Garbage Collection

Garbage Collection(GC) 概述

Nand Flash 空间使用到一定程度之后进行脏数据空间回收，同时要将有效数据搬移到新的位置，将原来脏数据所占用的空间重新释放出来。

两种类型 GC：

BGC： Backend GC, FTL idle 时 trigger

FGC： 前台 GC

选源

1. Wear leaving

   主要是选择 EPC counter 小的 SPB

1. free block 不足

   主要是选择 valid counter 小的 SPB ，如果 valid counter 相差不大（SPB DU 个数千分之一）但是 SN 差距较大就选择 SN 比较小的 SPB.

1. 不稳定 block

   主要选择有 read/write err 或者 BER risk block

主要流程

1. Find Valid PAA

GC 目的是将有效数据搬移到新的位置，所以在搬移前须明确哪些数据是有效。

1.
   1. User lu

      通过读取 candidate SPB P2L 建立

1.
    1.
        1. Read P2L
        2. Read L2P
            1. 通过 P2L 查找相关 L2P
            2. 每当搜集到的 L2P 到达一定个数后才会去 load L2P table
        3. L2P check
            1. L2P PAA 若属于当前正在处理的 P2L 则此 PAA 列为有效
    2. Mid lu

通过读取 candidate SPB L2P 建立，由于 mid lu size 已经不大且没有 P2L， 所以直接 scan mid lu 的所有 L2P 来 check 有效数据

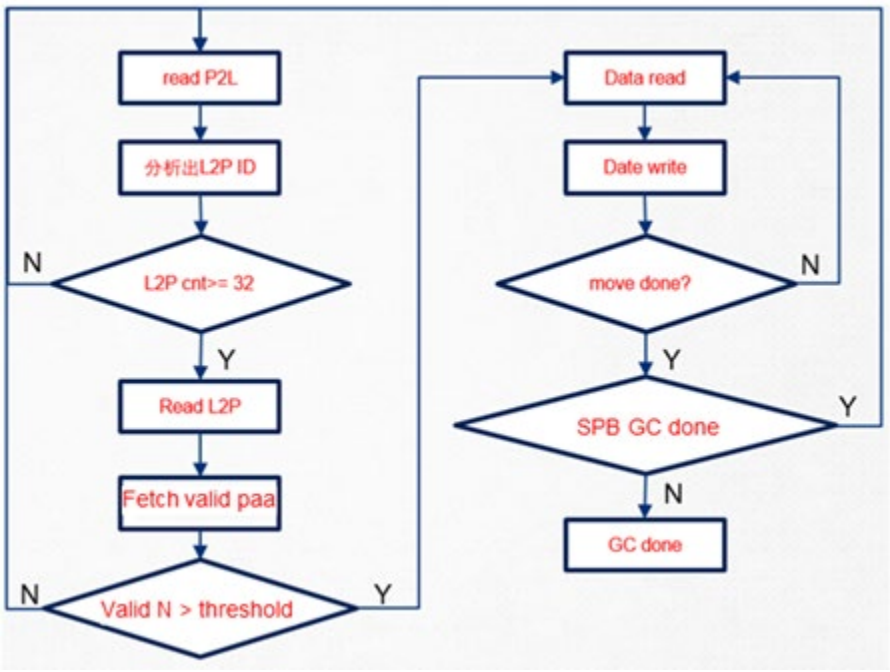
-
    - Sys lu

读取 lut table（in TCM）建立，sys lu 的 maaping size 很小（only 2K for 2T capacity），所以直接 sys lu lut table 可以直接决定有效数据

2. Data move

根据有效数据 向 BE 发 read->write 请求。每次读写 DU 个数需兼顾 resource 及 candidate SPB 的 WFU。

Fetch valid PAA data move



GC 过程中，user/mid/sys lu 的 gc 在 Date move 部分的实现 flow 相同，只有在 fetch valid paa 的过程 有差异。

GC write 更新 P2L 时为优化 host write performance，需要记录 GC destination SPB 所有 DU 的 source SPB，

当 merge GC SPB 的 P2L 时如果发现所记录的 source SPB 已经不是 L2P 中的 PDA 对应 SPB 即认为该 DU 在 GC 后又有 host 写入所以放弃该 du 的 merge，

优势是：host write merge 时不需要再去检查 GC spb 的 P2L。

Schedule

User/mid/sys 三个 lu 的 gc 如果全部都触发，则按照 sys > mid > user 的优先级顺序调度然后发给送给 folding，

不过鉴于 resource 原因同一时间只允许一个 GC 运行，mid/sys GC 可以抢占 user lu GC。

异常处理

1. GC read err（读 PDA err）
    1. Mask 坏块

2. 通过 P2L 或者 L2P 找到 err PDA 对应 LDA
3. Check 这个 LDA 是否已经/正在被上层写/trim（此检查可以删掉）
    1. 如果是， skip
    2. 如果否，insert err PDA of LDA（P2L 如何实现）
4. Data 处理
    1. User LU

Data 还是会继续写入 nand，但是要保证修改 mapping

1.
    1.
        1. Internal LU
            1. Buffer set 为 err pda，写入 nand
2. 读 P2L err

放弃 P2L， 改为读 LU 所有的 L2P 来 fetch valid paa

3. Abort/cancel/stop

三个 LU 的 GC req 可单独设置是否 abort/cancle/stop

GC 在运行时会随时关注是否有被 Abort/cancel/stop