第25讲 存储设备原理

数据是如何在计算机系统中持久保存的?

本讲内容:

- 存储系统原理
- 持久数据的存储:磁、光、电

1. 存储系统原理

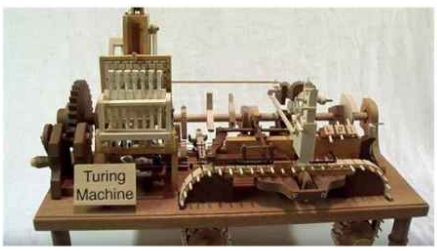
为什么会有内存和外存?

我们的状态机 (M, R)

- Memory 内存
- Registers 寄存器 (原则上可以不需要)

存储“当前状态”的需求

- 可以寻址
  - 根据编号读写数据
- 访问速度尽可能快
  - 越快就意味着越难 persist
  - 干存允许状态在掉电后丢失
    - 机械玩具就没有这个烦恼



开始持久化之旅

> Persistence: "*A firm or obstinate continuance in a course of action in spite of difficulty or opposition.*"
>
> 除了"当前状态"，我们希望更大、更多的数据能"留下来"(并且被操作系统有效地管理起来)

持久化的第一课：持久存储介质

- 构成一切文件的基础
  - 逻辑上是一个 bit/byte array
  - 根据局部性原理，允许我们按"大块"读写
- 评价方法：价格、容量、速度、可靠性
  - 再次见证人类文明的高光时刻！

2. 持久化存储：磁

持久化可能没有想象的那么困难——我们只需要一个"能反复改写的状态"

- 当然，要可寻址+用电路改写

磁带 (Magnetic Tape, 1928)

1D 存储设备

- 把 Bits “卷起来”
  - 纸带上均匀粘上铁磁性颗粒
- 只需要一个机械部件 (转动) 定位
  - 读取：放大感应电流
  - 写入：电磁铁磁化磁针

磁带：作为存储设备的分析

分析

- 价格
  - 非常低 - 都是廉价的材料
- 容量
  - 非常高
- 读写速度
  - 顺序读取：勉强 - 需要等待定位
  - 随机读取：几乎完全不行
- 可靠性
  - 存在机械部件、需要保存的环境苛刻

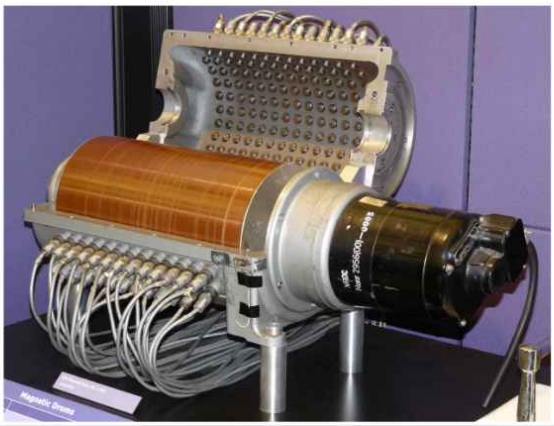
今天的应用场景

- 冷数据的存档和备份

磁鼓 (Magnetic Drum, 1932)

1D → 1.5D (1D x n)

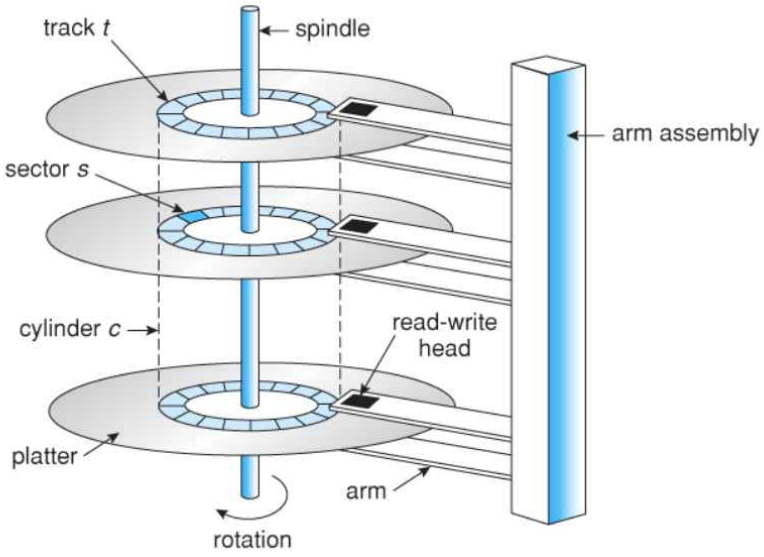
- 用旋转的二维平面存储数据
  - 无法内卷，容量变小
- 读写延迟不会超过旋转周期
  - 随机读写速度大幅提升



磁盘 (Hard Disk, 1956)

1D → 2.5D (2D x n)

- 在二维平面上放置许多磁带 (内卷)



磁盘：作为存储设备的分析

分析

- 价格
  - 低 - 密度越高，成本越低
- 容量
  - 高 (2.5D) - 平面上可以有数万个磁道
- 读写速度
  - 顺序读取：较高
  - 随机读取：勉强
- 可靠性
  - 存在机械部件，磁头划伤盘片导致数据损坏

今天的应用场景

- 计算机系统的主力数据存储 (~~海量数据，便宜才是王道~~)

磁盘：性能调优

为了读/写一个扇区

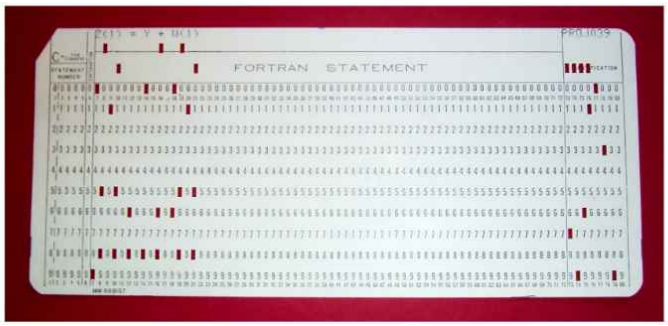
1. 读写头需要到对应的磁道
   - 7200rpm → 120rps → “寻道” 时间 8.3ms
2. 转轴将盘片旋转到读写头的位置
   - 读写头移动时间通常也需要几个 ms

通过缓存/调度等缓解

- 例如著名的 “电梯” 调度算法
- 现代 HDD 都有很好的 firmware 管理磁盘 I/O 调度
  - `/sys/block/[dev]/queue`
  - `[mq-deadline] none` (读优先；但写也不至于饿死)

3. 持久化存储：坑

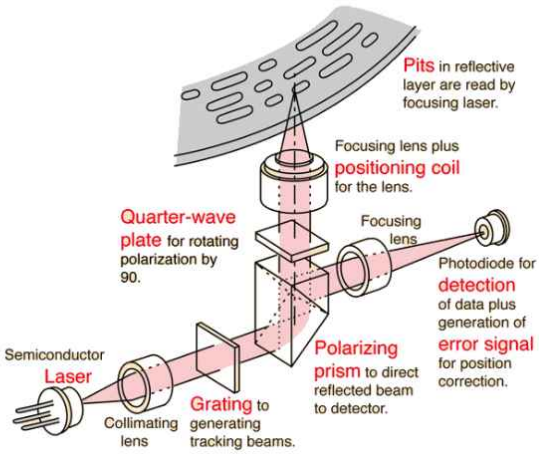
坑：天然容易“阅读”的数据存储



Compact Disk (CD, 1980)

在反射平面 (1) 上挖上粗糙的坑 (0)

- 激光扫过表面，就能读出坑的信息来
  - 飞利浦 (碟片) 和索尼 (数字音频) 发明
  - ~700 MiB，在当时是非常巨大的容量



光盘：作为存储设备的分析

分析

- 价格
  - 很低 (而且很容易通过“压盘”复制)
- 容量
  - 高
- 读写速度
  - 顺序读取速度高；随机读取勉强
  - 写入速度低 (挖坑容易填坑难)
- 可靠性
  - 高

今天的应用场景

- 作为数字收藏

4. 持久化存储：电

不管是磁盘还是光盘，它都是把某一个介质以内卷的方式卷在一个平面上。而但凡是这样的结构的设计，那个最终的那个旋转的机械部件是逃不掉的。而机械部件逃不掉就意味着它至少有毫秒级的延迟，你不可能有一个每秒1000万转的电机。第一做不出来，第二很容易造成损坏。

如果你想要和我们的 CPU 和内存能够匹敌的速度的话，电就是唯一的解决方案。

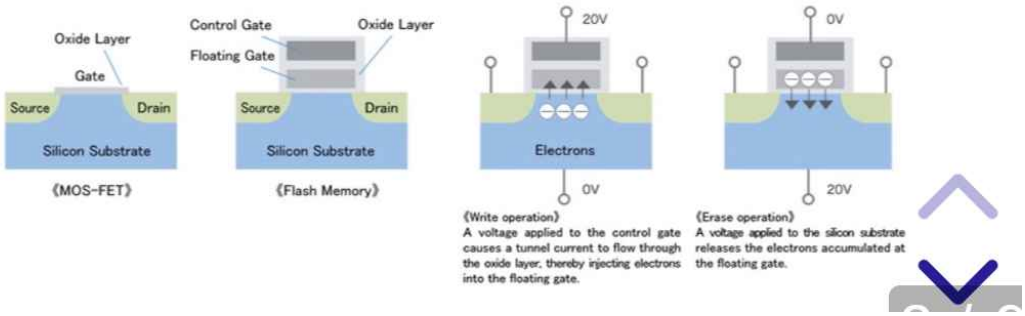
Solid State Drive (1991)

之前的持久存储介质都有致命的缺陷

- 磁：机械部件导致 ms 级延迟
- 坑 (光): 一旦挖坑，填坑很困难 (CD 是只读的)

密度和速度：最后还得靠电 (电路) 解决问题

- Flash Memory “闪存”
  - Floating gate 的充电/放电实现 1-bit 信息的存储



我们的磁盘是一个 2.5D 的存储。在一个平面，上面内卷了很多个磁带，然后又有好多个平面。

而我们的 Flash memory，你基本上 3D 或者也可以说是2.5D。在 2.5D 的空间里面， XYZ 的坐标你可以放很多很多个bit。而这个 XYZ 它的精度或者说它的这个大小会比机械盘要小得多，就是因为它小，所以它的容量可以做得非常高，而且它的读写速度也很快，它可以直接通过电路来读写。

而且 SSD 它有一个不讲道理的特性。因为它是通过电路读写的，电路是并行的，所以它的容量越大，速度越快。你想一想，我有两个电路，那我就可以同时从两个电路里面读数据，对吧？所以你们会发现比如说你买一台电脑，它有这个 256G、 512G 和 1T 的三个版本，然后你会看评测的话，那个 1T 的版本，它的磁盘的读写速度会比 256G 更快。

Flash Memory: 几乎全是优点

分析

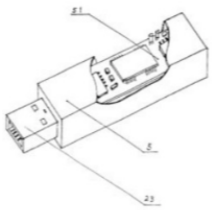
- 价格
  - 低 (大规模集成电路，便宜)
- 容量
  - 高 (3D 空间里每个 (x, y, z) 都是一个 bit)
- 读写速度
  - 高 (直接通过电路读写)
    - 不讲道理的特性：容量越大，速度越快 (电路级并行)
    - 快到淘汰了旧的 SATA 接口标准 (NVMe)
- 可靠性
  - 高 (没有机械部件，随便摔)

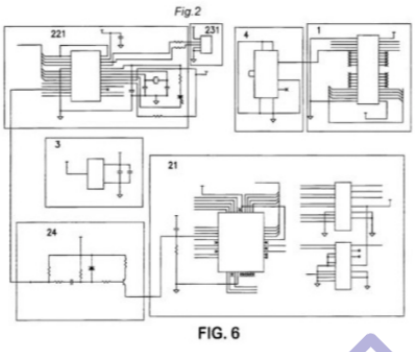
但有一个意想不到的缺点 (大家知道是什么吗？)

USB Flash Disk (1999)

优盘容量大、速度快、相当便宜

- 很快就取代了软盘，成为了人手 n 个的存储介质
  - Compact Flash (CF, 1994)
  - USB Flash Disk (1999, “朗科”)





朗科公司“用于数据处理系统的快闪电子式外存储方法及其装置”专利图节选(2004年) 美国专利号 6,829,672

放电 (erase) 做不到 100% 放干净

- 放电数千/数万次以后，就好像是“充电”状态了
- Dead cell; “wear out”
  - 必须解决这个问题 SSD 才能实用

所以只有解决了这个问题，SSD才能真正投入大规模使用。

典型的SSD上就放置了一个完整的计算机系统！

- 它有内存做缓存
- 它有一个cpu
- ……

也就是通过软件定义的磁盘，我们解决了SSD的那个严重的问题。

优盘和 SSD 的区别

优盘, SD 卡, SSD 都是 NAND Flash

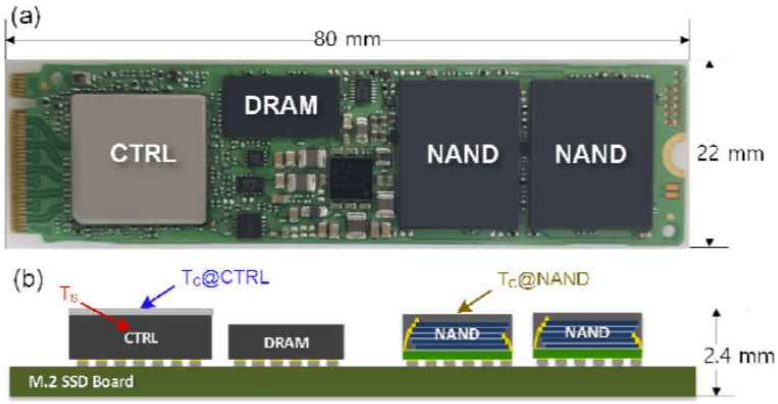
- 但软件/硬件系统的复杂程度不同，效率/寿命也不同
  - 典型的 SSD
    - CPU, on-chip RAM, 缓存, store buffer, 操作系统 ...
    - 寿命: ~1 PiB 数据写入 (~1,000 年寿命)
  - SD 卡
    - SDHC 标准未规定
      - 黑心商家一定会偷工减料
    - 但良心厂家依然有 ARM 芯片

一定不要用便宜的优盘保存重要数据

- × 宝 9.9 包邮的优盘，芯片一毛钱都能省.....

优盘和 SSD 的区别 (cont'd)

软件定义磁盘：每一个 SSD 里都藏了一个完整的计算机系统



- FTL: Flash Translation Layer
  - “Wear Leveling”: 软件管理那些可能出问题的 blocks
  - 像是 managed runtime (with garbage collection)

它有一个技术叫做wear leveling：你的闪存用着用着就坏掉，那我会根据你写入的负载，把你的写入均匀地分到闪存盘上的每一个闪存块上面。然后这部分闪存管理的核心和我们内存管理其实是一样的，它就是一个 VR 眼镜（本质上都是管理一个巨大的字节数组不是吗）。我们有一个巨大的字节数组，然后我们为了效率，不是以字节为单位管理，而是block。block是擦除的最小单位，一个block里面又有很多page，page是读取的最小单位。

Wear Leveling

再一次，VR 眼镜

- Logical block address (LBA) → Physical block address (PBA)

道理简单，实现困难

- SSD 的 Page/Block 两层结构
  - Page (读取的最小单位, e.g., 4KB)
  - Block (写入的最小单位, e.g., 4MB)
  - Read/write amplification (读/写不必要多的内容)
- Copy on write
- “VR 眼镜” 本身也需要更新
  - Coding for SSDs

那比如我们现在有很多block了。那么对于操作系统来说，它可以发出指令，往第三个块上执行1000次写。把第三块写一次，等你写完再来，再写一次，再写一次。在操作系统看起来它往编号为 3 的块里面写了 1000 次，但是实际上对于这个 SSD 来说，它有一个和虚拟内存一样的翻译层，它有一个数据结构f（理解为某种函数），你可以理解成是一个 VR 眼镜，然后戴上这个 VR 眼镜以后，我上一次写的编号为 3 的这一块可能映射到物理块，是第 100 块，下一次写的时候我就把它映射到第 101 块。所以我们的，SSD 里面的主控，它里面有缓存，会把你所有对磁盘的写均匀的分布到这个磁盘的各个位置上。

才我讲到 SSD 里面读的最小的单元是page，一个page可能比如说 4 kb，然后一个写入的最小单元其实是 4 mb，所以如果你要往里面写，哪怕是一个字节的数据的话，你都需要把这个 4 mb 给读出来，然后 4 mb 读出来以后，你要分配一个新的物理块，然后把这个 4 mb 改了，这就是写放大。同样也有读放大。我们的 SSD 要管理好所有的来自我们的系统的请求，然后把他们排好队，然后一个高效的处理。

你们可以想象的是现在的SSD，当你的写入到来的时候，他应该不会立即把它写进去——为了寿命。就和我们的缓存一样，我们的写操作到达了磁盘以后，尤其是SSD，包括今天的机械盘。其实如果你们有发现的话，如果你们写其哪怕写一个机械盘，甚至是直接写这个盘，不经过操作系统直接写这个盘，你都会发现在一开始的时候这个写入速度是很快的。然后这个写入速度很快，写 u 盘也是的，一开始写入速度很快，然后稍等一会以后，这个写入速度就会掉下去，就是因为不管是我们的 u 盘也好，还是我们的机械硬盘也好，还是闪存盘也好，它外面都有一个小的计算机系统，然后这个计算机系统会在你的请求到达磁盘以后立即给你一个反馈。好，我已经收到了，你可以再给我了。然后他在后面会悄悄的一点，再把这个请求慢慢给真正的落到磁盘上。

而我们的 SSD 是一个，尤其就它的这个主控是做的尤其复杂的，它的缓存做的尤其的大，因为它不仅要嗯说我只是一个 write buffer，它把你写进来的数据先放到内存里，这样我就可以返回了。

而且它还要管所有的逻辑块到物理块的映射，然后总的来说它是一个和我们刚才讲的 copy on write 写实复制的数据结构类似的一个思路，就是当你要写一块的时候，想写这一块的时候，你就会把这一块重新复制一块，对吧？也就是说旧的这一块还在系统里面，旧的这块里面没有消掉，但是我们把新的这一块，哎重新做一个映射，说我们的 3 号要映射到，比如说这个第 101 号物理块， 3 号就映射给它。然后这部分其实还挺复杂的。

然后当然 SSD 也带来很多有更有趣的问题，比如说刚才我们讲了写时复制，就意味着你原来的数据都还在——这是不是很危险啊？因为曾经有过一个影视明星因为修电脑发生过某个惨案。如果你们有任何的数据不希望被别人看到的话，你们当然会想我要把这个文件删掉。如果你们会想着如果这个数据实在是太不想被别人看到了，你会把这个磁盘给格式化。然后实际上把磁盘格式化，也不能真正的把文件删掉。

我们的马上布置的一个实验，你们会在一个实验里面去看到有一个文件系统，然后这个文件系统里面我会在里面放一些图片，然后我会把这个文件系统做一个格式化，然后格式化完了以后，嗯，你看起来如果你挂载那个虚拟磁盘镜像的话，你会看到里面一个文件也没有，但你还是可以把很多的图片给恢复出来。

然后在你理解了就知道这个 SSD 有这个 wear leveling，也就是说一块数据在 SSD 里，它绝对不会轻易再去写它一次，因为写一次就少一次，他要维护每一个，他有一张表维护了每一个block，它的寿大概寿命在哪里？所以你想这个数据在这里，他就绝对不会想去再写一次。所以你哪怕把它格式化了，你的 SSD 都这个数据都还在里面。也这也就意味着如果真的有一个人他可以 hack 这个主控的芯片的话，就他比如说他可以把你的这个SSD的主控下下来，换上他的主控，他就可以把你的 Flash 里面所有的数据都给找出来，那么你的数据就被他拿到了。

这个时候就引来了一些更有趣的问题，比如说你们的文件系统是加密的，一般比如说今天的手机上面的文件系统都是加密的。加密带来一个好处是你就算是你的盘物理性的落入了别人的手里，他也不可

能从里面恢复出数据，除非他有你的密钥。但同时也意味着如果你的盘真的发生了数据损坏的话，你也更难把它的数据给救出来。

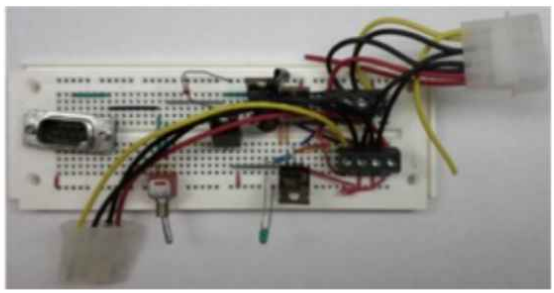
SSD 的可靠性：另一个故事

什么？硬件里的软件？

- 其实非常复杂：算法, cache; store buffer; ...

谁写出来的？那可得有 bug 啊！

- 让我们好好构造疯狂的 workloads，把它弄挂吧！
  - Understanding the robustness of SSDs under power fault (FAST'13)



5. Take away messages

无论是内存还是持久存储，最终胜出的仍然是电——它的密度和速度是其他介质难以比拟的。但同时我们也看到，NAND Flash 作为持久存储时有着巨大的缺陷——写入寿命。但我们也看到了工业界竟然敢于试制这样跨时代的产品，在十多年的争议中终究成为了今天存储的主角。如果更快的 non-volatile memory 到来，我们的计算机系统是否会发生翻天覆地的变化？