


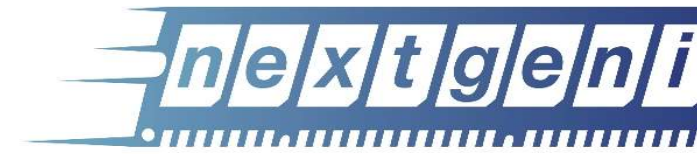


General I/O and Persistent Memory Hardware

Adrian Jackson
a.jackson@epcc.ed.ac.uk
@adrianjhpc 

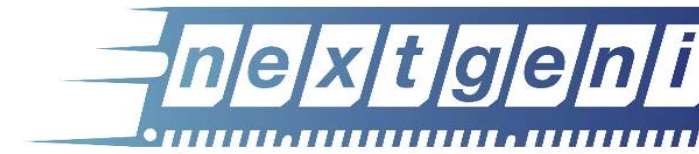


I/O



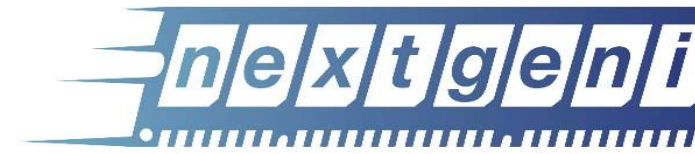
- I/O essential for all applications/codes
 - Some data must be read in or produced
 - Instructions and Data
- Basic hierarchy
 - CPU – Cache – Memory – Devices (including I/O)
- Often “forgotten” for HPC systems
 - Linpack not I/O bound
 - Not based on CPU clock speed or memory size
- Often “forgotten” in program
 - Start and end so un-important
 - Just assumed overhead

I/O



- Small parallel programs (i.e. under 1000 processors)
 - Cope with I/O overhead
- Large parallel programs (i.e. tens of thousand processors)
 - Can completely dominate performance
 - Exacerbate by poor functionality/performance of I/O systems
- Any opportunity for program optimisation important
 - Improve performance without changing program

Challenges of I/O

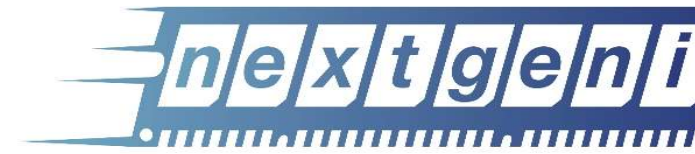


- Moves beyond process-memory model
 - data in memory has to physically appear on an external device
- Files are very restrictive
 - Don't often map well to common program data structures (i.e. flat file/array)
 - Often no description of data in file
- I/O libraries or options system specific
 - Hardware different on different systems
- Lots of different formats
 - text, binary, big/little endian, Fortran unformatted, ...
 - Different performance and usability characteristics
- Disk systems are very complicated
 - RAID disks, caching on disk, in memory, I/O nodes, network, etc...

Performance

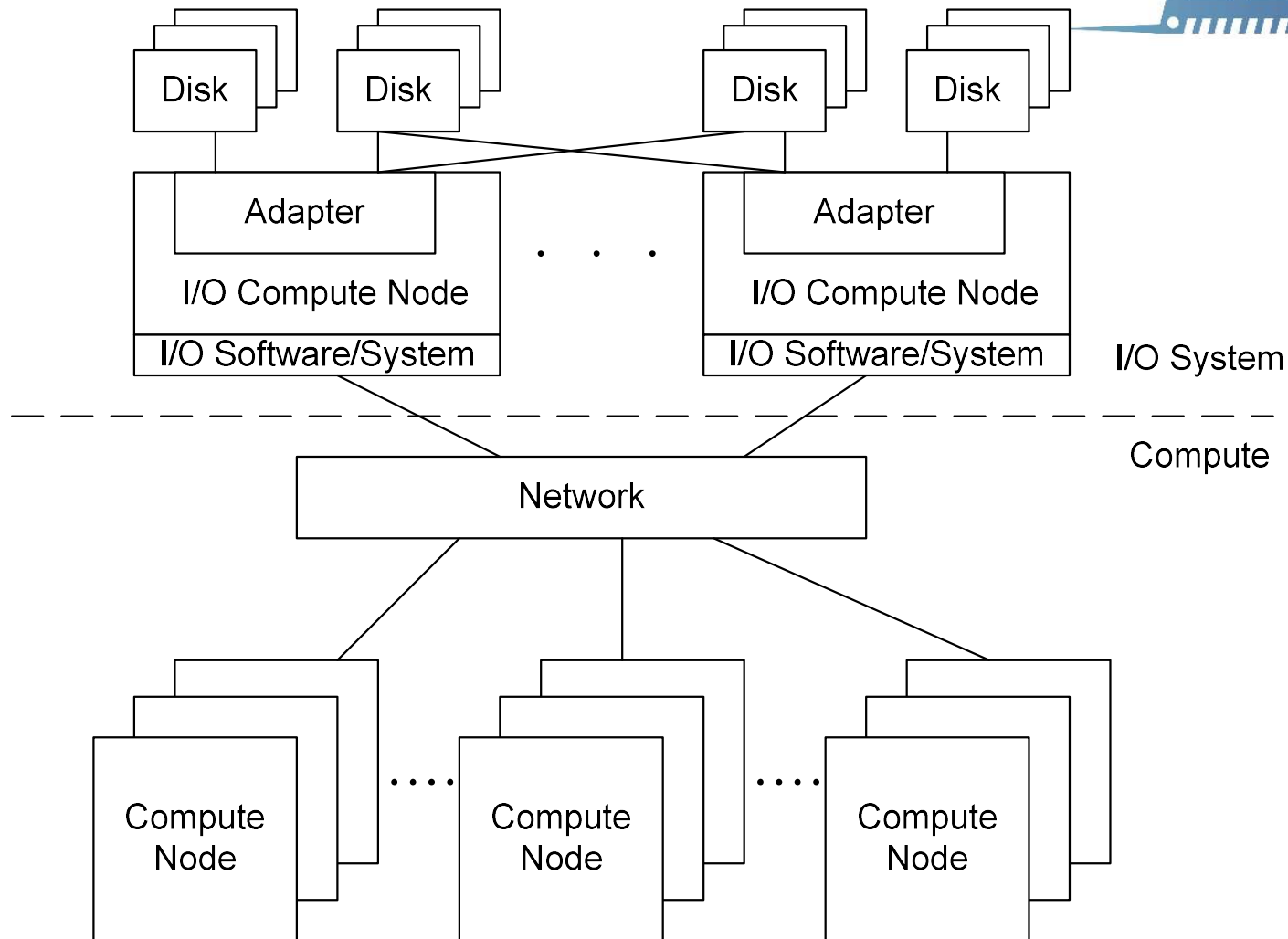
Interface	Throughput Bandwidth (MB/s)
PATA (IDE)	133
SATA	600
Serial Attached SCSI (SAS)	600
Fibre Channel	2,000
NVMe	2,000

High Performance or Parallel I/O

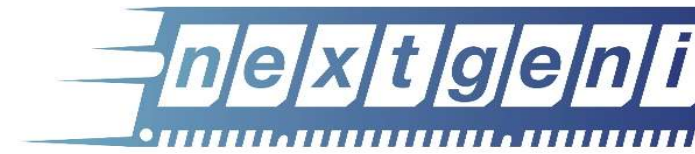


- Lots of different methods for providing high performance I/O
- Hard to support multiple processes writing to same file
 - Basic O/S does not support
 - Data cached in units of disk blocks (eg 4K) and is *not coherent*
 - Not even sufficient to have processes writing to distinct parts of file
- Even reading can be difficult
 - 1024 processes opening a file can overload the filesystem limit on file handles etc....
- Data is distributed across different processes
 - Dependent on number of processors used, etc...
- Parallel file systems may allow multiple access
 - but complicated and difficult for the user to manage

Hierarchy



Non-volatile memory



- Non-volatile memory stores data after power off
 - SSDs (NAND Flash) are common examples
 - Similar technology in memory cards for your phones, cameras, etc...
- These store data persistently but are generally slow and less durable than volatile memory technologies (i.e. DDR memory)

JEDEC NVDIMM standards



- NVDIMM-F

- Traditional flash solution with controller on board
- NAND flash performance and size

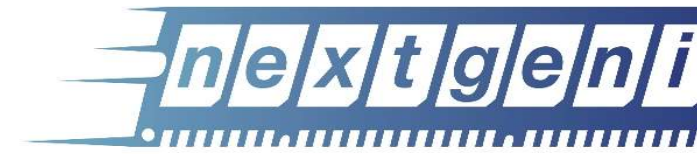
- NVDIMM-N

- DRAM with Flash for backup
- Separate power supply (i.e. super capacitors) allow data to be copied to flash on power failure
- Limited by DRAM size and capacitor sizes
- DRAM performance and size

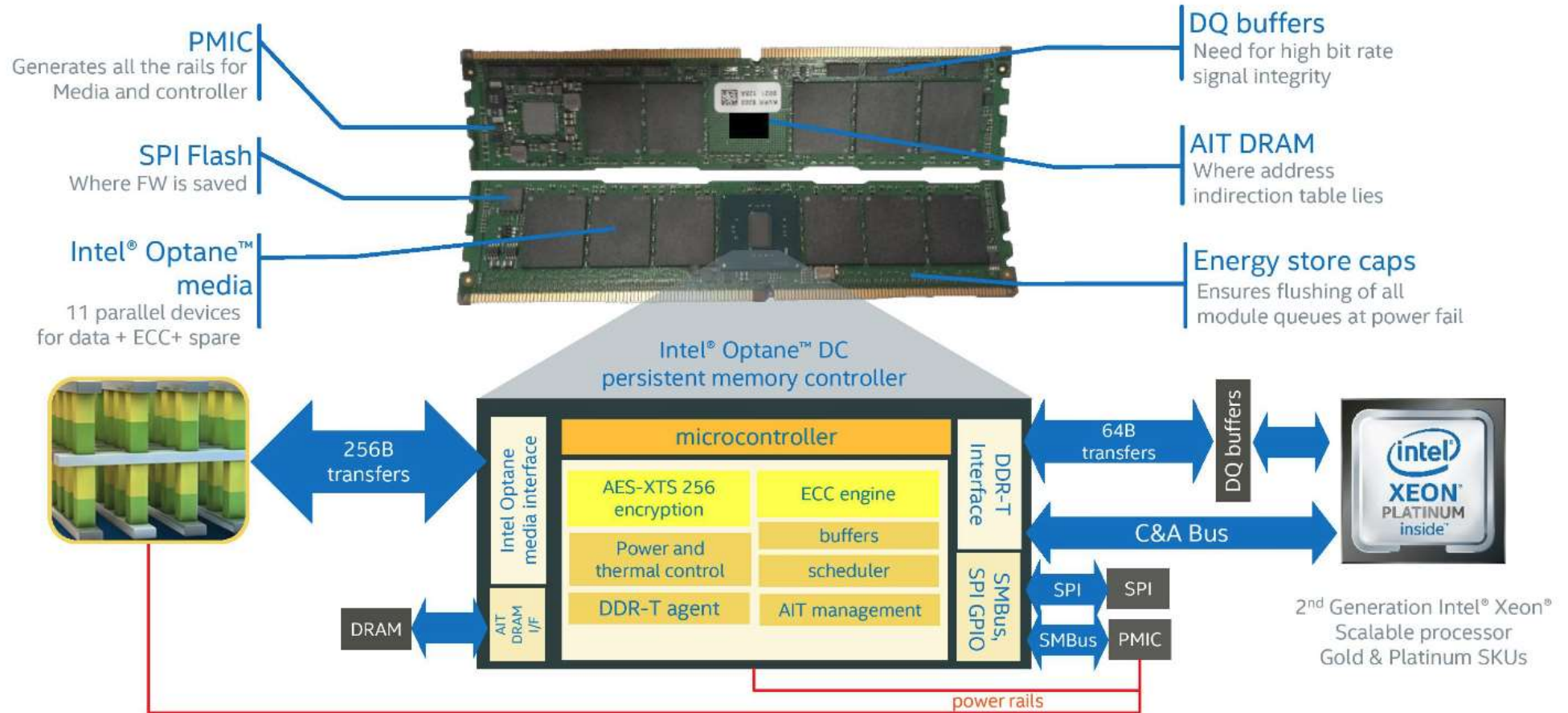
- NVDIMM-P

- Channel support for mixed memory types
- Protocol to enable transactional access
- Different access latencies allowed between media types
- Intel Optane DCPMM, technically, does not implement the NVDIMM-P standard but it is conceptually NVDIMM-P

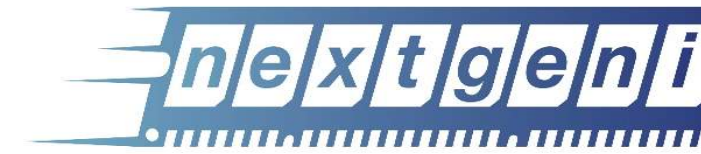
Intel Optane DCPMM



COMPLETE SYSTEM ON A MODULE

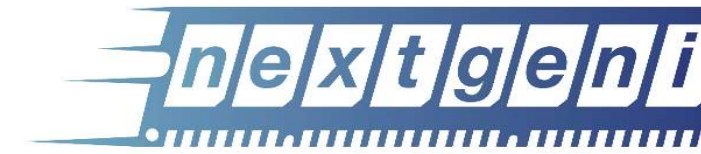


Intel Optane DCPMM

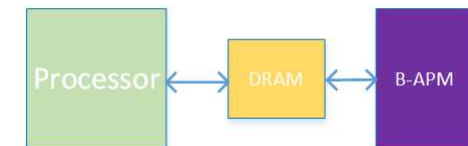
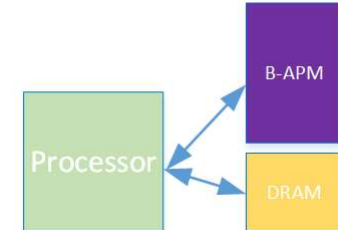


- Non-volatile RAM
 - Optane memory
- Much larger capacity than DRAM
 - Hosted in the DRAM slots, controlled by a standard memory controller
- Slower than DRAM by a small factor, but significantly faster than SSDs
- Read/write asymmetry and other interesting performance factors
- High endurance (5 year warranty)

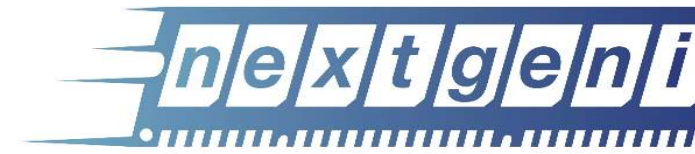
Intel Optane DC PMM



- Requires specific processors to support the hardware
 - PM-enabled memory controller required
 - Deal with different latency memory bus traffic
- Has different modes of operation
- 1LM – App Direct
 - Both memories are visible to the program
 - Using the PM requires program changes
- 2LM – Memory Mode
 - DRAM used as Last Level Cache (LLC) for PM
 - Transparent exploitation but no persistence

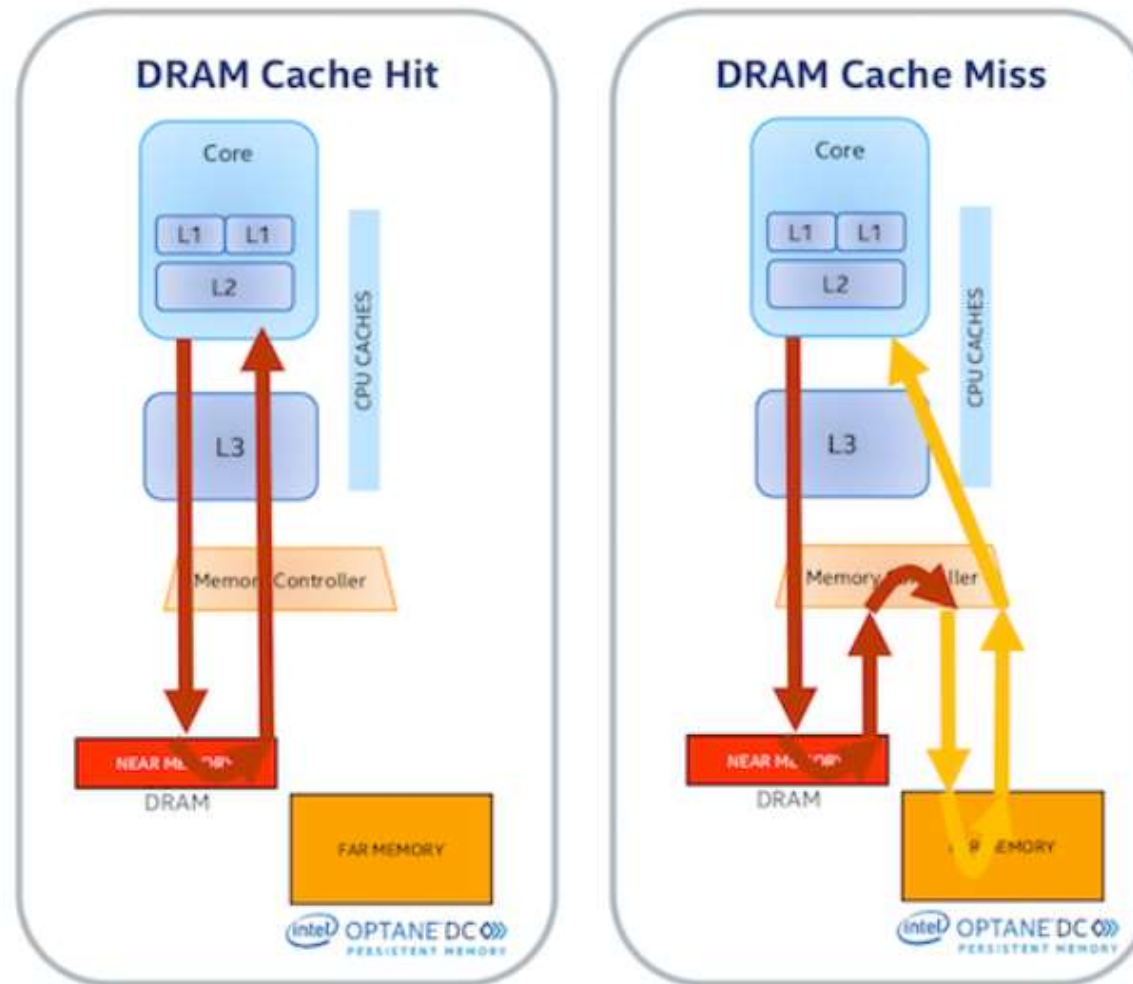
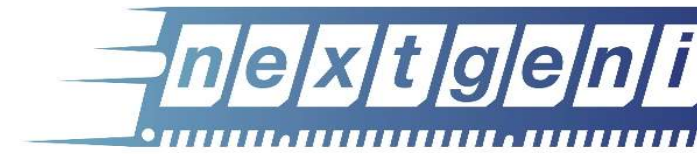


Optane DCPMM

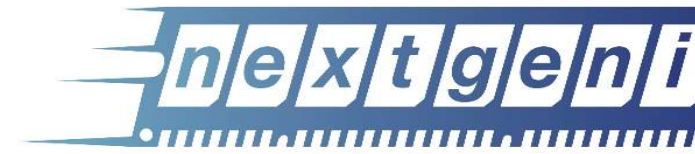


- Cache coherent data accesses
 - Byte addressable (cache line)
- Requires reboot before switching platform now
 - Memory mode (2LM)
 - App direct (1LM)
 - fsdax
 - Filesystem block device for creating namespace
 - devdax
 - Character device, no namespace
 - Performance identical once file is created
- Can partition system to have both memory and app direct spaces

Memory mode

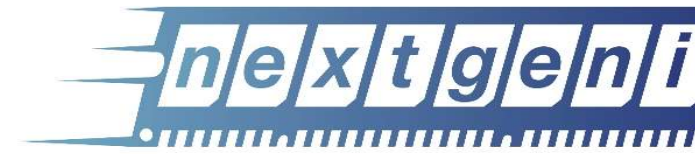


Optane DCPMM



- Socket based systems means NUMA when not a single socket system
- Performance dependent on using local memory
- Factor $\sim 4x$ write performance for using local memory when fully populating nodes
- Factor $\sim 2x$ read performance for using local memory when fully populating nodes

NUMA programming



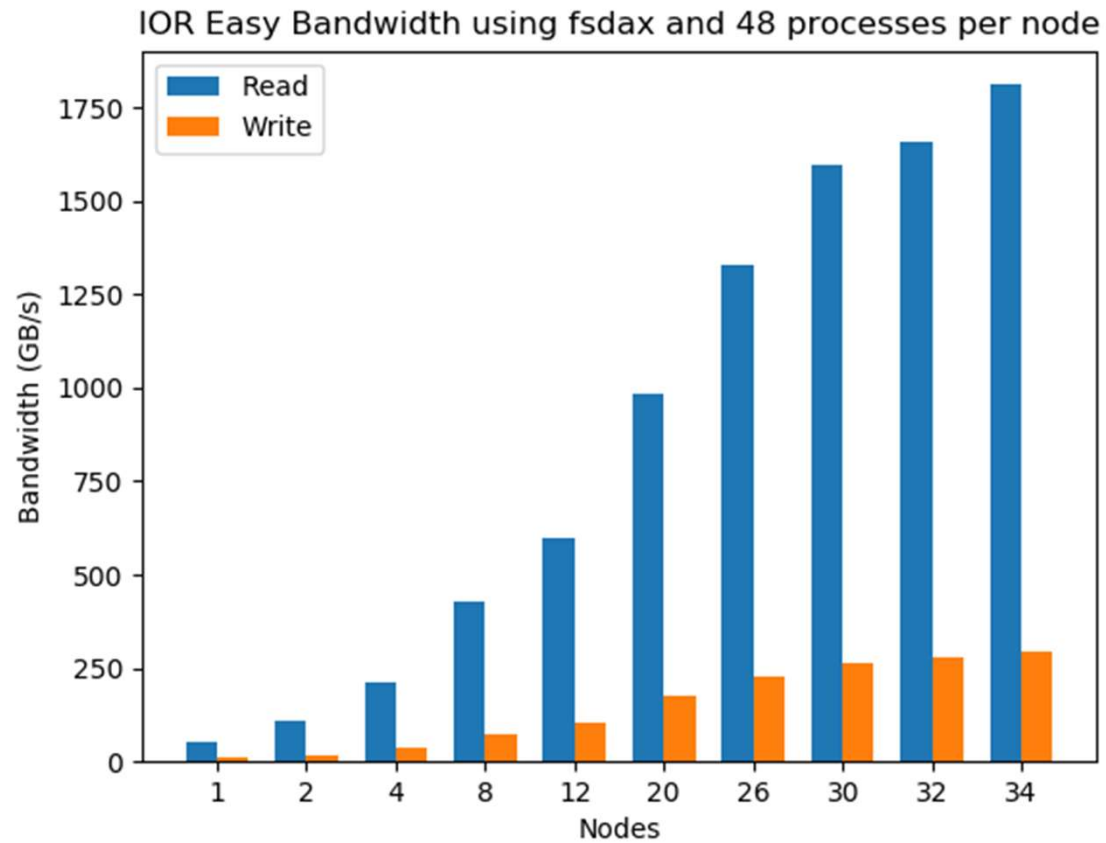
- Intel specific:

```
unsigned long GetProcessorAndCore(int *chip, int *core){
    unsigned long a,d,c;
    __asm__ volatile("rdtscp" : "=a" (a), "=d" (d), "=c" (c));
    *chip = (c & 0xFFF000)>>12;
    *core = c & 0xFFF;
    return ((unsigned long)a) | (((unsigned long)d) << 32);;
}
```

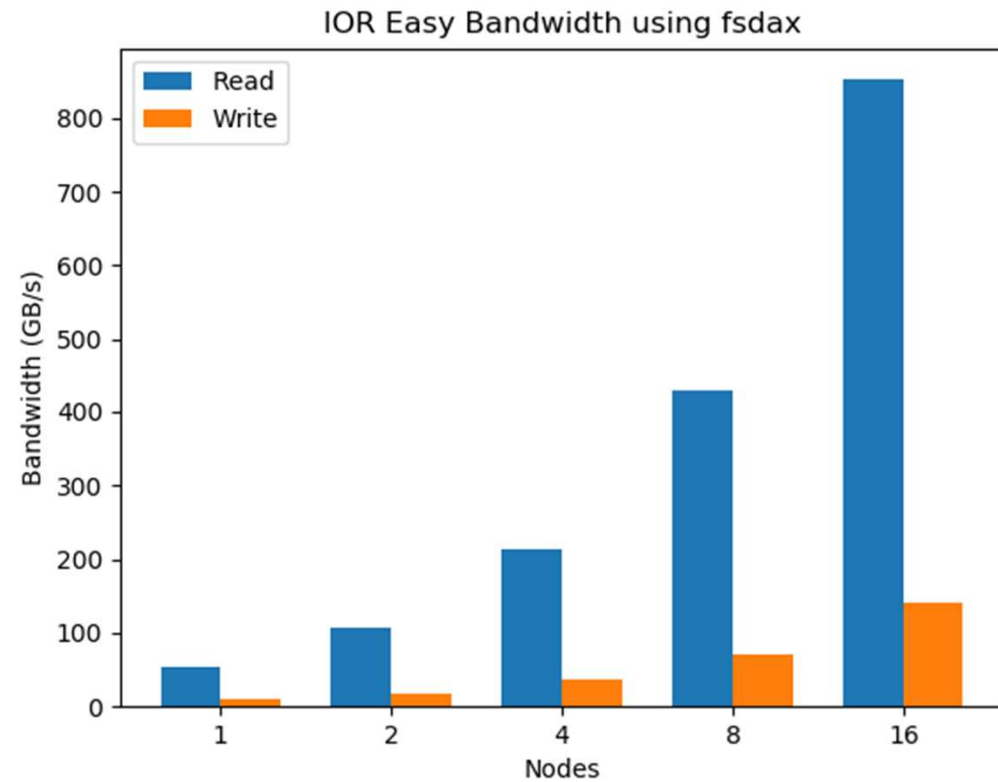
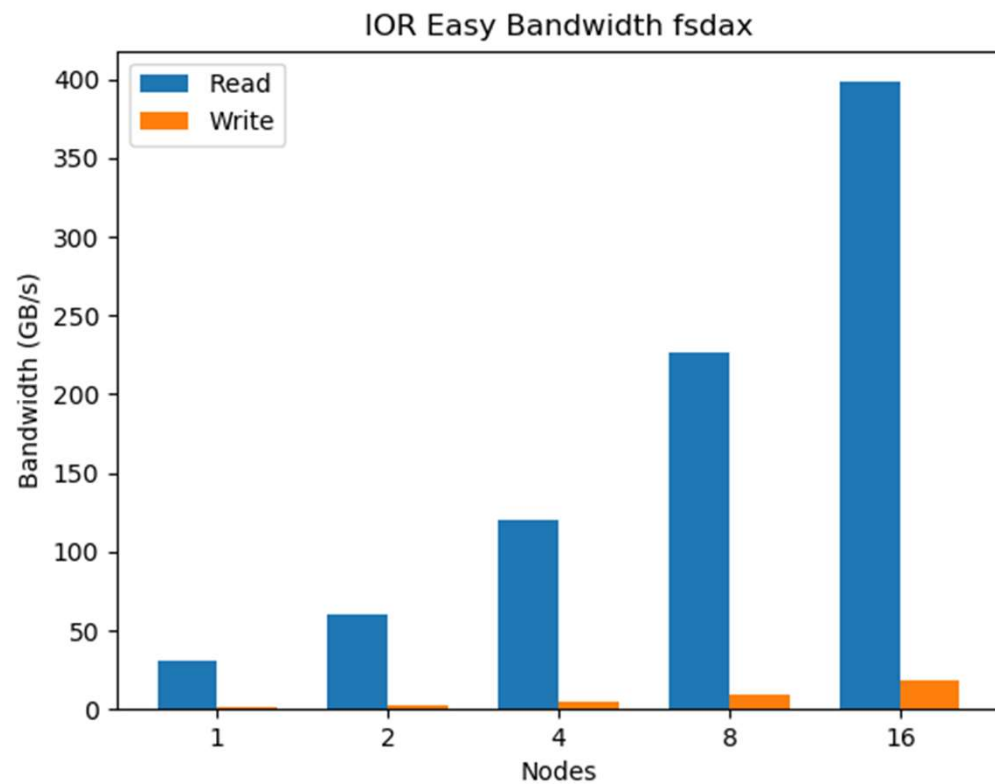
- Arm specific:

```
unsigned long GetProcessorAndCore(int *chip, int *core){
    return syscall(SYS_getcpu, core, chip, NULL);
}
```

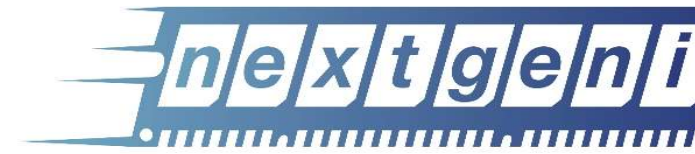

Overall performance



NUMA performance



Performance asymmetry



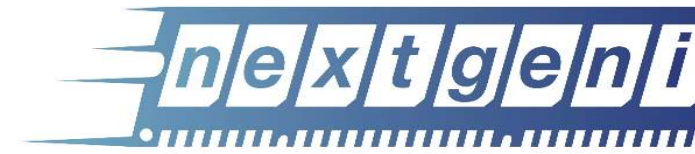
- Read is $\sim 3x$ slower than DRAM
- Write is $\sim 7x$ slower than DRAM
- Read is 4x-5x faster than write for Optane
- Write queue issues can mean variable performance
 - Optane has active write management
 - On-DIMM controller
- Accesses are coalesced into blocks of i.e. 256 bytes

INTEL® OPTANE™ DC PERSISTENT MEMORY PERFORMANCE DETAILS

- Intel® Optane™ DC persistent memory is programmable for different power limits for power/performance optimization
 - 12W – 18W, in 0.25 watt granularity - for example: 12.25W, 14.75W, 18W
 - Higher power settings give best performance
- Performance varies based on traffic pattern
 - Contiguous 4 cacheline (256B) granularity vs. single random cacheline (64B) granularity
 - Read vs. writes
 - Examples:

Granularity	Traffic	Module	Bandwidth
256B (4x64B)	Read	256GB, 18W	8.3 GB/s
256B (4x64B)	Write		3.0 GB/s
256B (4x64B)	2 Read/1 Write		5.4 GB/s
64B	Read		2.13 GB/s
64B	Write		0.73 GB/s
64B	2 Read/1 Write		1.35 GB/s

Volatile Memory



- Two principal volatile memory technologies: SRAM and DRAM.
- DRAM (dynamic random access memory) is used for main memory of almost all current machines.
- SRAM (static random access memory) predominantly used for caches.
- DRAM uses a single transistor for each bit.
 - reading the bit can cause it to decay
 - needs to be refreshed periodically
- SRAM uses 4-6 transistors per bit
 - no need to refresh

Wide memory path

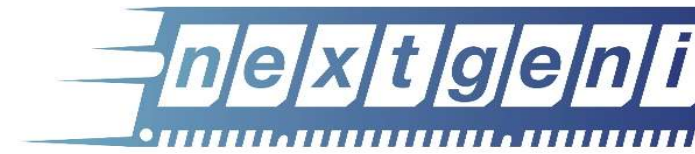
- Instead of the connection from cache to memory being one word wide (32 or 64 bit), it can be many words wide (typically 32 or 64 bytes).
- Allows, for example, a level 3 cache line to be loaded all at once.
- Memory is normally organised in **banks**
- Different banks can service read or write requests concurrently.
- Addresses are striped across banks to minimise the possibility that two consecutive accesses go to the same bank.

Bank 0	Bank 1	Bank 2	Bank 3
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Virtual memory

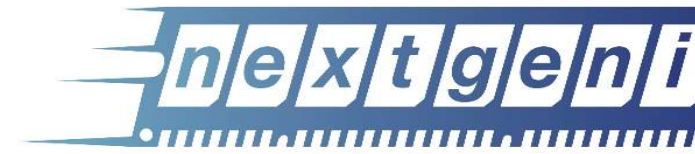
- Allows memory and disk to be seamless whole
 - processes can use more data that will fit into physical memory
- Allows multiple processes to share physical memory
- Can think of main memory as a cache of what is on disk
 - blocks are called **pages** (4 to 64 kbytes)
 - a miss is called a **page fault**
- CPU issues **virtual** addresses which are translated to physical addresses.
- Pages can be placed anywhere in memory
 - like a fully associative cache
 - approximate LRU replacement strategy
 - write back, not write through
- Mapping from virtual to physical address is stored in a **page table** in memory
- Page table lookup is relatively expensive
- Page faults are very expensive
 - requires system call (~1ms)

Summary



- Optane hardware is complicated
- Performance is workload dependent (when isn't it?)
- Targeted usage will be required for the best performance
- I/O has been problematic for a while anyway

Pricing



- Don't trust these numbers
- Availability does not seem to be there at the moment
- Info from tomshardware.co.uk (old)

	128 GB	256 GB	512 GB
Intel guide price	\$577	\$2125	\$6751
Shop1	\$892	\$2850	
Shop2	\$695	\$2,595	\$8,250
Shop3	£755		
DRAM	~\$4500	N/A	N/A

Pricing



- List prices

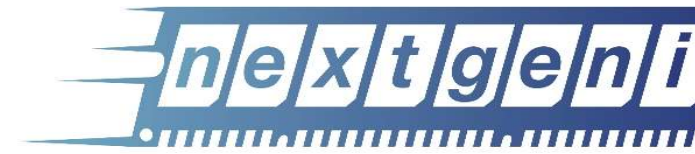
Optane DPCMM

Size (GB)	Cost (€)	€/GB
128	2223	17.36
256	7638	29.83
512	23199	45.31

DRAM

Size (GB)	Cost (€)	€/GB
8	347	43.37
16	444	27.78
32	855	26.71
64	2063	32.23
128	4959	38.74

Practical



- Take IOR and make it NUMA aware
- Adding in function call to work out which socket a process is running on
 - Create file based on this
- Two mnt points are available
 - /mnt/pmem_fsdax0
 - /mnt/pmem_fsdax1