

# NVMe Better File System

中期报告

# 目 录

- 什么是NVMe? 什么是文件系统?
- 文件系统为什么需要针对NVMe优化?
- 怎么做?

# 什么是NVMe?

# 什么是文件系统?

1. 什么是NVMe?
2. NVMe 队列管理
3. 文件系统I/O队列机制

什么是NVMe?

# HDD与SSD



什么是NVMe?


# HDD与SSD



什么是NVMe?

# SATA与NVMe

**SAMSUNG**



读取550 MB/s 写入520 MB/s

M.2 接口

企业购更优惠

三星 (SAMSUNG) 500GB SSD固态硬盘 M.2接口(SATA总线) 860 EVO (MZ-N6E500BW)

【存储品牌日特惠直降】升级优选！品牌铸造品质，智能兼容，性能稳定！5年质保！活动查看>

京东价 **¥ 519.00** 降价通知

累计评价  
50万+

增值业务  
配送至

重量 0

选择颜色

选择版本

增值保障

白条分期

**SAMSUNG**



SSD 980  
NVMe M.2

Read Speed 3500MB/s

企业购更优惠

❤ 关注 < 分享 1 对比

举报

新品 三星 (SAMSUNG) 500GB SSD固态硬盘 M.2接口(NVMe协议) 980 (MZ-V8V500BW)

980超级新品，兼具速度与可靠性！DRAM-less消费级固态硬盘！读速高达3500MB/s，全功率模式！活动点击！查看>

京东价 **¥ 519.00** 降价通知

累计评价  
100万+

增值业务 **高价回收，极速到账**

配送至 安徽省合肥市包河区包公街道 有货

**京东物流** 京准达 | 211限时达 | 预约送货

由 京东 发货，供应商提供售后服务。23:54前下单，预计明天(04月24日)送达

重量 0.08kg

服务支持 **放心购** 自助装机

可配送海外99元免基础运费

选择颜色

970 EVO Plus | 探索NVMe无限潜力

新品870EVO | 性能升级款SATA

860 PRO | 守候MLC颗粒的孤傲

新品980 | NVMe PCIe3.0\*4

新品980 PRO | NVMe PCIe 4.

860 EVO | M.2接口 (SATA总线)

970 EVO | 在性能的路上探索进取

970 PRO | 惊艳性能只愿为MLC绽放

新品870 QVO | 让TB容量普及

860 EVO | 稳定/兼容让信赖变依赖

选择版本

【250G/256G】

【500G/512G】

【1TB】

【2TB】

【4TB】

【8TB】

增值保障

1年全保换新 ¥39.00

2年升级换新 ¥39.00

标准版安装 ¥198.00



什么是NVMe？

# SATA与NVMe

商品介绍	规格与包装	售后保障	商品评价(50万+)	商品问
主体	型号	MZ-N6E500BW		
	系列	860 EVO 系列		
	品牌	三星(SAMSUNG)		
规格	顺序写入	最大520 MB/s		
	缓存	512M		
	闪存类型	TLC		
	顺序读速	最大550 MB/s		
特性	TBW	300TBW		
	工作温度	0 - 70 °C		
	产品尺寸 (mm)	Max. 80.15 x Max. 22.15 x Max.2.38 (mm)		
	保存温度	-45°C to 85°C		

包装清单 硬盘 × 1 说明书 × 1, 包装彩盒\*1

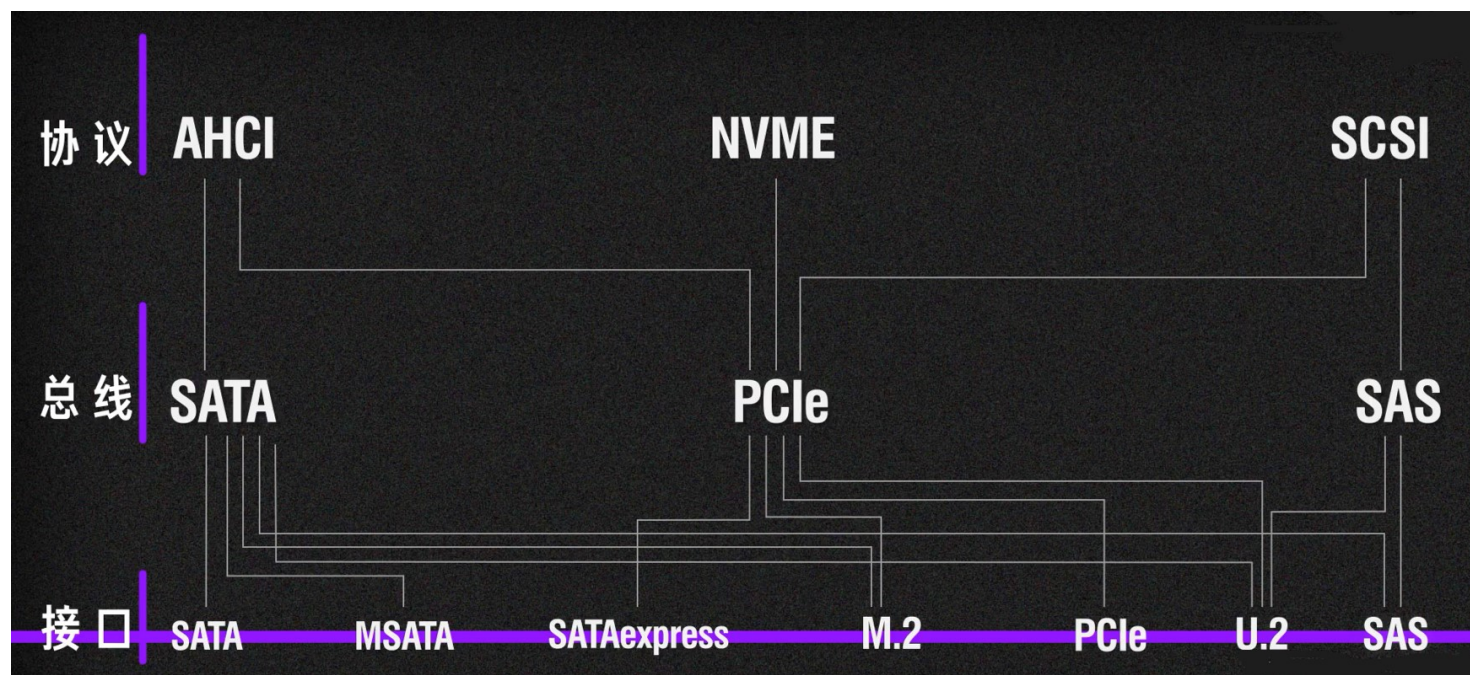
商品介绍	规格与包装	售后保障	商品评价(100万+)	商
主体	型号	MZ-V8V500BW		
	系列	980		
	品牌	三星(SAMSUNG)		
规格	顺序写入	高达2600 MB/s		
	缓存	无缓存		
	闪存类型	TLC		
	顺序读速	高达3100 MB/s		
特性	TBW	300		
	工作温度	0 - 70 °C		
	产品尺寸 (mm)	80.15 x 22.15 x 2.38 (mm)		
	保存温度	-40~85°C		

包装清单 硬盘 × 1, 说明书 × 1, 包装彩盒×1



什么是NVMe?

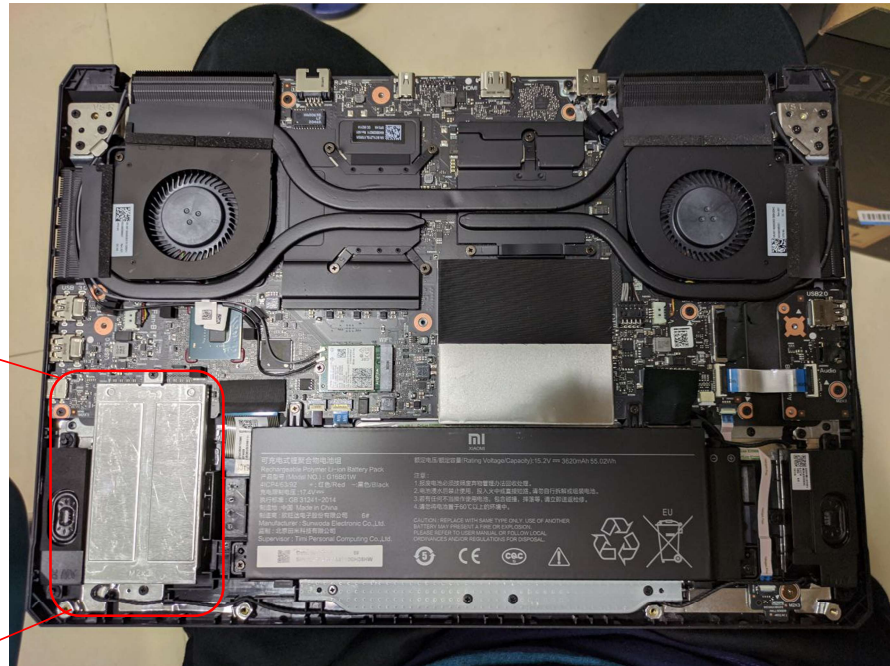
# 协议、总线、接口





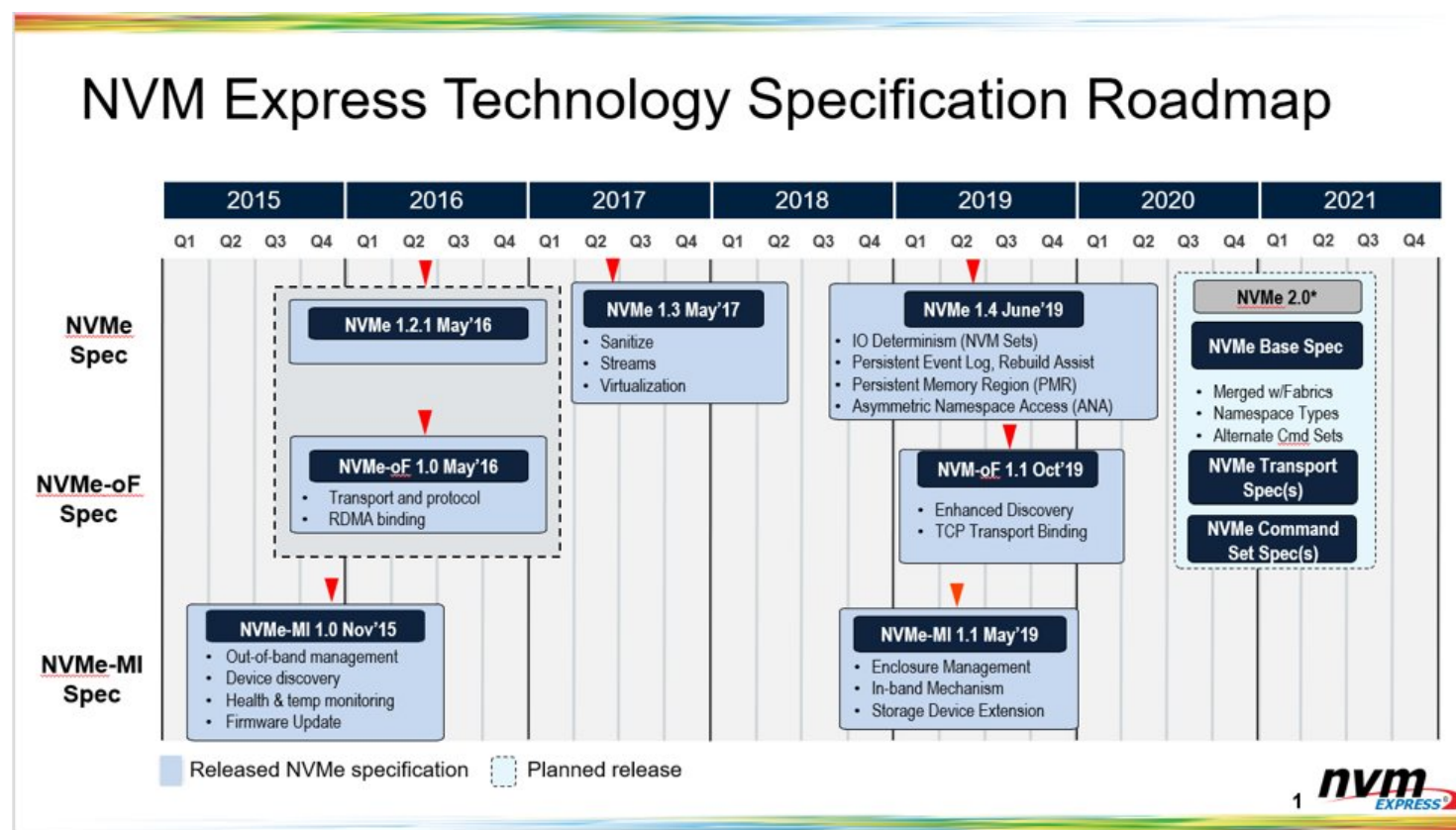
什么是NVMe?

# NVMe



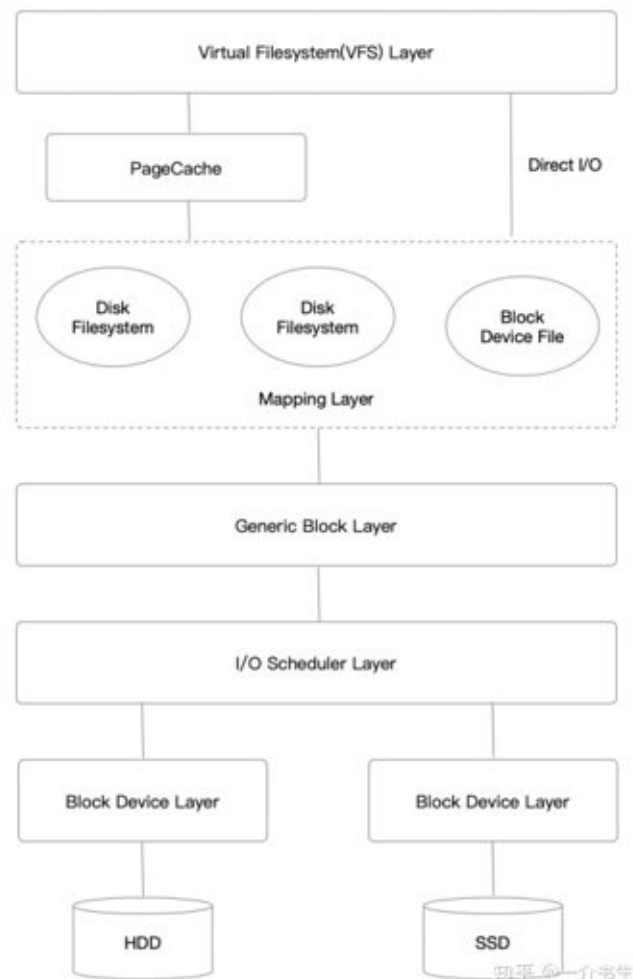
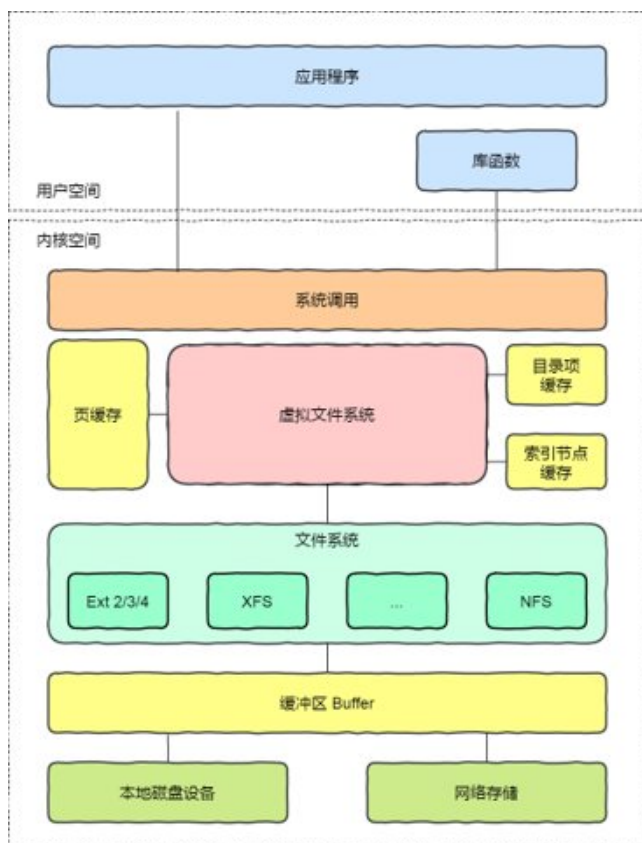
什么是NVMe?

# NVMe



什么是文件系统？

# 文件系统 I/O 队列机制




# NVMe定义的命令

Admin Commands
Create I/O Submission Queue
Delete I/O Submission Queue
Create I/O Completion Queue
Delete I/O Completion Queue
Get Log Page
Identify
Abort
Set Features
Get Features
Asynchronous Event Request
<i>Firmware Activate (optional)</i>
<i>Firmware Image Download (optional)</i>
<i>Format NVM (optional)</i>
<i>Security Send (optional)</i>
<i>Security Receive (optional)</i>

NVM I/O Commands
Read
Write
Flush
<i>Write Uncorrectable (optional)</i>
<i>Compare (optional)</i>
<i>Dataset Management (optional)</i>
<i>Write Zeros (optional)</i>
<i>Reservation Register (optional)</i>
<i>Reservation Report (optional)</i>
<i>Reservation Acquire (optional)</i>
<i>Reservation Release (optional)</i>

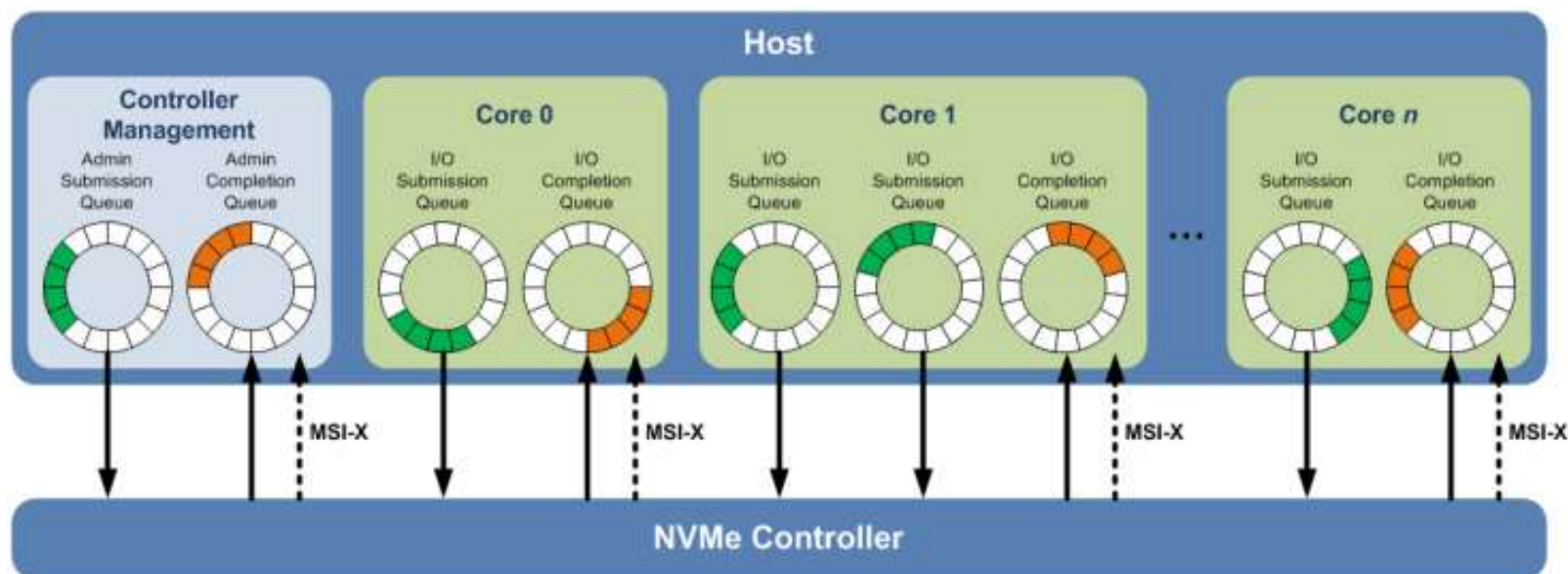


# NVMe特性

	AHCI	
<b>Uncacheable Register Reads</b> Each consumes 2000 CPU cycles	4 per command 8000 cycles, ~ 2.5 $\mu$ s	0 per command
<b>MSI-X and Interrupt Steering</b> Ensures one core not IOPs bottleneck	No	Yes
<b>Parallelism &amp; Multiple Threads</b> Ensures one core not IOPs bottleneck	Requires synchronization lock to issue command	No locking, doorbell register per Queue
<b>Maximum Queue Depth</b> Ensures one core not IOPs bottleneck	1 Queue 32 Commands per Q	64K Queues 64K Commands per Q
<b>Efficiency for 4KB Commands</b> 4KB critical in Client and Enterprise	Command parameters require two serialized host DRAM fetches	Command parameters in one 64B fetch

NVMe的队列管理

## Admin SQ/CQ 和 IO SQ/CQ



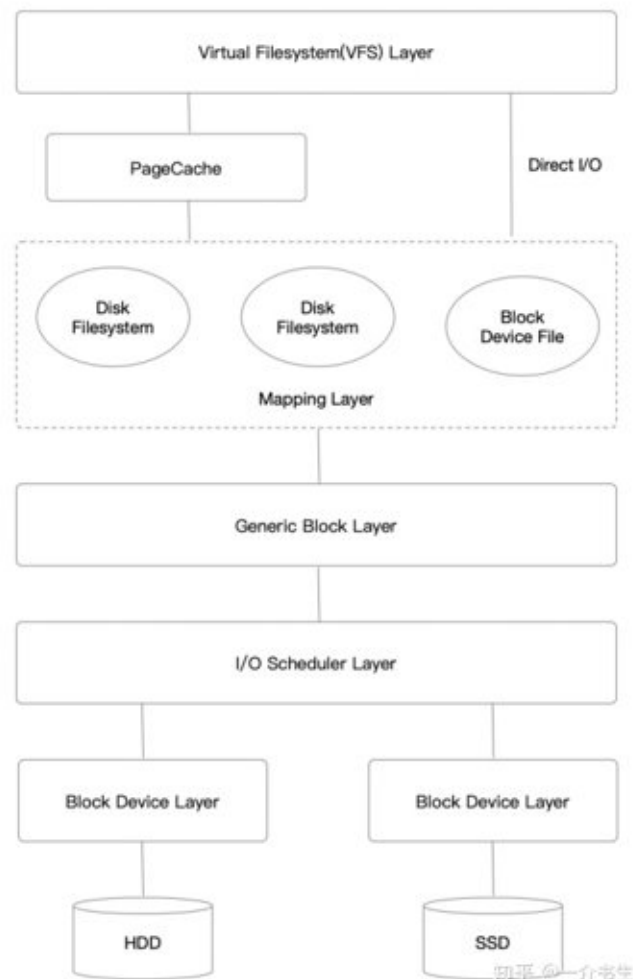
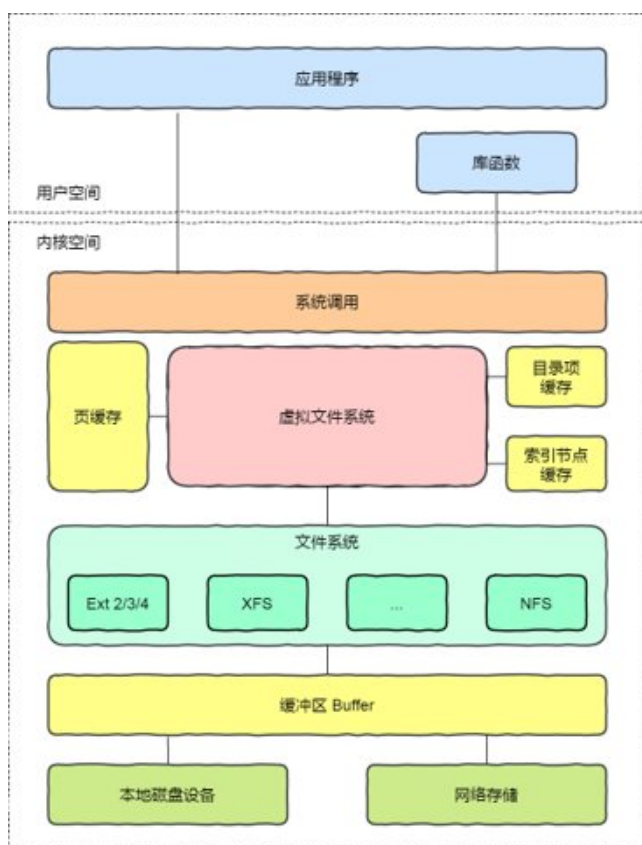
# 为什么需要针对NVMe优化？

当前文件系统不足



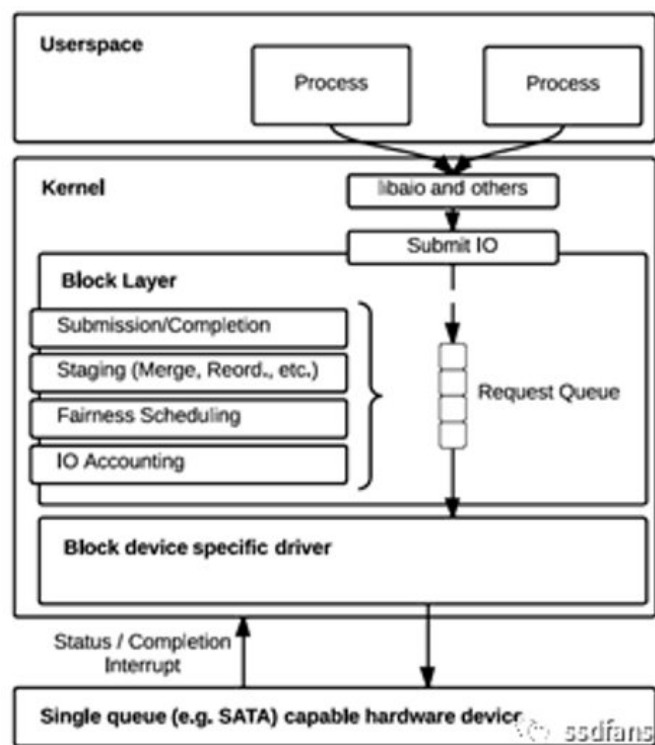
为什么需要针对NVMe优化?

# 文件系统I/O队列机制

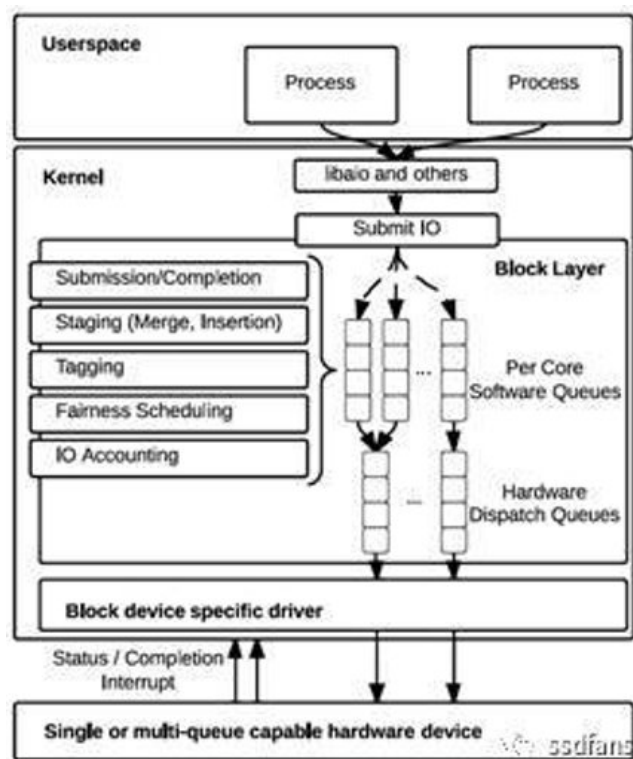


为什么需要针对NVMe优化?

## 文件系统I/O队列机制



单队列机制

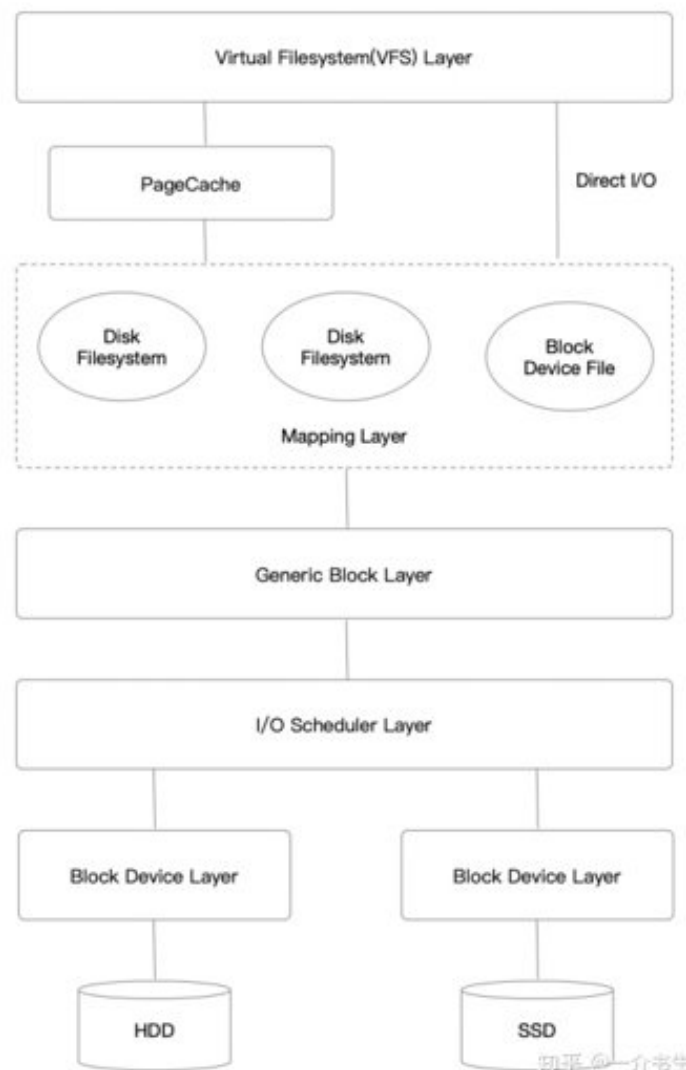


多队列机制

为什么要做针对NVMe优化的文件系统？

## 当前文件系统不足

1. 中断
2. 长I/O
3. 落后的“优化”
4. 队列

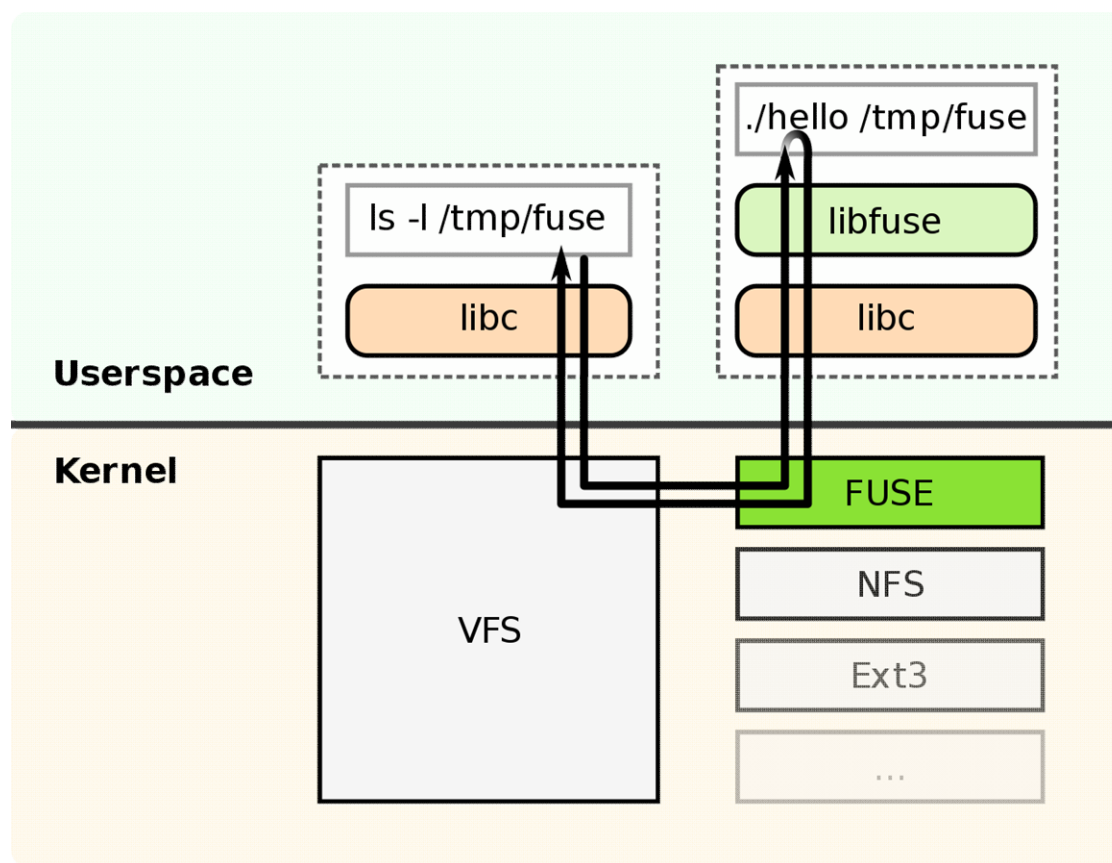


# 怎么做？

1. 用户态文件系统
2. SPDK
3. 测试

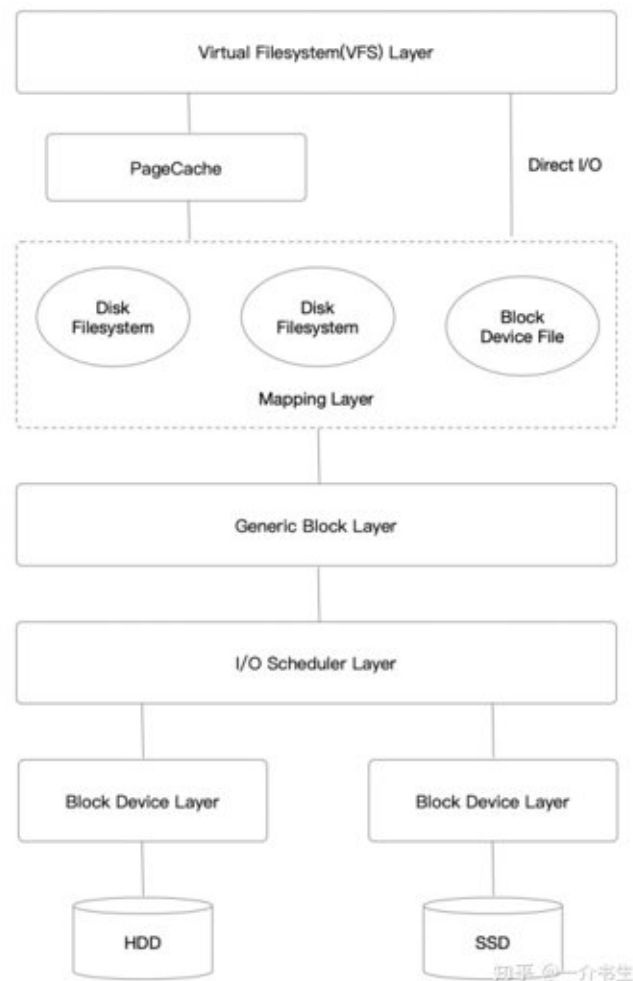
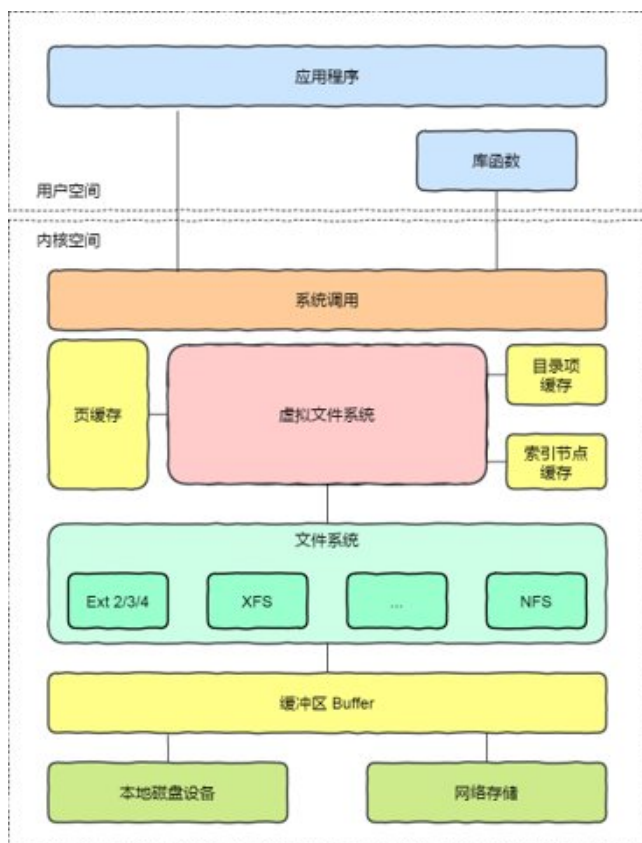
怎么做?

# FUSE (Filesystem in Userspace)



怎么做?

# 文件系统I/O队列机制



怎么做？

# 使用SPDK搭建用户态文件系统



- 用户态
- 中断->轮询



怎么做？

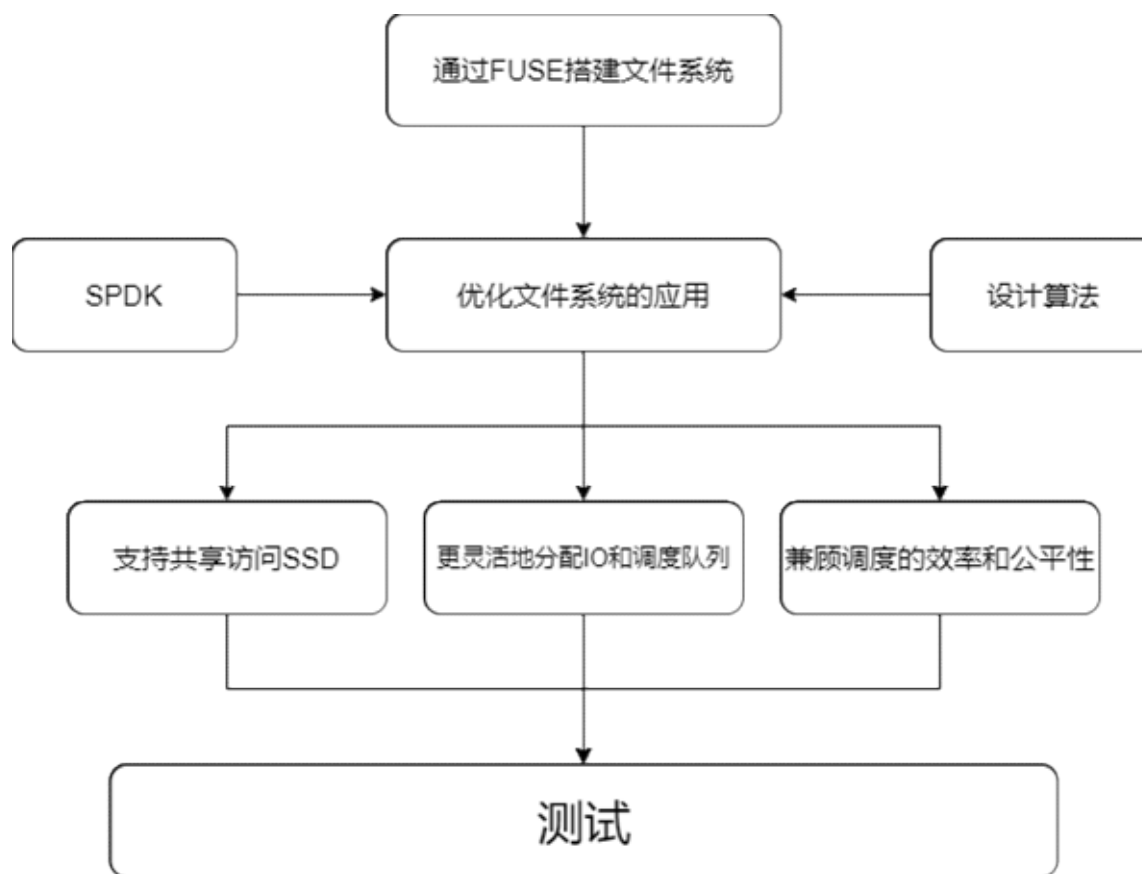
# 测试

以下是一些文件系统测试工具：

1. **pjd-fstest (posix 接口兼容性测试)**: fstest 是一套简化版的文件系统 POSIX 兼容性测试套件，它可以工作在 FreeBSD, Solaris, Linux 上用于测试 UFS, ZFS, ext3, XFS 和 NTFS-3G 等文件系统。fstest 目前有3601个回归测试用例，测试的系统调用覆盖 chmod, chown, link, mkdir, mkfifo, open, rename, rmdir, symlink, truncate, unlink。
2. **IOZone (读写模式测试)**: IOZone 是目前应用非常广泛的文件系统测试标准工具，它能够产生并测量各种的操作性能，包括 read, write, re-read, re-write, read backwards, read strided, fread, fwrite, random read, pread, mmap, aio\_read, aio\_write 等操作。IOZone 目前已经被移植到各种体系结构计算机和操作系统上，广泛用于文件系统性能测试、分析与评估的标准工具。
3. **FIO (顺序、随机IO测试)**: flexible I/O tester。FIO 可以模拟给定的IO工作负载而无需编写量身定制的测试案例。它支持13种不同类型的 I/O引擎 ( sync, mmap, libaio, posixaio, SG v3, splice, null, network, syslet, guasi, solarisaio 等)，I/O priorities(for newer Linux kernels), rate I/O, forked or threaded jobs 等等。fio 可以支持块设备和文件系统测试，广泛用于标准测试、QA、验证测试等，支持 Linux, FreeBSD, NetBSD, OS X, OpenSolaris, AIX, HP-UX, Windows 等操作系统。
4. **Filebench (文件系统应用负载生成测试)** Filebench 是一款文件系统性能的自动化测试工具，它通过快速模拟真实应用服务器的负载来测试文件系统的性能。它不仅可以仿真文件系统微操作 (如 copyfiles, createfiles, randomread, randomwrite)，而且可以仿真复杂的应用程序 (如 varmail, fileserver, oltp, dss, webserver, webproxy)。Filebench 比较适合用来测试文件服务器性能，但同时也是一款负载自动生成工具，也可用于文件系统的性能。
5. **IOR/mdtest (利用并行IO来测试文件系统的IO性能和元数据性能)** IOR 是并行IO基准，可用于使用各种接口和访问模式来测试并行存储系统的性能。IOR 存储库还包括 mdtest 基准测试，该基准测试专门测试不同目录结构下存储系统的峰值元数据速率。这两个基准测试均使用通用的并行I/O抽象后端，并依赖 MPI 进行同步。
6. **dd-benchmark** dd 是 linux 内核程序，但可以用其测试文件系统的各种性能 (因此不是自动测试程序，需要自己定义测试内容)

## Summary

# 实现路线



# 谢谢！

## 祝大家劳动节快乐！