Gordon:

Using Flash Memory to Build Fast, Power-efficient Clusters for Data-intensive Applications

Adrian M. Caulfield, Laura M. Grupp, Steven Swanson

Department of Computer Science and Engineering
University of California, San Diego

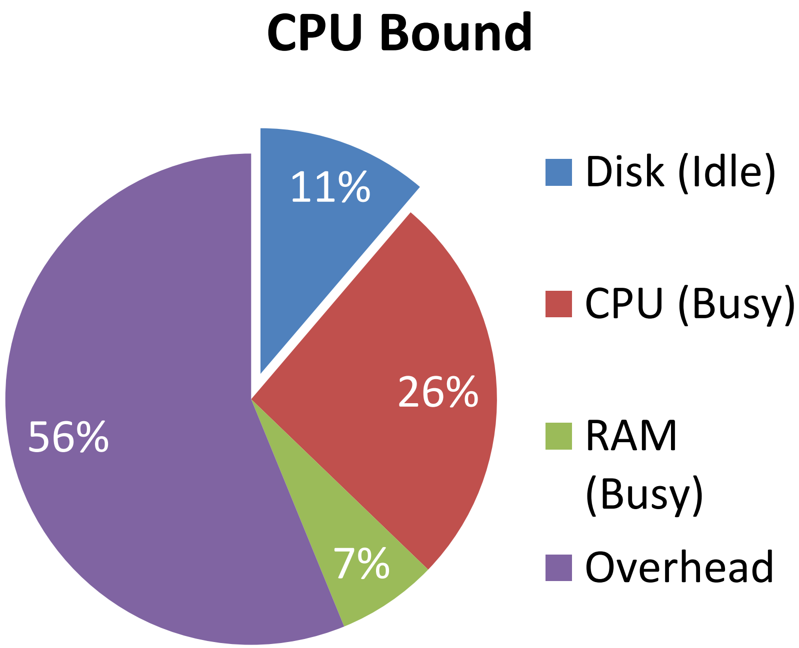
Data Centric Computing

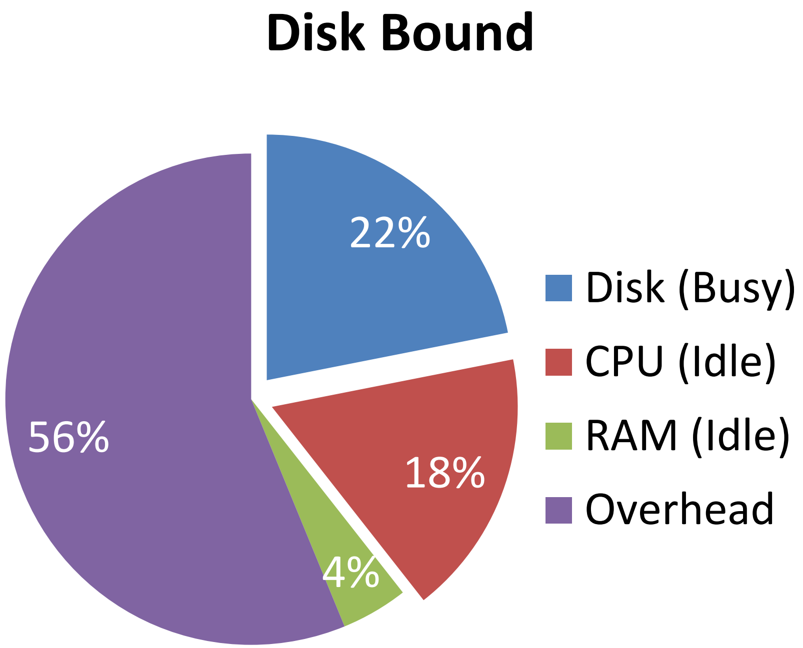
- Becoming ubiquitous
  - Retail sales data
    - 2.5PB at Wal-Mart
  - Indexing billions of web pages
  - Social Networking
    - Facebook has 175M users, 25M joined last month
  - Desktop search
- Terabyte scale, parallel
- I/O bound
- Power

Data Centric Frameworks

- Several frameworks exist
  - Map Reduce (Hadoop)
  - Dryad
  - Sawzall
- Typically run on commodity systems
  - Slow disks + I/O bound
  - Optimizing rest of system makes little sense

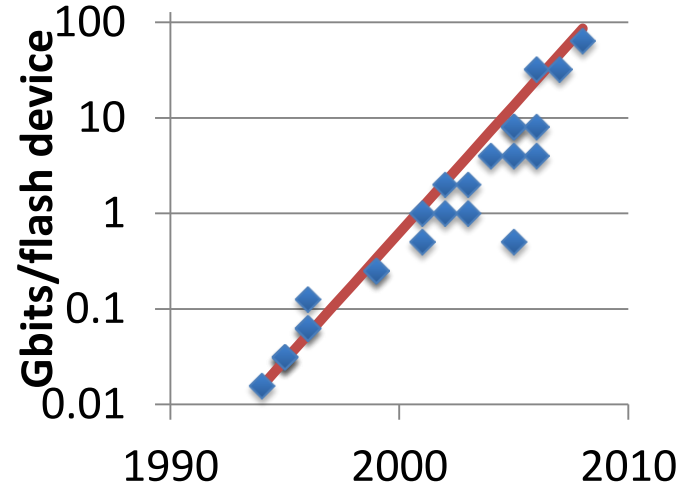
Server Power Breakdown

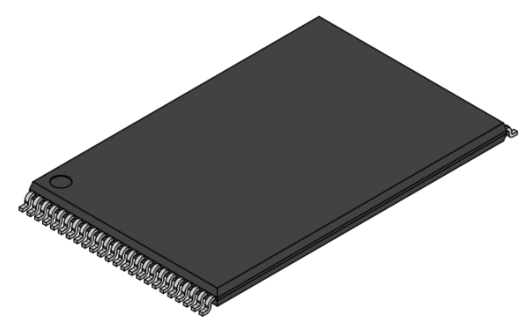




NAND Flash Memory: A Power Efficient Storage Technology

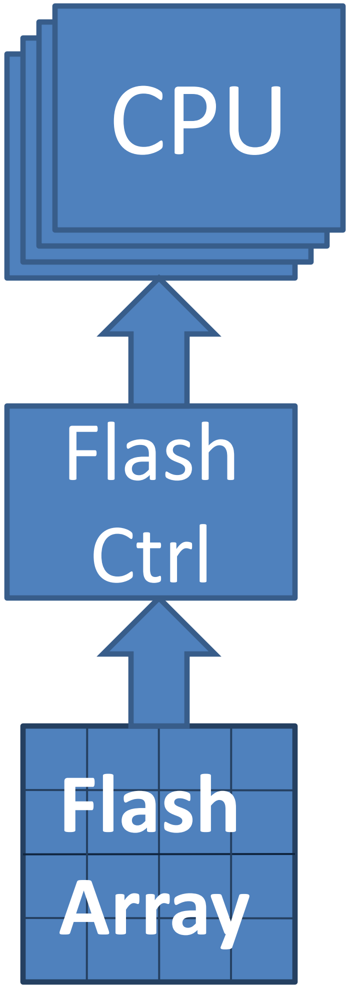
- New on the architecture scene
  - Density competitive with disk
- 256GB Next Gen. NAND Flash
  - Low Latency
    - 250x better than disk
  - High Bandwidth
    - 900MB/s possible
  - Low power
    - Active: 12x better than disk (1.28W)
    - Idle: 1% of disk idle power (0.096W)
  - Density scaling at 85%/year





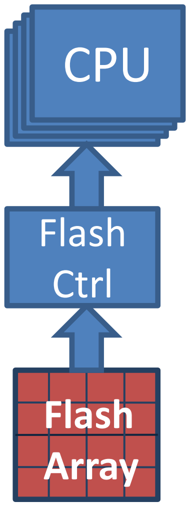
Gordon Design Process

- System architecture designed around a flash array per node
  - Start with flash array
  - Optimize the flash controller
  - Select a processor based on a design space exploration
- Cluster based
  - Each node runs full OS

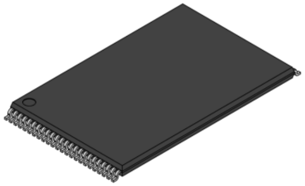


Gordon Overview

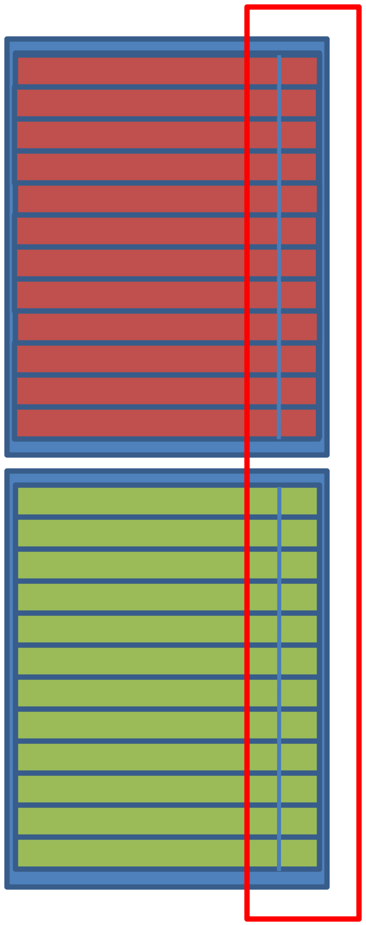
- Introduction
- Flash storage systems
  - Flash basics
  - Optimizing flash for data centric computing
- Design space exploration
- Gordon discussion



NAND Flash Overview



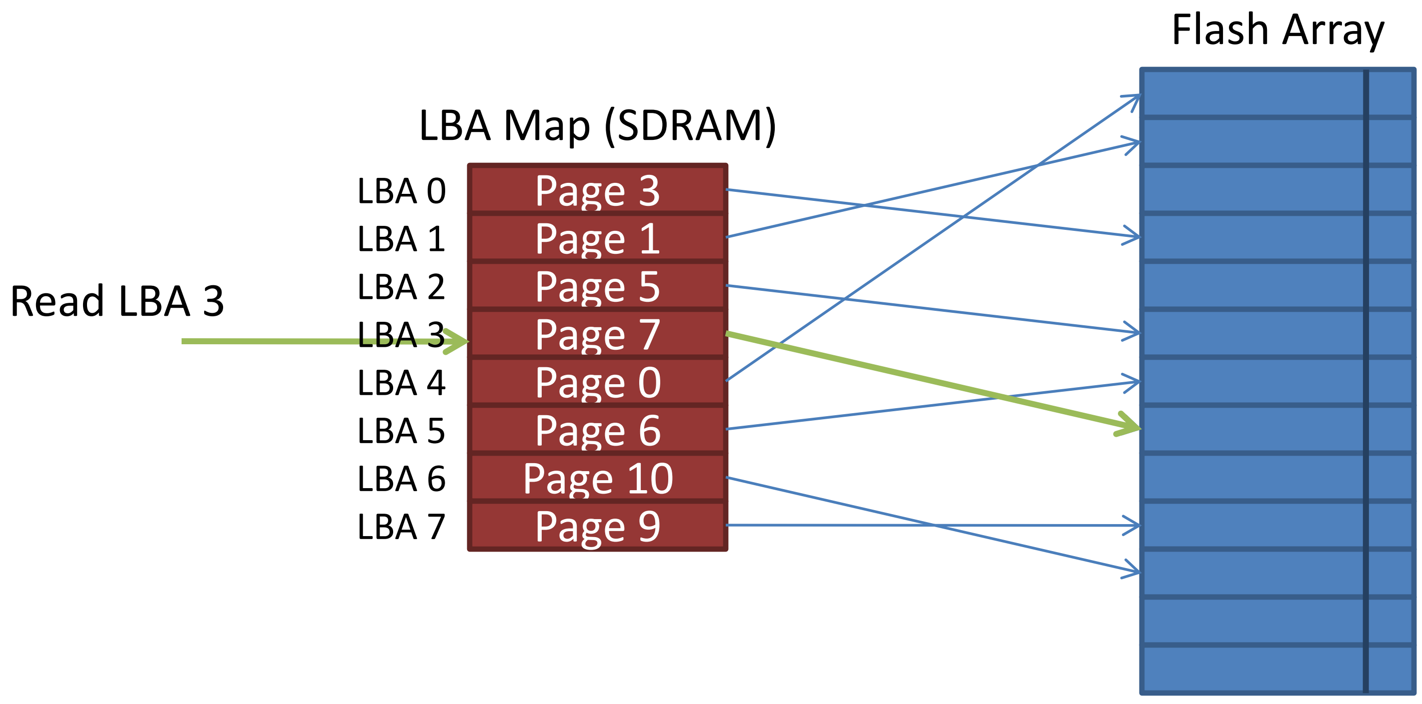
- Data arranged in blocks
  - Blocks contain pages (64 typical)
  - Each page has metadata section
- Flash Idiosyncrasies
  - Pages must be programmed in order
  - Read/Write and Erase size disparity
    - Pages are smallest read/write unit
    - Erases only at block level
  - Blocks wear out
    - Wear leveling required



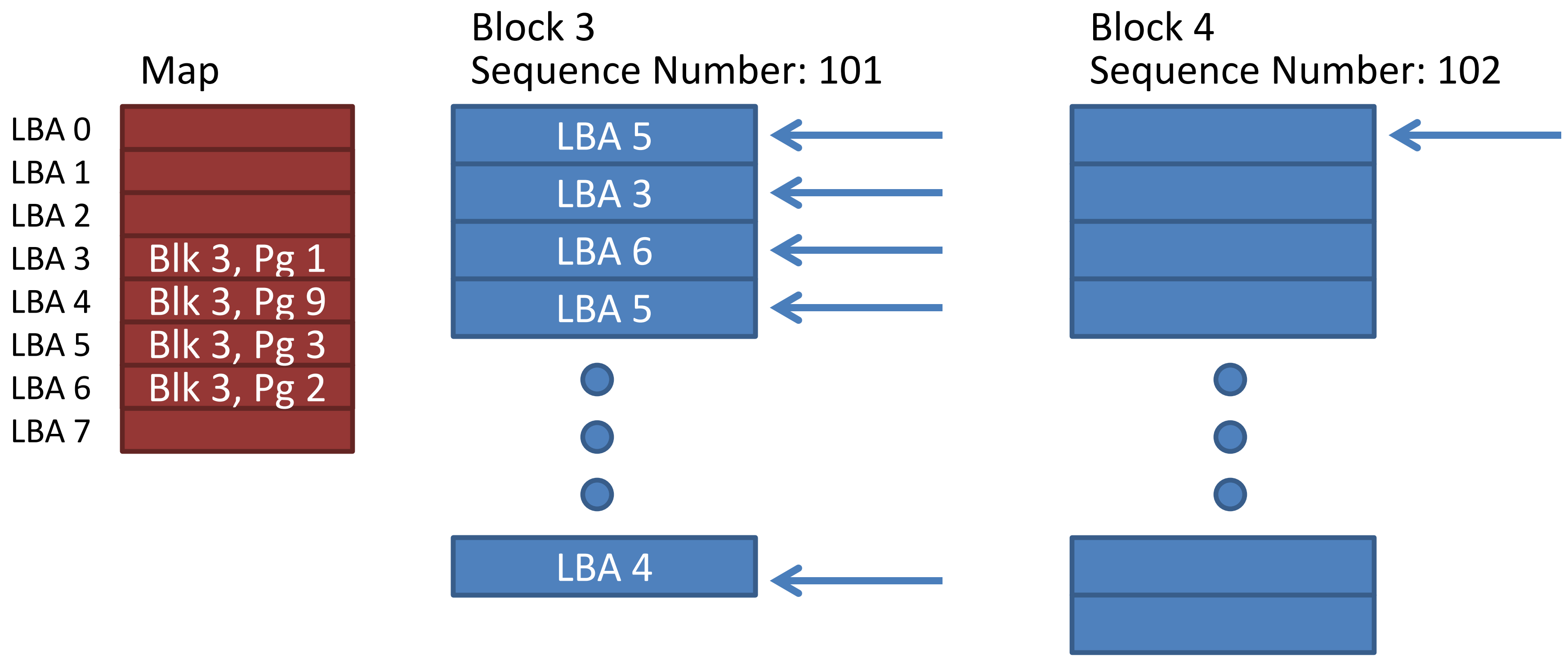
Extra metadata space at end of each page

Flash Translation Layer (FTL)

- Based on design from Microsoft Research [Birrel, 2007]
- Maps a Logical Block Address (LBA) to a Physical Address
- Map stored in SDRAM
  - Reconstructed from Metadata section of pages on power-up

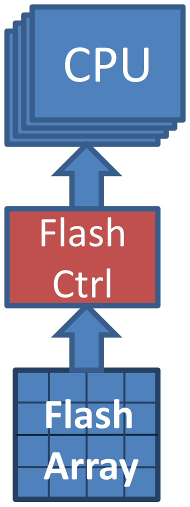


The Write Point



Gordon Overview

- Introduction
- Flash storage systems
  - Flash basics
  - Optimizing flash for data centric computing
- Design space exploration
- Gordon discussion

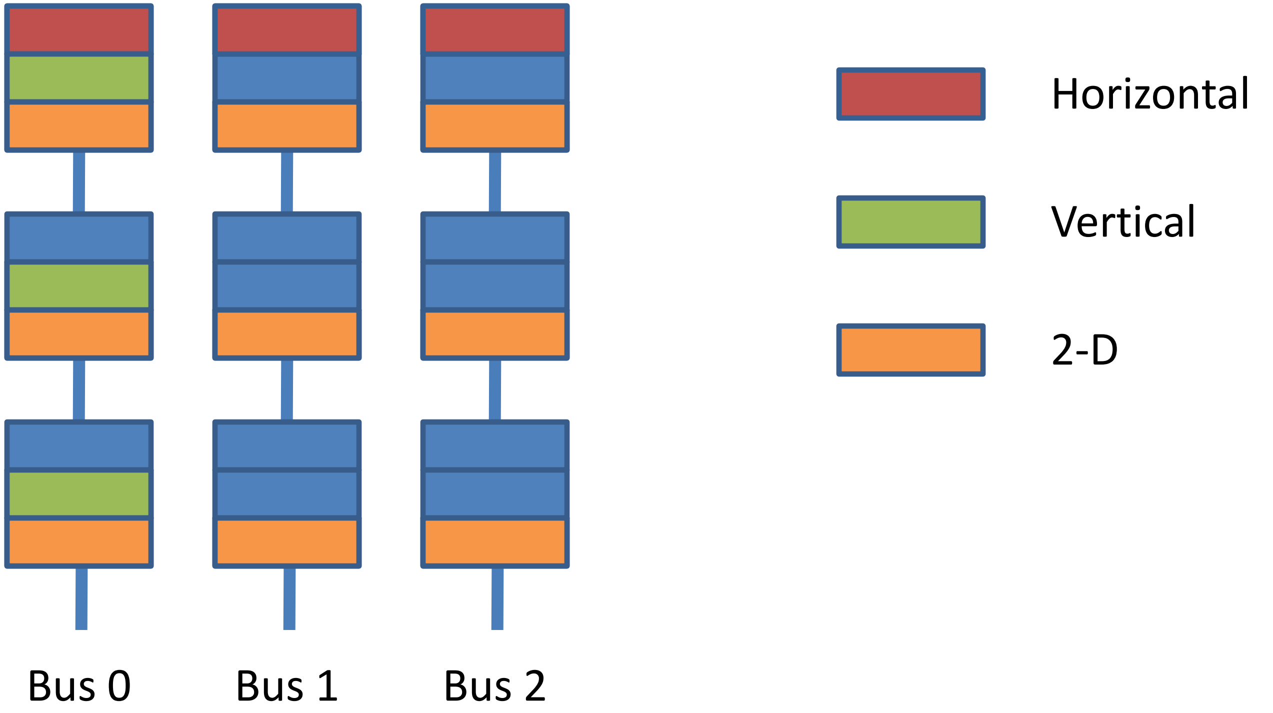


Gordon FTL Enhancements

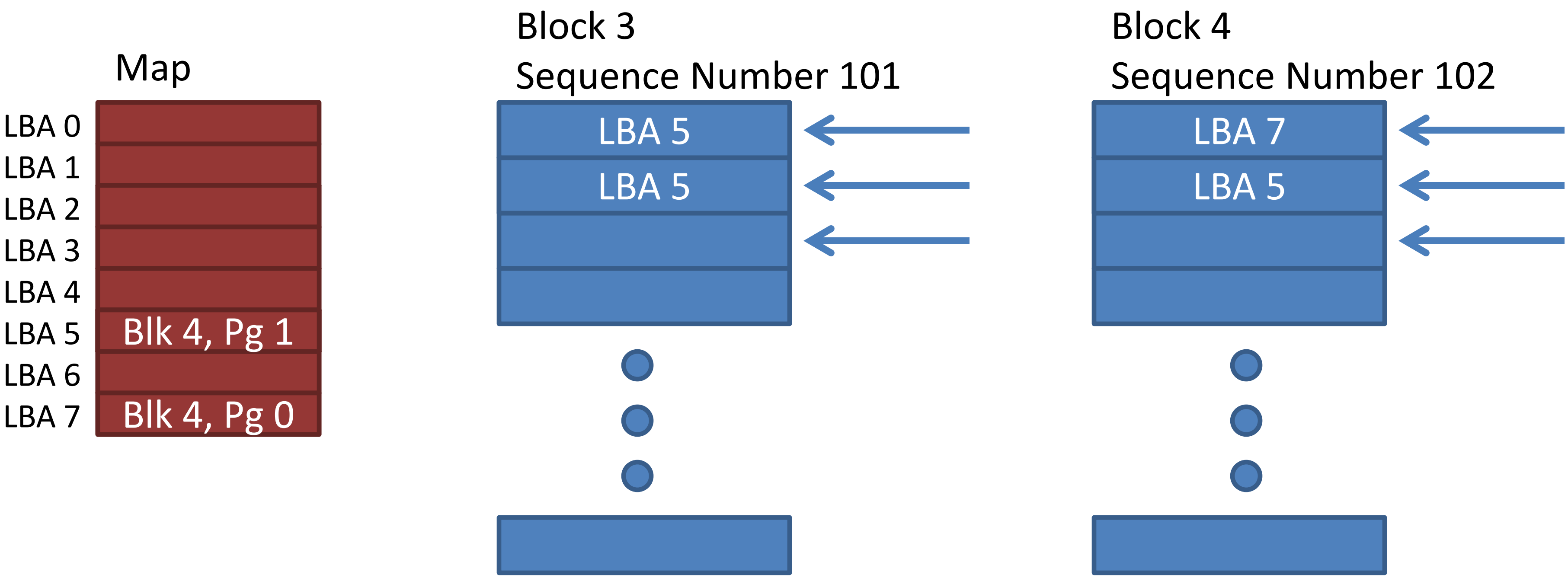
- Exploit parallelism in flash array
  - Tuning architecturally visible block size
  - Multiple write points
- Access pattern optimizations
  - Write combining
  - Bypassing

Super Pages

- Stripe data across several chips and/or busses
- Handle larger units of data
- Reduces metadata overhead



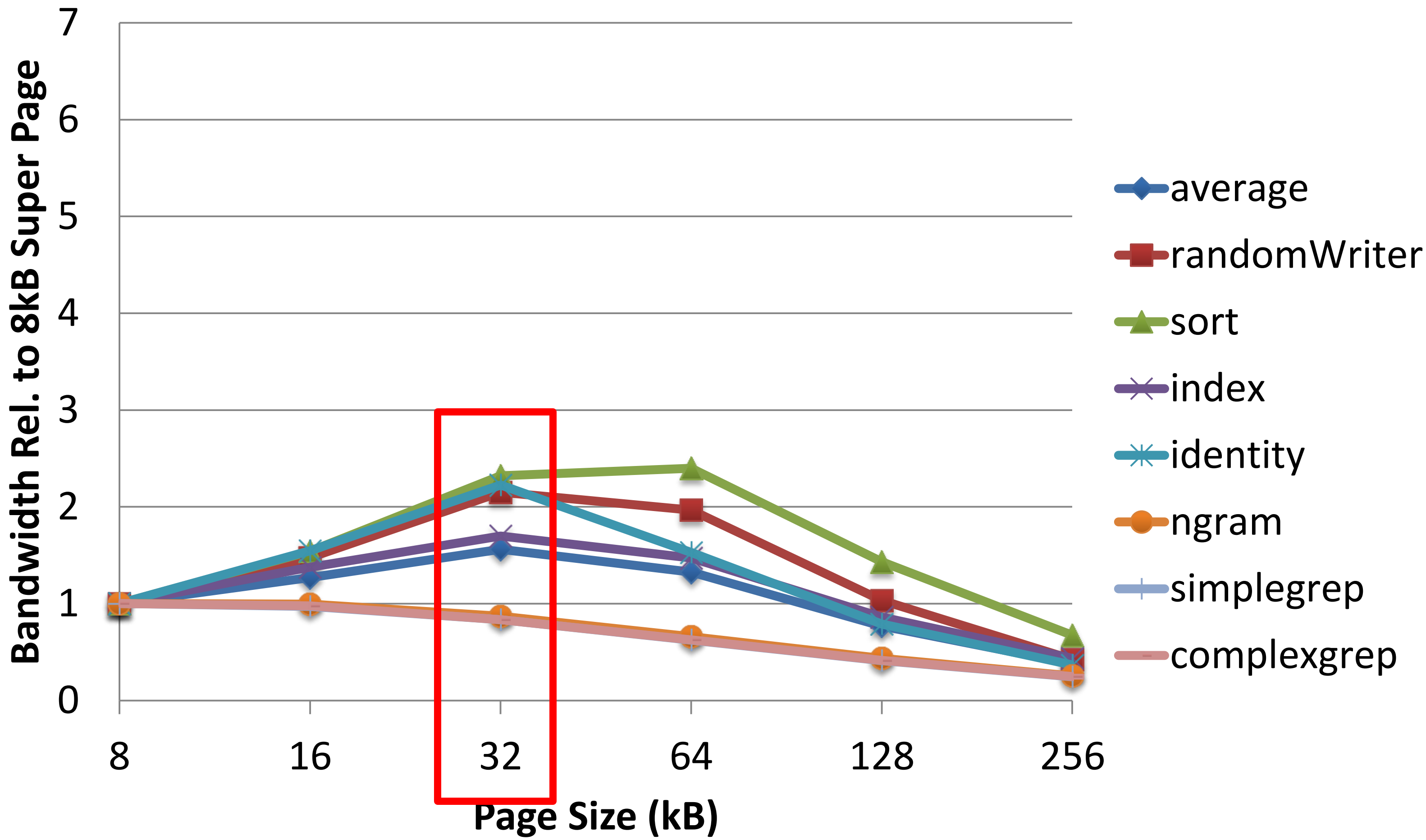
Multiple Write Points



Workload Details

Name	Description	Size	Disk Read	Disk Write
Random Writer	Output random data	10GB	0.4 GB	26.9 GB
Identity	Copy all inputs to outputs	15GB	45.1 GB	103.7 GB
Sort	Sort random numbers	1GB	1.4 GB	5.7 GB
SimpleGrep	Search for “the” in multi-lingual text	8GB	8.4 GB	0.5 GB
ComplexGrep	Complex Regex search in multi-lingual text	8GB	9.2 GB	1.0 GB
N-Gram	Find frequently occurring N-word phrases in multi-lingual text	4GB	40.1 GB	90.7 GB
WebIndex	Indexing of web pages	13GB	18.9 GB	62.8 GB

Super Page Performance

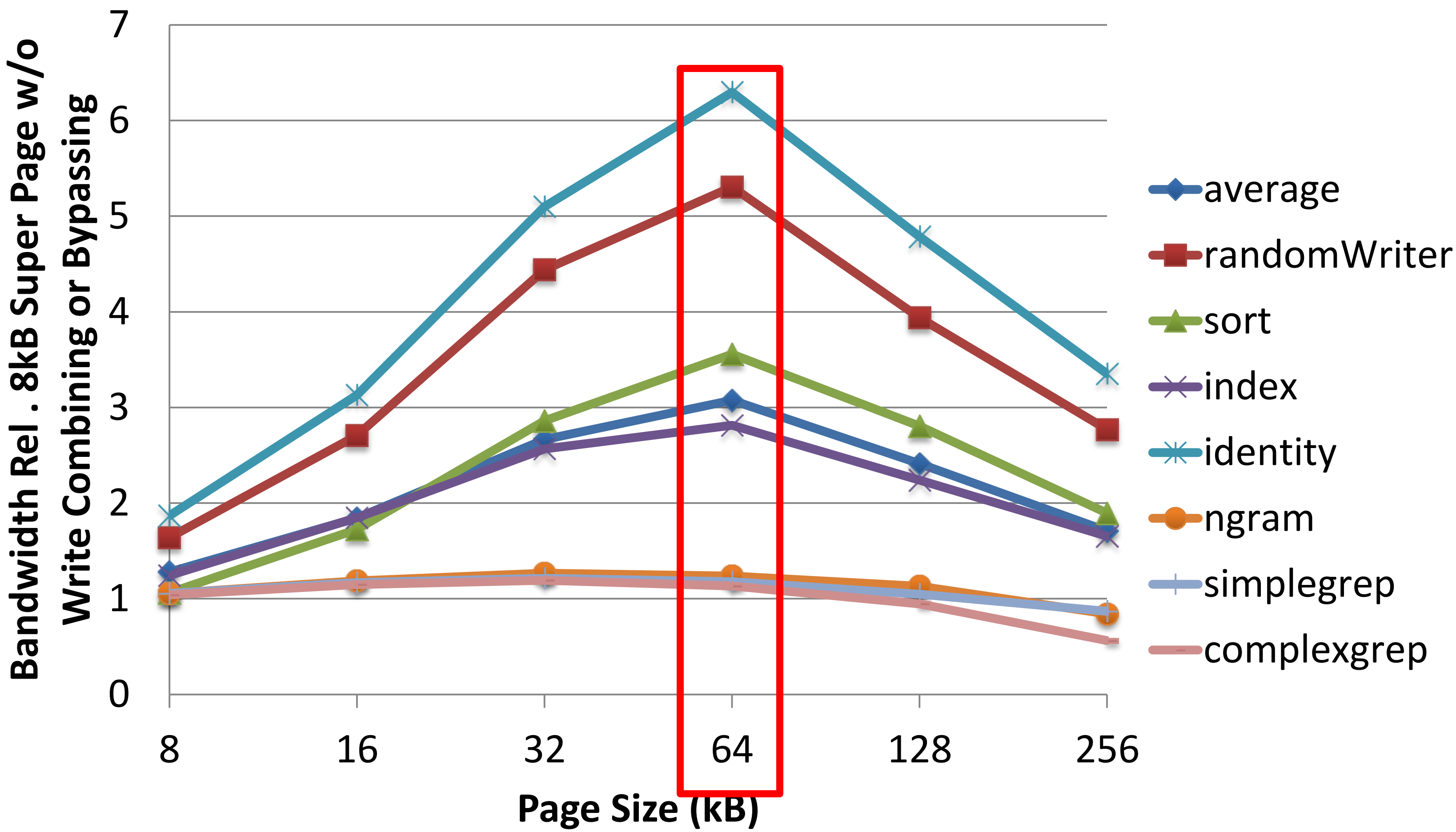


Write Combining and Bypassing

- Write Combining
  - Merge multiple writes to the same address when possible
- Bypassing
  - Merge incoming read requests
  - Cache last page read

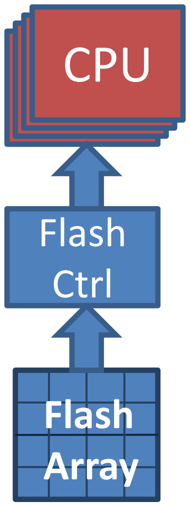
Super Page Results

With Write Combining and Bypassing



Gordon Overview

- Introduction
- Flash Storage Systems
- Design Space Exploration
  - Design Space
  - Methodology
  - Results
- Gordon Discussion



Design Space

Parameter	Values
Processor Types	Atom, Core 2
Processors	1, 2, 4
Atom Frequency (GHz)	0.8, 1.5, 1.9
Core 2 Frequency (GHz)	0.6, 1.2, 1.8, 2.4
Flash dies	0, 64
Hard drives	0, 1, 2, 4
Power Budget	300W

- 84 designs
- 32 node cluster

Methodology

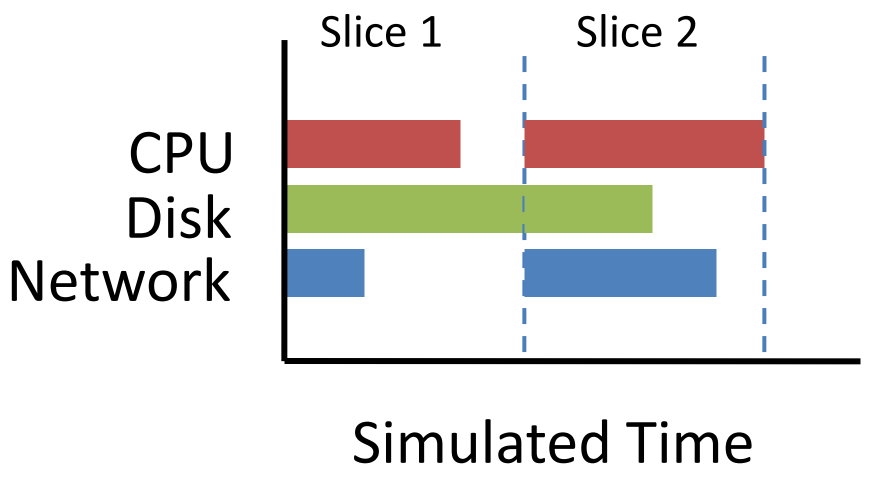
- Trace Based Simulator
  - Instruction count
  - L2 miss count
  - Network read/write usage
  - Disk read/write usage, I/O count
- Traces processed using our cluster simulator

Gordon Cluster Simulator

- Calculates both Power and Performance
  - Performance reported as total runtime
  - Storage times simulated using our flash simulator and DiskSim
- Power model
  - Estimate average power for a given trace slice using activity ratios

Calculating Performance

- Per component simulated times for each 1-second trace slice
- Maximum individual component time represents total simulated slice time



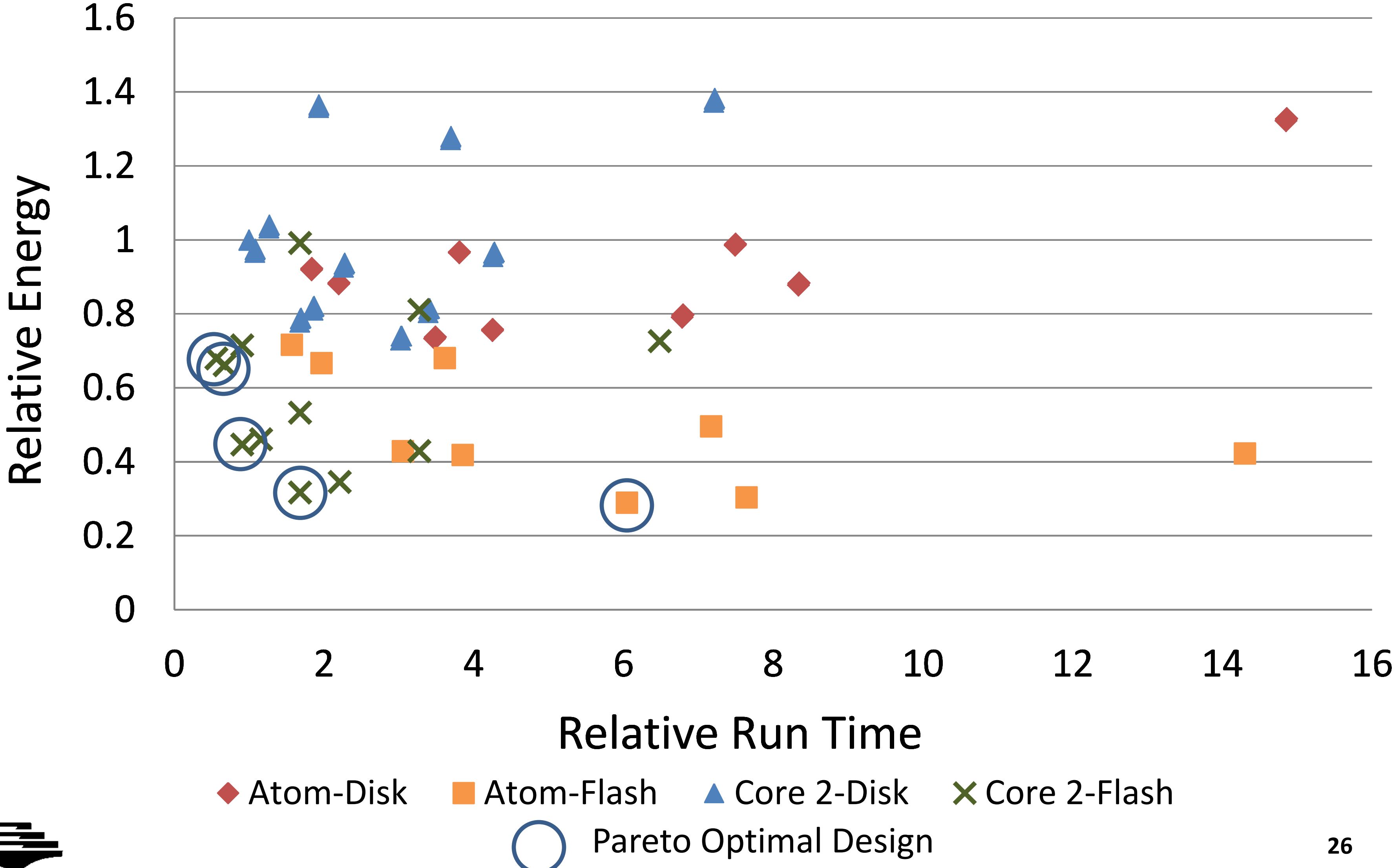
Power Model

- Calculate per second component activity factors
  - DRAM: L2 cache misses
  - CPU: Instruction count
  - Flash/Disk: Number of IO requests
- $P_{Total} = \sum (\%Active \cdot P_{Active} + \%Idle \cdot P_{Idle})$
- *Active* and *Idle* power measured on actual servers
  - Datasheet numbers when not possible
- CPU power scaled based on datasheet voltage range for different CPU frequencies $P = fv^2$

Gordon Overview

- Introduction
- Flash Storage Systems
- Design Space Exploration
  - Design Space
  - Methodology
  - Results
- Gordon Discussion

Design Space Survey Results

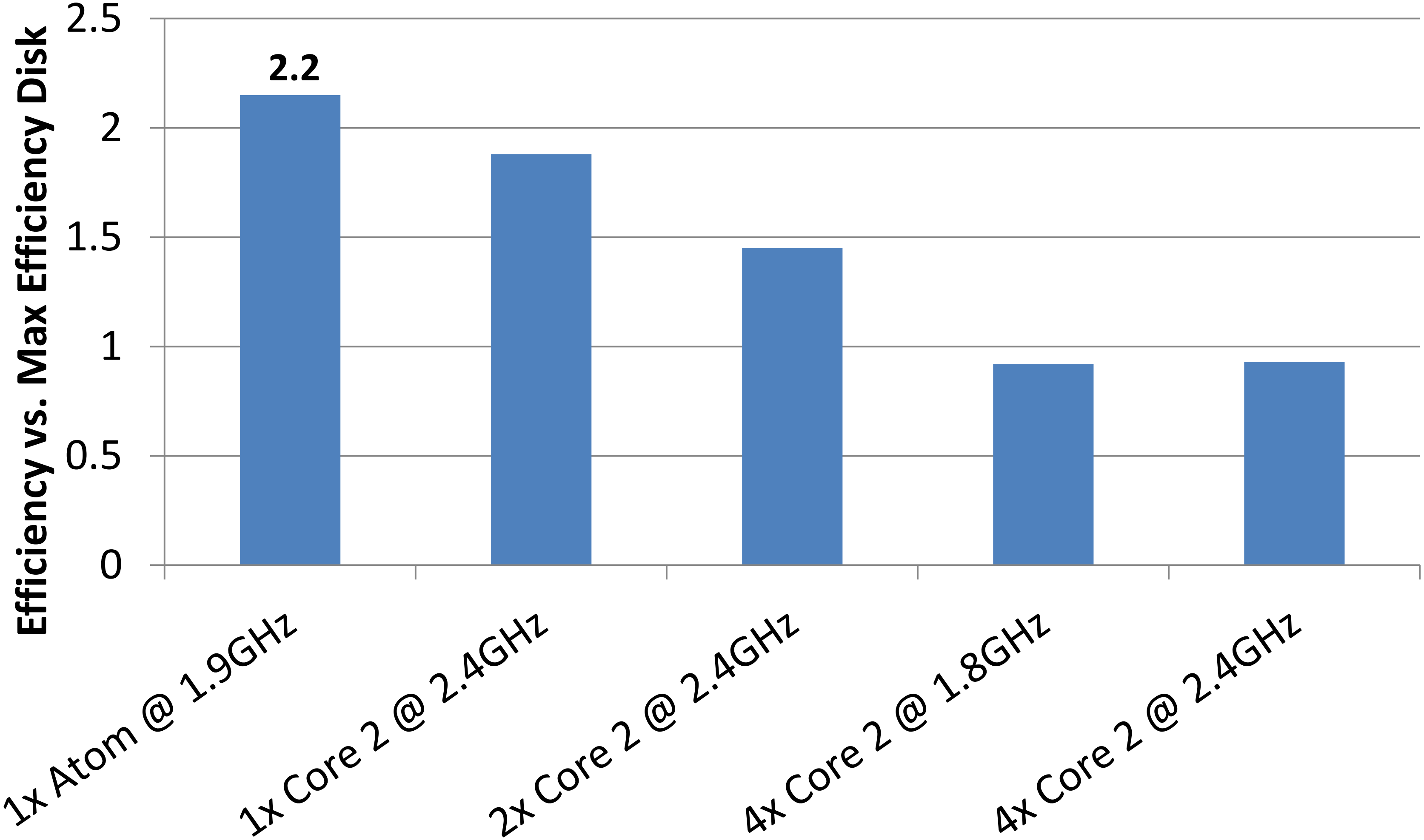


Pareto Optimal Designs

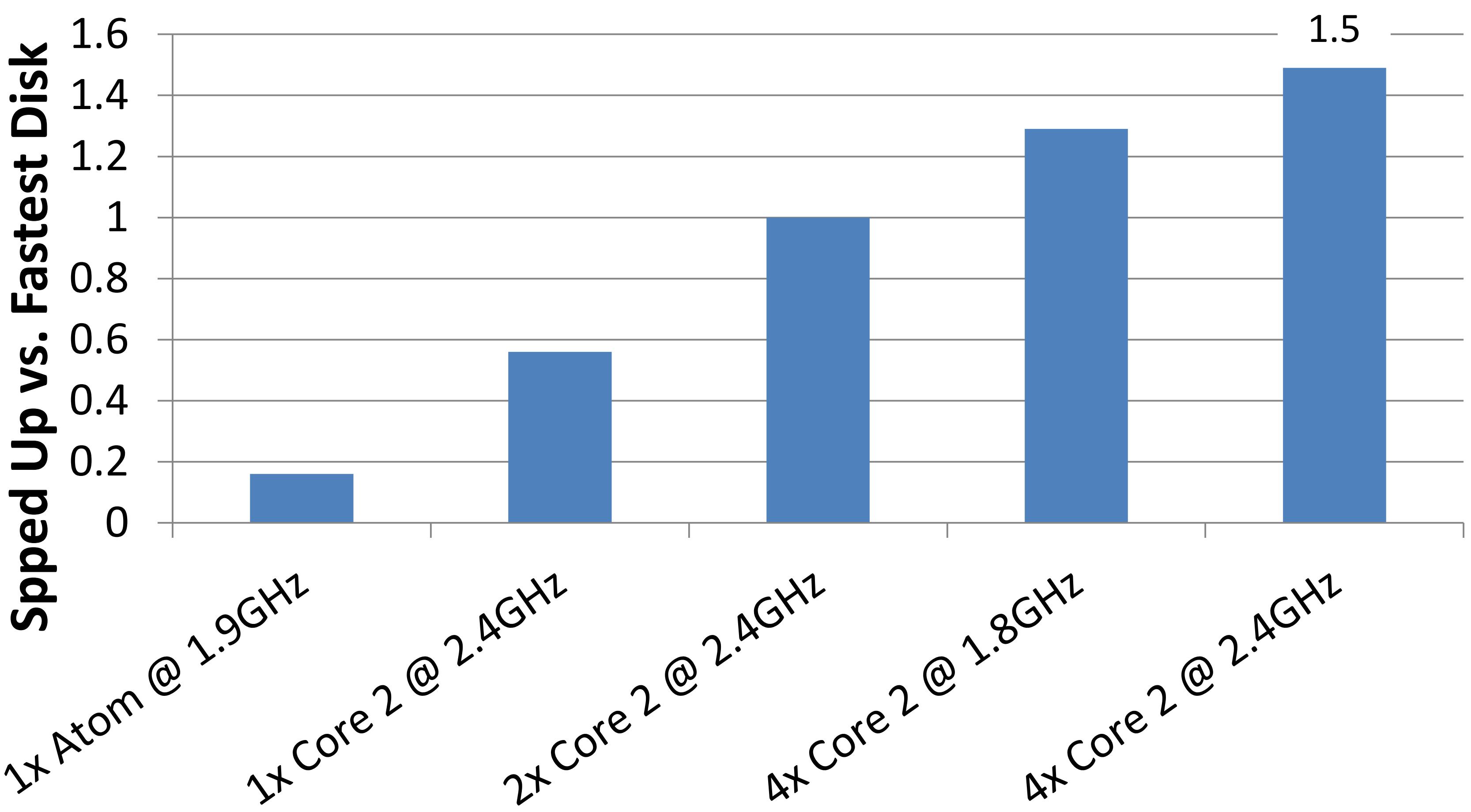
Processor			Average Power
Number of Cores	Type	Frequency	
1x	Atom	1.9 GHz	4.81 W
1x	Core 2	2.4 GHz	19.89 W
2x	Core 2	2.4 GHz	45.66 W
4x	Core 2	1.8 GHz	92.74 W
4x	Core 2	2.4 GHz	106.18 W

- All pareto optimal configurations use flash memory

Efficiency vs Most Efficient Disk

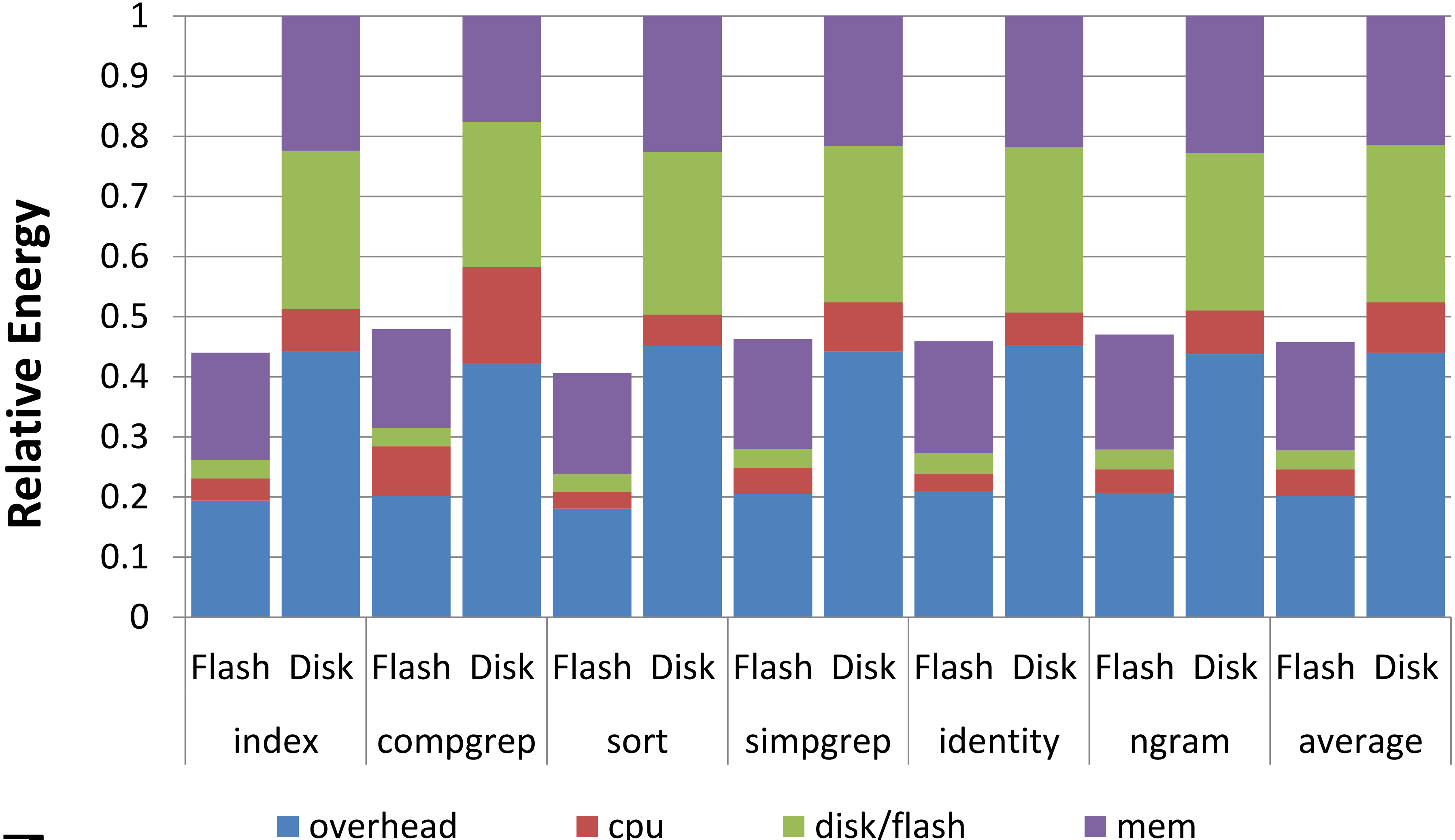


Speed Up vs Fastest Disk



Relative Energy Consumption

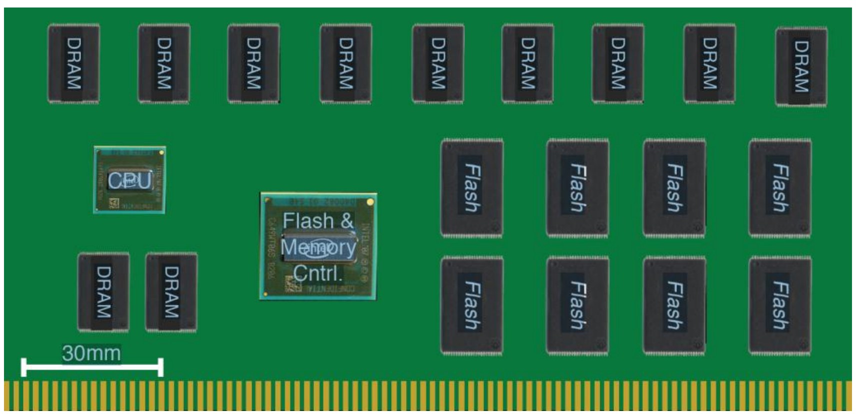
Most Efficient Flash vs Most Efficient Disk



Gordon Overview

- Introduction
- Flash Storage Systems
- Design Space Exploration
  - Design Space
  - Methodology
  - Results
- Gordon Discussion

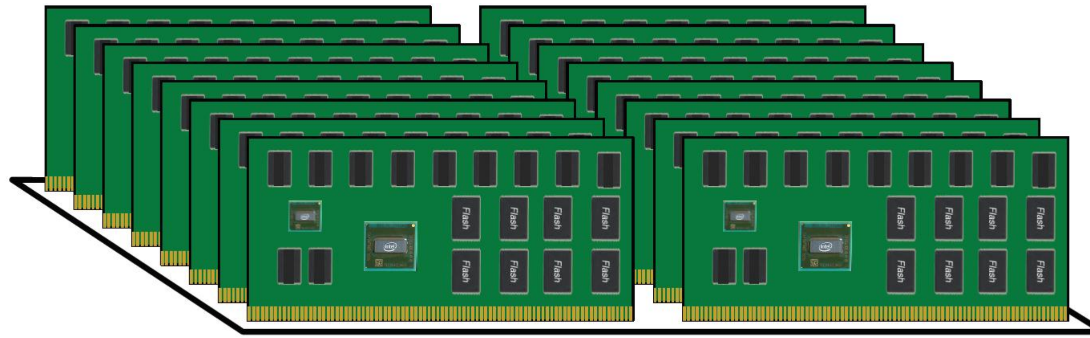
Most Efficient Gordon Node



- 256GB of Flash
- 2GB of DRAM
- 1.9GHz Atom Processor
- Flash Controller
  - 512MB of DRAM
- Power usage: 5W

System Architecture

- 16 Gordon nodes per backplane
  - 1Gbit network connectivity



- 16 backplanes per standard server rack
  - 256 Nodes
  - 64 TB of Storage
  - 230GB/s of aggregate I/O bandwidth
  - 1300 Watts power usage
- A data centric super computer

Gordon Cost

- Disk is cheaper per GB of storage
- Flash clear winner in Cost/MB/s
  - For 900MB/s bandwidth
    - Flash: $350, Disk: $4500
- Real value: Gordon enables new applications
  - Fast random I/O

Virtualizing Gordon

- Keep Gordon as busy as possible
- Gordon becomes a data intensive coprocessor
  - Large datasets stored on disk
  - Transferred to flash for processing
- Pipeline loading and computation

Conclusion

- Data centric applications increasingly common
- Flash memory provides low power, low latency, high bandwidth storage
- Optimized Flash Translation Layer
- Gordon enables fast, power efficient data centric processing
- Gordon is up to 2.2X more efficient and 1.5X faster than disk based designs

Thank You