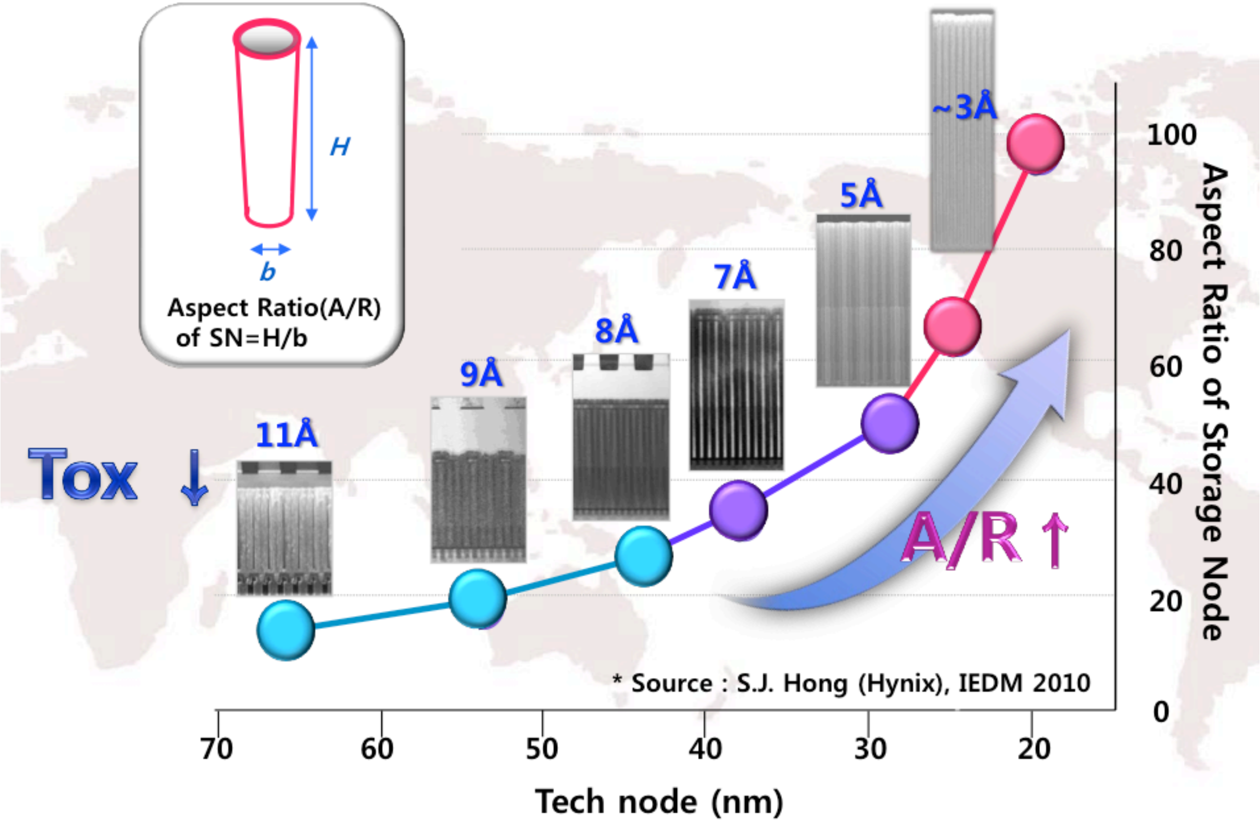
- 1T-1C DRAM
- Charge based sensing
- Volatile



Memory Scaling in Jeopardy

Scaling of semiconductor memories greatly challenged beyond 20nm

Example: DRAM

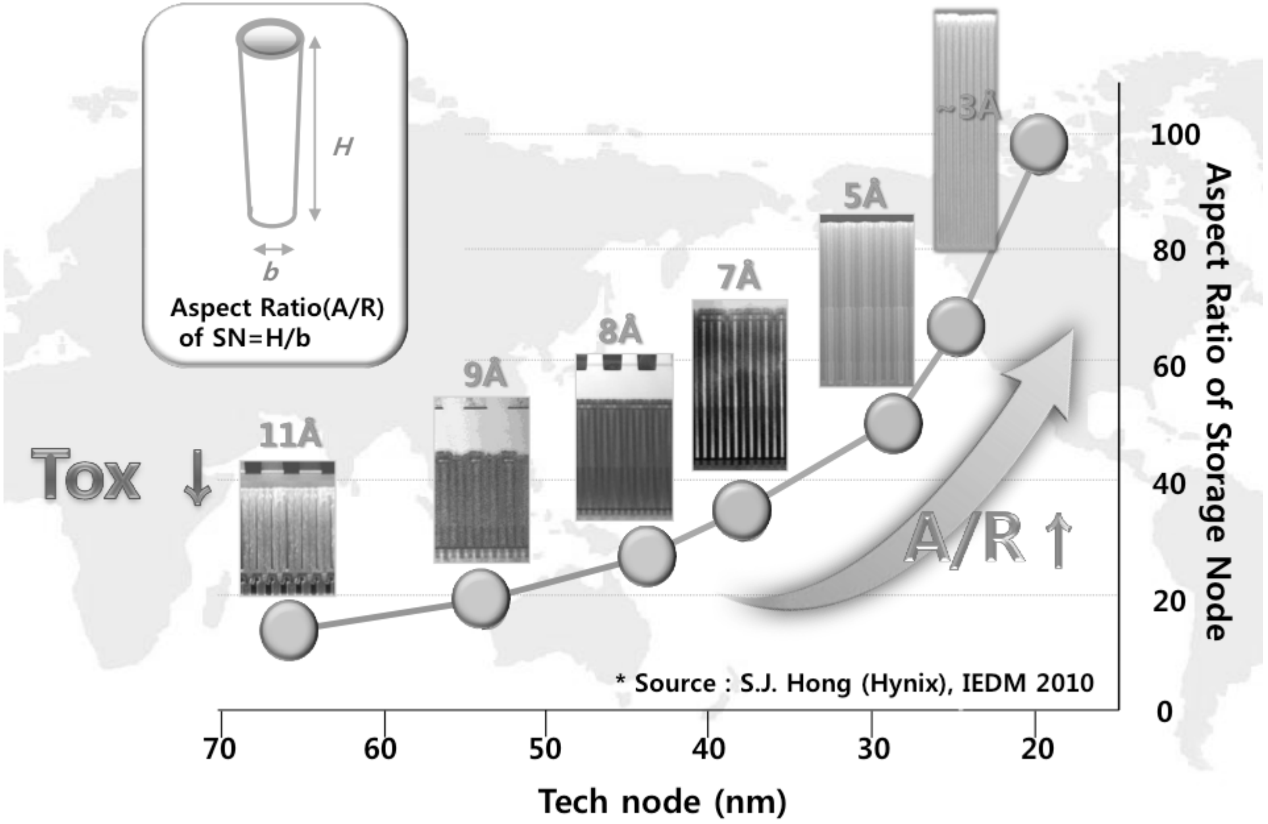


* Source : S.J. Hong (Hynix), IEDM 2010

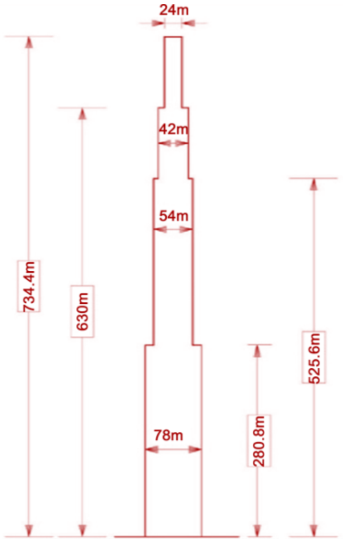
Memory Scaling in Jeopardy

Scaling of semiconductor memories greatly challenged beyond 20nm

Example: DRAM



* Source : S.J. Hong (Hynix), IEDM 2010

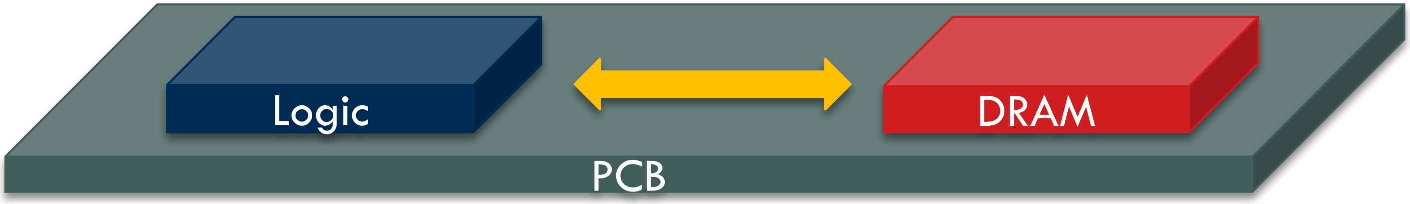


A/R < 10

Why DRAM Slow?

- Logic VLSI Process: optimized for better transistor performance

- DRAM VLSI Process: optimized for low cost and low leakage



How to reduce distance?

Processing-in-Memory

- Increasing bandwidth by placing processing units on same die with DRAM
- Not a new concept!
  - Merged Logic and DRAM (MLD)
    - IBM, Mitsubishi, Samsung, Toshiba, etc.
  - Other efforts
    - FlexRAM
    - IRAM
    - Active Pages
    - …

Historical PIM Challenges

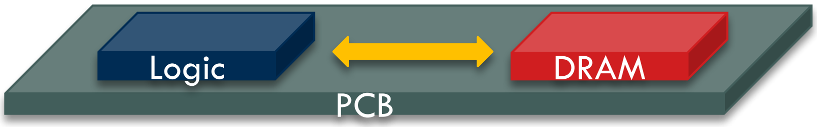
- Hard to program (no standard interface)

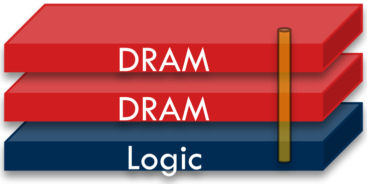
- Embedding logic on modified DRAM process
  - Substantially larger transistors
    - Reduce memory capacity
  - Slower logic and lower performance

- Embedding DRAM on modified logic process
  - Leaky transistors, high refresh rates, increased cost/bit
  - Increased manufacturing complexity

3D Die-Stacking

- Different devices are stacked on top of each other
- Layers are connected by through-silicon vias (TSVs)

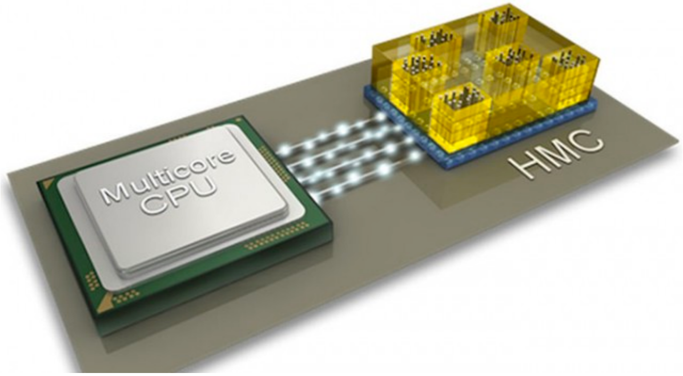




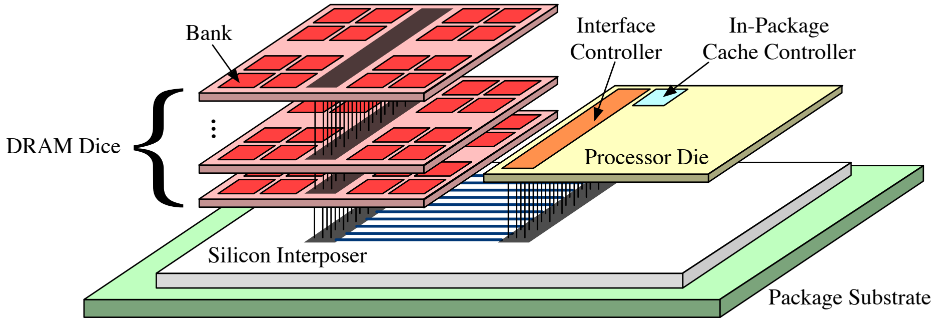
- Why?
  - Communication between devices bottlenecked by limited I/O pins
  - Integrating heterogeneous elements on a single wafer is expensive and suboptimal

3D Stacked Memory

- Hybrid Memory Cube (HMC)
  - A logic layer at the bottom



- High Bandwidth Memory (HBM)
  - Silicon interposer at the bottom


Bank
Interface Controller
In-Package Cache Controller
DRAM Dice
Processor Die
Silicon Interposer
Package Substrate