目　录

SATA2.0 协议综述 4

1.1 SATA2.0 协议规范 4

1.1.1 SATA2.0 物理层协议解析 4

1.1.2 SATA2.0 链路传输协议解析 5

1.1.3 SATA2.0 命令层协议解析 9

1.1.4 SATA2.0 NCQ 协议解析 12

1.2 SAPIS 接口标准规范 16

SATA2.0 协议综述

1.1 SATA2.0 协议规范

SATA（Serial Advanced Technology Attachment）是连接硬盘和计算机系统的新标准，是由 Intel、IBM、Dell、APT、Maxtor 和 Seagate 公司共同提出的串行硬盘接口规范。SATA 规范采用串行高速差分模式进行信号传输，使它相对于传统硬盘接口标准更具优势，从根本上解决了 PATA 接口存在的串扰问题。SATA2.0 规范可支持 3.0Gbit/s 的数据传输速率。相比传统并行接口技术 SATA2.0 具有以下优势：

- 传输速度更快：SATA2.0 串行 3.0Gbit/s，相当于并行 300MB/s；
- 针脚仅为 7 针，利于机箱散热，便于布局；
- 支持 Hot-plug、支持 NCQ（原生命令队列）技术；
- 支持数据和命令包的 CRC 校验，总线传输更可靠。

以上优势确保了采用 SATA 接口技术意味着拥有更快的传输速度、更便捷的连接方式、更可靠的数据传输、更高效的数据存储。图 2-1 是 SATA2.0 协议栈的构成，主要有应用层、传输层、链路层和物理层。以下对 SATA 规范的各层协议进行解析。

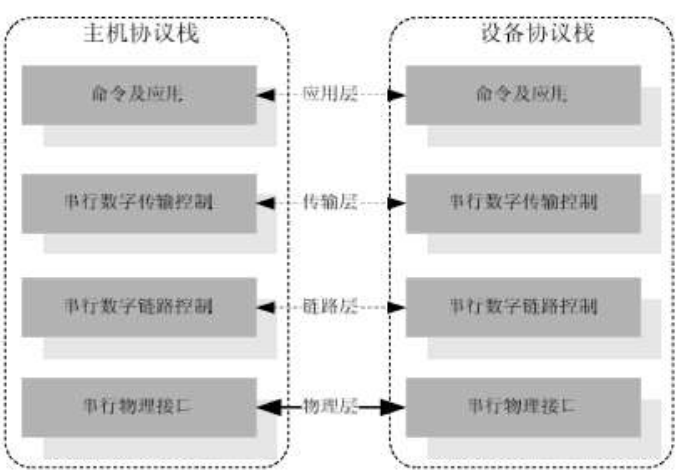

图 1-1 SATA2.0 协议栈

1.1.1 SATA2.0 物理层协议解析

SATA 物理层采用了诸如 LVDS NRZ 串行数据传输、逗号序列检测以及电源

管理等技术，有效的实现了 3.0 Gbp/s 串行数据流发送和接收、串行数据流定界等功能。物理层还包含了并串转换器，时钟恢复提取模块，同时提供带外 OOB 信号用于实现物理链路的上电初始化和速度协商功能。SATA2.0 物理层具有以下功能：

- 实现 3.0 Gbp/s LVDS NRZ 串行数据流传输；
- 可从串行数据流中提取数据和高精度时钟；
- 提供 100 欧姆阻抗匹配，支持可选的发送和接收阻抗匹配调节；
- 负责实现数据流的串并/并串转换；
- 利用逗号序列检测技术实现串行数据流定界；
- 提供带外 OOB（Out of Band）信号的检测和发送；
- 支持速度协商机制；
- 实时的向上层提供设备连接状态；
- 支持可选的电源管理；

从 SATA2.0 协议 7.1.2.1 节物理层的典型结构框图中可以看出，控制模块负责实现与链路层的通信，包括接收链路层的传输控制信号，向链路层反馈物理层的工作状态信息，从而协调控制整个物理层功能实现。模拟前端采用串行高速差分驱动器和接收器实现传输连接线的基本接口。而 OOB 信号处理模块负责实现 OOB 信号的传输和检测。同步字符源和同步字符检测模块负责串/并转换以及字符定界过程的同步实现。数据提取模块是从高速输入数据流中提取时钟和数据。

1.1.2 SATA2.0 链路传输协议解析

链路传输协议主要应用于 SATA 链路层和传输层帧传输过程中。帧传输在大多数情况下是软件通过发送特定的命令引起的，每个命令会引起主机适配器和设备之间进行一系列的帧结构信息交互。在某种情况下，帧结构信息是直接由应用层决定的。无论哪种情况，用于帧传输的链路传输协议在本质上是相同的。

SATA2.0 数据链路层和传输层协同完成帧的构造和传输。传输层构造要传输的帧，将其存储在发送缓冲区并通知链路层该帧进入传输挂起状态。链路层通过控制原语的传输和接收实现链路传输协议的大部分功能。图 2-2 是帧传输过程中涉及到的传输层和链路层组成单元示意图。

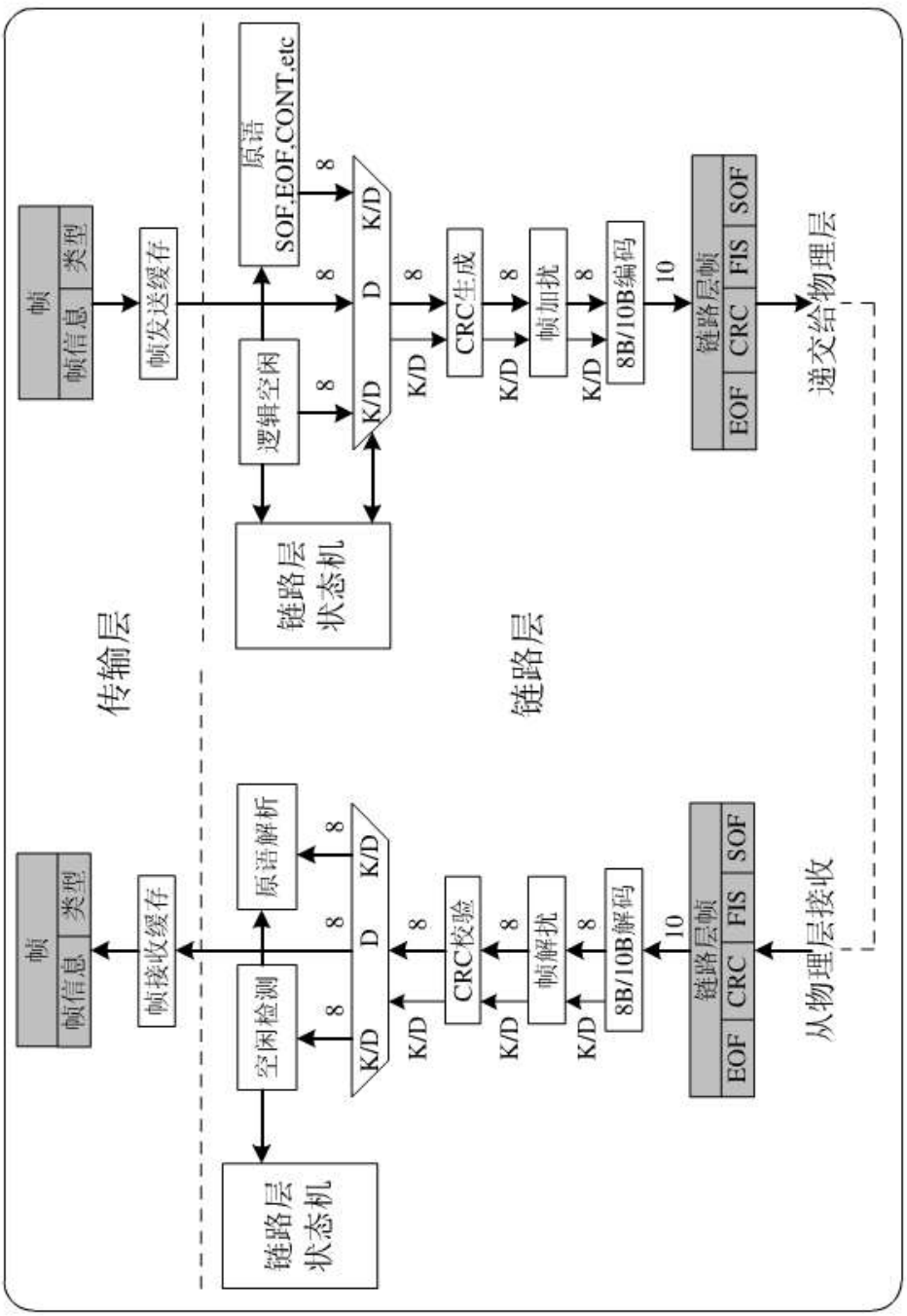

图 1-2　传输链路层协议框图

图 2-2 描述了与帧传输控制协议相关的单元。传输层介于链路层与应用层之间，负责帧的构造与解析，传输层无需知道帧是如何被发送和接收的。主机端和设备端通过传输层帧的交互来完成命令交互和数据传递。

需要发送帧时，传输层完成以下功能：

（1）根据帧类型组建帧，将帧信息按正确的顺序填充；

（2）接收链路层帧传输请求，将帧发往链路层；

（3）管理上层缓存，响应链路层流控；

（4）对于上层请求发送的帧，返回完成或错误状态；

当接收到帧时，传输层提供以下功能：

（1）接收来自链路层的帧；

（2）检测帧类型，根据帧类型区分帧内容并提取；

（3）返回帧接收完成或错误状态；

SATA2.0 传输层的主要工作是将要传输的数据封装成 FIS（Frame Information Structures），或者把接收到的 FIS 去除封装还原成数据，而不需要知道信息块是怎么样被发送和接收的。FIS 是以 Dword 为基本单元，用来在主机和设备之间传递信息的一种帧结构，控制原语 SOF 和 EOF 作为其的开始和结束标志，在传输过程中也可以用 CONT、HOLD\HOLDA 等原语来对其进行流量控制。主要 FIS 类型及功能见表 2-1。

表 1-1　主要 FIS 类型及功能

帧类型码	帧类型名称	帧描述
27h	寄存器型 H2D 帧	更新影子寄存器内容到设备端寄存器
34h	寄存器型D2H帧	更新设备端寄存器内容到影子寄存器
39h	DMA 激活帧	设备响应主机发起的 DMA 传输
41h	DMA 配置帧	配置 DMA 引擎
46h	数据帧	用于数据传输
58h	内建自测试激活帧	进入内建自测试模式
5Fh	PIO 配置帧	PIO 传输模式参数传递
A1h	寄存器位更新帧	更新影子寄存器中位
C7h	用户自定义帧	用户自定义

链路层主要负责发送和接收帧信息。通过传输层的控制信号发送和接收原语信号并向传输层提交。链路层并不需要关注所传输帧的内容，而是实现与帧传输相关的事件序列控制，包括原语的生成与接收，以及当传输层请求发送帧时，链接层做出及时响应。有帧要发送时，其主要实现如下功能：

（1）实现点对点链路层对等协商，避免主机和设备同时请求发送数据。若出现冲突时，设备传输优先；

（2）从传输层接收帧信息并添加帧头、帧尾以及 CRC 校验码等信息；

（3）对数据信息进行加扰、8B/10B 编码；

（5）发送帧，进行流量控制；

（6）接收对等链路层的帧收接信息，向传输层报告传输完成或错误信息。

当接收从物理层送来的帧时，链接层实现如下功能：

（1）从物理层接收帧数据，并进行流量控制；

（2）去除帧头、帧尾信息。对数据进行解码、解扰并进行 CRC 校验。

（3）向对等链路层响应帧接收信息，向传输层报告接收正确或错误信息。

原语在链路层产生，在帧传输过程中提供流量控制和状态信息反馈，同时也可提供电源管理功能。表 2-2 是 SATA2.0 协议中规定的各种原语。原语由一个十位控制码与紧跟其后三个连续的数据码组成。SATA2.0 协议规定物理层完成初始化后，链路只能发送原语和帧信息。

表 1-2　SATA2.0 协议中的原语

原语名称	原语编码(低字节在前)	原语描述
ALIGN	{D27.3, D10.2, D10.2, K28.5}	成对发送，用于链路状态或流控原语复现
CONT	{D25.4, D25.4, D10.5, K28.3}	原语长串发送时，用于对上个原语的重复
DMAT	{D22.1, D22.1, D21.5, K28.3}	请求发送节点停止 DMA 数据的传输
EOF	{D21.6, D21.6, D21.5, K28.3}	标识帧结束，其上一个双字是帧内 CRC
HOLD	{D21.6, D21.6, D10.5, K28.3}	数据未准备好或当前无空间接收数据
HOLDA	{D21.4, D21.4, D10.5, K28.3}	接收到 HOLD 原语则回应该原语
PMACK	{D21.4, D21.4, D21.4, K28.3}	允许进入功率管理状态的原语回复
PMNAK	{D21.7, D21.7, D21.4, K28.3}	拒绝进入功率管理状态的原语回复
PMREQ_P	{D23.0, D23.0, D21.5, K28.3}	请求进入低功耗管理状态的原语发送
PMREQ_S	{D21.3, D21.3, D21.4, K28.3}	请求进入休眠状态的原语发送
R_ERR	{D22.2, D22.2, D21.5, K28.3}	节点检测到数据传输错误时发送该原语
R_IP	{D21.2, D21.2, D21.5, K28.3}	节点当前处于数据正确接收状态
R_OK	{D21.1, D21.1, D21.5, K28.3}	帧被正确接收时，节点返回此原语
R_RDY	{D10.2, D10.2, D21.4, K28.3}	节点准备接收帧
SOF	{D23.1, D23.1, D21.5, K28.3}	标识帧开始
SYNC	{D21.5, D21.5, D21.4, K28.3}	总线空闲
WTRM	{D24.2, D24.2, D21.5, K28.3}	等待接收节点返回帧的接收状态
X_RDY	{D23.2, D23.2, D21.5, K28.3}	节点准备发起帧的传输

1.1.3 SATA2.0 命令层协议解析

SATA2.0 协议中数据是在命令的执行过程中完成传输的，每一个命令的执行都伴随着若干特定帧的交互。命令层协议负责控制这些帧的交互过程。表 2-3 是 SATA2.0 协议支持的命令类型。

表 1-3 SATA2.0 协议支持的命令

命令类型	数目
Non-Data	34
PIO Data-In	13
PIO Data-Out	14
DMA-In	2
DMA-Out	2
DMA-In queued	2
DMA –Out queued	2
Packet (ATAPI)	1
Sevice	1
Device Reset	1
Execute Device Diagnostics	1

在本节中，会对 SATA 接口环境下常用到的基于数据传输的命令交互过程进行解析。这些命令大部分在传统 PATA 接口协议中也会被执行，但有的命令是新加入到 SATA 命令协议中的，这些命令支持新的特性，提高了数据传输的效率。

每个命令序列开始执行时，首先由主机通过寄存器 H2D 帧将命令传输给设备。一旦设备接收并识别该命令，主机与设备之间的命令协议便被建立起来。以下是具体命令的执行过程。

PIO 读命令的执行过程如图 2-3，首先主机驱动软件会通过写影子寄存器组中的命令寄存器来发起 PIO-Read 命令的执行，然后双方按如下过程进行交互。

- 主机适配器将 BSY 位置 1 并向设备发送寄存器帧。
- 设备读取该命令并在向主机发送数据之前，发送 PIO 配置帧给主机。
- 接收到 PIO 配置帧后，主机适配器分配缓存区保存该帧内容。
- 设备在 PIO 配置帧后紧接着发送数据帧到主机。
- 接收到数据帧后，主机适配器将 PIO 配置帧内容更新到影子寄存器组，将

状态寄存器中 DRQ 置位并清除 BSY，根据中断标志向主机软件请求中断。

- 主机控制器开始将数据帧中的数据读入数据寄存器并由主机软件读取，同时会将读取数据的数目与 PIO 配置帧中标记的数据数目进行对比。若读完，主机将 PIO 配置帧中结束状态标识更新到状态寄存器。若没有数据继续传输则清除 DRQ 标志，产生命令执行完成状态；若仍有数据要传输，则置位 BSY 位，然后设备继续发送 PIO 配置帧并重复以后的执行过程。

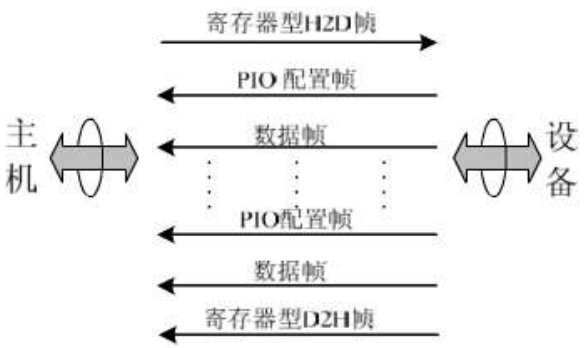

图 1-3 PIO 读命令

PIO 写命令的执行过程如图 2-4 所示，其过程与 PIO 读命令基本相同，只是数据帧传输的方向是由主机到设备。该方向信息可通过 PIO 配置帧中 D 位进行判断。当扇区寄存器中所要求传输的数据被传输完成后，设备应该通过发送寄存器 D2H 帧向主机反馈命令执行完成状态并清除 BSY 位。

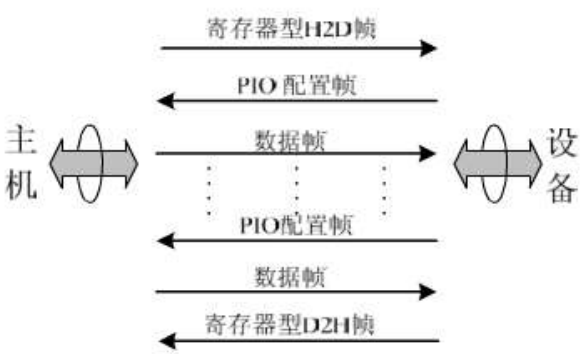

图 1-4 PIO 写命令

DMA 读命令执行过程如图 2-5 所示，在发送命令之前，主机软件会对主机适配器中的 DMA 控制器进行初始化配置，包括分配内存空间并指明传输方向。然后按以下步骤执行命令。

- 主机驱动软件写命令寄存器，主机适配器将状态寄存器中 BSY 置位，然

后发送寄存器 H2D 帧到设备。

- 设备接收并识别命令，准备数据。然后通过一个或若干个数据帧将数据传输给主机。主机适配器识别数据帧后，由 DMA 控制器将数据读入特定内存区域。
- 数据传输完成后，设备向发送寄存器 D2H 帧，反馈命令的执行状态并清除 BSY 位，若中断标志有效，则向主机产生中断信息。

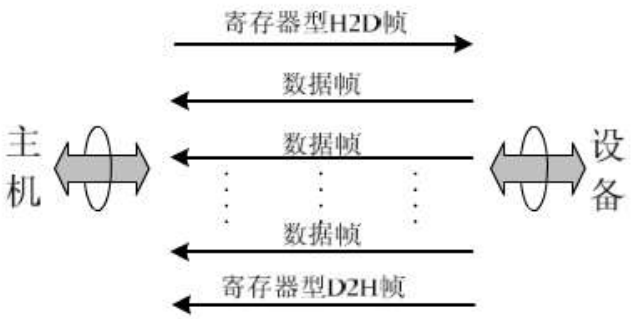

图 1-5　DMA 读命令

DMA 写命令的执行过程与 DMA 读命令基本相同。区别在于数据帧的传输方向，还有就是在每一个数据帧发送之前设备会发送 DMA 激活帧来激活主机端 DMA 控制器。图 2-6 是 DMA 写命令执行过程示意图。

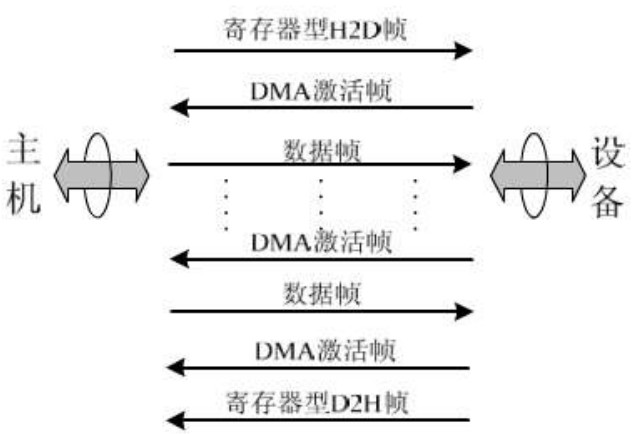

图 1-6　DMA 写命令

NCQ（原生命令队列）读/写命令是 SATA 协议独有的，其命令执行过程在 SATA2.0 NCQ 协议解析一节中将做详细介绍，这里不再叙述。图 2-7 和图 2-8 是其执行过程示意图。

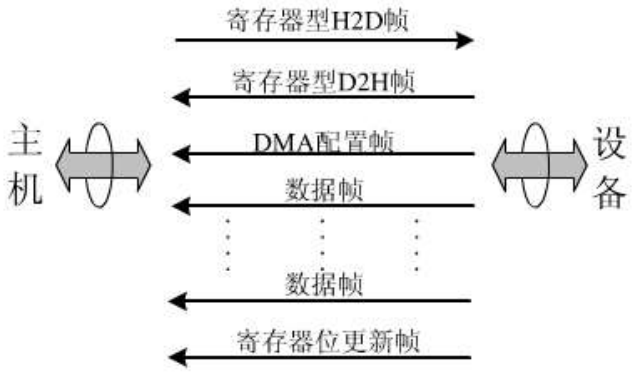

图 1-7 NCQ（原生命令队列）读命令

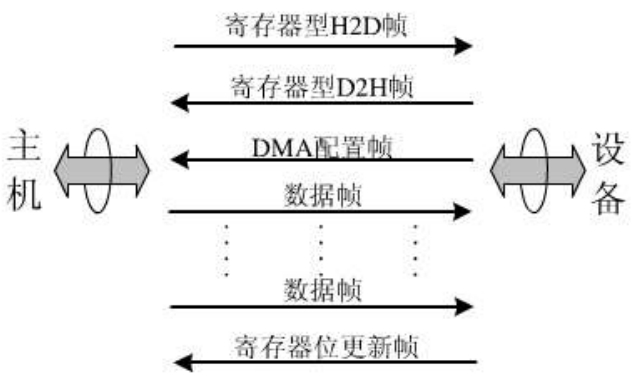

图 1-8 NCQ（原生命令队列）写命令

1.1.4 SATA2.0 NCQ 协议解析

命令队列使硬盘可以从主机控制器接收多个命令，然后重新组织这些命令的执行次序，从而实现最大化的数据吞吐。硬盘在执行命令时，大部分时间花费在将磁头正确定位到待传输数据上。因此，硬盘可以通过采用寻道优化策略选择下一个将要执行的命令来减小硬盘寻道正确传输的数据所花费的延时。

硬盘可以通过选择这样的命令来实现磁头旋转优化策略，即选择待传输最靠近当前磁头位置的命令作为下一个执行的命令；可以通过对比当前磁头所在柱面与下一个命令磁头所在柱面位置来实现定位优化策略。命令队列最大的优点就是命令的发布和数据定位可以同时进行，即磁头执行当前命令数据定位寻道的时间里，硬盘可以同时发起另一个不同命令的数据传输。例如，主机发送了一个新的命令到硬盘，此时硬盘可以为另一个不同的命令来寻道并定位其要传输的数据位置。本质来讲，就是命令的发布时间被节省了，因为他和另一个命令的寻道时间是重叠的，类似于 CPU 指令的流水执行机制。

图 2-9 显示了主机控制器与硬盘通信示意图，每一个队列命令都有一个标识号。无论是在该队列命令的数据传输阶段还是命令执行完成阶段，主机都会用到此标识号。在执行一个命令的数据传输时，硬盘会将该命令的标识号发给主机，然后主机会启动 DMA 操作，并指向主机为该命令在内存中分配的区域。当该命令完成时，硬盘会将该命令的标识号也发给主机。标识号的使用使得主机可以避免其内存空间被硬盘直接操作。同时，硬盘在命令执行完成后会及时向主机发送命令的执行状态及错误信息。

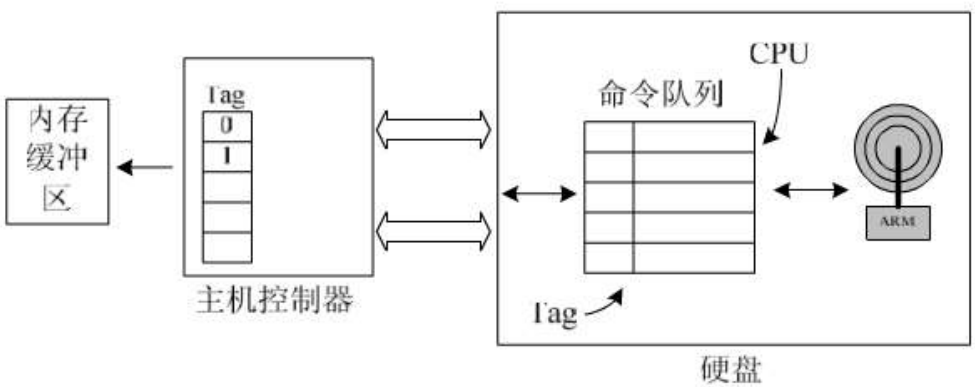

图 1-9 主机控制器与硬盘通信示意图

SATA 原生命令队列（NCQ）是为了更有效的使用 SATA 协议和队列命令数据传输的流水特性而设计的一种队列协议。原生命令队列协议对 SATA 协议没有任何改变，该命令协议仅使用 SATA1.0 协议的部分帧和原语。它有两种命令 Read DMA Queued 和 Write DMA Queued。

SATA 接口要使用原生命令队列命令，需要满足以下条件：

1. 为了使能 First Party DMA 命令和高效的完成队列命令，SATA 主机控制器必须支持 DMA Setuo FIS 和 Set Device bits FIS 的传输。

2. 软件驱动必须支持 Read DMA Queued 和 Write DMA Queued 命令的传输。

3. 硬盘必须支持 SATA2.0 原生命令队列传输协议。2003 下半年以后生产的硬盘大部分支持原生命令队列传输。

主机控制器端罗列的 SATA 接口所连接的硬盘中可能有部分不支持 SATA2.0 原生命令队列协议，所以软件驱动应该能够检测出那个接口上连接的硬盘支持 NCQ，在发送 Read DMA Queued 和 Write DMA Queued 命令时，只能发送到支持 NCQ 的端口上。这样在同一个硬盘存储子系统中既可以有支持 NCQ 的硬盘，也可以有不支持 NCQ 的硬盘。硬盘在其 Idetify Device 页中给出是否支持 NCQ 的标

示。Read DMA Queued 和 Write DMA Queued 作为 ATA/6 标准命令的补充。

原生命令队列（NCQ）最多可支持 32 个命令同时传输，命令队列深度可以通过主机端驱动或 RAID 控制器设置。

SATA 原生命令队列协议提供了高效的数据流传输以及状态反馈机制。NCQ 的这种高效性能的提升得益于 SATA 原生命令队列协议中的状态返回竞争机制、中断聚合以及 First Party DMA 等特性。

Read FPDMA Queued 是 First Party DMA 队列命令的一种读命令，此命令只支持 48 位 LBA 地址模式。表 2-4 是 Read FPDMA Queued 命令的结构，其中主机软件在扇区计数寄存器部分填充的是命令标识号，Reserved 部分填充 0。设备通过发送 Set Device Bits FIS 更新主机队列命令的完成状态，该帧会产生中断，从而主机软件可以执行中断服务程序完成命令的执行。主机驱动软件会查询 SActive 寄存器来确认命令是否执行完毕。

表 1-4 Read FPDMA Queued 命令结构

寄存器	7	6	5	4	3	2	1	0
特征	扇区数目 7:0							
特征（EXP）	扇区数目 15:8							
扇区数目	Tag					Reserved		
扇区数目（EXP）	Reserved							
扇区号	LBA7:0							
扇区号（EXP）	LBA31:24							
柱面低字节	LBA15:8							
柱面低字节（EXP）	LBA39:32							
柱面高字节	LBA23:16							
柱面高字节（EXP）	LBA47:40							
设备/磁头	FUA	1	Res	0	Reserved			
命令	60h							

Write FPDMA Queued 是 First Party DMA 队列命令读命令的一种，其命令组成结构，以及命令执行状态的反馈和执行过程类似于 Read FPDMA Queued。

下面描述了多个 SATA 原生对列命令执行的机制，例子表明，各个命令执行完成的顺序与命令发布的顺序是不一致的。

- 主机发送一个标识号为 0 的对列读命令

（1）首先将主机端 SActive 寄存器中位 0 置 1。

（2）主机发送标识号为 0 的对列读命令。

（3）该读命令通过 Register H2D 帧发送给设备，同时主机控制器的状态寄存器中 BSY 位置 1，表示没有新的命令要发送。

（4）设备接收到命令后，清 BSY 位并发送 Register D2H 帧给主机控制器。

- 主机发送一个标识号为 5 的对列读命令

（1）首先将主机端 SActive 寄存器中位 5 置 1。

（2）主机发送标识号为 5 的对列读命令。

（3）该读命令通过 Register H2D 帧发送给设备，同时主机控制器的状态寄存器中 BSY 位置 1，表示没有新的命令要发送。

（4）设备接收到命令后，清 BSY 位并发送 Register D2H 帧给主机控制器。

（5）设备发送针对标识号为 5 的命令发送 DMA Setup FIS。

- 标识号为 5 的对列命令的数据传输

（1）将标识号 5 对应的 PRD 指针下载到主机控制器中 DMA 引擎中。

（2）设备发送标识号 5 命令对应的数据。

（3）主机控制器 DMA 引擎将接收到的数据导入主机中标识号 5 所对应内存区域。

（4）设备发送 Set Device Bits 帧，并将其 I 位和 SActive 位 5 置 1 以表示对应标识号 5 的命令已经被执行完成。

（5）主机控制器将 SActive 寄存器中位 5 清零，并产生中断。

- 标识号为 0 的对列命令的数据传输

（1）设备发送针对标识号为 0 的命令发送 DMA Setup FIS。

（2）将标识号 0 对应的 PRD 指针下载到主机控制器中 DMA 引擎中。

（3）设备发送标识号 0 命令对应的数据。

（3）主机控制器 DMA 引擎将接收到的数据导入主机中标识号 0 所对应内存区域。

（4）设备发送 Set Device Bits 帧，并将其 I 位和 SActive 位 0 置 1 以表示对应标识号 0 的命令已经被执行完成。

（5）主机控制器将 SActive 寄存器中位 0 清零，并产生中断。

SATA 原生命令队列协议允许同时进行多达 32 个命令发送，极大优化了主机与设备间的数据传输。设备通过对命令进行重新排序，达到了最大的数据吞吐率。正如所讨论，SATA 原生命令队列协议在流水机制上比 PATA 队列协议做的要好，

利用硬件逻辑自动建立 DMA 传输所需的内存空间，最小化数据传输所需的中断数目。总之，SATA 原生命令队列协议所采用智能数据处理机制将满足诸如下一代接入服务器、网络存储、高性能 PC 机等领域对高性能数据传输的需求。

1.2 SAPIS 接口标准规范

SAPIS(SATA PHY 接口规范)定义了一组 SATA PHY 与链路层之间通信的标准接口，旨在降低 SATA 物理层与链路层设计之间的耦合度。在系统设计时只要遵循规范进行接口设计便可保证不同层次之间通信的正确性。图 2-10 是 SAPIS 接口功能框图。

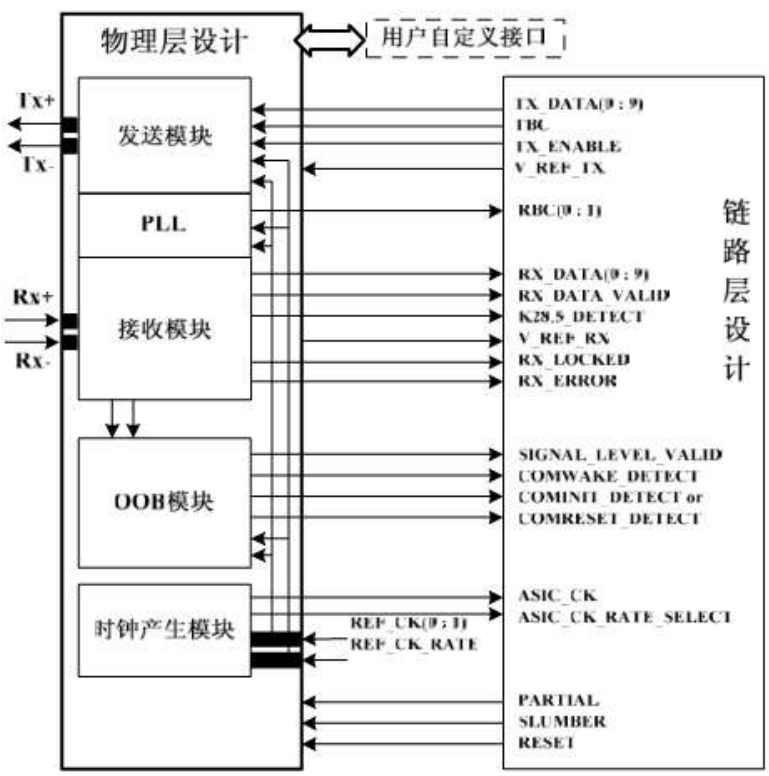

图 1-10　SAPIS 接口功能框图

表 2-5 描述了 SAPIS 标准接口信号，其中输入输出是相对应 PHY 一端来说的。在 SATA2.0 加解密系统物理层设计时充分参考了 SAPIS 标准接口设计，同时又切实考虑实际设计需求，因此对表 2-5 中的接口信号进行精简和必要的修改。

表 1-5 SAPIS 标准接口信号

接口名	接口方向	接口描述
TX_DATA(0 : 9)	输入	PHY 发送端接收到链路层传过来的数据
TBC	输入	发送端数据传输同步时钟
TX_ENABLE	输入	使能 SATA 发送端链路
V_REF_TX	输入	发送端信号参考电平
RBC(0 : 1)	输出	用于同步 PHY 到链路层数据传输的时钟
RX_DATA(0 : 9)	输出	PHY 接收端发送给链路层的数据
RX_DATA_VALID	输出	RX_DATA 数据有效指示
K28.5_DETECT	输出	RX_DATA 是 K28.5 控制字符时拉高
V_REF_RX	输出	接收端信号参考电平
RX_LOCKED	输出	接收端接收到对齐的非 OOB 信号锁定时置高
RX_ERROR	输出	数据接收错误指示
SIGNAL_LEVEL_VALID	输出	用于唤醒链路层信号或处于挂起的时钟信号
COMWAKE_DETECT	输出	检测到 COMWAKE 信号
COMINIT_DETECT	输出	检测到 OMINIT 信号
COMRESET_DETECT	输出	检测到 COMRESET 信号
ASIC_CK	输出	链路层模块时钟
ASIC_CK_RATE_SELECT	输出	用于选择链路层采用的时钟频率
PARTIAL	输出	用于控制 PHY 进入低功耗状态的信号
SLUMBER	输出	用于控制 PHY 进入休眠状态信号
RESET	输出	复位 PHY 内部模块到初始化状态

注：上表所列的接口信号包括数据接口、流控接口、指示接口、时钟接口，在实际设计中接口定义应视具体情况而定。不必严格按照上表中所列的信号定义。