

# Operating System Support for NVM+DRAM Hybrid Main Memory

Jeffrey C. Mogul, Eduardo Argollo,  
Mehul A. Shah, Paolo Faraboschi  
HP Labs, Palo Alto & Barcelona



# The problems with DRAM

- Not dense enough
  - DRAM capacity limited by space, power, wire lengths
- Costs too much
  - Price increases non-linearly with density
- Takes too much power
  - Significant fraction of server power
- Can't get enough of it
  - Many applications are hungry for main memory
- Volatile – but please ignore that for this talk

# Good things about Non-Volatile Memory

For example, Flash:

- Can be denser than DRAM
- Could be cheaper/bit than DRAM
- Takes no refresh power
- Non-volatile – but ignore that!

# So, why not just use Flash instead of DRAM for main memory?

- That's crazy talk!
  - Flash reads are slower than DRAM
  - Flash writes are really, really, really, really slow
  - Flash has to be erased
  - Flash wears out pretty quickly
    - $10^5$  erase cycles – if you're lucky
    - Wear-out lifetime can decrease with increasing density

# It's so crazy, it just might work!

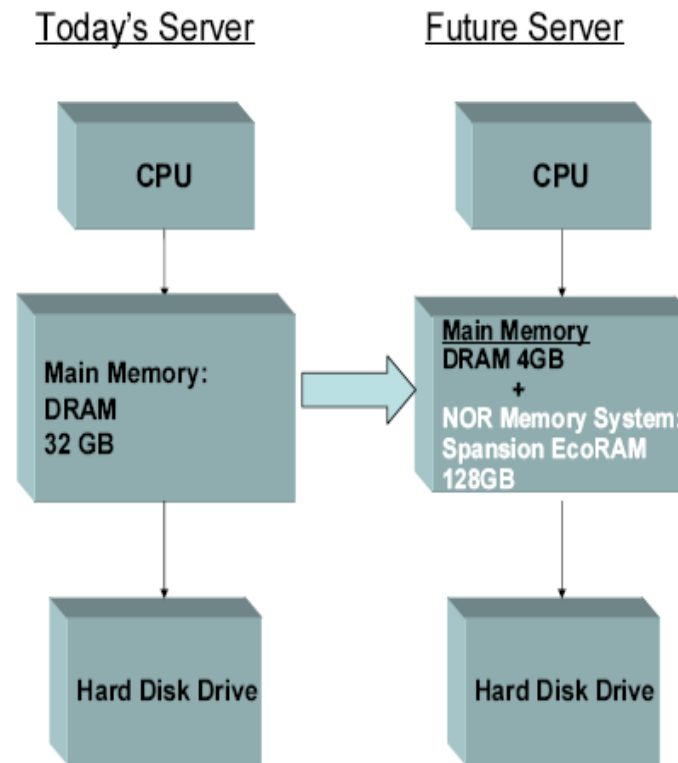
## What you should remember from this talk:

- Changes in memory technology will change the way we build main memory
  - Buzzword: “Universal Memory”
- Wear-out is the big problem
  - And slow writes/erases, too.
- The operating system is the best place to solve those problems

# It's so crazy, it just might work!

E.g., Spansion's "EcoRam™"

- Replace some DRAM with NOR flash
- Spansion claims:
  - Lower capital costs because you can use fewer servers
  - Lower operating costs:
    - slightly lower power/server
    - fewer servers
  - For 160TB-RAM Data Center
    - 48% lower CapEx
    - 75% lower OpEx
    - (Mostly R/O workload)



(5000x in "160TB Data Center")

Source: Spansion Inc.

Source: USING SPANSION® ECORAM™ TO IMPROVE TCO AND POWER CONSUMPTION IN INTERNET DATA CENTERS, Frost & Sullivan

# FLAM: a hybrid of Flash and DRAM

## Our proposed straw-man design:

- Replace part of DRAM with “FLAM DIMMs”
- Migrate pages from DRAM to Flash (in FLAM)
- On write attempts, fault page back to DRAM
- Use OS knowledge to manage migration policy
  - In particular, estimate Time To Next Write (TTNW)
  - Especially for complex workloads (not trivially read-only)

# Goal of this talk

- Semi-convince you that FLAM is a good idea
  - or at least a plausible idea
  - although flash isn't ideal for this use
  - maybe PC-RAM or something more exotic?
- Convince you that FLAM requires OS knowledge
  - To estimate TTNW for candidate pages
  - To optimize garbage-collection



# The role of the Operating System

- Use OS-level knowledge to avoid wear-out
- Migrate pages to FLAM when they are:
  - Hot for future reads (optimize scarce DRAM resources)
  - Cold for future writes (avoid wear-out & overheads)

# Memory characteristics

Tech	Density	Lifetime (erase cycles)	Rand. read time	Write time	Erase time	Erase size	Idle "on" power
DRAM	6-8 F <sup>2</sup>	10 <sup>15</sup>	~40-60 ns	~40-60 ns	---	---	~0.1—0.2W /DIMM
NAND flash	4-5 F <sup>2</sup>	10 <sup>5</sup> ..10 <sup>6</sup>	5—50 us	200 us /page	2 ms	512KB (e.g.)	~0
NOR flash	10 F <sup>2</sup>	10 <sup>5</sup> ..10 <sup>6</sup>	70 ns	1 us	1 sec	512KB (e.g.)	~0
PC-RAM	8-16 F <sup>2</sup>	10 <sup>8</sup> ..10 <sup>11</sup>	60 ns?	100—1000 ns	---	---	~0

(F = feature size)

# Memory characteristics

## why not NAND Flash?

Tech	Density	Lifetime (erase cycles)	Rand. read time	Write time	Erase time	Erase size	Idle "on" power
DRAM	6-8 F <sup>2</sup>	10 <sup>15</sup>	~40-60 ns	~40-60 ns	---	---	~0.2W /DIMM
NAND flash	4-5 F <sup>2</sup>	10 <sup>5</sup> ..10 <sup>6</sup>	5—50 us	200 us /page	2 ms	512KB (e.g.)	~0
NOR flash	10 F <sup>2</sup>	10 <sup>5</sup> ..10 <sup>6</sup>	70 ns	1 us	1 sec	512KB (e.g.)	~0
PC-RAM	8-16 F <sup>2</sup>	10 <sup>8</sup> ..10 <sup>11</sup>	60 ns?	100—1000 ns	---	---	~0

# Memory characteristics

## What's good about NOR Flash?

Tech	Density	Lifetime (erase cycles)	Rand. read time	Write time	Erase time	Erase size	Idle "on" power
DRAM	6-8 F <sup>2</sup>	10 <sup>15</sup>	~40-60 ns	~40-60 ns	---	---	~0.2W /DIMM
NAND flash	4-5 F <sup>2</sup>	10 <sup>5</sup> -10 <sup>6</sup>	5—50 us	200 us /page	2 ms	512KB (e.g.)	~0
NOR flash	10 F <sup>2</sup>	10 <sup>5</sup> -10 <sup>6</sup>	70 ns	1 us	1 sec	512KB (e.g.)	~0
PC-RAM	8-16 F <sup>2</sup>	10 <sup>8</sup> -10 <sup>11</sup>	60 ns?	100—1000 ns	---	---	~0

Does Spansion have double-density NOR cells?

# Memory characteristics

## What's a problem with NOR Flash?

Tech	Density	Lifetime (erase cycles)	Rand. read time	Write time	Erase time	Erase size	Idle "on" power
DRAM	6-8 F <sup>2</sup>	10 <sup>15</sup>	~40-60 ns	~40-60 ns	---	---	~0.2W /DIMM
NAND flash	4-5 F <sup>2</sup>	10 <sup>5</sup> -10 <sup>6</sup>	5—50 us	200 us /page	2 ms	512KB (e.g.)	~0
NOR flash	10 F <sup>2</sup>	10 <sup>5</sup> -10 <sup>6</sup>	70 ns	1 us	1 sec	512KB (e.g.)	~0
PC-RAM	8-16 F <sup>2</sup>	10 <sup>8</sup> -10 <sup>11</sup>	60 ns?	100—1000 ns	---	---	~0

# Solving the problems with NOR-based FLAM

- Low endurance:
  - Don't write pages with low ETTNW to FLAM
- Slow writes:
  - Buffer CPU's writes via small DRAM
    - CPU writes pages to DRAM buffer on FLAM DIMM
    - Simple controller in the FLAM DIMM manages copy to NOR
  - Don't write pages with low ETTNW to FLAM
- Large erase-block size:
  - Steal good ideas from garbage-collection people
  - Allocate pages with similar ETTNW to same erase block

# Solving the problems with NOR-based FLAM

- Low endurance:
  - Don't write pages with low ETTNW to FLAM
- Slow writes:
  - Buffer CPU's writes via small DRAM
    - CPU writes pages to DRAM buffer on FLAM DIMM
    - Simple controller in the FLAM DIMM manages copy to NOR
  - Don't write pages with low ETTNW to FLAM
- Large erase-block size:
  - Steal good ideas from garbage-collection people
  - Allocate pages with similar ETTNW to same erase block

# How the OS could help FLAM: Estimating per-page time-to-next-write

## Information that the OS has about pages:

- Page types (ANON, MAP2DSK, etc.)
- File types (executable, ...)
- File modes ("temporary", sequential)
- Application-supplied hints
  - E.g., "I'm a database and this is my read-mostly index"
- Dynamic information based on history:
  - Classified based on file names?
  - History tracked per-page??



# Memory characteristics

## What if PC-RAM becomes a reality?

Tech	Density	Lifetime (erase cycles)	Rand. read time	Write time	Erase time	Erase size	Idle "on" power
DRAM	6-8 F <sup>2</sup>	10 <sup>15</sup>	~40-60 ns	~40-60 ns	---	---	~0.2W /DIMM
NAND flash	4-5 F <sup>2</sup>	10 <sup>5</sup> -10 <sup>6</sup>	5—50 us	200 us /page	2 ms	512KB (e.g.)	~0
NOR flash	10 F <sup>2</sup>	10 <sup>5</sup> -10 <sup>6</sup>	70 ns	1 us	1 sec	512KB (e.g.)	~0
PC-RAM	8-16 F <sup>2</sup>	10 <sup>8</sup> -10 <sup>11</sup>	60 ns?	100—1000 ns	---	---	~0

4 F<sup>2</sup> in the future?

# How well would FLAM work?

## Limits on acceptable mean TBWP

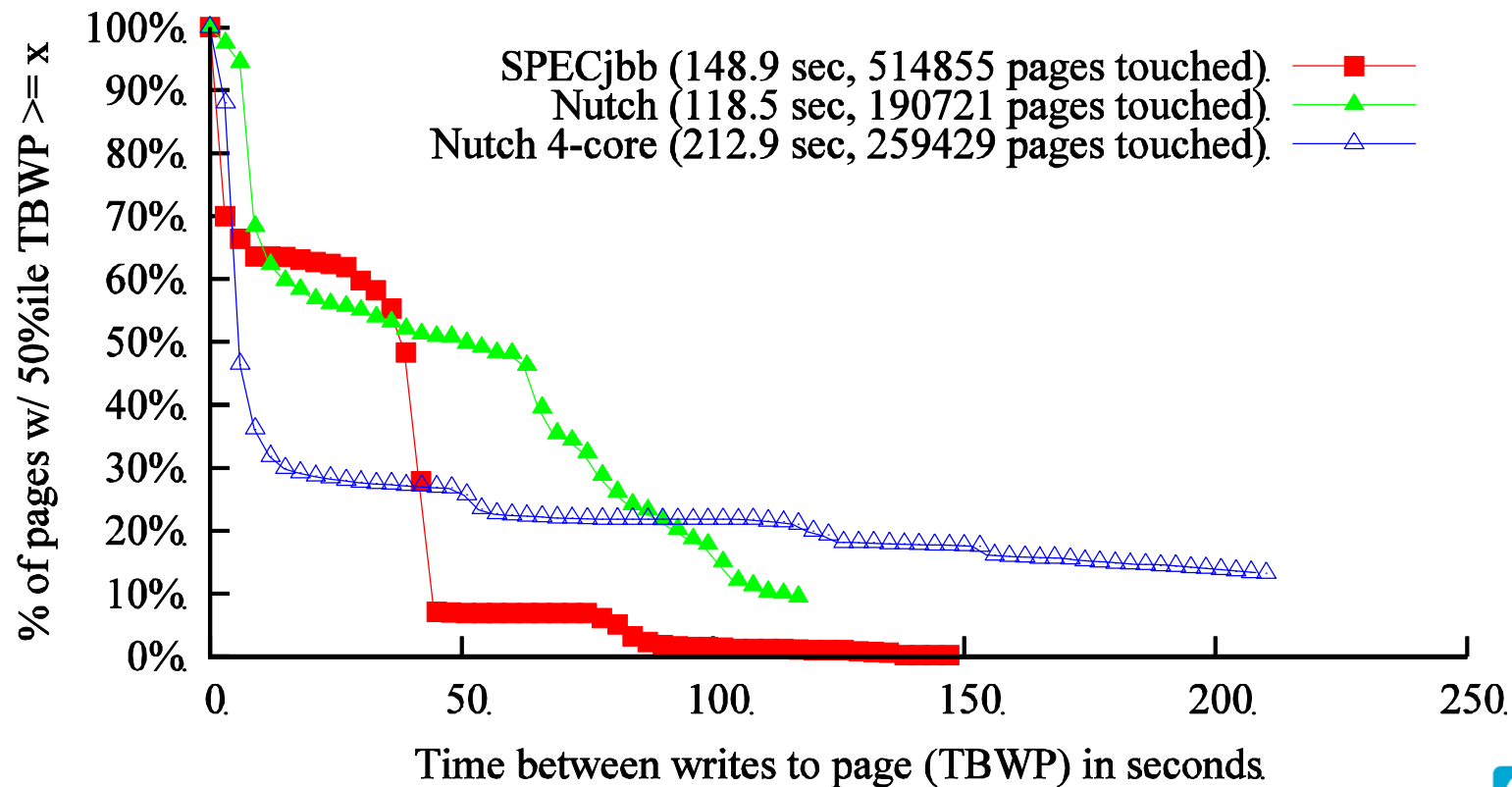
TBWP = Time Between Writes to a Page

- Assume a target lifetime of 5 years =  $1.58e8$  sec.
- Assuming  $10^6$ -erase lifetime for NOR:
  - Target mean TBWP = 158 sec = 2.6 min
- (Assuming  $10^5$ -erase lifetime for NOR:
  - Target mean TBWP = 1580 sec = 26 min)
- Assuming  $10^8$ -erase lifetime for PC-RAM:
  - Target mean TBWP = 1.58 sec

# How well would FLAM work?

## Preliminary experiments -- simulation

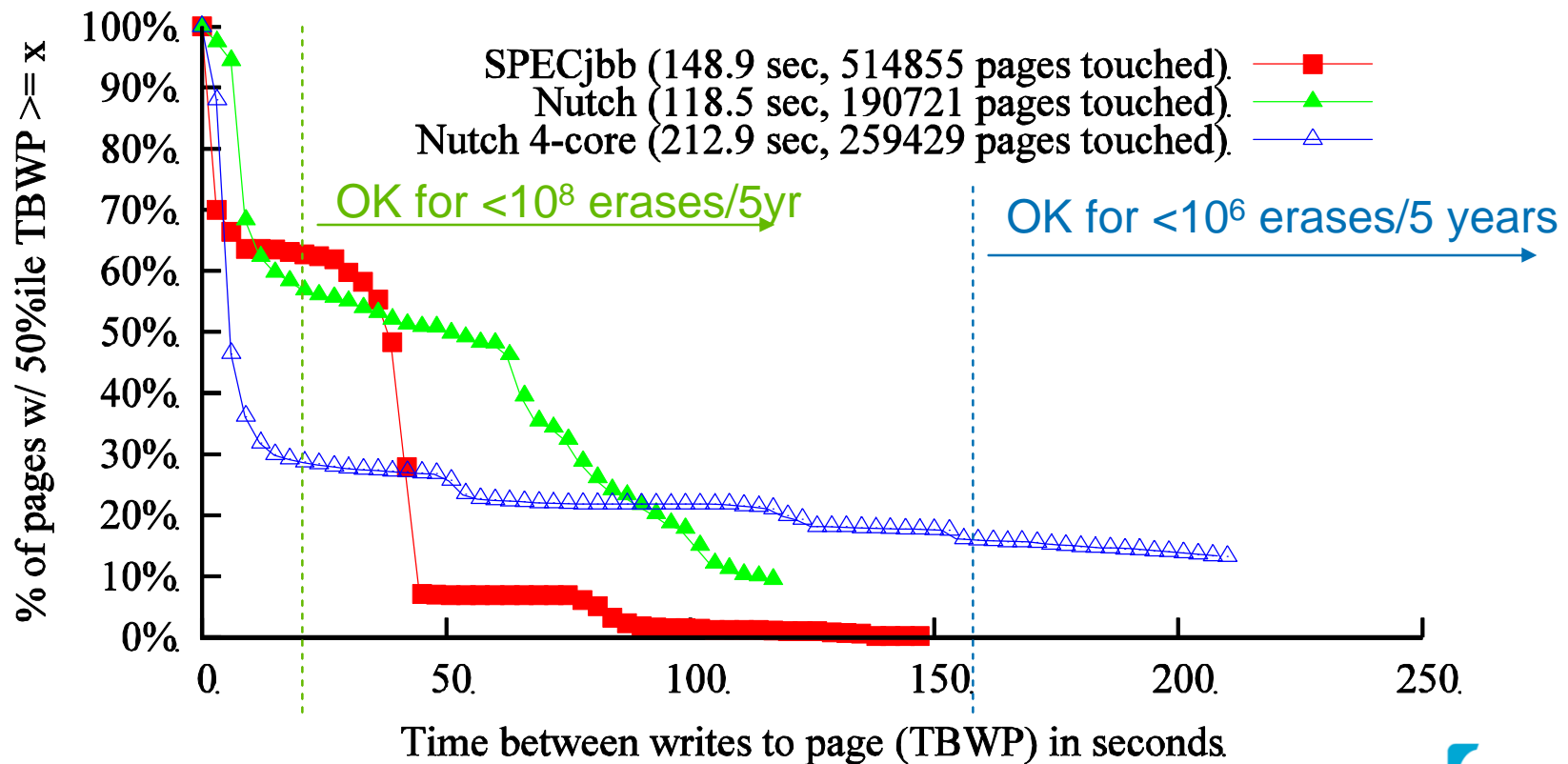
- Simulated whole system using COTSon
  - Opteron, Linux 2.6.15, Nutch or SPECjbb-like
  - Trace all L2\$ writebacks – 150-200 secs takes 2 weeks



# How well would FLAM work?

## Preliminary experiments -- simulation

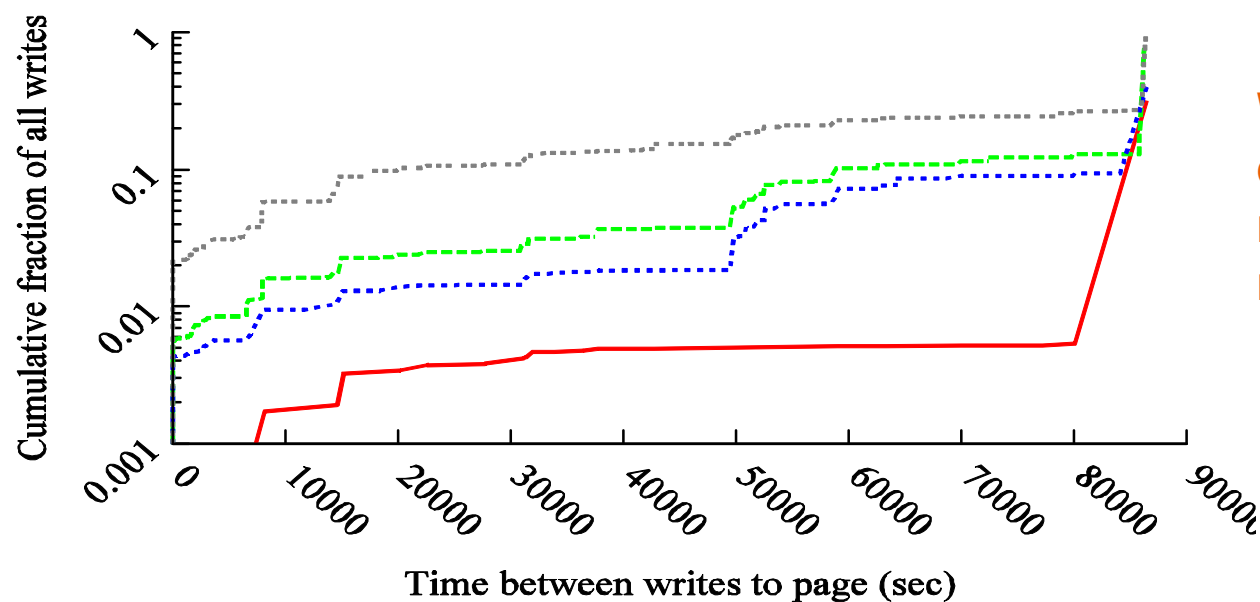
- Simulated whole system using COTSon
  - Opteron, Linux 2.6.15, Nutch or SPECjbb-like
  - Trace all L2\$ writebacks – 150-200 secs takes 2 weeks



# How well would FLAM work?

## Preliminary experiments -- Tracing

- Traces on actual hardware at (nearly) full speed
  - Linux 2.6.28.5, ran hacked SPECjbb for a whole day
  - Slightly hacked VM code tracks "PageHasBeenDirty" bit
  - Slightly hacked /proc/kpageflags, user code polls every 10 sec.



**WARNING: not clear if these results are meaningful!**


# Stuff we haven't done yet

- **Modify OS (e.g., Linux) to manage FLAM**
  - Could do this using DRAM as “fake FLAM” for testing
  - Could get realistic performance results with enough RAM
    - Would still have to model power consumption
  - Linux VM system is a bit scary
- **Characterize which applications might exploit FLAM**
  - Especially: where will extra read-mostly memory help performance?
- **Prototype FLAM hardware**
  - Will Spansion sell us what we want?
  - Is PC-RAM a better choice?
- **Think about exploiting non-volatility, too**
  - But flash isn't as reliable as you would hope/expect

# Summary

Using NVM for main memory is a crazy idea

- but it might work!
- and if it does work, the OS is the best place to make migration and placement decisions



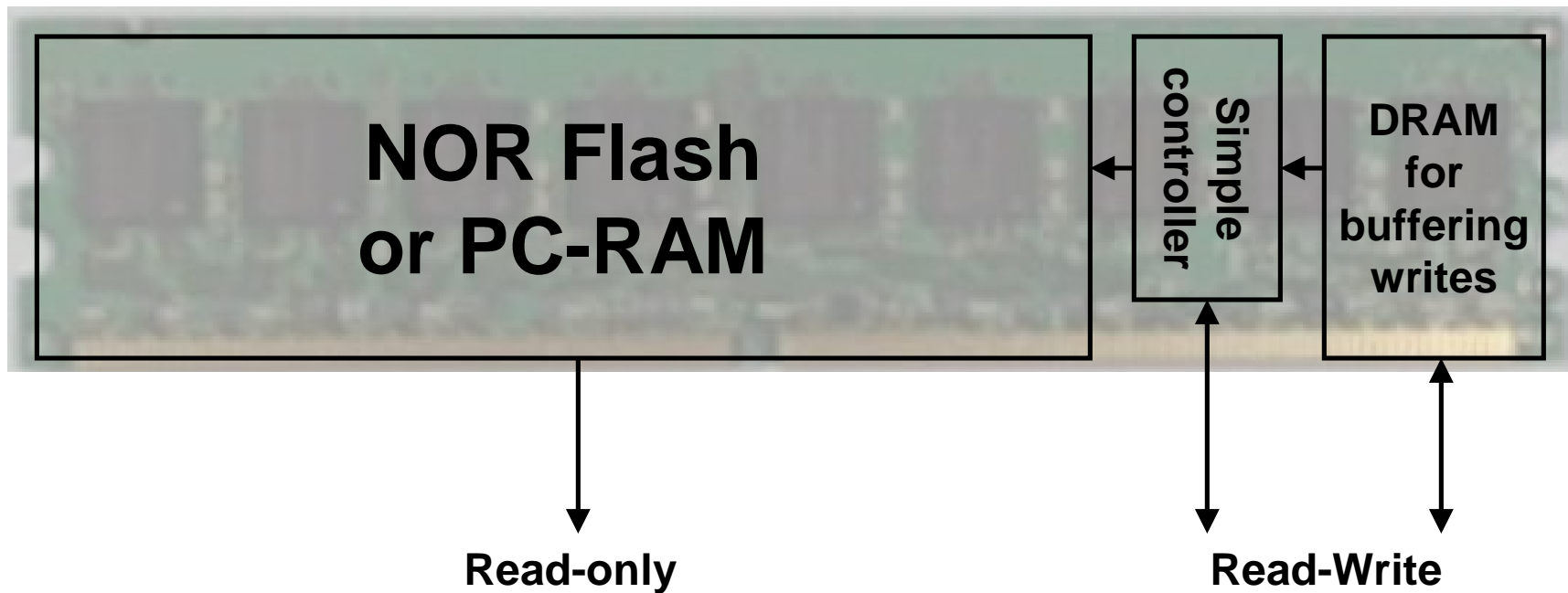
# Additional material



# Comparison of various NVM technologies

Memory Element	Density	CMOS Integration	Switch Mechanism	Bipolar /Unipolar	Write Power	Scaling	Ultimate Scaling Limit	Set-reset Times	Maturity
PCM (PC-RAM)	4F <sup>2</sup>	Demonstrated	Temperature	Unipolar	Poor	Fair	Stable nanocrystal size (~10nm)	Good	Prototype
Flash	4F <sup>2</sup>	Excellent	E-field	N/A	Good	Fair	Capacitor size	Fair	Product
FeRAM	4F <sup>2</sup>	Demonstrated	E-field	Bipolar	Good	Poor	Domain size (20nm)	Good	Product
MRAM	4F <sup>2</sup>	Poor (Fe)	B-field	Bipolar	Poor	Poor	Domain size (10nm)	Good	Specialty product

# Crudely-drawn design of a FLIM



# Security aspects of FLAM

- Avoid storing keys & plaintext in NVM
  - Increases chance of compromise
- Can the OS do this automatically?
  - Might require API to mark data as “please forget ASAP”
  - Or will DIFC make this work?

# Related products (recently announced): Flash-hybrid support for memcached

- memcached: “distributed memory object caching system ... intended for use in speeding up dynamic web applications by alleviating database load.”
- [gear6.com](http://gear6.com): hybrid DRAM-flash architecture for memcached
  - “allows for 5-10x more memcache memory / rack unit”
  - “cuts memory costs by 50%”
  - up to 320GB
- [schoonerinfotech.com](http://schoonerinfotech.com):
  - appliances for memcached, SQL acceleration
  - 512GB flash, 64GB DRAM, Intel CPUs
- Both use: NAND flash, networked access