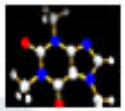
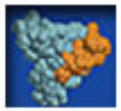
SciDAC
Scientific Discovery through Advanced Computing





ASCR

Office of Science

Futures Panel

HEC File System & IO Workshop, Aug 3, 2010

Garth Gibson
Carnegie Mellon University

NSF

Carnegie Mellon
Parallel Data Laboratory

pdsi

Panel Charge

- What do you think parallel FS (hw & sw) will look like in 10 years?
- What do you wish parallel FS (hw & sw) will look like in 10 years?
- What do you think will be the top 3 challenges in the next 10 years to provide IO and associated infrastructure into Exa- and enormous data analytics?
- Where will HPC be able to find good external tools to leverage (e.g. Google, LSST, etc)?
- How does the hype about clouds and virtualization add challenges or opportunities to file system and storage SLAs both at future exa as well as at current more moderate scale?
- Is it time to toss the file systems concept entirely and go to some other kind of abstraction?

Crystal Ball

- What do you think parallel FS (hw & sw) will look like in 10 years?
  - 10X more clients means 10X more IO server nodes
    - 10% IO nodes means next decade's IO subcluster is today's entire cluster
  - Scalable bandwidth works, but total disk bandwidth >> IO bandwidth (active?)
  - Much more failure, failure tolerating codes, online repair of many specialized faults
  - Flash becomes common and used for many purposes, almost all is SSD
    - Checkpoint, cheap DRAM, metadata store, cache for hot random access data
  - POSIX/VFS largely unchanged, though "eventual consistency" available
  - pNFS creates new competition in HPC, new penetration of HPC into mainstream
  - More layers above file system doing more complex packing of data in files, leading to more common and more obscure performance problems
  - Disk is bulk of online capacity, but are *shingled* (see next)
- What do you wish parallel FS (hw & sw) will look like in 10 years?
  - Tape is (all but) gone; (powered down) disk is tape
  - Phase change memory replaces Flash as NV, significant replacement of DRAM
  - BigTable-like, database-like analytics support available to (and from) parallel FS
  - API much richer, including location/topology/replication awareness

Shingled Disk Slides

Big "Three"

- What do you think will be the top 3 challenges in the next 10 years to provide IO and associated infrastructure into Exa- and enormous data analytics?
- 1) coping with failure rates, specialized failure cases
- 2) dealing with storage mgmt & pseudo-db apps on 1 trillion files
- 3) systematic avoidance of (rare) code bugs as "silent" errors
- 4) revising the partitioning of function between traditional middleware layering (MPI-IO, file format, database, runtime, file system,….)

Cloudy Issues

- Where will HPC be able to find good external tools to leverage (e.g. Google, LSST, etc)?
  - Not Google, but Google's competitors doing open source tools at scale
  - Database researchers and companies fighting to be relevant in largest data
  - Machine learning community

- How does the hype about clouds and virtualization add challenges or opportunities to file system and storage SLAs both at future exa as well as at current more moderate scale?
  - If HPC is leader, not follower, in DISC & clouds, infusion of young talent/energy
  - Outsourcing more use of less customized systems (performance variance)
  - Virtualization used to cope with all the cores, but doesn't virtualize capacity (memory or disk) well, so space sharing is more important
  - SLA increasingly looked too for solutions, but problems get harder

Clean sheet is a cope out

- Is it time to toss the file systems concept entirely and go to some other kind of abstraction?
- Yes
  - In a trillion files, human structured namespace is unworkable as sole solution
  - Database techniques will start to penetrate to manage attributes
  - Sensor data more common with more fundamental provenance semantics
  - Many middleware services will add/change service semantics layered on “objects”
- No
  - Partitioning schemes will be hierarchical, named variations on name tree
  - Humans demand familiar things, so folders persist as organization forever
  - Unstructured blobs will remain dominant use of capacity
  - Linux is not done, will still be growing strong with all current code for many years
  - Legacy code lasts longer than a decade

Carnegie Mellon

Directions for Shingled-Write and TDMR System Architectures: Synergies with Solid-State Disks

Garth Gibson

www.pdl.cmu.edu

May 7, 2009

Short Bio

Co-author, A Case for RAID, 1988

Professor, CS & ECE, CMU, 1991-

Systems Thrust Leader, DSSC, CMU, 1990s

Founder, Parallel Data Lab, CMU, 1993

Founder & CTO, Panasas Inc, 1999

- HPC storage @ Los Alamos, BP, Intel, Boeing, NIH, Ferrari, Citadel

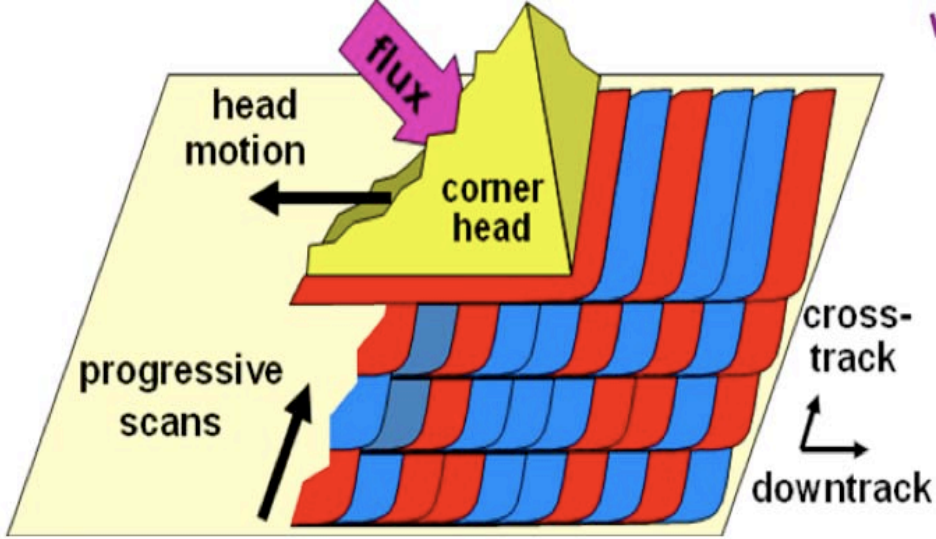
Co-Instigator, SCSI OSD & IETF Parallel NFS stds

Storage Networking Industry Tech Council, 2000s

Steering Cmte, File & Storage Tech (FAST) Conf

PI, DOE Petascale Data Storage Inst., 2006-

Shingled-Writing



Garth's simple world view

- HAMR, BPMR:
  big changes in fab/assembly
- Shingled-writing does not need big changes

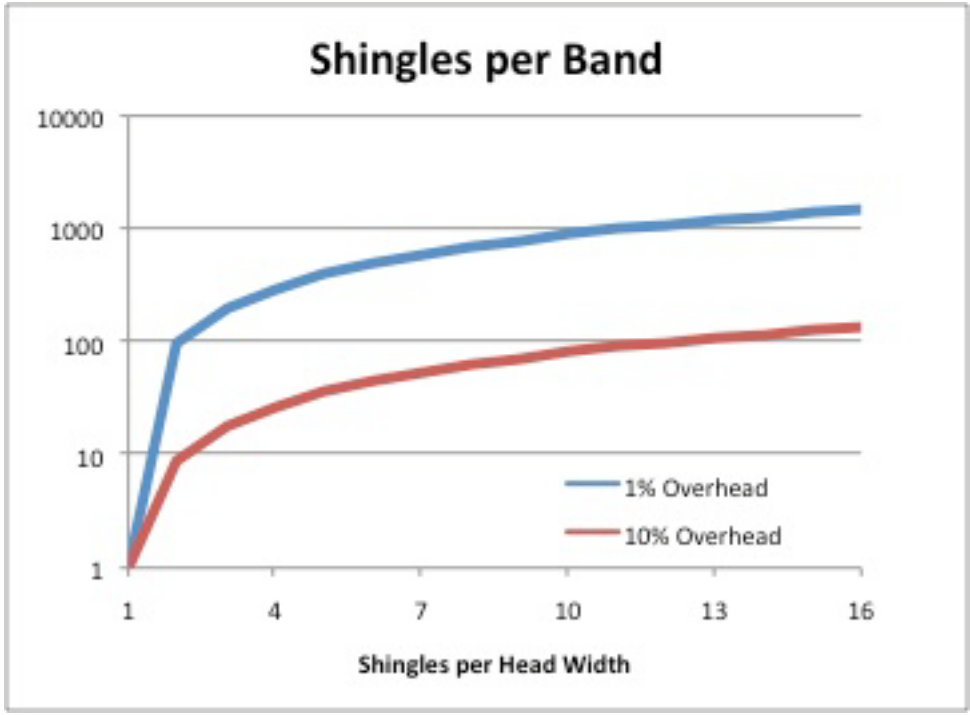
Shingle-writing means

- Partially overwriting tracks, for closer pitch
- Inability to modify one embedded sector without rewriting cross-track neighbors

Loss of Update-in-place

Banding of shingles

- Last track is wider, capacity overhead
- Tracks per band (@ 90% overlap):
- 1% ov => 1000 & 10% ov => 100



Modifying a random sector in a band of 100 tracks

- Avg. of 50 revs to rewrite overlapped tracks!

Writing System Model

Shingled-write disk is N bands, each of order 1 GB

Append to end of a band has today's performance

Overwriting non-end of band "deletes" rest of band

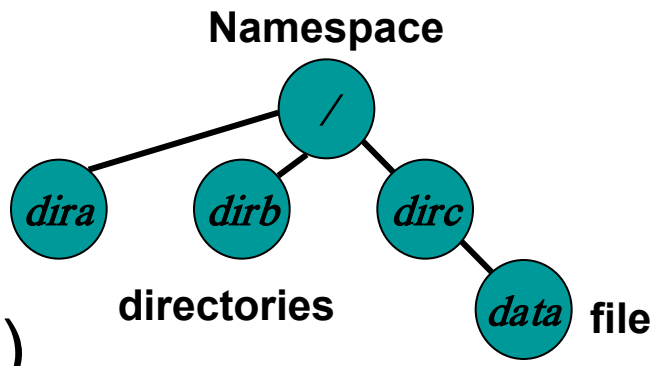
Writing start of band deletes prior content

Performance prohibitive to update-in-place at all

Can systems software cope with this?

No

File Systems 101



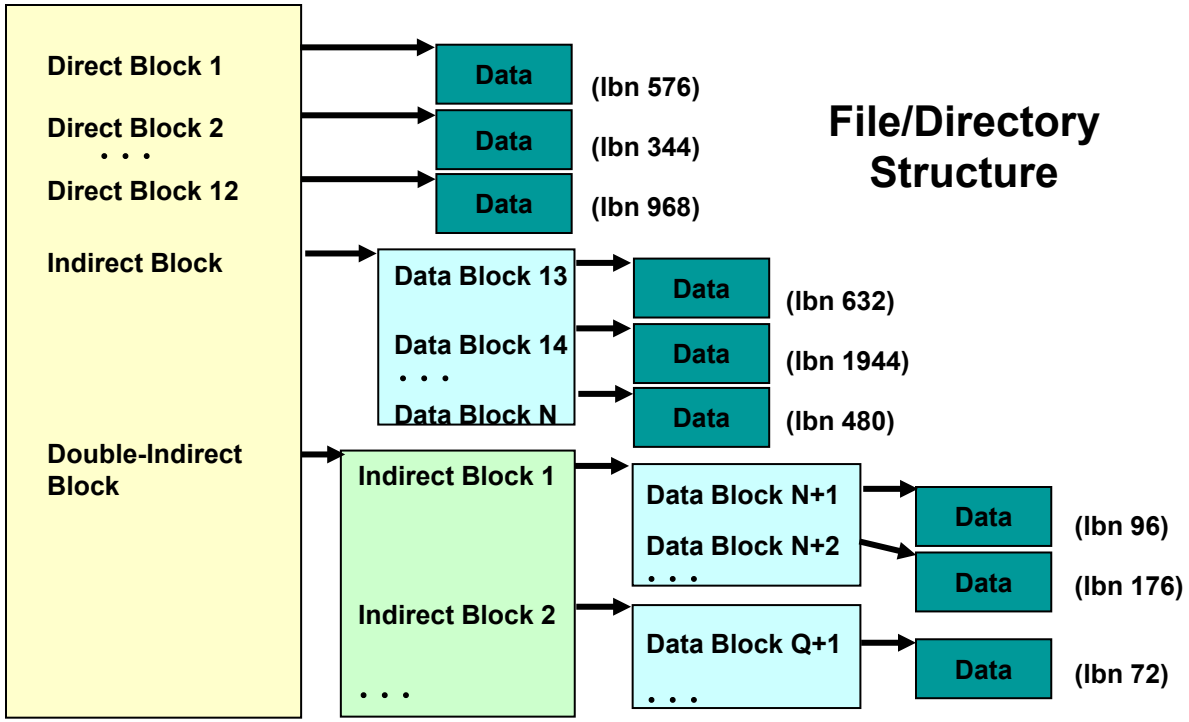
File systems store structured data

Metadata (block lists, attributes, …) are generally small

Page-at-a-time from OS

Disk fragments with delete

Small writes b/c Metadata! Hole filling



Carnegie Mellon
Parallel Data Laboratory

& Files are Small

CDF of general file size

Historically

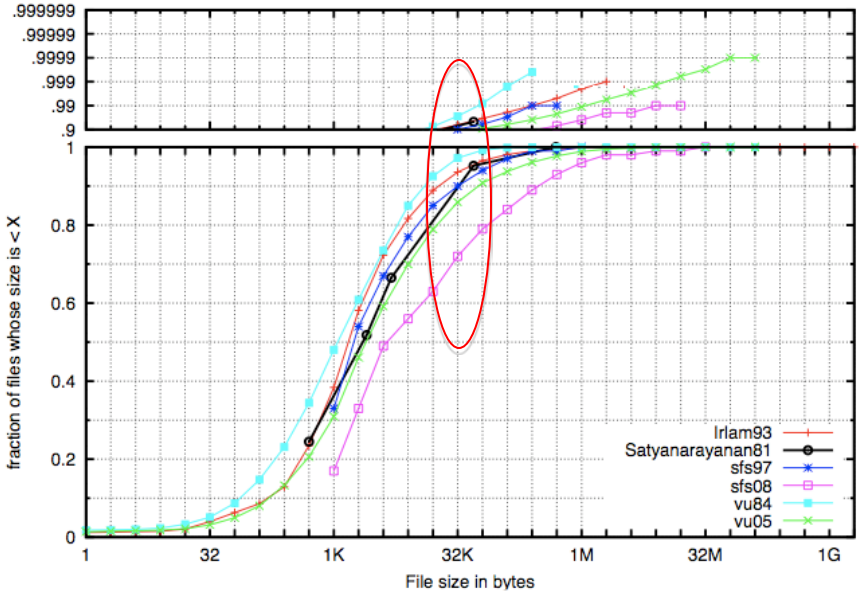
> 75% < 32KB

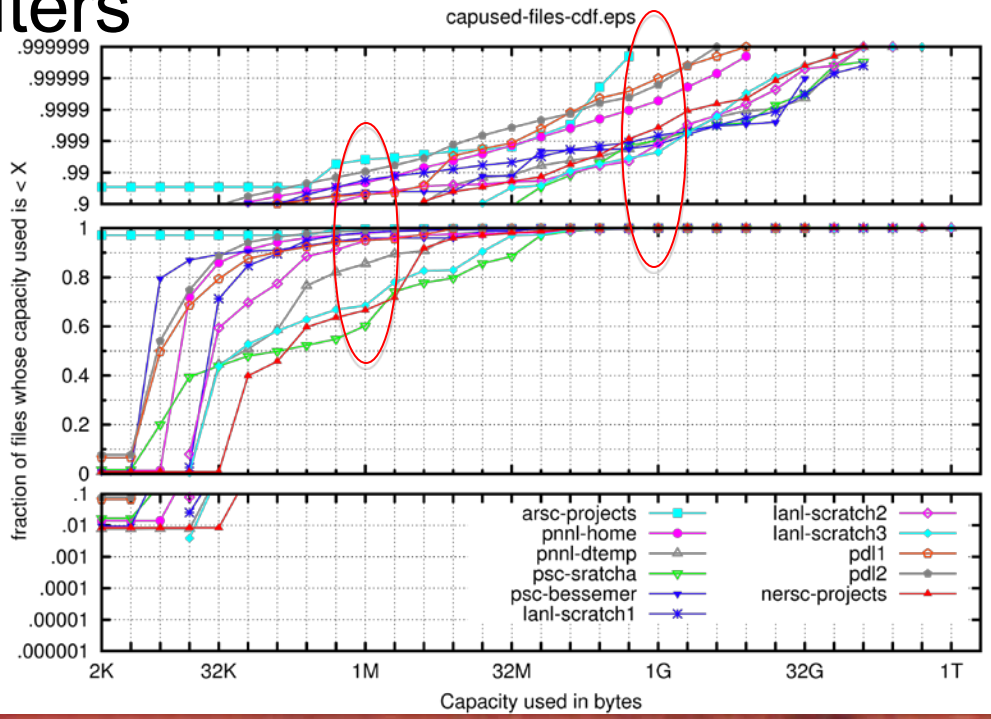
Today's supercomputers

60-99% < 1MB

< 0.1% > 1GB

Most space in large files, but no avoiding the small ones





Carnegie Mellon
Parallel Data Laboratory

System Model for Hard Disks

Hard disk is a memory model: billions of sectors

File system allocation is search for free sectors

To avoid “losing” space, small holes written

Durability/fault tolerance forces prompt writing

Metadata is small and often written

Storage performance improvement is always:

“Make disk writes larger by merging data”

But can’t fundamentally avoid small writes

Same Problem for Flash

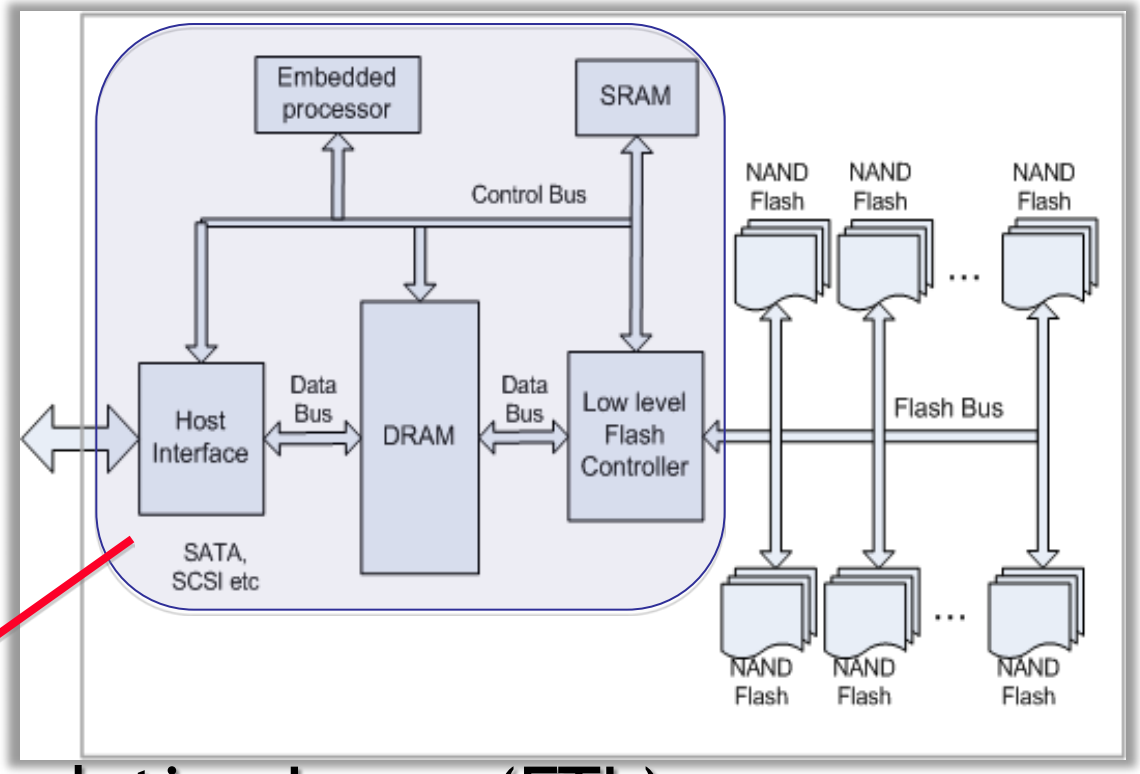
Flash SSD organized as “bands” of “sectors”

Must pre-erase band before programming data

Hide erase in FTL

Simple products
rewrite band
on all writes

Smart products
remap LBN
dynamically



Flash Translation Layer (FTL)

Shingled-write needs “FTL”

Use embedded processor to translate full SCSI/ATA command set to “append” & “rewrite”

Host “overwrite” is append and record new location

- Prior location is now “wasted space”
  - Overprovision space to absorb waste
- Background cleaning rewrites live part of bands
  - Same as today’s defrag tools
  - New TRIM command to expose waste

Not new: 1992 Log-structured file system paper

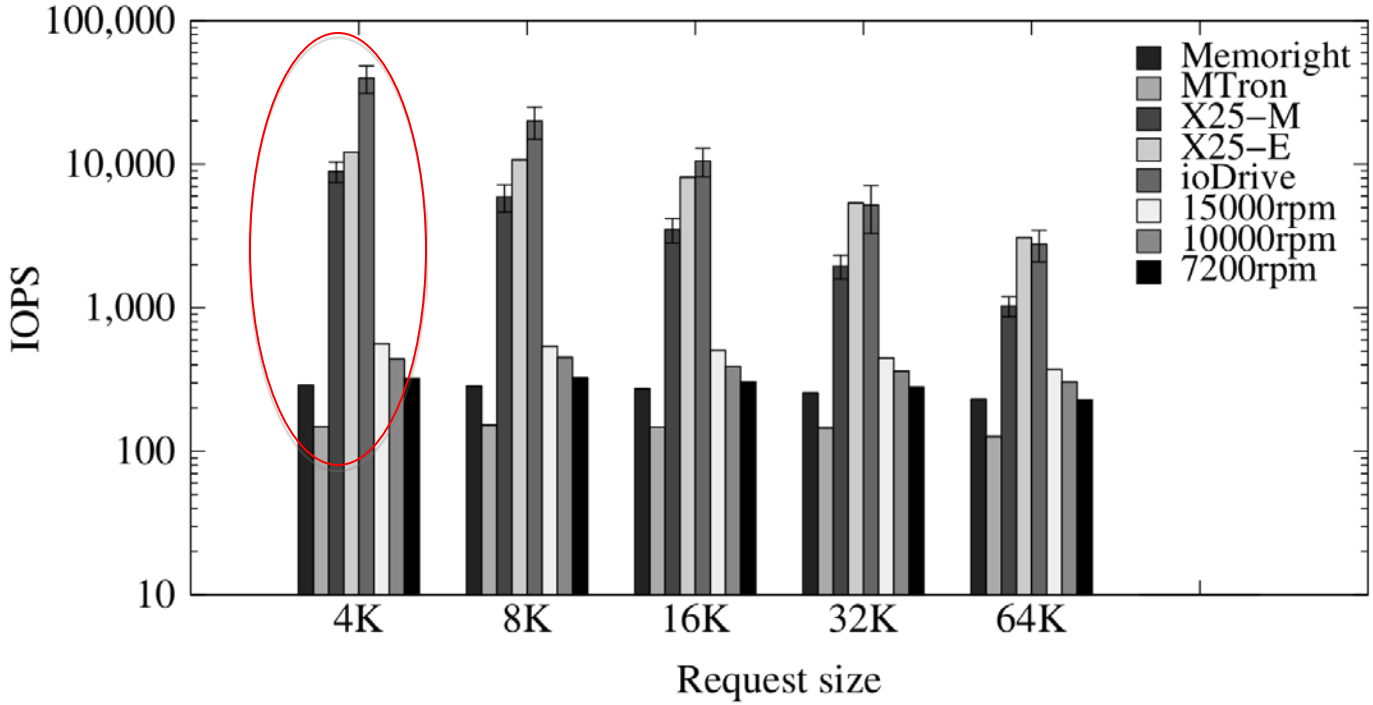
- NetApp, Panasas use remapping disk layout

Example: Flash Write Speeds

Measuring today's simple and smart flash SSDs

100x – 1000x more small writes per second

Remapping can rescue Shingled-writing disks!



Carnegie Mellon
Parallel Data Laboratory

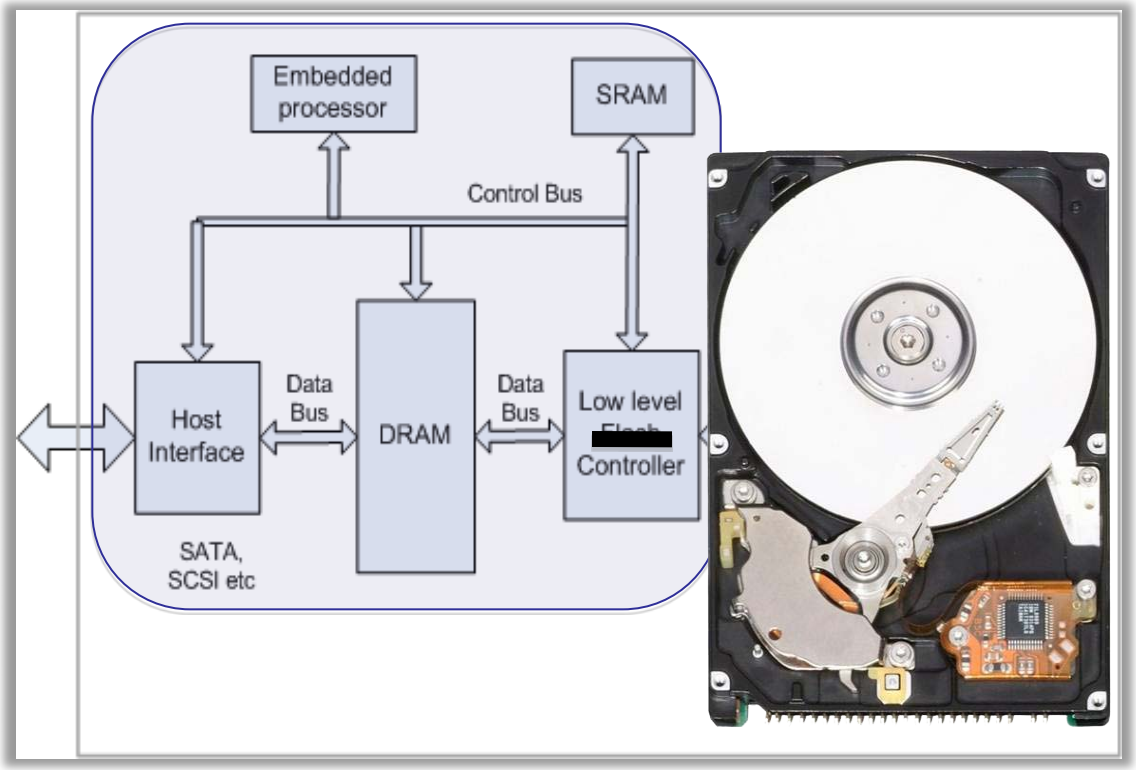
Shingled-write w/ translation

Its just code ☺

Okay, that means a faster CPU and more DRAM and Complexity!

But you can start with flash translation code

Hire from FusionIO alumni ☺



What About Reading?

Reading a shingle involves signal processing in two dimensions (TD) – down and cross track

One approach to TDMR involves gathering signal from 1-2 adjacent tracks on both sides

Means 3 to 5 revs to read a single sector

3x – 5x lower small random read rates

Remapping on write probably doesn't help

Read traffic depends more on applications than on system software/translation layer

Summary

Shingled-written disk is N bands of sequentially written sectors, each of order GB

Disk can still offer normal commands, write speed using “translation layer” embedded code

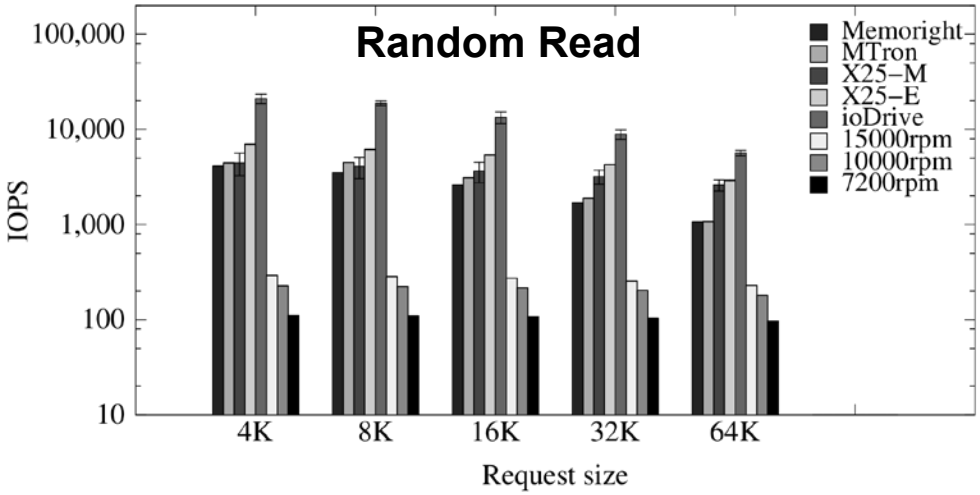
- Take Flash SSD FTL as starting point
- Flash-inspired TRIM command helps

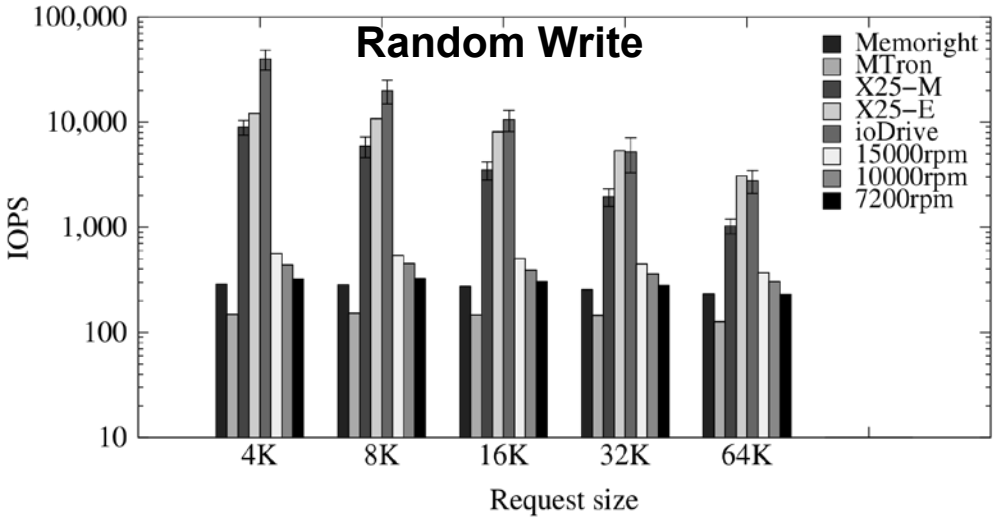
TDMR reading a bigger problem

- 3-5 revs per small read hard to hide
- This could reduce market acceptance

A Little More on SSD & Disks

- SSD performance !!
  - Big impact on systems coming
- Hybrid SSD+Disk
  - Cost of Disk bits
  - Speed of SSD
  - Compelling!
- SSD hybrid could "solve" TDMR speed issues





Carnegie Mellon®

www.pdl.cmu.edu

A few references

Rosenblum, M., J. Ousterhout, “The Design and Implementation of a Log-Structured File System,” ACM Trans. on Computer Systems, v10, n1, 1992.

Gal, E., Toledo, S., “Algorithms and data structures for flash memories,” ACM Computing Surveys, v37, n2, June 2005.

Agrawal, N., Prabhakaran, V., Wobber, T., Davis, J. D., Manasse, M., Panigrahy, R., “Design tradeoffs for SSD performance,” USENIX 2008 Annual Technical Conference, Boston MA, June 2008.

Polte, M., J. Simsa, “Enabling Enterprise Solid State Disk Performance,” Integrating Solid-state Memory into the Storage Hierarchy (WISH09), 2009.

www.pdl.cmu.edu and www.cs.cmu.edu/~garth