Sunday, May 31, 2015

Solid State Drive (SSD)

SSD基本结构：
Die -> Plane -> Block-> Page(包含关系)
Pages size: 2 KB, **4 KB**, 8 KB or 16 KB.
Most SSDs have blocks of **128 or 256 pages**, which means that the size of a block can vary between **256 KB and 4 MB.**

SSD相对于HDD
主要优点：
一个是省电，另一个是没有seek time，意味着small random access表现好, 原因见上面连接
主要缺点：
一个是单位价格贵，另一个是数据相对于HDD并不那么endurable和writable的(lifetime)

BenchMark中常出现的几个概念：
Throughput = IOPS x I/O size(Chuck size)
Sequential I/O和Random I/O
Sequential Write VS. Random Wirte
referring to accesses as being **"sequential" or "random".** An I/O operation is said to be **sequential** if its **starting logical block address (LBA) directly follows the last LBA of the previous I/O operation.** If this is not the case, then the operation is said to be random.

SSD最常用的两个interface: **Serial ATA (SATA), PCI Express (PCIe)**
SATA 3.0 interface：
data can be transferred up to **6 Gbit/s**, which in practice gives around **550 MB/s, support NCQ**
PCIe 3.0 interface：
data can be transferred up to **8 GT/s per lane**, which in practice is roughly **1 GB/s** (GT/s stands for Gigatransfers per second). SSDs on the PCIe 3.0 interface are **more than a single lane**. With four lanes, PCIe 3.0 can offer a maximum bandwidth of 4 GB/s, which is eight times faster than SATA 3.0.

Pre-conditioning
持续30分钟左右的random write会使SSD的IOPS(throughput)骤降，avg latency剧升，因为garbage collection机制需要通过erase block来应对新的写入指令，这个机制本来是后台执行，但随着愈发繁忙要调到前台来和host command指令(后续write operation或read operation)进行竞争，从而降低了SSD的throughput.

基本操作
Reads (page-aligned size)
Write (page-aligned size):
即便只写1byte, 整个page都要被写. Writing more data than necessary is known as **write amplification**
这也是为啥page size不能太大的原因，同时可以用**RAM**来buffer several small write来实现one big write
Pages cannot be overwritten:
每一个page有一个state, 标记free的才能被写入。如果data变了，其内容会load到internal register里，然后更新其data, 最后写入一个新的"free" page, 这个操作叫做 **"read-modify-write".** 之前的page 被标记为"stale", it will remain as such until it is erased.
Erases (block-aligned size):
如果需要回收 stale pages，SSD controller 的garbage collection会自动发出erase指令

Wear leveling
如果反复读写同一部分blocks，到了P/E上限，是怎样的体验？ 其实就是500GB的SSD变成了250GB的容

量，因此**SSD controller的FTL要distributes P/E cycles as evenly as possible among the blocks**

FTL(Flash Translation Layer) 主要功能：
(1) logical block mapping (Map LBA to PBA)
a. page-level mapping(mapping table要占用不少RAM空间，从而增加了制造成本)
b. block-level mapping(考虑到很多small update的时候，会increase write amplification, 因为block太大了)
c. **log-block mapping(similar to log-structured file systems):**
写操作以**sequentially**的方式写到**log blocks. When a log block is full, it is merged with the data block associated to the same logical block number (LBN) into a free block.**
(2) garbage collection (GC):
清除掉stale page并标记成free page, 因为delay比较高，所以在后台运行，而且是idle time的时候运行，这就是为什么sustained small random write 30min之后SSD性能骤降的主要原因

Split cold and hot data for GC：
如果一个page含有部分cold data同时含有一些hot data，这样的话在GC 为了实现wear levering的时候，cold data会随着hot data一同拷贝，同时又增加了write application，因为cold data是不必要的写入。这个问题可以通过把cold data 和 hot data放到不同的page里来解决，但又引入了另一个蛋疼的问题，cold data pages 不经常被erase, 因此我没还需要时不时的swap cold data page和hot data page来保证wear leveling. 这样可以使GC的工作量降低

Flash != SSD，但大多数SSD都是基于NAND flash**(其特点是write in pages, erase in blocks of pages)**
NAND FLASH的类型(type) (从上到下越来越便宜，endurance越来越差):
SLC(Single Level Cell): 1bits/cell(**density**最低，所以**endurance** 最好，也是最**expensive**的）100k P/E
eMLC(enterprise Multiple Level Cell): 2bits/cell
MLC(Multiple Level Cell): 2 or more bits/cell (大名鼎鼎的Pure storage用的就是MLC) 5K P/E
TLC(Triple Level Cell): 3bits/cell
program/erase cycles, or P/E cycles

提一个比较高大上的概念：**NAND shrinking lithography**，上个数据体会下：
体积越小越容易出错，对**ECC**的要求越高，所以实际存储空间在**shrinking**
Current generation
32nm SLC: 8 bit ECC for each 512 bytes
32nm MLC: 24 bit ECC for each 1024 bytes
Previous generation SLC
54nm SLC: 1bit ECC for each 512 byte

SLC和MLC的trade off：
总的来说就是MLC的缺点主要有两点：
(1) **shorter life span:**
SLC: 100,000 write cycles; MLC: only 3,000-5,000
(2) **higher error correction overhead**
MLC在每一个write cycle 中会产生相对多的erorrs, 所以对ECC(error correction codes)的要求较高，意味着需要reserve更多的space来存ECC
但现在的**flash controller** 已经基本可以弥补这些缺陷
总结就是，**first tier**还是用**SLC**比较好，**middle tier**就可以用**MLC**了，当然要具体问题具体分析，比如：
For **high-update workloads, SLC** is the best choice, whereas for **high-read and low-write workloads (ex: video storage and streaming), then TLC** will be perfectly fine.

Parallelism:
SSD controller has 4-10 channels into storage, but only one thing to do, so it underperforms

介绍parallelism前需要知道几个概念
Cluster Block:
我们知道每个chip有多个plane，每个plane又有多个block，cluster block 就是每个plane中的number一样的block(stripe)

Trim:
其实就是SSD controller并不知道logical data什么时候被user删除的(OS overview是logical location, SSD device overview 是physical location, 所以当os标记logical location free掉了之后，SSD controller其实并不知道physical location已经被delete了)，只有同一个physical location再次被写入的时候，SSD controller才知道这块地方要被回收(GC)，但是从OS overview上，OS要把空间report给用户，但又不能每次再overwrite同一个logical block的时候再report，所以要发送trim指令给controller，让他提前把对应的physical location erase掉。
用trim的另一个原因就是，如果controller并不知道那块数据已经失效了，GC在运行时会触发wear leveling机制，把无用的数据搬来搬去，so, really bad for performance.

Over-provisioning:
生产厂商预留空间作为替换坏块用的，这就是为什么到手的256G SSD的你看到的容量只有240G左右。我们知道在sustained random write的时候，在硬盘block块都被占用的时候，GC会十分繁忙，而且erase block的latency很高，此时write指令还在源源不断的接受，而controller要不断erase，此时OP就可以当做buffer(absorb high throughput write workloads) 来为GC erase block来争取时间

Native Command Queueing (NCQ)
Feature of **Serial ATA**
同时接受多个用户请，通过internal parallelism机制实现 concurrency

Power-loss protection用supercapacitor来蓄电保证commit I/O requests

Channel-level parallelism. The flash controller communicates with the flash packages through multiple channels. Those channels can be accessed independently and simultaneously. Each individual channel is shared by multiple packages.
Package-level parallelism. The packages on a channel can be accessed independently. Interleaving can be used to run commands simultaneously on the packages shared by the same channel.
Chip-level parallelism. A package contains two or more chips, which can be accessed independently in parallel. Note: chips are also called “dies”.
Plane-level parallelism. A chip contains two or more planes. The same operation (read, write or erase) can be run simultaneously on multiple planes inside a chip. Planes contain blocks, which themselves contains pages. The plane also contains registers (small RAM buffers), which are used for plane-level operations.

Access Patterns
一般情况下, random write比sequential write慢，但是如果random write的size是cluster block(通常是16～32MB) 的倍数的话，就可以有和sequential write相当的表现(parallelism)

Single large write OR many small concurrent write?
通常情况下，他们的throughput相同, 但考虑到latency, large write的response time更快，所以前者好
但如果 I/O is small and cannot be grouped, 后者更好

To improve the read performance, write related data together
read performance和write pattern息息相关，write related data in the same page, block, or clustered block, 可以使得后续read 变快，得益于internal parallelism.

A single large read is better than many small concurrent reads
Concurrent random reads 无法充分利用readahead(pre-fetching buffer) mechanism. 另外，多个 Logical Block Addresses可能会map到一个chip里，并行并不能被充分利用. 相反, a large read operation如果是访问地址的话就会利用上readahead buffer. 因此，single large read更好

Separate read and write requests
A mix of small interleaved reads and writes will prevent the internal caching and readahead mechanism to work properly, and will cause the throughput to drop. It is best to avoid simultaneous reads and writes, and perform them one after the other **in large chunks, preferably of the size of the clustered block**

From **Design Tradeoffs for SSD performance**

SSD的**performance**和**lifetime**是workload-sensitive, 也就是说这俩属性和workload息息相关
SSD：
(1) producing exceptional bandwidth 和 random I/O performance
(2) savings in power budget 以及提高了system stability

allocation pool => how an SSD allocates flash blocks to service write request.
Four variables
(1) Static map:
A portion of each LBA constitutes a fixed mapping to a specific allocation pool.
(2) Dynamic map(需要查询key):
The non-static portion of a LBA is the lookup key for a mapping within a pool.
(3) Logical page size(1Kb ~ 256Kb => 4 page size ~ one flash block):
The size for the referent of a mapping entry might be as large as a flash block (256KB), or as small as a quarter-page (1KB) .
(4) Page span(类似cluster block或stripe):
A logical page might span related pages on different flash packages thus creating the potential for accessing sections of the page in parallel.

Three constraints
(1) Load Balancing:
Optimally, I/O operations should be evenly balanced between allocation pools
(2) Parallel access:
LBA => PA 应尽量不要和 LBAs' parallel access 相互干扰. 例如, 如果 LBA0..LBAn 总是被同时访问，they should not be stored on a component that requires each to be accessed in series.
(3) Block erasure:
Flash pages cannot be re-written without first being erased. **Only fixed-size blocks of contiguous pages can be erased.**

Trade off 举例：
(1) 如果一个large portion of LBA space 是statically mapped，这样的话就削弱了load balancing的效率
(2) 如果一个contiguous range of LBAs is mapped to the same physical die, performance for sequential access in large chunks will suffer.
(3) Page size大小的trade off

Small logical page size:
More work will be required to eliminate valid pages from erasure candidates. 就是增加了GC的工作量
Large logical page size(block size):
Erasure is simplified because the write unit and erase unit are the same, however all writes smaller than the logical page size result in a read-modify-write operation involving the portions of the logical page not being modified.
前半句好理解，当写和擦除的单位相同时，对Erase的工作要求降低了，后半句个人的理解就是，因为一个logical page size太大了(相当于一个block size), 那么smaller than logical page size的写操作会造成相对面积叫大的read-modify-write operation. 举个例子，一个ssd的logical page size是4Kb， 另一个是256Kb (block级别的大小)， 假设我们都要更新2kb的data，第一个ssd我们需要把一个4kb的page加载到cache中(read-modify-write)，期间只加载了2kb不需要update的data，而第二块ssd需要加载254Kb不需要update的data.

References:

18746 slides:
http://www.ece.cmu.edu/~ganger/746.spring15/lectures/Lecture5-flash.pdf

SLC, eMLC, MLC, TLC:
http://www.speedguide.net/faq/slc-mlc-or-tlc-nand-for-solid-state-drives-406

Coding for SSDs:
http://codecapsule.com/2014/02/12/coding-for-ssds-part-1-introduction-and-table-of-contents/

Garbage Collection and TRIM in SSDs Explained:
http://www.thessdreview.com/daily-news/latest-buzz/garbage-collection-and-trim-in-ssds-explained-an-ssd-primer/

https://www.usenix.org/legacy/events/usenix08/tech/full_papers/agrawal/agrawal.pdf