Operating Systems

Permanent Storage

Me

November 17, 2016

Блочные устройства

- Блочные устройства - устройсва с поблочным доступом
  - общение с устройством идет порциями кратными некоторому фиксированному размеру - сектору
  - типичный размер сектора - 512 байт;
  - внутри устройство может работать с блоками произвольного размера, но интерфейс зачастую в 512 байтных блоках.
- Типичные примеры блочных устройств:
  - жесткие диски (HDD) и твердотельные накопители (SSD);
  - оптические диски (CD, DVD, etc);
  - даже USB Flash накопители...

Механические диски (HDD)

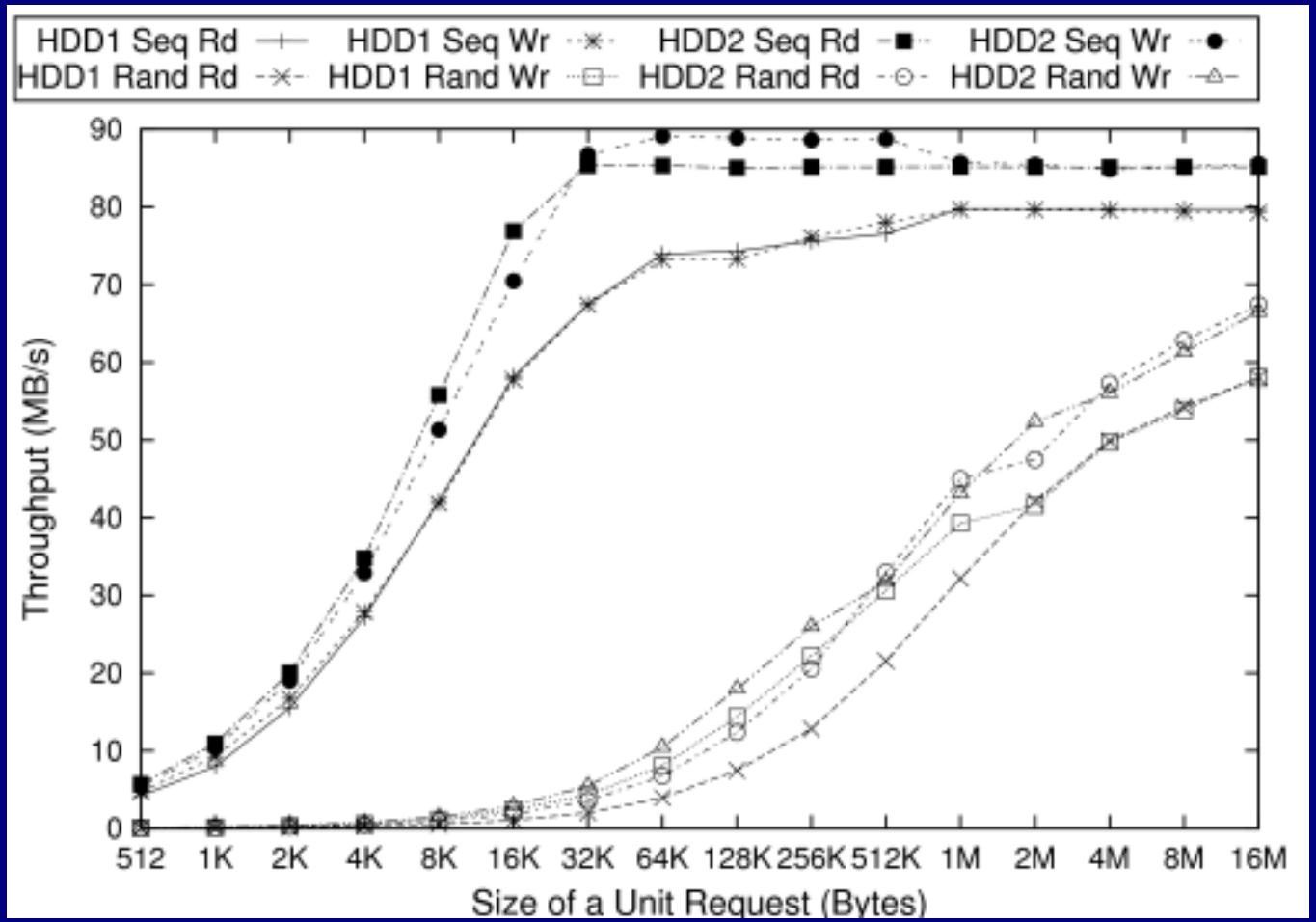
Три основных части:

- вращающиеся диски (platters);
- подвижная рука (arm);
- читающие/пишушие головы (heads);

Скорость HDD

- Скорость вращения дисков HDD:
  - при скорости 7200 оборотов в минуту - один оборот 8-9 мс;
  - чтобы записать/прочитать данные нужно поставить голову над/под нужным цилиндром и подождать, пока нужное место диска "доедет" до головы.
- Скорость позиционирования читающей/пишушей головы:
  - время определяется опять же миллисекундами.
- Скорость работы диска доминируется поиском:
  - скорость вращения диска + позиционирование головы;
  - random IO гораздо медленнее sequential IO.

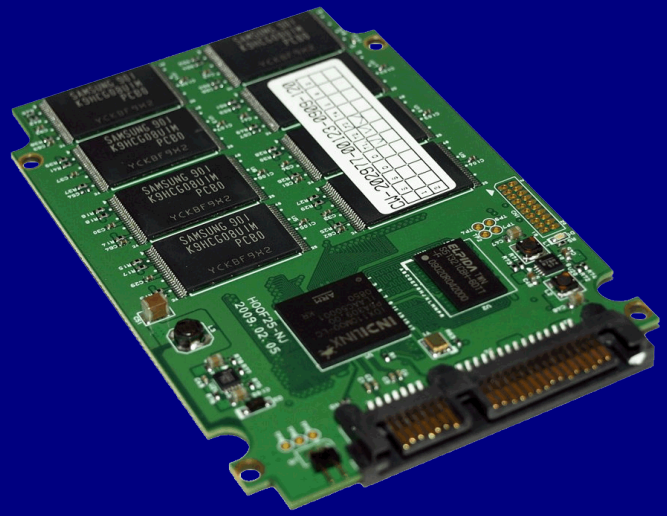
Скорость HDD



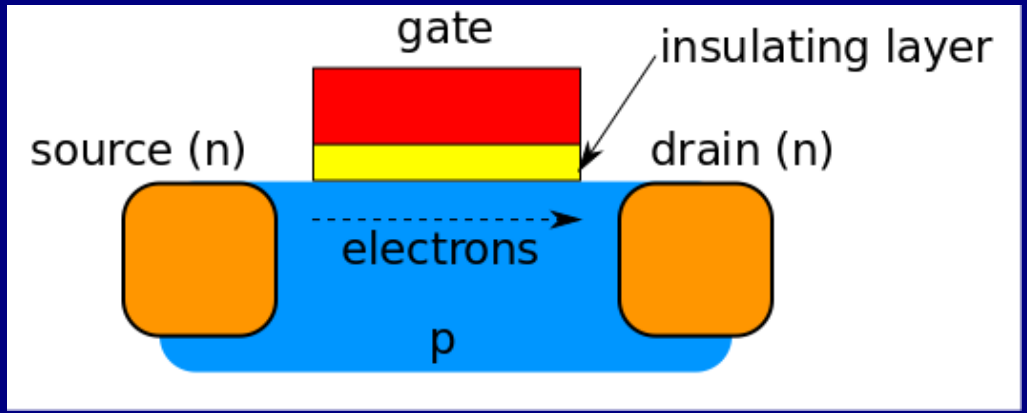
Твердотельные накопители (SSD)

Нет подвижных механических частей:

- нет замедления;
- хорошо переживают тряску.

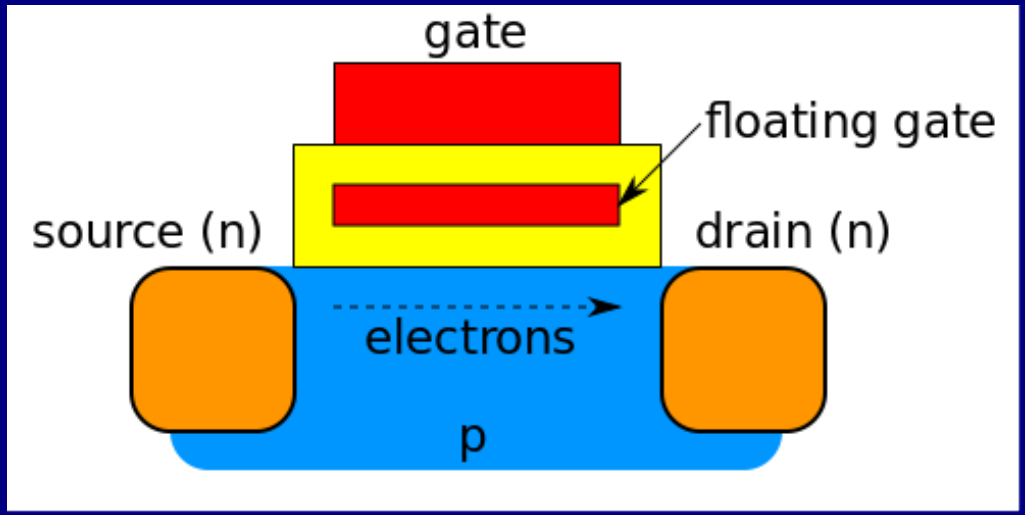


MOSFET



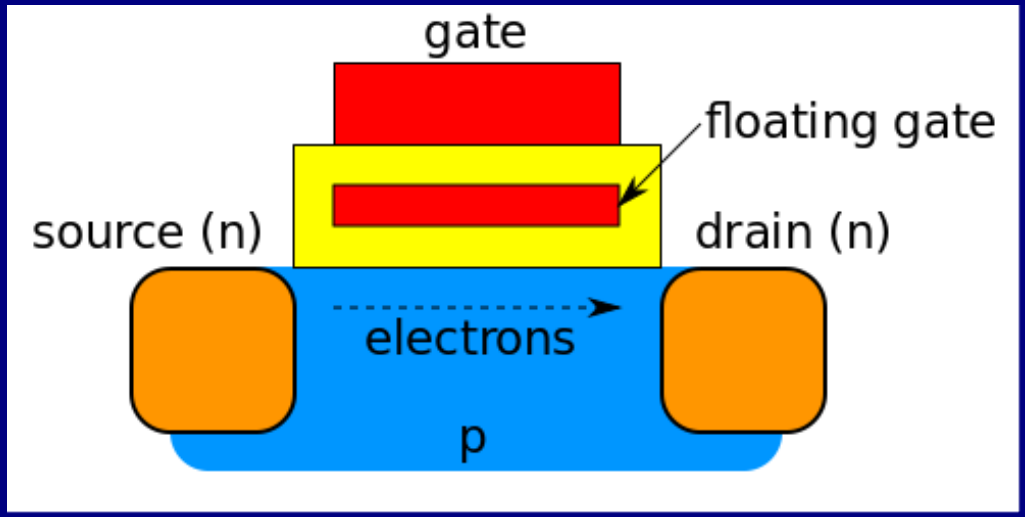
- В обычных условиях транзистор "заперт":
  - даже если между source и drain есть напряжение, тока все равно нет.
- Напряжение между source и gate "открывает" транзистор:
  - если между source и drain есть напряжение будет и ток.

FGMOS



- MOSFET + дополнительный floating gate:
  - floating gate изолирован от всего, но есть способ его зарядить;
  - floating gate позволяет "нейтрализовать" эффект gate-a.
- FGMOS может хранить один бит информации:
  - если floating gate заряжен - с FGMOS мы читаем 0;
  - если разряжен - читаем 1.

Изнашивание FGMOS



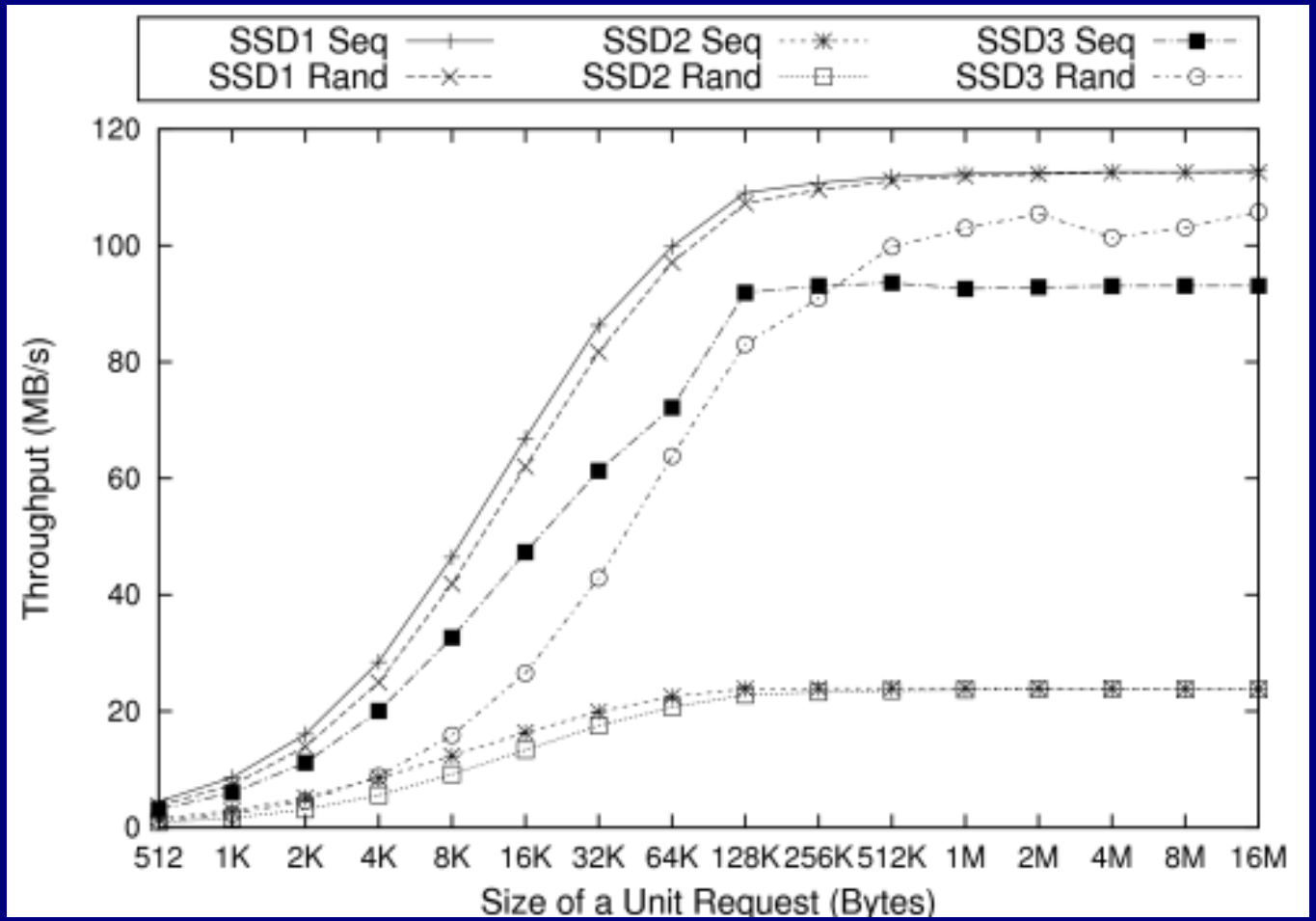
- Изолятор вокруг floating gate изнашивается:
  - чем больше раз запрограмировали/стерли бит хранимый в FGMOS тем сильнее изнашивается изолятор;
  - ячейка памяти с изношеным изолятором становится ненадежной.

NAND Array

- FGMOS-ы группируются вместе, чтобы их можно было разместить максимально плотно:
  - минимальная единица адресации чтения - страница (типично 8-16 Kb);
  - несколько страниц объединяются в блок - минимальная единица записи (типично 32-256 страниц).
- При этом SSD диск выглядит как обычный диск, т. е. интерфейс использует 512 байтные блоки:
  - SSD firmware скрывает от нас параметры NAND Array;
  - firmware следит за тем, чтобы страницы изнашивались равномерно;
  - для этого внутри SSD работает трансляция адресов + GC + компрессия + дедупликация + много других страшных слов.

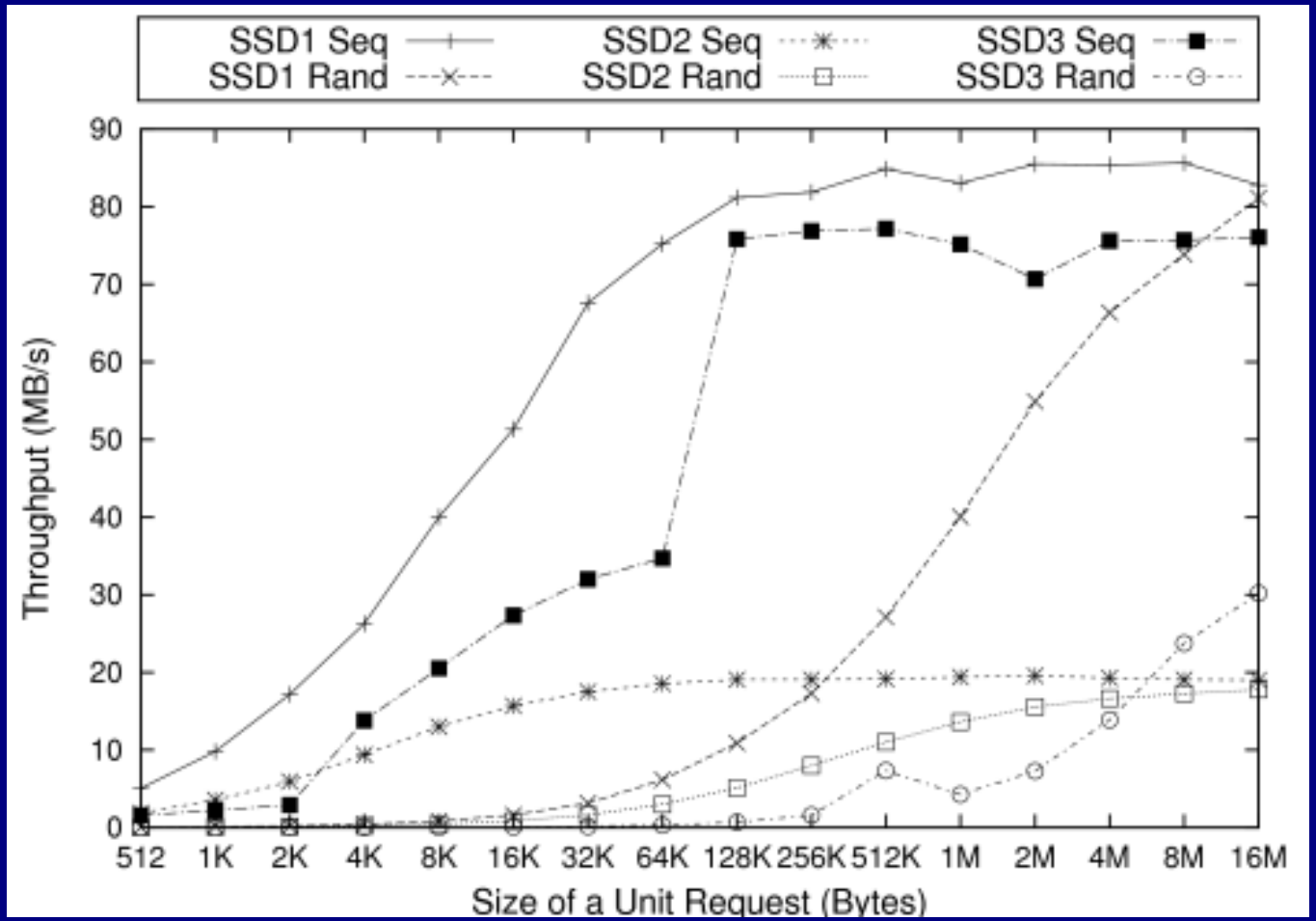
Скорость SSD

Чтение



Скорость SSD

Запись



Интерфейс для работы с дисками

- Современные диски содержат встроенные контроллеры:
  - т. е. вам не нужно заботиться о физической организации диска (на каком диске, в каком цилинде, в каком секторе хранится блок данных и тд.);
  - вам нужно сформировать корректную команду и передать ее диску.
- Есть два более или менее распространенных набора команд:
  - ATA - получил популярность благодаря использованию в PC;
  - SCSI - используется в производительных системах хранения.

IO Scheduler

- Маленькие запросы в случайные места диска приводят к потерям производительности
  - в основном это справедливо для HDD;
  - для SSD это справедливо только для записи.
- Для частичного решения проблемы в ОС не редко присутсвует IO Scheduler:
  - IO Scheduler определяет в каком поряде отдавать запросы диску;
  - IO Scheduler накапливает запросы и пытается "слить" смежные запросы в один большой;
  - т. е. если приложение пишет последовательно, но небольшими порциями, то планировщик может улучшить производительность.

Скорость CPU, памяти и дисков

- Некоторое время назад (80-90-ые года) скорость CPU росла довольно быстро
  - вместе со скоростью CPU росли объемы оперативной памяти и дисков.
- Однако скорость оперативной памяти и дисков не успевала за ростом:
  - разрыв между скоростью памяти и CPU так или иначе заполняется кешами;
  - но между оперативной памятью и дисками просто пропасть.

Массивы независимых (недорогих) дисков

- RAID (Redundant Array of Independent (Inexpenisve) Disks) - объединение из нескольких физических устройств в одно логические устройство:
  - несколько дисков позволяют достигать большего объема;
  - при дублировании данных на несколько дисков мы можем читать данные с любого из них - несколько параллельных запросов повышают производительность;
  - имея копии данных мы можем переживать поломки дисков/порчу данных.

Redundancy is important

- В спецификации дисков зачастую указывают параметры MTTF/MBTF/AFR:
  - MTTF (Mean Time to Failure) - для SSD где-то в районе 2 млн. часов;
  - для массива из 1000 одинаковых дисков получим что-то вроде 2000 часов;
  - чем больше независмых устройств - тем больше вроятность отказа системы.
- Для массива из дисков полезен еще дополнительный параметр MTTR:
  - MTTR (Mean Time To Repair) - время необходимое на обнаружения сбоя диска, его замену и восстановление данных на нем;
  - никакой RAID вас не спасет, если вам нужно 100500 лет чтобы заменить плохой диск.

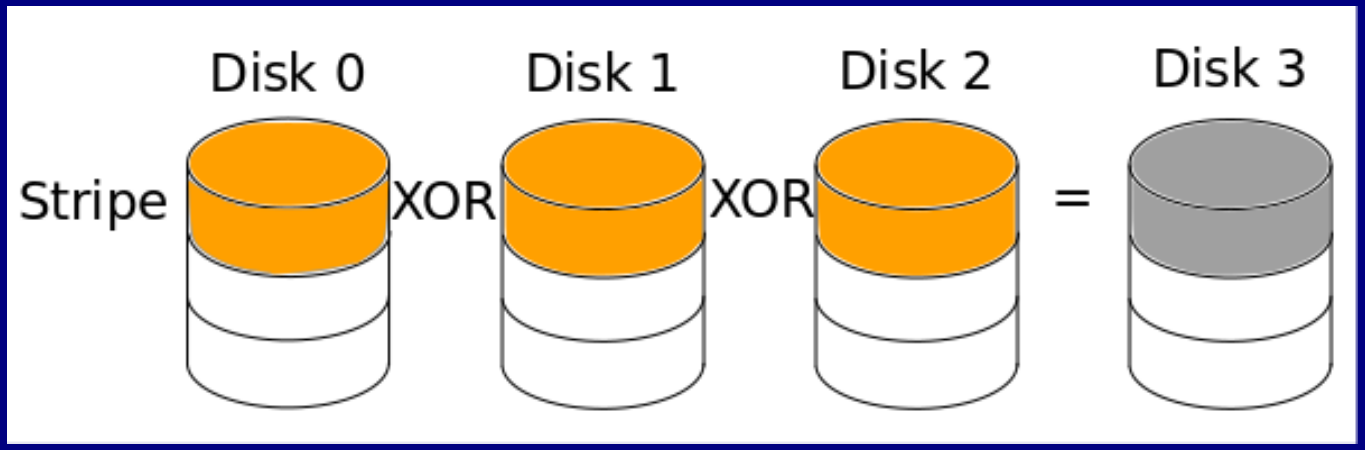
RAID

Mirroring

- Самый очевидный способ повысить надежность - простое дублирование:
  - вы просто пишите каждую порцию данных на 2 (N) дисков параллельно;
  - при таком подходе можно переживать 1 (N-1) отказ диска.
- Минусы:
  - требуется в 2 (N) раза больше пространства;
  - "хвосты" - запись со скоростью самого медленного диска.
- Плюсы:
  - чтение в 2 (N) раза быстрее - любая порция данных может быть прочитана с любого из дисков (если он в строю);
  - т. е. мы можем обрабатывать несколько запросов на чтение параллельно.

RAID

Parity



- Вместо полного дублирования мы можем использовать проверку четности:
  - пусть у нас будет N дисков с данными - на каждый из них пишется своя порция информации;
  - добавим к ним еще один диск - диск четности;
  - диск четности хранит XOR соответствующих бит дисков с данными;

RAID 2

Matrix Form

$$\begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix} \times \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix} = 0$$

- Мы можем записать условие четности в матричном виде:
  - сложение - XOR;
  - умножение - AND;
  - т. е. XOR всех бит данных с битом четности должен давать 0.

RAID

Generalized Parity

$$\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \times \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

- Мы можем проверять четность для некоторого подмножества бит
  - если каждый диск входит хотя бы в одно подмножество;
  - и для каждой пары дисков есть подмножество, в которое входит только один из них;
  - то мы легко можем исправлять сразу две ошибки.

RAID

- В общем случае нужную систему уравнений нам дают коды Хэмминга:
  - для C дисков с битами четности можно использовать до $2^C - 1 - C$ дисков с данными;
  - т. е. всего $2^C - 1$ дисков;
  - и переживать потерю любых двух дисков.
- Недостатки:
  - нужно сравнительно много дисков, чтобы получить какой-то выигрыш по сравнению с Mirroring нужно как минимум 7 дисков;
  - запись происходит со скоростью самого медленного диска.
- Достоинства:
  - дополнительные расходы дискового пространства уменьшаются экспоненциально.

Финальные замечания про RAID-ы

- Мы предполагали, что диски могут выходить из строя целиком
  - если диск вышел из строя целиком, то это легко обнаружить, т. е. мы всегда знаем где ошибка;
  - что если диск все еще работает, но некоторые данные на нем испортились?
  - добавим к каждой порции данных на диске контрольную сумму (хеш).

Финальные замечания про RAID-ы

- Мы рассмотрели возможность исправления 1 или 2 ошибок, при условии, что мы знаем где эти ошибки произошли
  - в общем случае можно построить код позволяющий исправить любое количество ошибок;
  - например, коды Боуза-Чоудхури-Хоквингема в общем и коды Рида-Соломона как частный случай.

Q&A