下面是Nand Flash介绍的第二节，主要是介绍下Nand Flash里面结构和一些特性。

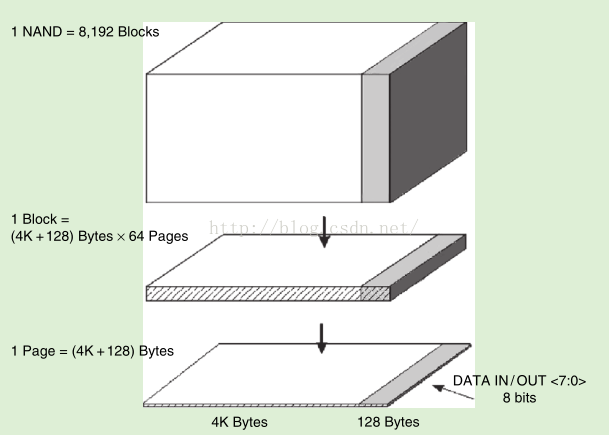
Nand Flash结构

图1

上图为Nand Flash的结构，图中这片Flash有8192个块(Block)，一个块包含有64个页(Page)，一个页有4K+128Bytes。但为什么一个页会有4K+128Bytes？那是因为需要一些冗余的空间（如上面128Bytes）用来存放错误纠正码(ECC)的和其它一些FTL管理信息。

现在的Flash页大小一般是8K、16K了，之前听说可能有32K的（不确认，没接触过）,一个块也可能有192、256、384、512个页。而且还有Multi Planes等结构，以后机会再说，本系列主要想讲FTL的，Nand Flash介绍只是作为铺垫。

Nand Flash特性

基本操作

Nand Flash基本操作有读、编程（写）、擦除。编程、读是以页为单位的，擦除以块为单位。

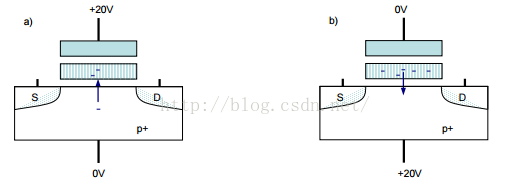


图2

擦除操作，是把电子“吸”出浮动门，擦除后每个Cell状态为1(这里把擦除后状态定义为1，编程后状态定义为0，因为擦除后读出就为1)。擦除操作最小单位为1个块，在图1中，即64个页。图2a为擦除操作。

编程(写)操作，是把电子“打入”Cell中，即从1变0。以页为单位进行操作。图2b就是编程操作。

读操作，检测浮动门中电势，判断是0 or 1。以页为单位。

因此在擦除后，Cell只能从状态1变为0，而不能从0变1，所以在写之前，块需要擦除。但如果恰好某几个Cell状态为“1101”，你需要编程“0001”，这样是可以的，可以节约擦除时间，但是操作起来异常复杂，而且Cell 0、1不均衡，其实也会不稳定，应该没有人会这样做。

P/E次数限制

P/E：Program/Erase。Nand Flash中每个块有一定的擦除、写次数限制，因为浮动门充放电会降低二氧化硅绝缘能力，到了一定阈值后，不能保证Cell中电荷数量。下图是不同种类和制程的Nand Flash P/E情况，其中有意思的是，制程的提高降低了P/E次数。MLC、TLC的P/E次数比SLC少。

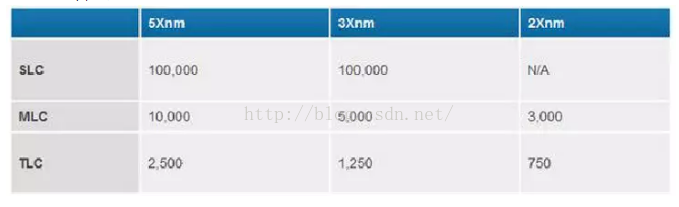


图3

Data Retenion

Data Retenion：即数据保存的能力，往Flash中写入数据，数据并不可以永久保存。随着时间的推移，Cell中电子会跑掉，严重的话影响Cell的电压阈值。如果在一个页中，异常的Cell超过了ECC纠错阈值，那么这个页中的数据就会损坏。Data retention和当前数据保存块的P/E次数（浮动门周围的二氧化硅绝缘能力降低）和温度有关。

Write/Read Disturb

Write Disturb(编程干扰)：指当对页（page）进行写入时，由于阈值电压接近的关系，相邻的位（bit）也被升高，从而造成相邻的位出错。闪存电荷非常不稳定，相邻存储电荷的悬浮门间会相互干扰，造成相邻悬浮门间的bit错误，MLC由于存在4组接近的电压，与SLC相比更容易受到干扰。

Read Disturb(读干扰)：所使用的闪存读取方式随着时间的推移会导致在同一区块中相近的记忆单元内容改变（变成写入动作）。这即是所谓的读取干扰。会导致读取干扰现象的读取次数门槛介于区块被抹除间，通常为100,000次。假如连续从一个记忆单元读取，此记忆单元将不会受损，而受损却是接下来被读取的周围记忆单元。为避免读取干扰问题，闪存控制器通常会计算从上次抹除动作后的区块读取动作总次数。当计数值超过所设置的目标值门槛时，受影响的区块会被复制到一个新的区块，然后将原区块抹除后释放到区块回收区中。原区块在抹除动作后就会像新的一样。若是闪存控制器没有即时介入时，读取干扰错误就会发生，如果错误太多而无法被ECC机制修复时就会伴随着可能的数据丢失。

（说人话！）Read disturb简单说就是，在某个块中重复读某个页的数据，可能会造成这个块内某些页数据受损(Cell电势改变)，严重的情况下，被干扰页出错会超过ECC纠错能力，造成数据损坏。

总结

由于Nand Flash读、写、擦除粒度不同，造成了写放大的情况（写放大：实际写操作的量超过请求的写数据量。比如说某款Flash页粒度为8K，现在要写1K数据，由于读、写以页为单位，所以需要先把Flash页其它7K读出来，加入新的1K数据，再重新写进去。所以简单的写1K数据，涉及到了读、写8K的操作。还有更恐怖的情况：垃圾回收，在以后再说）；由于有P/E的限制和Data Retention，所以我们需要增加磨损均衡和坏块管理的算法；由于W/R Disturb，所以我们需要增加对重复读操作进行特别处理。 而且在存储设备使用的过程中，又要考虑大数据读写、小数据读写、顺序读写、随机读写、持续读写、多线程读写、设备异常掉电等等等等…所以FTL还是很有挑战的。

因为Nand Flash有上述的BT属性和各种各样的操作需求，所以需要我们开发FTL去更加高效管理它，在主要的操作Case下发挥高性能的情况下，平衡每个块的磨损，设备可以使用更久。

————————————————

版权声明：本文为CSDN博主「忠s」的原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/ibanezjem/article/details/51742520