3Dsim 用户手册

3Dsim 简介

固态盘模拟器可以给固态盘的相关研究人员提供一个前期功能验证的手段，通过固态盘的模拟器模拟，可以验证所设计软件算法的有效性，硬件结构的可靠性，从而设计出满足预定目标的固态盘产品。目前开源的固态盘仿真平台主要有以下几个：微软开发的 disksim 的固态盘模块，宾夕法尼亚州立大学开发的固态盘模拟器 flashsim，汉阳大学在虚拟机平台下高并行性固态盘模拟的 VSSIM 和华中科技大学开发的多通道并行性固态盘模拟器 SSDsim。然而目前常见的开源固态盘模拟器大多数针对的是基于平面结构闪存介质的固态盘进行模拟，基于三维堆叠的闪存的固态盘逐渐成为闪存市场的发展趋势，开发并开源一种针对三维堆叠的闪存的固态盘模拟器对固态盘的相关研究人员很有必要。

本文介绍的是一种针对三维堆叠闪存(3D flash)的固态盘模拟器，称为 3Dsim。该模拟器具有自主配置 3D flash 的组织结构、提供多并行通道、适配目前常见的 3D flash 高级命令，以及模块化 FTL 算法等功能，可用于研究固态盘硬件设计结构和软件算法的有效性。

3Dsim 设计思路

固态盘是一个既有固件软件，又有系统硬件的嵌入式系统，固态盘的模拟器需要对固态盘进行模拟，就需要对固态盘上的软件算法，硬件行为进行模拟。模拟器不涉及实际数据的读取和写入，所有的数据都存在于内存中。3Dsim 作为一个模拟软件，其上的软件算法与真实系统的系统软件基本类似，因此，模拟的关键在于对固态盘系统硬件的模拟。而固态盘的系统硬件是由多个通道、多个芯片组成，3D 闪存作为固态盘硬件的核心，是 3Dsim 模拟的关键点。它的主要所做工作包括以下几点：

1、 按照时序模拟 3D flash 的各种高级命令行为；
2、 维护状态：data buffer、cache、FTL、flash 等；
3、 统计数据：统计数据：缓存命中率，读、写、擦除次数，高级命令使用次数，读写平均响应延时等。

3Dsim 的仿真功能包含两个：时间模拟、高级命令模拟 ，分别为：

1、 时间模拟：按照 3D flash 芯片实际参数手册，由读写擦操作时序以及操作次数的统计，计算固态盘各种操作的延时；
2、 高级命令模拟：按照 3D flash 芯片实际参数手册，匹配各种高级命令的时序以及使用条件，分析固态盘中各种高级命令的使用场景。

时间模拟

在高性能固态盘中，存在三类主要的部件，即固态盘中的嵌入式处理器，内存颗粒，和闪存颗粒。处理器的时钟通常大于百兆，单个时钟的处理时间小于 10 纳秒甚至更短；内存的读写操作时间是纳秒级；与此形成对比的是，闪存颗粒的读写时间是微秒级。因此在 3Dsim 中，只考虑对闪存的读写擦除操作的时间开销，忽略处理器、内存的时间开销。对于一个独立的读、写、擦除操作，可以根据以下的方式获得准确的时间开销。表 1 中介绍了三个公式

中出现的变量的含义，所有的参数具体值都来自 Hynix 256Gb 3D TLC 数据手册。

单页读操作延时：$7\times t_{WC}+t_R+P\times t_{RC}$

单页写操作延时：$7\times t_{WC}+P\times t_{WC}+t_{PROG}$

单次擦除操作延时：$5\times t_{WC}+t_{ERASE}$

三个页一次性写操作延时：$7\times t_{WC}+P\times t_{WC}+t_{PROGO}$

表 1　时间开销公式中的主要变量

简称	含义	时间
t_R	数据从目标物理页中读到分组的寄存器所消耗的时间	90 微妙
t_{PROG}	数据从分组的寄存器写到目标物理页所消耗的时间	1100 微妙
t_{ERASE}	目标物理块的擦除时间	10 毫秒
t_{WC}	通过数据总线上向寄存器传输一个字节的数据所消耗的时间	5 纳秒
t_{RC}	通过数据总线从寄存器向外传输一个字节的数据所消耗的时间	5 纳秒
t_{PROGO}	三个页数据从分组的寄存器写一次性到目标物理页所消耗的时间	1100 微妙
P	传输的数据量	NA

图 1 是一个 3D flash 普通单页读操作过程。从图 1 中可以看到，一个读操作分成三个阶段：输入命令地址阶段、访问介质阶段、数据传输阶段。上述公式中的三个部分分别对应这三个阶段。第一个部分是输入命令地址阶段耗费的时间。一个读请求，首先花费一个t_{WC}时间通过总线传输一个起始命令（00h），之后花费 5 个t_{WC}时间通过总线传输读操作的地址，紧接着是花费一个t_{WC}时间传输结束命令（30h），因此公式中第一个部分是 7 个t_{WC}；公式的第二个部分是闪存将数据从介质中读到分组的数据寄存器上，这个过程图 1 中忙（busy）状态所指。在读操作中这个时间是t_R；公式的第三个部分是数据的传输部分，在这个部分，上一阶段读出的数据将从数据寄存器通过总线传输到通道控制器的缓存中，这个部分的传输时间与传输的数据量有关，因此在公式的的三个部分需要考虑传输的数据量 P 大小。

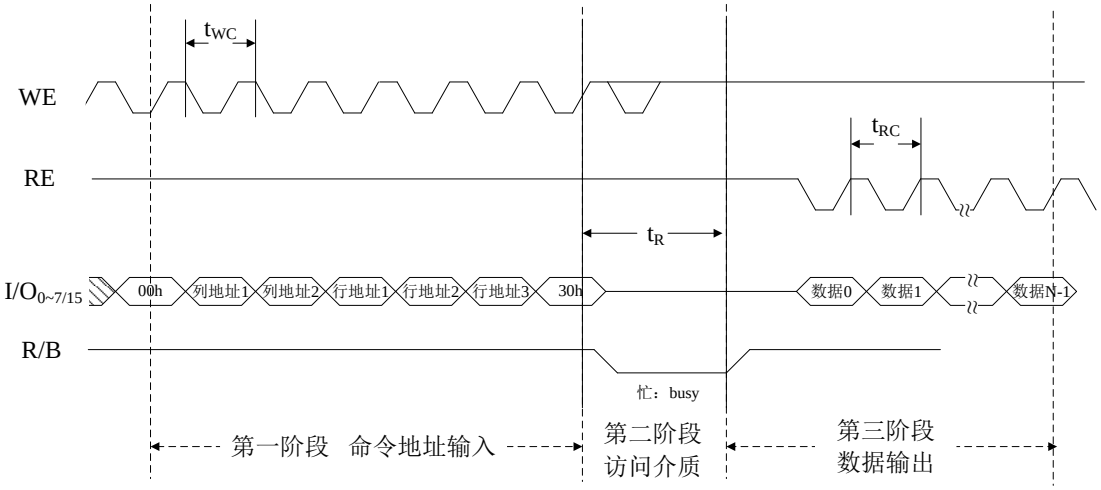


图 1　闪存读操作流程

图 2 是 3D flash 普通单页写操作过程。从图 2 中可以看到，一个写操作分成三个阶段：输入命令地址阶段、数据传输阶段、访问介质阶段。公式中的三个部分分别对应这三个阶段。第一个部分是输入命令地址阶段耗费的时间。一个写请求，首先花费一个 tW 时间通过总线传输一个起始命令（80h），之后花费 5 个 t_{WC} 时间通过总线传输写操作的地址，在传输完数据之后将花费一个 t_{WC} 时间传输结束命令（10h），因此公式中第一个部分是 7 个 t_{WC}；公式的第二个部分是将数据从通道控制器的缓存传输到分组的数据寄存器上的传输阶段，这个部分的传输时间与传输的数据量有关，因此在公式的的第二个部分需要考虑传输的数据量 P 大小；公式的第三个部分是访问介质阶段，在这个部分，上一阶段传入的数据将从写到介质中，这个过程图 2 中忙（busy）状态所指，在写操作中这个时间是 t_{PROG}。

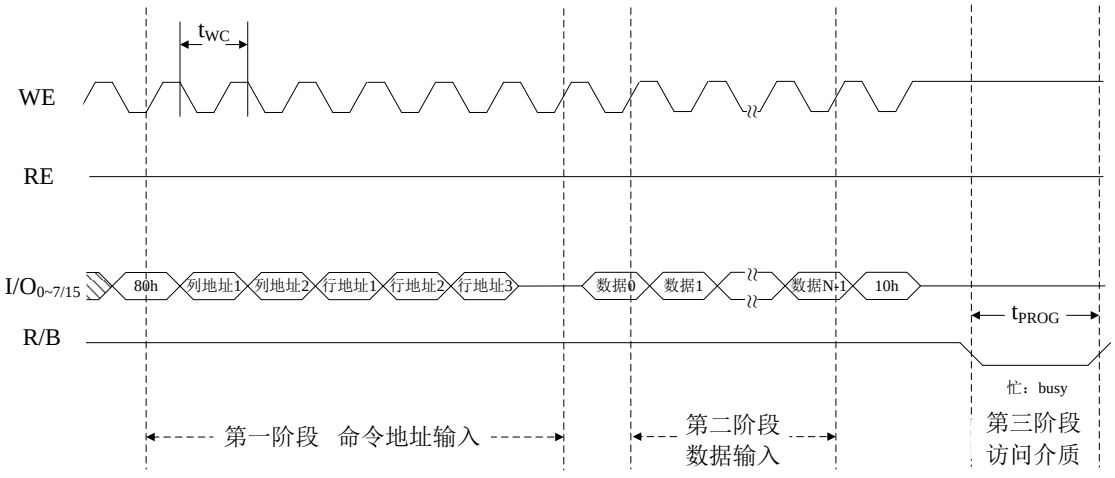


图 2　闪存写操作流程

驱动方式

3Dsim 属于模拟器的一种，一般来说，模拟器的驱动方式分为三种：时间驱动、事件驱动、请求驱动。3Dsim 的模拟行为是类似于 SSDsim 的，区别于上面的其它仿真器，是一种基于事件驱动的模拟器。事件驱动即是根据系统内部各个部件的状态改变、跳转来修改系统时间，是模拟的时间线不断向前仿真，直至结束。

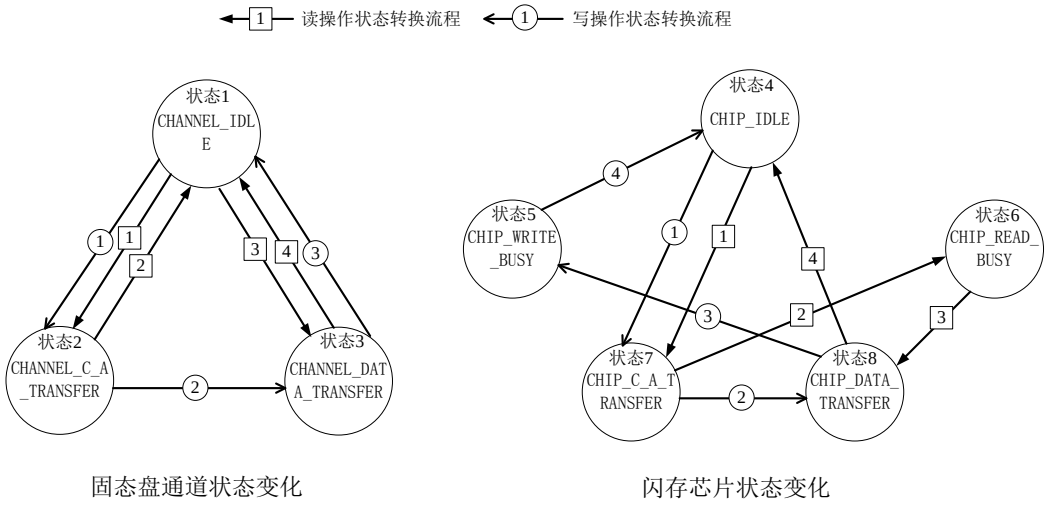
考虑到独立服务和并行性，3Dsim 中闪存颗粒和通道称为独立单元，每个独立单元的状态转变称为一个事件，每个事件的发生推动 3Dsim 的系统时间，使仿真不断进行直至结束。

闪存颗粒是固态盘中的独立服务单元，闪存通道相互之间也是独立的，在 3Dsim 中，将这些单元称之为独立单元。在 3Dsim 中，每个独立单元设置四个时间和状态变量：当前状态、当前状态时间、下一状态、下一状态预计时间。当前状态表示这个独立单元的当前所处的状态，下一状态表示这个独立单元从当前状态将进化到的下一状态。宏观上来看，固态盘中的闪存颗粒和闪存通道都存在两种状态，既工作状态（busy）和空闲状态（idle）。从微观上看，根据读、写、擦除的状态变化，闪存颗粒存在四种状态，包括空闲状态、接收命令地址状态、介质操作状态和数据传输状态，后三种状态使得闪存颗粒处于工作状态；闪存通道存在三种状态，包括空闲状态、数据传输状态和命令地址传输状态，后两种状态使得闪存通道处于工作状态。下表 2 中列出了这几种状态的名称和含义。

表 2：状态的各种名称和含义，其中 1-3 表示通道的状态，4-8 表示闪存颗粒的状态

	独立服务单元状态	**含义**		**停留时间**
1	CHANNEL_IDLE	通道	空闲	(0,∞)
2	CHANNEL_C_A_TRANSFER		命令地址传输	$7\times t_{WC}$
3	CHANNEL_DATA_TRANSFER		数据传输	传输的数据量$\times t_R$
4	CHIP_IDLE	芯片	空闲	(0,∞)
5	CHIP_WRITE_BUSY		介质操作：写	t_{PROG}
6	CHIP_READ_BUSY		介质操作：读	t_R
7	CHIP_C_A_TRANSFER		接收命令地址	$7\times t_{WC}$
8	CHIP_DATA_TRANSFER		数据传输	接收或发送的数据量$\times t_R$

图 3 画出了各个独立服务单元状态的跳转图：



固态盘通道状态变化　　闪存芯片状态变化

图 3 状态转换图

3Dsim 的仿真器的状态跳转按照上述的独立单元的状态机进行跳转。例如，对于一个执行读操作的内存颗粒而言，假设当前状态是接受命令地址状态（CHIP_C_A_TRANSFER），则下一个状态必然是读介质状态（CHIP_READ_BUSY），如图 3 从状态 7 到状态 6 的转变。表 2 给出了各个状态的停留时间，独立单元的下一状态预计时间就是依据每个状态的停留时间计算获得。在 3Dsim 中，状态的转变就是一个事件，每一个事件的发生将招致 3Dsim 的系统时间向前推进。具体来说，时间的推进方法是：已知每个独立单元的当前状态，和当前状态时间，同时下一状态和下一状态的预计时间可以通过图 3 和表 2 获得，在所有独立单元中寻找最先到达的下一个状态的时间，作为 3Dsim 的新的系统时间，周而复始地使模拟不断向前推进。通过对所模拟的固态盘内的每个状态进行观测，可以准确地得到对每条外部请求的响应时间，从而完成 3Dsim 的时间模拟。

3Dsim 总体框架

基本架构

3Dsim 系统架构如下图 4 所示，3Dsim 中始终维护有一个时间线和状态机，通过状态机的事件跳转，3Dsim 处理来自上层应用发出的对 3D flash 执行读写负载请求，解析请求，请求的数据通过缓存调度，经过 FTL 层的地址映射、垃圾回收等操作后，转换成为闪存可以响应的命令，在此模块可以判断是否可以组成高级命令，通过不同高级命令对 3D flash 进行操作，3D flash 响应操作，更改自己的状态表并输出统计信息到文件，最后修改时间线，反馈给状态机，进行新一条 trace 的读取，如此反复。

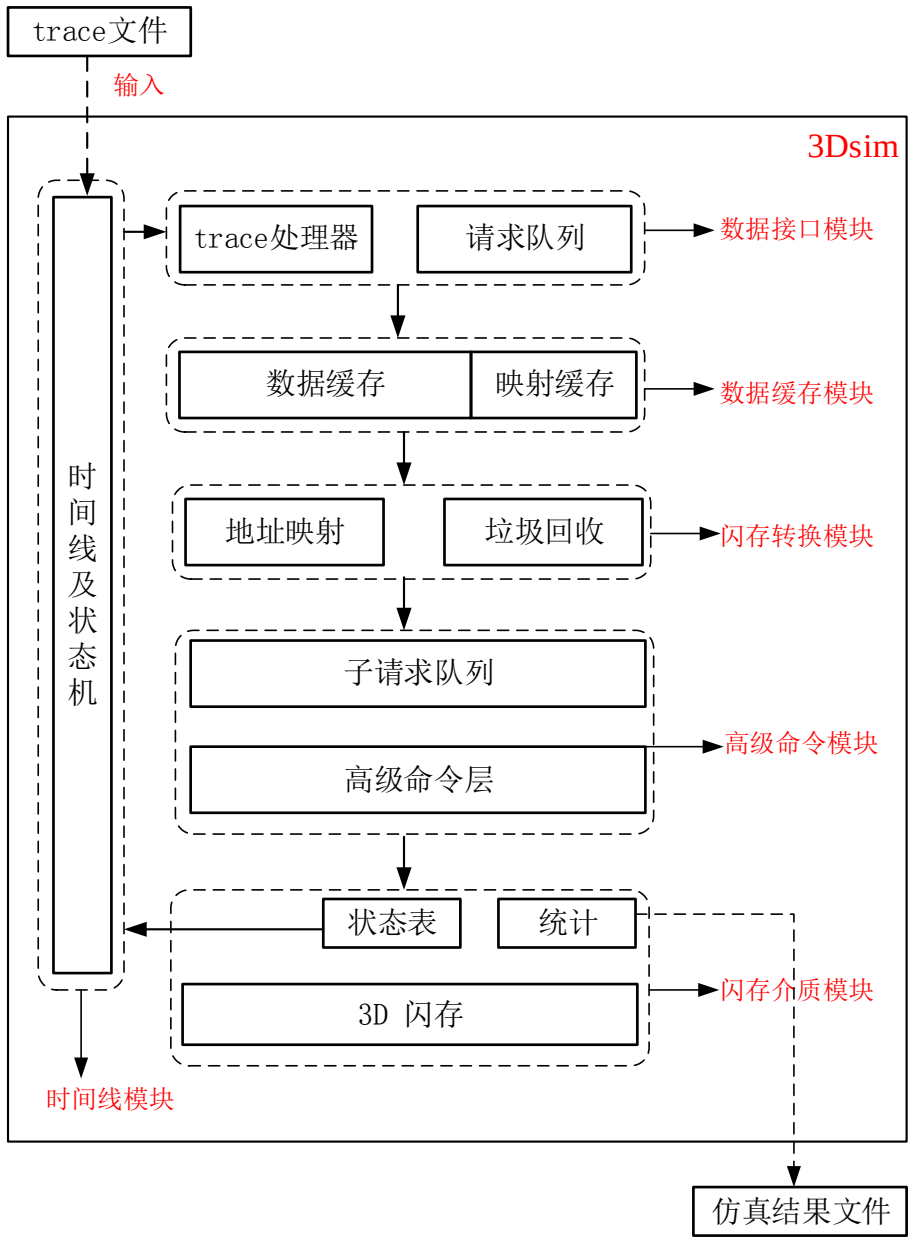


图 4 3Dsim 的基本架构图

模块划分

3Dsim 用来模拟固态盘的硬件结构（包括闪存颗粒，固态盘内部通道等），闪存转换层（包括数据缓存区、地址映射、垃圾回收及损耗均衡等）。3Dsim 分成 5 个逻辑模块，分别为时间线模块、数据接口模块、数据缓存模块、闪存转换模块、高级命令模块、闪存介质模块。下面对每个模块进行一个简单的介绍：

时间线模块

整个模拟器共维护四个时间线：current time(系统时间)、channel time（通道的时间）、chip time（芯片的时间）、 sub request time（子请求的时间）。时间线的推动步骤如下图 5 所示：

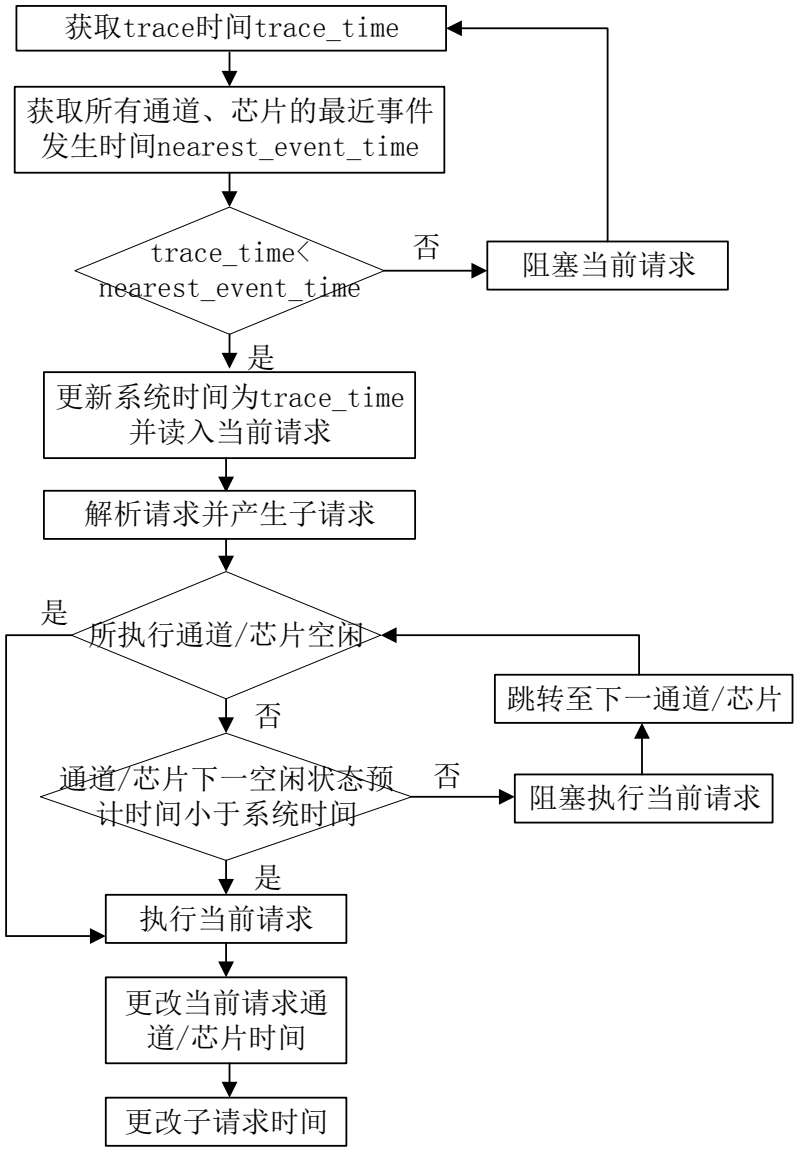


图 5 时间线推动流程图

数据接口模块

该模块模拟与主机单的数据通信，即接受下发的 trace，从 trace 文本中读取 trace，根据 trace 时间戳判断是否阻塞，并挂载在请求队列上，供数据缓存层使用。如图 6 所示：

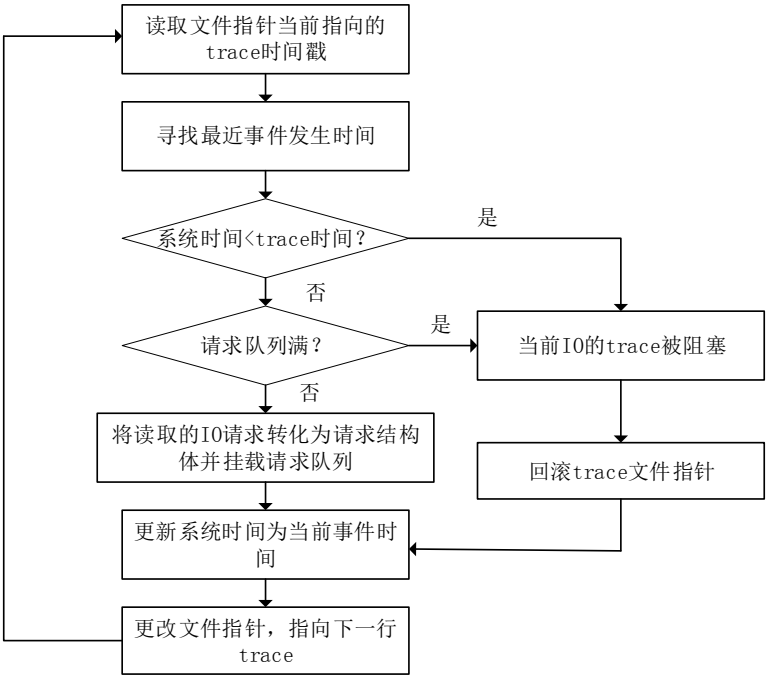


图 6　数据接口模块流程图

trace 处理器将读取的 trace 请求转换为请求结构体，每个结构体包含请求的完整信息：到达时间，逻辑设备号，访问起始扇区号，请求长度，操作类型。并用链表结构保存请求结构体，等待下一步处理。

数据缓存模块

缓存层主要是模拟真实 SSD 中 RAM 的功能，3Dsim 中的缓存层由两部分组成，一部分是数据缓存，用于缓存请求队列中的数据；另一部分是映射缓存，用于缓存逻辑地址到物理地址的映射条目。

请求队列上的请求结构在进入缓存之前会按照 3Dsim 的缓存大小可以通过参数文件进行配置，其中数据缓存部分以闪存页为逻辑节点单位，但是其真实操作的最小单位是闪存页的子页，它使用平衡二叉树进行缓存节点的查找和插入，以 LRU 算法维持一个双向链表进行缓存结点的替换，将未在缓存中命中的数据写入缓存并将该节点提到 LRU 链表的头节点，将 LRU 链表尾结点替换形成子请求写回闪存。由于 3Dsim 采用的是页映射，映射缓存部分存储的是全部逻辑页到物理页之间的映射条目。缓存结构如图 7 所示，数据缓存 LRU 链表逻辑结构如图 8 所示，buffer_group 代表一个逻辑节点，每次未命中的逻辑页写入都会在 LRU 链表首部插入一个新的结点，当链表长度达到最大限制会先替换写回链表尾部的结点。

使用者可以根据自己的需求修改缓存层算法，实现自己所需的功能或进行相关模拟仿真实验。

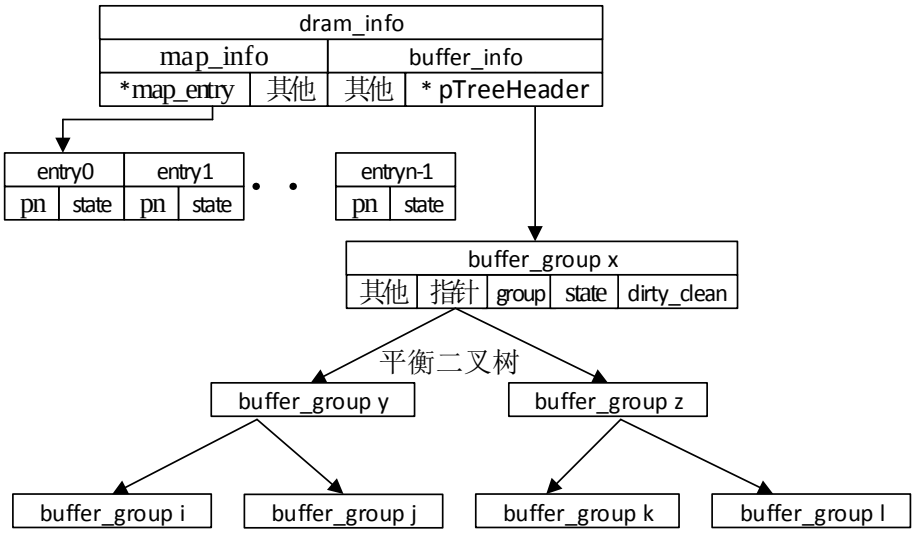


图 7 3Dsim 缓存结构图

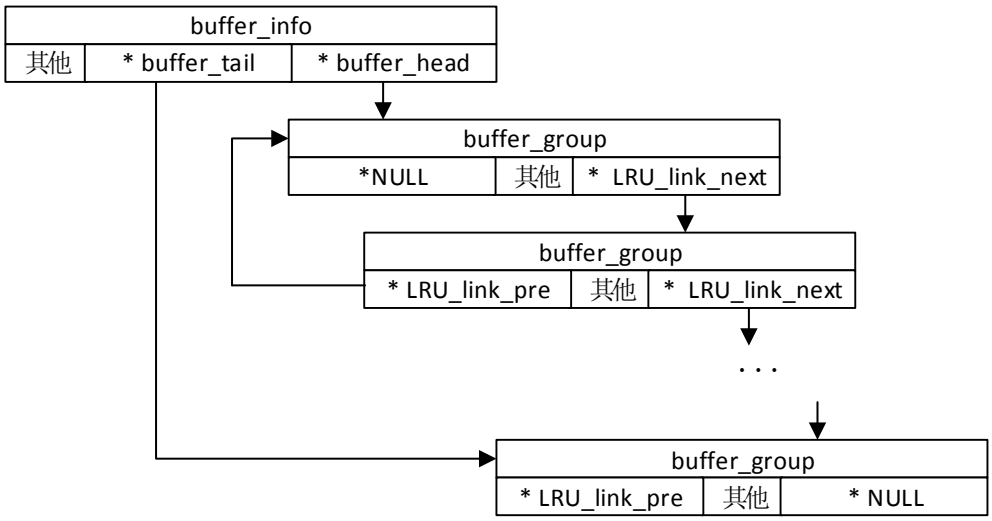


图 8 数据缓存 LRU 链表逻辑结构

闪存转换模块

在由于闪存的独特性质，真实固态盘中存在一个闪存转换层(Flash Translation Layer: FTL)，用于翻译上层文件系统的读写命令、管理闪存的各种操作，3Dsim 中也包含 FTL 层用来模拟真实 SSD 内部 FTL 层的各种算法，包括地址映射、垃圾回收等。

3Dsim 的闪存转换层采用简单的页映射算法，即每一个逻辑页对应一个物理页，逻辑页与物理页之间是全相连，映射缓存保存全部的映射条目，当闪存上的数据更新或者进行数据迁移时映射表也需要更新。

3Dsim 固态盘通道内空闲页数量低于阈值时，控制器产生垃圾回收请求，挂载在垃圾回收请求队列上，等待合适的时候进行垃圾回收操作。3Dsim 的闪存转换层采用基于超级闪存块的贪心垃圾回收算法，超级闪存块指的是同一晶圆的两个分组内相同偏移地址的两个闪存块组成的单位，3Dsim 以此作为垃圾回收的基本单位，贪心算法是选取无效页最多的超级闪

存块进行垃圾回收操作。

高级命令模块

高级命令层用于模拟 SSD 中高级命令的使用，通过请求调度，让多个子请求组成对应的高级命令，并向闪存介质层发送相应的高级命令，使用者可以在配置文件中设置支持高级命令与否以及支持何种高级命令。3Dsim 目前实现的高级命令包括：

Multi plane:同一晶圆内，同时对多个分组进行相同的读写擦操作；

Half page read:半页读，从介质中只读取半个页，仅支持单个 Plane；

One shot read:一次性读，一次从介质中读三个页，仅在 TLC 模式下支持；

One shot write:一次性写，一次向介质中写三个页，仅在 TLC 模式下支持；

Erase suspend/resume:擦除挂起/恢复命令，可以挂起正在执行的擦除操作，去执行同 Plane 内的读操作，读操作完成后恢复擦除操作；

3Dsim 在高级命令模块逻辑会完成能否组成高级命令对 3D flash 操作进行判断，具体的判断流程如图 9；

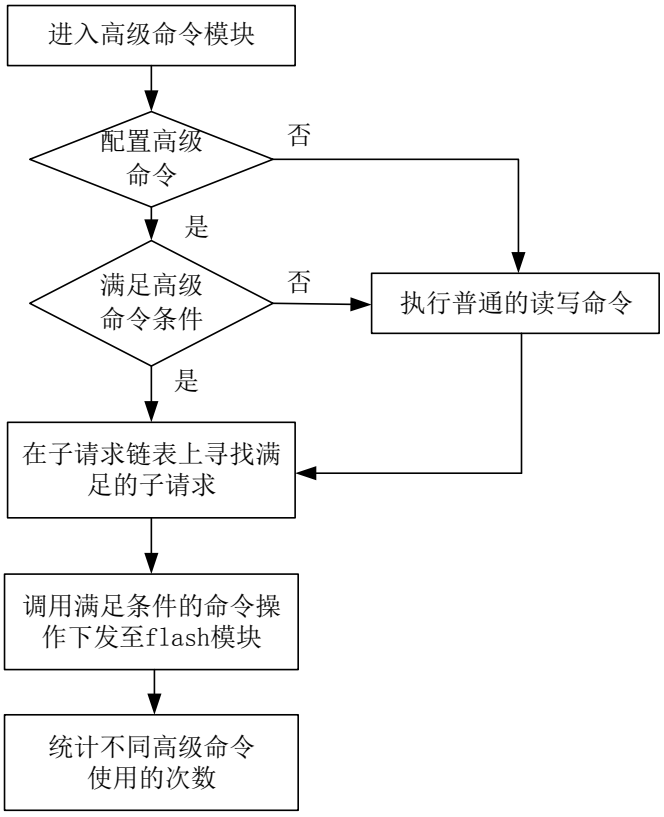


图 9 高级命令判断流程图

闪存介质模块

闪存介质模块模拟flash的读写擦操作，并记录其延时。3Dsim中存储体模拟不涉及真实数据存储，只记录相关统计信息。闪存阵列被组织成多通道（channel），每个通道有一个控制器，不同的通道可以并行访问。每个通道内部包含多个闪存芯片，这些芯片分时复用通道的总线。闪存芯片由外到内由：芯片（chip）— 晶圆（die）— 分组（plane）— 块（block）

— 页（page）五个层次构成，芯片内部结构如图10所示。

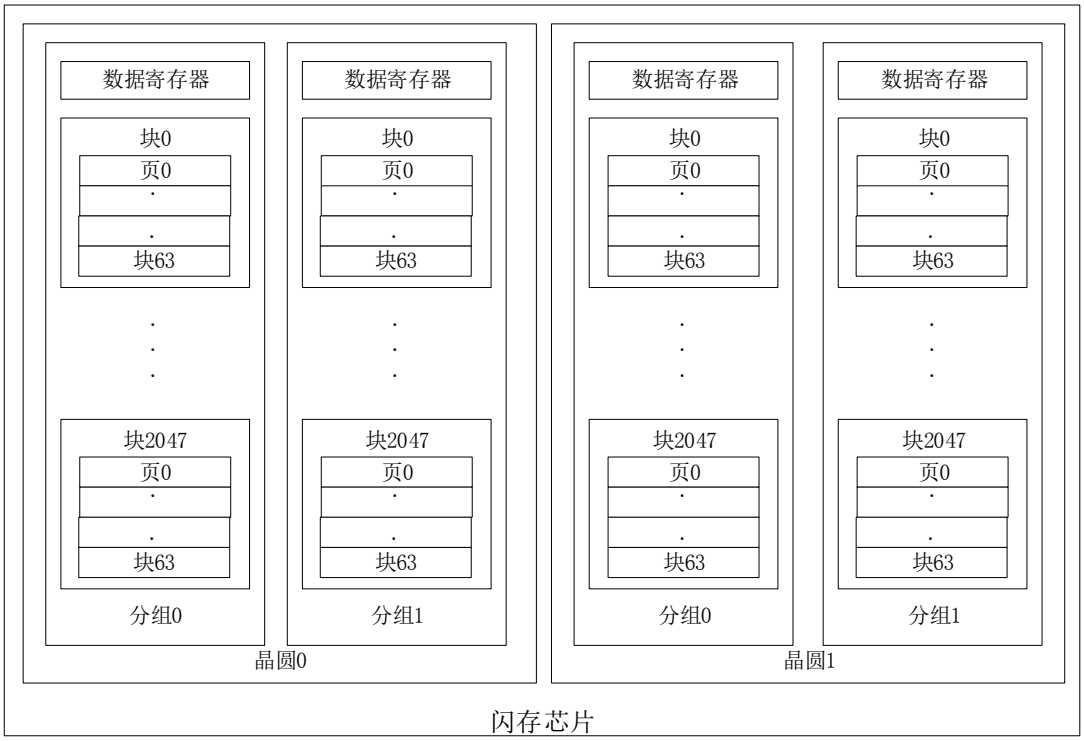


图 10 闪存芯片内部结构

3Dsim 使用结构体对该层次结构进行模拟，其在 3Dsim 中的结构如图 11 所示。对于每个结构单元的数目可以在参数文件中根据使用者需求进行具体配置。

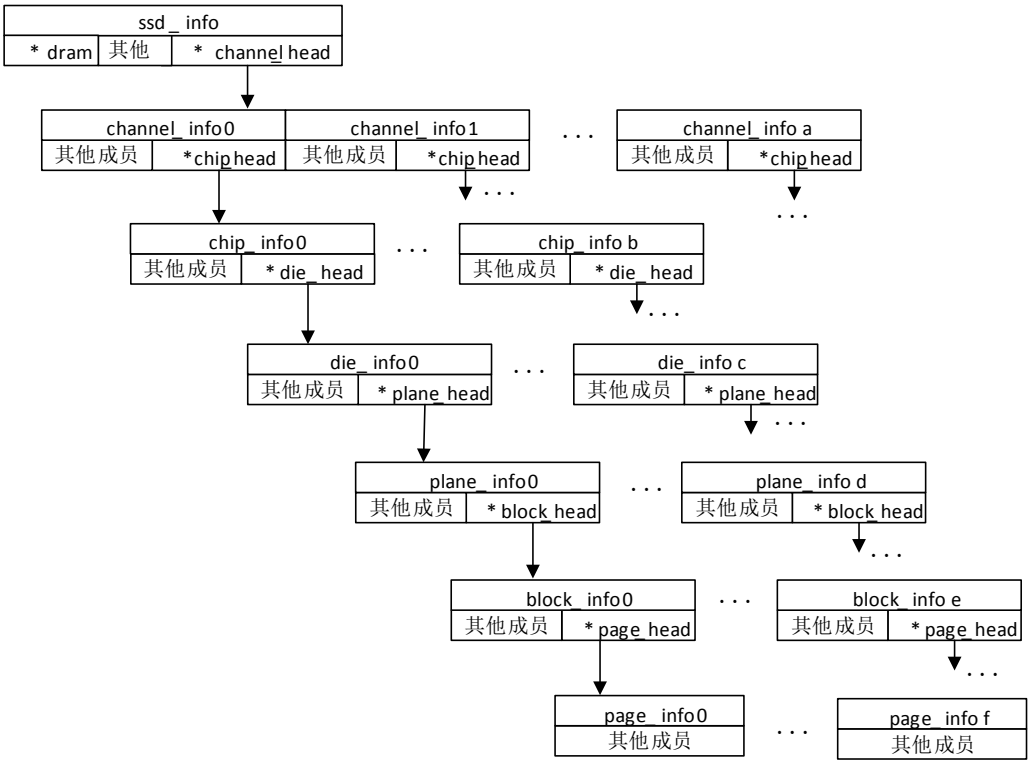


图 11　3Dsim 存储体结构图

基本流程

3Dsim 的基本操作流程如下图 12 所示：

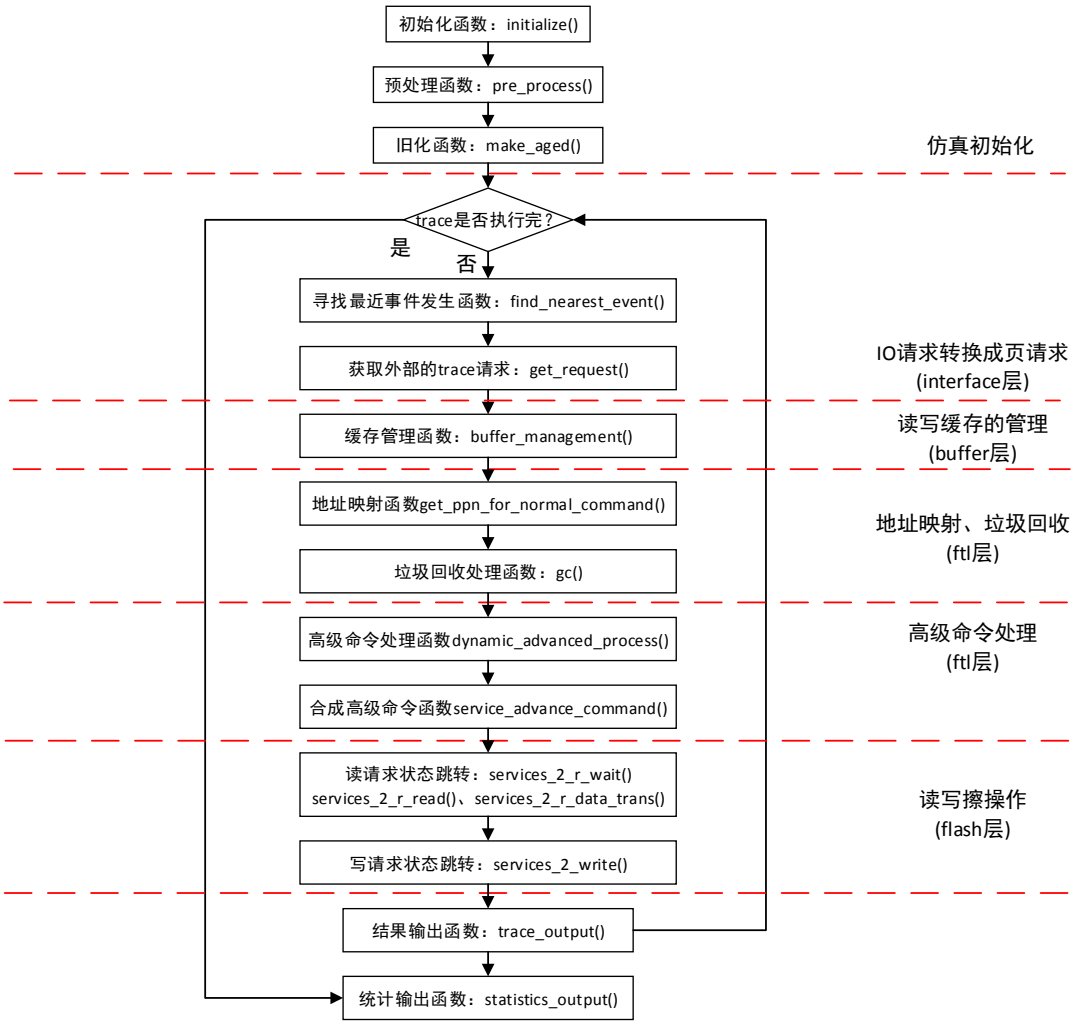


图 12 仿真基本流程图

3Dsim 从仿真流程下可以分为 5 个层次进行，首先进行仿真初始化，进行预处理操作，预处理函数将所有的读请求更改为写请求，构建映射表，这一操作保证读请求能读到有效的数据；然后进行正式仿真，正式仿真对应上面的各个模块，首先将 IO 请求转换为页请求，经过读写缓存的过滤，进行地址映射和垃圾回收，判断能否合成高级命令，使用普通命令或者高级命令操作对 flash 进行模拟，最后返回统计结果。

3Dsim 数据结构

在 3Dsim 中，存在三种类型的数据结构，他们分别是系统结构类、参数类、操作类。

系统结构类数据结构

系统结构类数据结构组成 3Dsim 的基础平台。在这一类数据结构中，每一种数据结构表示一种固态盘的逻辑或者物理单元，如表 3 所示：

表 3：系统结构类数据结构

编号	命名	模拟对象	主要成员
1	ssd_info	固态盘	channel_head dram current_time request_queue 、request_tail、{子请求}、{统计}
2	channel_info	通道	chip_head、{状态+时间}、{子请求}、{统计}
3	chip_info	芯片	die_head、{状态+时间}、{统计}
4	die_info	晶圆	plane_head
5	plane_info	分组	blk_head、add_reg_ppn、free_page
6	blk_info	物理块	page_head erase_count free_page_num invalid_page_num、last_write_page
7	page_info	一个物理页	valid_state、free_state、lpn
8	dram_info	固态盘中内存	map、buffer
9	map_info	映射关系区域	{统计}、map_entry
10	buffer_info	数据缓存区域	{队列}、{统计}、pTreeHeader
11	buffer_group	数据缓存节点	{队列}、group、stored、dirty_clean
12	entry	映射关系	pn、state

其中 1-8 是固态盘中的物理单元的数据结构，9-12 是固态盘中的逻辑单元的数据结构。1-6 的结构体中第一个成员是构成模拟平台的关键成员（channel_head、chip_head、die_head、plane_head、blk_head、page_head）。固态盘中存在多个独立的通道，每个通道上存在多个独立的芯片，每个芯片包括多个晶圆，每个晶圆中有多个分组，

每个分组由多个块组成，多个页组成一个块。在 3Dsim 中，结构体 channel_info、chip_info、die_info、plane_info、blk_info、page_info 分别表示通道、芯片、晶圆、分组、块、页。所要模拟的固态盘是什么样的硬件结构，3Dsim 中就由这些结构体来表示这个固态盘的组成。

在固态盘中，除了上述闪存的物理结构，还有一个物理结构就是内存，在 3Dsim 中的 ssd_info 结构体中，使用结构体 dram_info 来表示这个物理结构。在 dram_info 中，存在两个成员，即 map_info 结构体和 buffer_info 结构体。map_info 结构体的成员 map_entry 结构指针指向一个名为 entry 的 map_entry 结构体数组，这个固态盘中有多少映射关系就有多少个 entry 成员。buffer_info 结构体的成员 pTreeHeader 指向一个平衡二叉树的根节点，这个平衡二叉树的节点是 buffer_group 结构体，每个 buffer_group 表示一个数据缓存区节点。系统结构类结构体的关系如下图 13 所示：

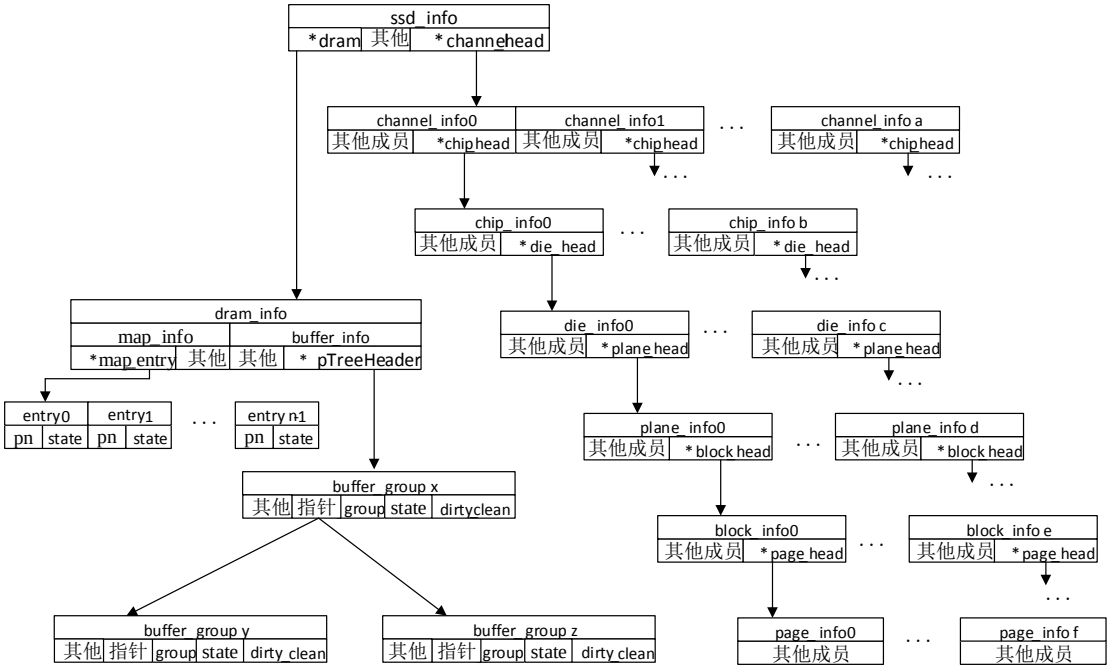


图 13 系统结构类结构体

参数类数据结构

因为 3Dsim 需要模拟不同的硬件结构，软件算法，所以需要提供外部输入不同参数的方法。在 3Dsim 中，通过参数文件的方式，将这些参数输入，输入的参数需要记录在模拟程序中，以便 3Dsim 在模拟过程中随时查询。所以在 3Dsim 中存在一类数据结构：参数类数据结构，用作记录这些参数，具体如下表 4 所示：

表 4：参数类数据结构

编号	命名	意义	主要成员
1	ac_time_characteristics	时序参数	tWC、tRC、tR、tPROG 等
3	parameter_value	整体参数	其他算法参数

其中第一个数据结构记录闪存的时序参数，第二个数据结构记录 FTL 算法的一些参数。

操作类数据结构

在 3Dsim 中，无论是外部请求还是内部产生的子请求，均由操作类结构体完成，该机构体主要维护多个队列以及用于缓存查找的平衡二叉树，具体如下表 5 所示：

表 5：操作类数据结构

编号	命名	意义	主要成员
1	request	外部请求	类型、LSN、大小、到达时间
2	sub_request	子请求	类型、LPN
3	local	物理地址	通道号、芯片号、晶圆号、分组号、块号、页号
4	gc_info	垃圾回收信息统计	{统计}
5	direct_erase	可直接擦除块号	块号

6	gc_operation	垃圾回收操作请求	local
7	suspend_location	挂起操作保留地址	通道号芯片号、晶圆号、分组号、块号

3Dsim 操作流程

3Dsim 用来模拟固态盘的行为，根据外部输入的请求流，得出在特定硬件结构，闪存转换层算法及数据缓存算法下固态盘的性能表现。为达到这个目的，3Dsim 在开始模拟前，需要输入三个文件：参数文件、负载踪迹文件（trace）、输出文件。

参数文件(page.parameters)

3Dsim 的参数配置文件为 page.parameters，从 SSD 的组织结构、flash 介质的操作时间、SSD 软件算法策略三个方面进行配置，其参数列表 6 如下：

表 6-1：SSD 组织结构参数

SSD 组织结构参数	
dram capacity	缓存大小，单位 byte
channel number	channel 总个数
chip number	chip 总个数
die number	一个 chip 包含 die 的个数
plane number	一个 die 包含 plane 的个数
block number	一个 plane 包含 block 的个数
page number	一个 block 包含 page 个数
subpage page	一个 page 包含子页的个数
page capacity	页大小，单位 byte
subpage capacity	子页大小，单位 byte

表 6-2：flash 介质时间参数

flash 介质时间参数	
t_R	数据从目标物理页中读到分组的寄存器所消耗的时间,单位 ns
t_PROG	数据从分组的寄存器写到目标物理页所消耗的时间,单位 ns
t_BERS	目标物理块的擦除时间,单位 ns
t_WC	通过数据总线上向寄存器传输一个字节的数据所消耗的时间,单位 ns
t_RC	通过数据总线从寄存器向外传输一个字节的数据所消耗的时间,单位 ns
t_DBSY	高级命令传输数据地址时间间隔，单位 ns
t_WB	数据总线忙 busy 时间，单位 ns
t_PROGO	高级命令 one shot program 三个页写时间，单位 ns

表 6-3：软件算法策略参数

软件算法策略参数	
erase limit	记录 block 最大次数擦写次数
overprovide	op 空间大小
requset queue depth	请求队列深度
scheduling algorithm	记录使用哪种调度算法，1:FCFS
buffer management	缓存策略：0：sub_page 拼凑

address mapping	映射表策略：1：page；2：4kb_map
wear leveling	WL 算法
gc	gc 策略：1：superblock
gc hard threshold	gc 硬阈值大小，当 plane 内无效页个数超过此阈值时，触发 gc 操作
allocation	分配方式，0：动态分配，1：静态分配
dynamic_allocation	动态分配方式类型，0:全动态，1:channel 静态,chip，die，plane 动态，2:channel，package，die 静态,plane 动态
dynamic_allocation_priority	动态分配方式优先级，0：channel>chip>die>plane,1：plane>channel>chip>die
advanced command	高级命令支持，用二进制表示，无(00000)、mutli plane(00001), half-page-read(00010), one shot program(00100),one shot read(01000),erase suspend/resume(10000)
greed MPW command	multi-plane 贪心算法，0：非贪心，1：贪心
aged	旧化处理，0：non-aged，1：aged
aged ratio	旧化率
flash mode	flash 支持模式，0：slc,1：tlc

负载追踪文件(example.ascii)

负载踪迹文件既通常所说的 trace 文件，用作记录特定的负载的所有的请求的到达时间，请求的类型，请求的逻辑地址，请求的大小。负载踪迹文件的所有请求都是按照时间顺序排列。3Dsim 根据这个踪迹文件，可以重现出这个特定负载或者应用环境下的请求流，通过参数文件设定的硬件结构和算法，可以得到在这种负载下该硬件结构和算法的性能，从而寻找出在这个负载下最优的硬件结构和或算法。Trace 文件在仿真器中以 xxx.ascii 文件命名，其基本结构如下表 7：

表 7：负载文件格式

请求到达时间	逻辑设备号	请求起始扇区号	请求大小	操作类型

输出文件(statistic10.dat/ex.out)

输出文件是用来记录每个请求的响应时间,以及特定硬件结构和软件算法在服务这个负载时的性能统计结果。每个请求的队列等待时间，服务时间和响应时间都依次输出到输出文件中，在模拟完成后输出服务整个负载的平均响应时间、读写请求数等统计信息。3Dsim 的输出文件总共包含两种，一种以.dat 为后缀，是记载简单的最后统计信息，一种是以.out 为后缀,记录了完整了每一条请求的执行情况。这其中统计分为三个部分:请求的延时统计，高级命令执行次数的统计、缓存命中率的统计。

请求延时统计：3Dsim 请求采用多种分配方式，trace 下发送的请求转换成子请求并分配到各个 die 上执行，当所有子请求执行完成后，该请求才执行完从请求队列上去除，所有子请求中的最大完成时间减去最小开始时间即为当前请求的延时。

读写平均响应延时计算公式如下：

$$\mathrm{read}_{\mathrm{avg\_time}} = \sum \frac{\mathrm{sub}_{\mathrm{endtime}_{\max}} - \mathrm{sub}_{\mathrm{begintime}_{\min}}}{\mathrm{read}_{\mathrm{count}}}$$

$$\mathrm{write}_{\mathrm{avg\_time}} = \sum \frac{\mathrm{sub}_{\mathrm{endtime}_{\max}} - \mathrm{sub}_{\mathrm{begintime}_{\min}}}{\mathrm{write}_{\mathrm{cont}}}$$

高级命令执行次数的统计：普通的读擦写及各种高级命令所有的统计参数都挂载在 ssd_info 结构体上。具体如下表 8 所示：

表 8-1：高级命令统计内容

read_count	普通读次数
update_read_count	更新读次数
gc_read_count	垃圾回收迁移读次数
program_count	普通写次数
pre_all_write	预处理写次数
update_write_count	更新写次数
erase_count	擦除次数
m_plane_prog_count	Mutli plane 写次数
m_plane_read_count	Mutli plane 读次数
half_page_read_count	半页读次数
mutliplane_oneshot_prog_count	Mutli plane one shot 组合写次数
one_shot_read_count	One shot 读次数
one_shot_mutli_plane_count	Mutli plane one shot 组合读次数
resume_count	擦除恢复命令使用次数
suspend_count	擦除挂起命令使用次数
suspend_read_count	擦除挂起读操作使用次数

表 8-2：缓存统计参数

缓存命令的统计：主要集中在一级缓存的读写命中率，二级缓存较小，具体如下表所示：

read_hit	读请求命中次数
read_miss_hit	读请求未命中次数
write_hit	写请求命中次数
write_miss_hit	写请求未命中次数

执行环境

表 9 测试系统的环境配置

配 置	参数说明
操作系统	Windows10 专业版
CPU	Intel(R) Core(TM) i5-6500 CPU @3.20GHz
内存	8GB
编译器	Visual Studio 2013