Nor Flash与NAND Flash的详细对比

特性（Characteristics）

**存储密度**：

1. 1. **Nor Flash**: 较低的存储密度。
   2. **NAND Flash**: 较高的存储密度，适用于大容量数据存储。

**读取速度**：

1. 1. **Nor Flash**: 读取速度快，因为它允许随机访问。
   2. **NAND Flash**: 相对较慢，通常需要按块或页进行读取。

**写入和擦除速度**：

1. 1. **Nor Flash**: 写入和擦除速度相对较慢。
   2. **NAND Flash**: 写入和擦除速度快。

**耐用性**：

1. 1. **Nor Flash**: 通常具有较低的写入/擦除周期。
   2. **NAND Flash**: 更高的耐用性和更多的写入/擦除周期。

**电源需求**：

1. 1. **Nor Flash**: 通常需要更多的电源引脚和更高的电源电压。
   2. **NAND Flash**: 电源需求相对较低。

用法（Usage）

**Nor Flash**：

1. 1. 代码存储（如BIOS或固件）
   2. 实时系统
   3. 需要随机访问的应用

**NAND Flash**：

1. 1. 数据存储（如SD卡，USB驱动器）
   2. 大文件存储
   3. 流媒体应用

经典用例（Classic Use Cases）

**Nor Flash**：

1. 1. 嵌入式系统的引导加载器（BootLoader）
   2. 车载信息娱乐系统
   3. 实时控制系统

**NAND Flash**：

1. 1. 智能手机和平板电脑的内部存储
   2. 数码相机和摄像机
   3. 企业级和数据中心的SSD存储解决方案

其他维度

**价格**：

1. 1. **Nor Flash**: 由于其复杂性和低存储密度，通常更昂贵。
   2. **NAND Flash**: 由于高存储密度和简单的制造过程，通常更便宜。

**接口和协议**：

1. 1. **Nor Flash**: 通常使用并行接口。
   2. **NAND Flash**: 通常使用串行接口，如ONFI或Toggle Mode。

总结

Nor Flash和NAND Flash各有优缺点，选择哪种取决于应用需求。Nor Flash更适用于需要快速随机读取和代码存储的场景，而NAND Flash则更适用于需要高存储密度和快速写入/擦除的应用。

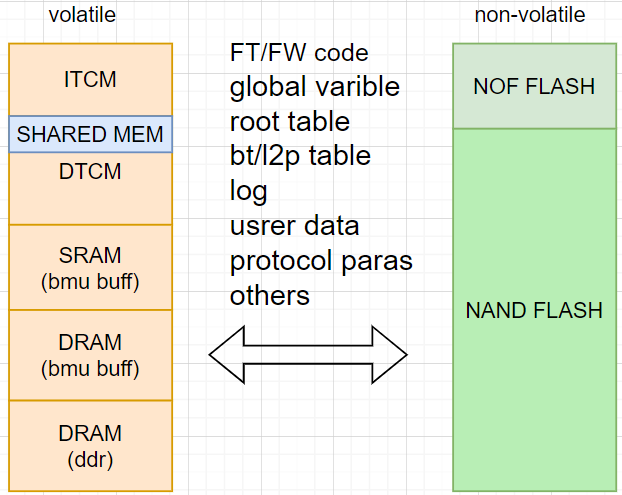
具体使用用例

1.1. **NAND&NOR空间管理-SPOR--daniel**

1.1.1. **功能描述**

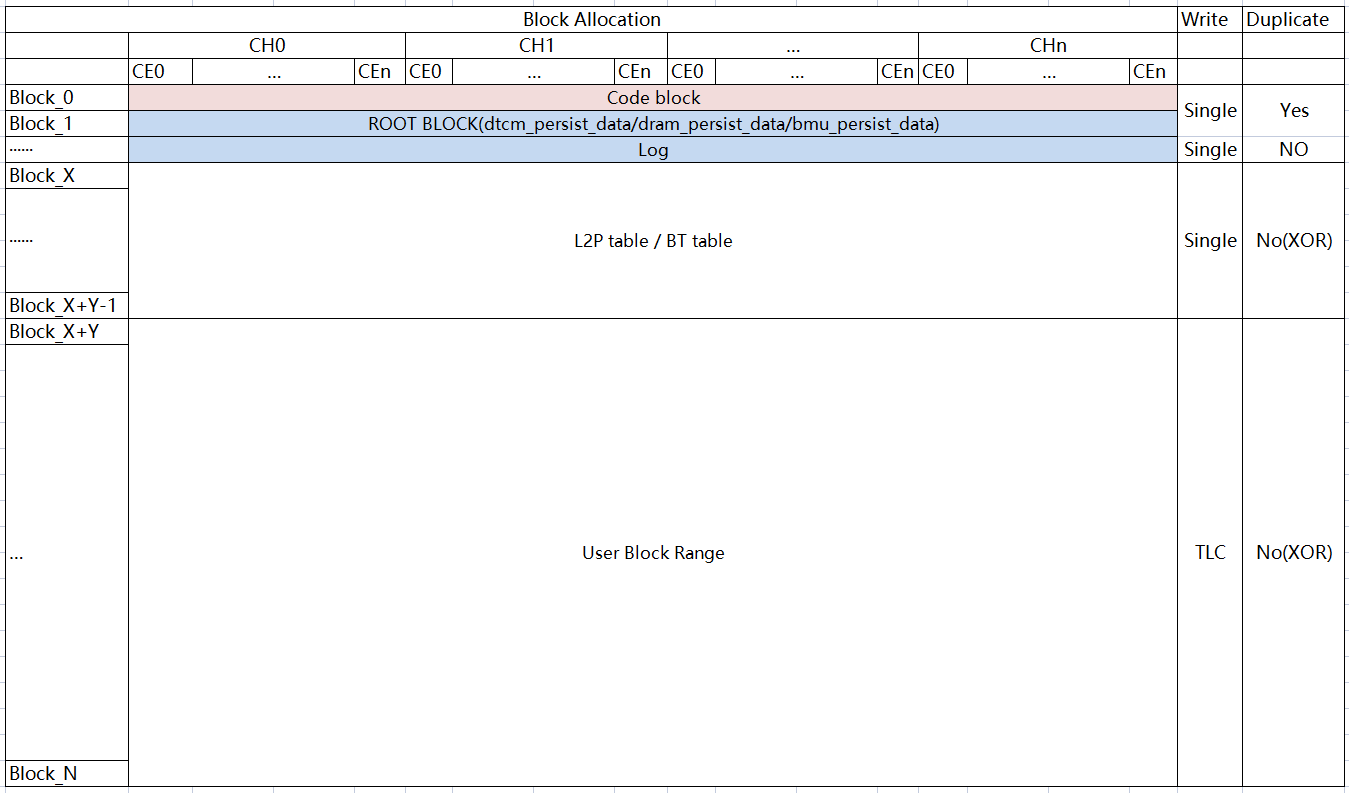
1.1.1.1. **存储空间分类**

存储数据的空间分为易失性和非易失性。易失性存储空间：ITCM/DTCM/SRAM/DRAM，非易失性存储空空间：NOR FLASH/NAND FLASH。如下图所示。



在运行过程中，用户写入的数据会以SRAM/DRAM作为BUFF，再存入NAND FLASH。其余在易失性存储空间中的数据，需要持久化的部分在运行过程中、或者掉电流程触发写入NAND FLASH或者NOR FLASH保存。由于各种持久化数据的更新频率、大小、写入时机等存在差异，所以对NAND和NOR的空间进行划分，以对持久化的数据进行分类保存。

1.1.1.2. **NAND上数据的分类**



NAND FLASH是SSD盘最大的持久化存储空间，按照数据的分类划分为以下几个区域(数据分类示意)：

1、Code Block区域：存储FW的代码；

2、Root Block区域：存储以下几类数据：

(1) dtcm\_persist\_data：驻留在DTCM中，在运行过程中会频繁读取、更新的数据，下电时需要保存的元数据，对应的NAND位置指针保存到NOR FLASH固定位置；

举例：supper block链表、L2P&BT的cur pa、BLOCK Table、bad block信息等。Bad block信息如果更新不频繁，可以考虑归类到running\_persist\_data，但是由于读取频繁，仍需要占用DTCM或者SRAM的空间。

(2) dram\_persist\_data：驻留在DRR中，在运行过程中会读取、更新的数据，但频率较低，更新后需要做持久化写入NAND；

举例：前端协议数据(feature/pmr/log page/FW Code升级缓存buff等)

(3) bmu\_persist\_data：如果dtcm存储空间不足，外溢申请bmu的buff来存放的持久化数据。

该区域的数据以multiblock为粒度进行分配，分配空间时预留与系统区super block行数相同的multiblock用于坏块预留、DIE失效处理。

上述3中类型的持久化数据，按照持久化的时机分为：running\_persist\_data(运行态做持久化操作)、por\_persist\_data(掉电做持久化操作)。

3、Log区域: 持久化的日志数据；

4、L2P/BT Table区域：存储L2P和BT表数据；

5、User Block区域：用户数据；

1.1.1.3. **NAND空间的分配**

目前规划需要支持NAND颗粒型号为Bics8(4plane/die)、YMTC\_X3\_9070(6plane/die)、YMTC\_X3\_XXXX(8plane/die)，系统区NAND空间的分配需要根据3种颗粒单die的plane数据、block大小采取差异化的分配策略，在满足需求的情况下，以尽可能的节省NAND空间。

系统区NAND空间的分配重点考虑的因素：

1、需要支持DIE失效，所以按照multiblock整数倍进行分配；

2、CODE、RUNNING、POR区域由于不做XOR，需要使用镜像备份；CODE和RUNNNING使用不同DIE上的BLOCK镜像备份，POR使用同一片空间横向切分为两片区域，按DIE错开镜像备份。

3、CODE、RUNNING、POR在备份数据时，一定要按DIE错开备份，以支持DIE失效场景下，数据能够搬移到预留的multiblock；

4、LOG区域不做镜像备份，如果出现坏块，自行隔离，不再分配新的预留block；

5、按照每行系统区预留一个multiblock的规格支持DIE失效处理，需要注意预留的multiblock最好不与有效数据在一个DIE上；

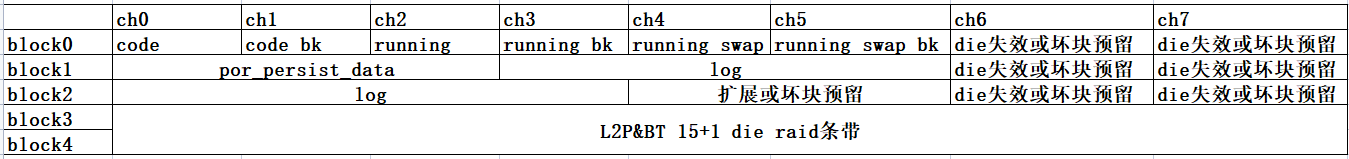
6、单行BLOCK系统区没有用完的multiblock可以给IO区域做坏块预留；

按照上述分析，系统区分为如下区域：

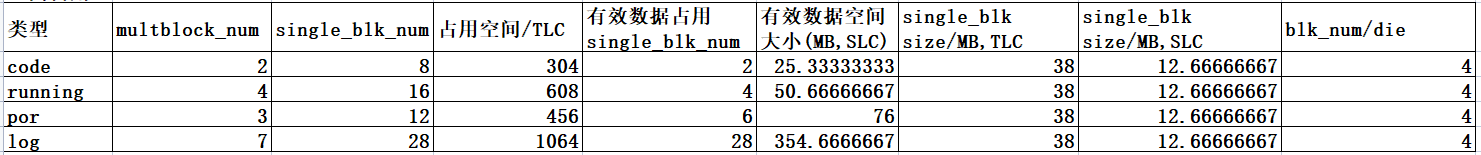
* CODE
* RUNNING
* POR
* LOG
* DIE失效预留或坏块预留
* L2P&BT

##### Bics8系统区分配

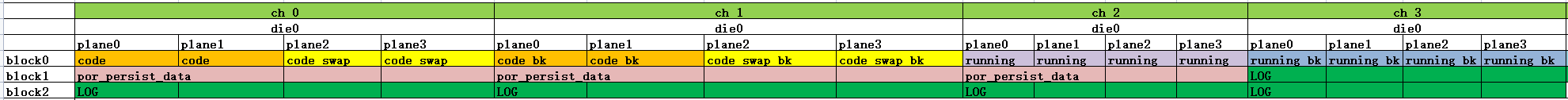
Bics8(4plane/die)整体空间分配如下：

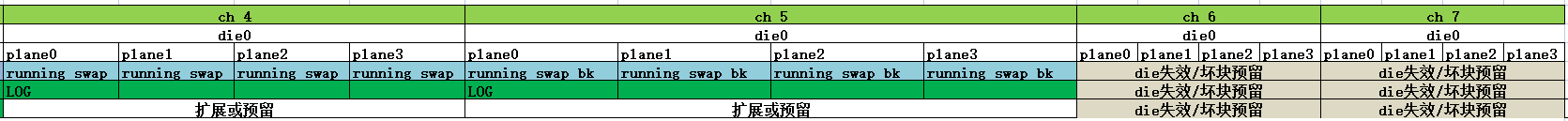


空间占用如下：



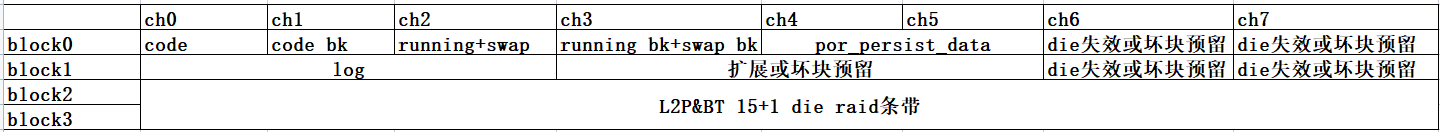
详细位置分布：



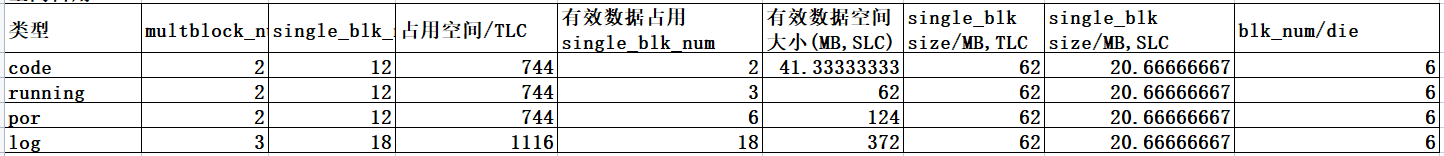


##### YMTC\_X3\_9070系统区分配

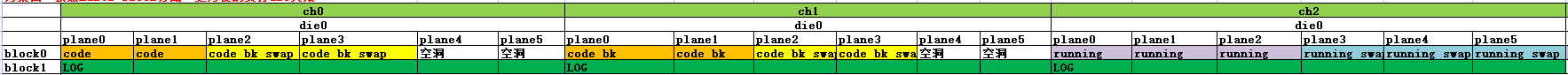
YMTC\_X3\_9070(6plane/die)整体空间分配如下：

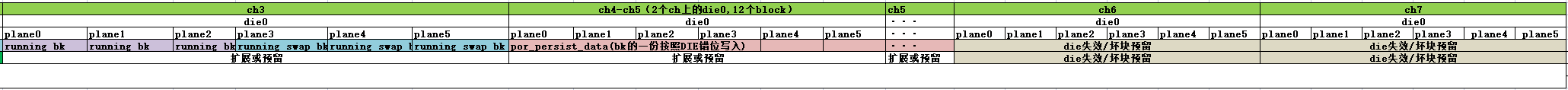


空间占用如下：



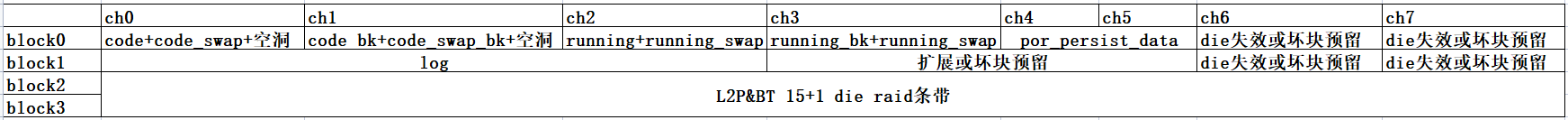
详细位置分布：



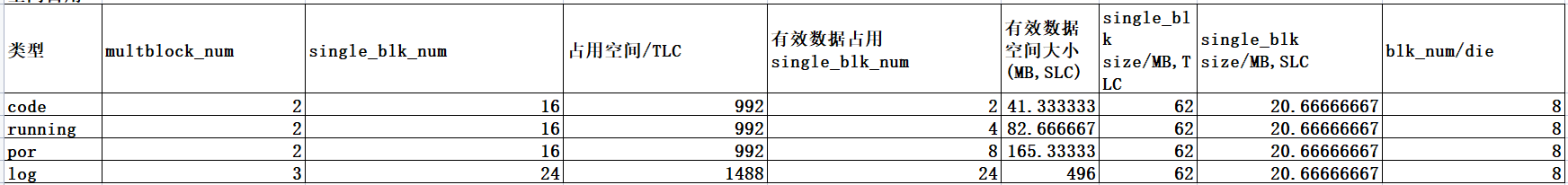


##### YMTC\_X3\_XXXX系统区分配

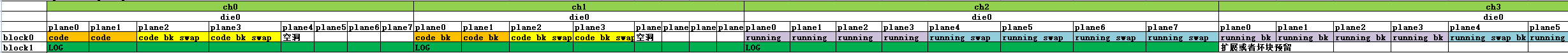
YMTC\_X3\_XXXX(8plane/die，厂商规划中)整体空间分配如下：

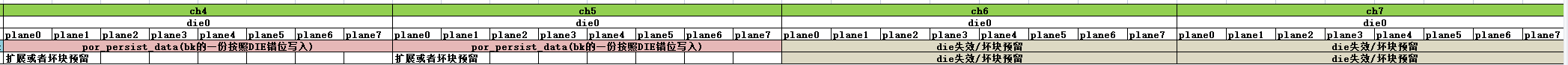


空间占用如下：



详细位置分布：





1.1.1.4. **NAND空间的划分**

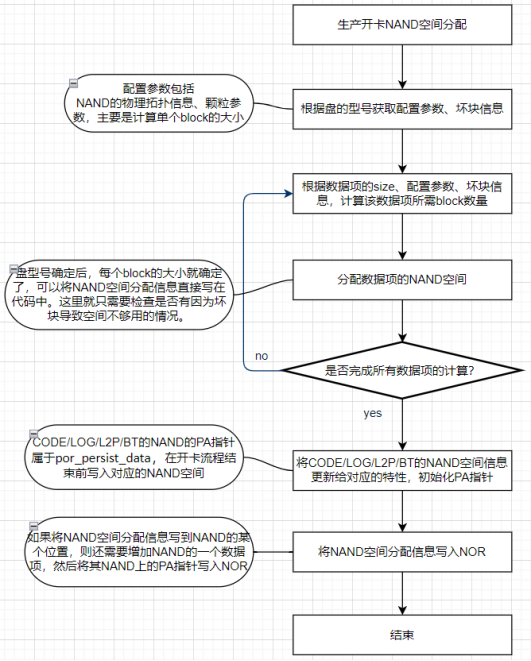
在生产开卡的一个环节，根据盘型号对应的配置参数(含各个数据项所需的空间大小)、出厂坏块&扫描坏块的信息，计算出各类数据所占用的NAND空间的位置信息，并写入NOR FLASH固定位置。

如果位置信息的数据较多，读写的时间不能满足下电上电要求，可以写入NAND FLASH(多备份)，再将NAND的位置指针写入NOR FLASH固定位置。

使用空间的特性决定数据项的size(作为盘的一个配置项)，再根据盘型号和size计算出空间分配表的其它变量值。

因为盘型号确定后，每个NAND的拓扑、颗粒规格就确定了，所以，也可以将NAND空间分配信息全部算好后，直接写在代码的空间分配表中。

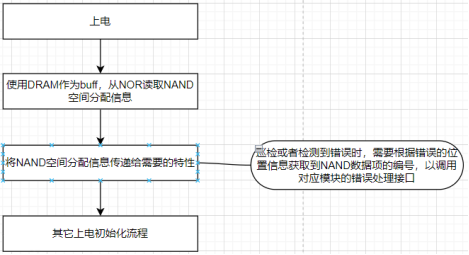
1、生产开卡NAND空间分配流程：



0514：图需要更新：原始坏块扫描-》老化测试-》新增坏块扫描-》空间分配。

2、上电恢复流程：

    在上电流程中，需要把开卡时形成并存储在NOF FLASH中的空间分配信息读出，保存到DRAM空间中（DRAM的global area）。在巡检、错误发生时，需要通过NAND的位置信息查询到对应的数据项，以通知使用该NAND空间的特性做响应的错误处理。



1.1.1.5. **坏块所在SUPER BLOCK空间利用**

    NAND FLASH上会出现某些BOCK行的坏块较多，无法组成一个USER SUPER BLOCK。可以将这类BOCK行剩余的好的BLOCK分配给系统区使用，以提高NAND空间的利用率。

1、分配给Root Block/Log区域使用(共用一个BLOCK行)；

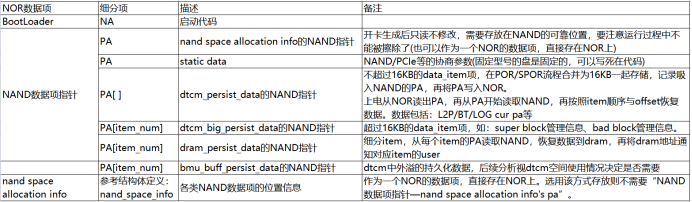
2、分配给L2P/BT作为错误处理时数据搬移的预留块；

   这样虽然提高了空间利用率，但是也会导致每块盘存放系统区数据的位置随坏块的分布变化，增加后期维护难度。

   详细流程待补充。

20230518：可以采用逆向思维，将系统区用不完的block作为坏块预留使用。

1.1.1.6. **NOR空间划分和数据分类**



待进一步细化补充。

1、增加上电恢复流程又掉电场景，执行的stage信息写入NOR；

1.1.2. **可靠性设计**

本特性涉及的可靠性设计、异常处理，注意与系统可靠性功能做区分。

1、除开User Data区域之外，系统区采用SLC以提高读写、数据的可靠性；

2、系统区除Log区域外，Code Block、Root Block采用双备份保证数据的可靠性，L2P/BT区域双备份占用空间较大，采用RAID5保证数据可靠性；

3、系统区需要纳入巡检范围，及时搬移错误bit数超过阈值的数据(需要产生和使用数据的模块/特性进行处理)；

1.1.2.1. **系统区数据镜像备份**

系统区code/running\_persist\_data/por\_persist\_data按照镜像备份的multiblock数量进行分配，数据保存时都做双备份。出现坏块隔离时，申请一个全局预留的multiblock，继续做双备份写入。

分配出去之后的multiblock由申请的模块继续维护和管理，不在回收。

1.1.2.2. **巡检错误处理**

   巡检的错误包括以下场景：

1、read的错误bit数超过阈值，巡检模块向NAND空间管理模块查询使用该block的模块ID，然后调用对应模块的错误处理接口进行处理。

2、read错误，巡检模块向NAND空间管理模块查询使用该block的模块ID，对应模块的错误处理接口中，将原错误multiblock做隔离处理，再向NAND空间管理模块申请预留的multiblock写入备份数据。

1.1.3. **接口设计**

模块间的接口设计（包括与芯片的接口）。

1、NAND数据项定义：

typedef tag\_nand\_data\_item {

    NAND\_DATA\_CODE, // upgrade flush

    NAND\_DATA\_RUNNING, // running flush,bad block, dram->nand, 需要再细分数据item

    NAND\_DATA\_POR, // por&spor flush, dtcm&bmu buff->nand, 需要再细分数据item

    NAND\_DATA\_LOG, // running flush, sram->dram->nand

    MULTI\_BLCK\_RSV, // DIE BAD

#    NAND\_DATA\_L2P, // running flush, dram->nand, 需要按15+1组条带，与前面4个分配算法不同，单独处理

#    NAND\_DATA\_BT, // running flush, sram->dram->nand, 需要按15+1组条带，与前面4个分配算法不同，单独处理

    NAND\_DATA\_MAX

} nand\_data\_item;

typedef enum tag\_nand\_type {

    KAIXIA\_BICS8, // 38MB/block,4plane/die

    YMTC\_X3\_9070, // 62MB/block,6plane/die

    YMTC\_X3\_XXXX, // 62MB/block,8plane/die

    NAND\_TYPE\_MAX

} nand\_type;

2、NAND数据项空间分配表：

typedef struct tag\_sys\_area\_multi\_blk\_space {

    uint8\_t ch\_id;

    uint8\_t die\_id;

    uint8\_t blk\_id;

} sys\_area\_multi\_blk\_space;

typedef struct tag\_blk\_allocation\_info {

    uint32\_t size; // MB,通过multi\_blk\_num算出来的,没有特性使用可以删除

    uint32\_t multi\_blk\_num; // 使用空间的特性决定需要几个multi\_blk

    sys\_area\_multi\_blk\_space multi\_blk[MAX]; // MAX的值取决于multi\_blk\_num的最大值

} blk\_allocation\_info;

// 先确定multi\_blk\_num的值，size根据multi\_blk\_num大小算出来，multi\_blk\_pa需要根据分配算法来确定

blk\_allocation\_info g\_blk\_allocation\_info[NAND\_TYPE\_MAX][NAND\_DATA\_MAX] = {

   {{38\*2\*4, 2, pa[]}, {38\*4\*4, 4, pa[]}, {38\*3\*4, 3, pa[]}, {38\*7\*4 = 354 MB, 7, pa[]}},//size算的是TLC

   {{62\*2\*6, 2, pa[]}, {62\*2\*6, 2, pa[]}, {62\*2\*6, 2, pa[]}, {62\*3\*6 = 372 MB, 3, pa[]}},//size算的是TLC

};

3、提供接口通过P4K地址查询nand\_data\_item的ID，错误检查模块检查到写NAND错误场景，通知对应的nand\_data\_item模块进行错误处理；

nand\_data\_item get\_nand\_data\_item(PHY\_ADR cur\_pa);

1.1.4. **性能设计**

1、running\_persist\_data属于管理类数据，每次只写入一个data\_item，控制data\_item的大小，即可控制对性能的影响；

2、por\_persist\_data在掉电流程写NAND，需要满足电容的时间要求。后续根据实际的写入数据量，调整por\_persist\_data占用的multiblock数量，提高写NAND的并发，以提供写入性能；

3、CODE、LOG、BT/L2P写NAND在对应的特性中考虑性能。

1.1.5. **可维护性设计**

特性的调试、定位手段等设计。

1、提供接口读出存放在NAND/NOR FLASH的空间分配表；

1.1.6. **可测试性设计**

包括白盒测试、黑盒测试、自动化测试的方法、命令等。

1、白盒测试

（1）UT用例在详细方案中设计；

（2）设计IT用例，以每种型号盘的配置参数为输入，验证空间分配表是否正确；

（3）设计IT用例，验证不同坏块分布的场景下，验证空间分配表是否正确；

2、黑盒测试

属于FT测试的一部分，在FT测试中设计。FPGA开发阶段，提供接口模拟开卡流程划分NAND空间的流程。

1.1.7. **需求分解**

|  |  |  |  |
| --- | --- | --- | --- |
| **类型** | **需求标题** | **需求描述** | **特性/模块** |
| **SR** | **NAND空间管理** | **NAND空间按照用途划分、查询区域信息。** | **SPACE** |
| AR | 支持NAND空间的划分 | 按照使用NAND空间的用途，根据盘/颗粒型号、坏块信息划分NAND空间，记录NAND空间划分表。 |  |
| AR | 支持NAND空间划分表写入NOR FLASH | 将NAND空间划分表产生后不再修改，写入NOR FLASH固定位置，在上电流程使用。 |  |
| AR | 支持查询NAND空间划分表 | 支持查询完整的NAND空间划分表、单个使用NAND空间的数据的分配信息。 |  |
| AR | 支持通过P4K指针查询对应的数据项 | 在巡检、错误发生时，需要通过NAND的位置信息查询到对应的数据项，以通知使用该NAND空间的特性做响应的错误处理。 |  |
| **SR** | **NOR空间管理** | **NOR空间按照用途划分、查询区域信息。** | **SPACE** |
| AR | 支持NOR空间的划分 | NOR空间主要存放bootloader、开卡静态信息、NAND数据项的PA指针、其它一些少量重要的状态信息。 |  |
| AR | 支持查询NOR空间划分表 | 支持查询完整的NOR空间划分表。 |  |
| **SR** | **支持模拟开卡流程划分持久化空间** | **FPGA开发阶段，提供接口模拟开卡流程划分NAND、NOR空间的流程。** | **DFT** |
| AR | 支持模拟开卡流程划分NAND空间 | 支持模拟开卡流程划分NAND空间 |  |
| AR | 支持模拟开卡流程划分NOR空间 | 支持模拟开卡流程划分NOR空间 |  |