RAID Introduction

Naming

- RAID:
- first: **Redundant Arrays of Inexpensive Disks**
- modern: **Redundant Arrays of Independent Disks**
- best: **disk array**

Definition

- Disk arrays organize multiple, independent disks into a large, high-performance logical disk

Motivation

- **Capacity boost**
- **Performance boost**
- **Reliability boost**

History

- in the early 1960s, **first disk**, scientists at IBM in San Jose, California
- In **1978. RAID** was first patented by IBM
- In 1988, RAID levels 1 through 5 were formally defined
  - by David A. Patterson, Garth A. Gibson and Randy H. Katz
  - in the paper, "A Case for Redundant Arrays of Inexpensive Disks (RAID)"
  - (http://www-2.cs.cmu.edu/~garth/RAIDpaper/Patterson88.pdf). This was published in the SIGMOD Conference 1988: pp 109–116.
  - The **term "RAID" started with this paper**.

History

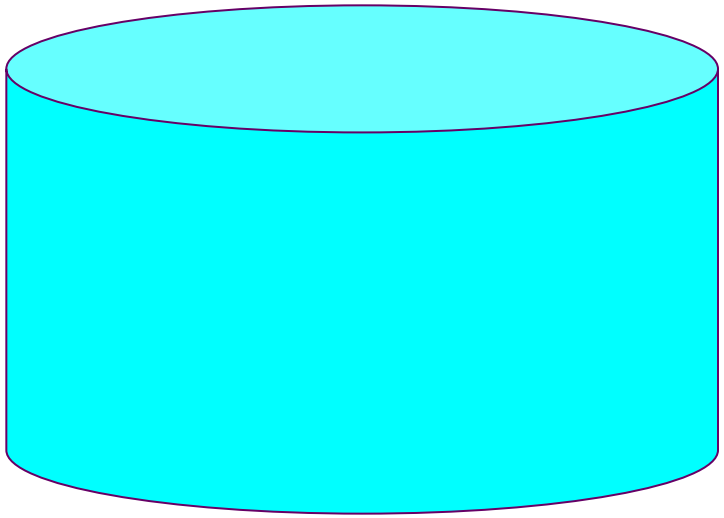
- This RAID prototype in 1992,
- Designed and built by University of Berkeley graduate students.
- Housing 36 320MB disk drives,
- Iits total storage was less than the disk drive in the cheapest PC only six years later.
- *Image courtesy of The Computer Museum History Center,*

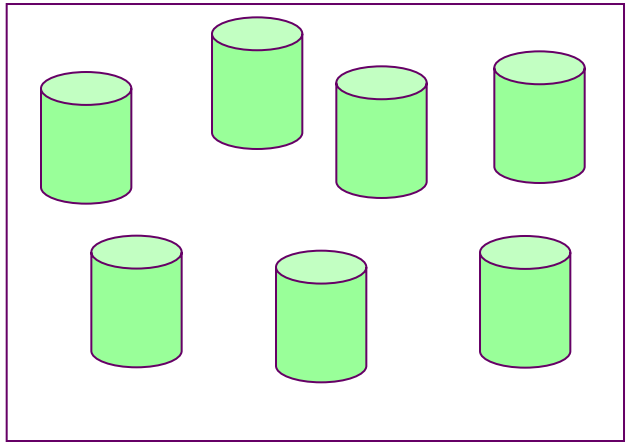
12 GB

36 x 320MB

RAID

- In the beginning
- N disks = as an large disk





- But…No large, too expensive disk
  - ultra large capacity (Terabytes)
  - very high performances
  - very high reliability

RAID

RAID features

- **Performance boost**
  - **parallelism of disk operations for large requests**
    - **few disks work in parallel for one request**
  - **concurrency in time for small requests**
    - **few requests on different disks**
- **Reliability boost**
  - **RAID** – multiple disk drives provides **reliability** via **redundancy**.
  - **Two base schemes**
    - mirroring
    - parity information
- **RAID is arranged into 6 different levels. (7)**
- **Multiply (nested) RAID level**

RAID (cont)

- Several improvements in disk-use techniques involve the use of multiple disks working cooperatively.
- Disk striping uses a group of disks as one storage unit.
- RAID schemes **improve performance** and **improve the reliability** of the storage system by storing redundant data.
  - ***Mirroring*** or *shadowing* keeps duplicate of each disk.
  - ***Block interleaved parity*** uses much less redundancy.

Conventional data placement v stripping

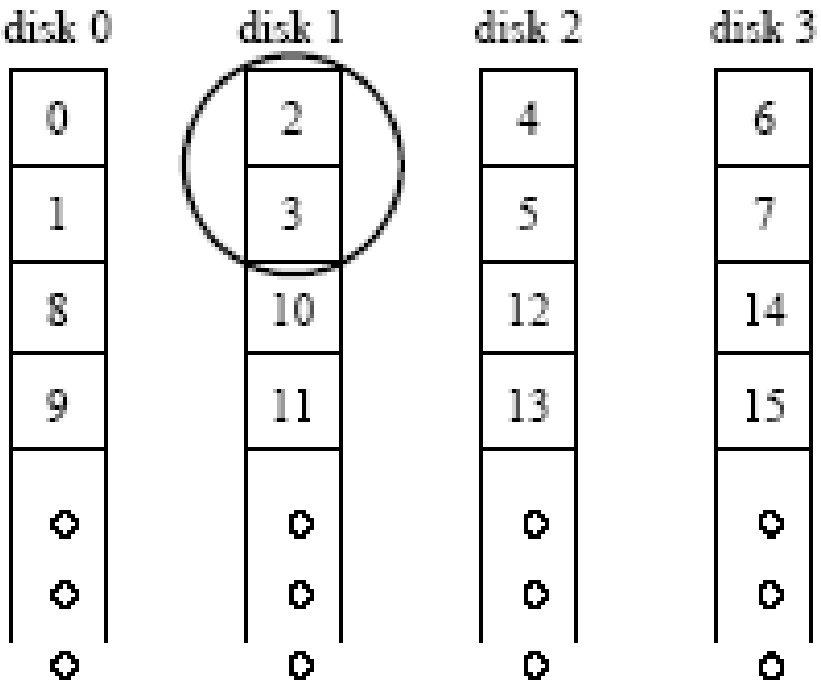
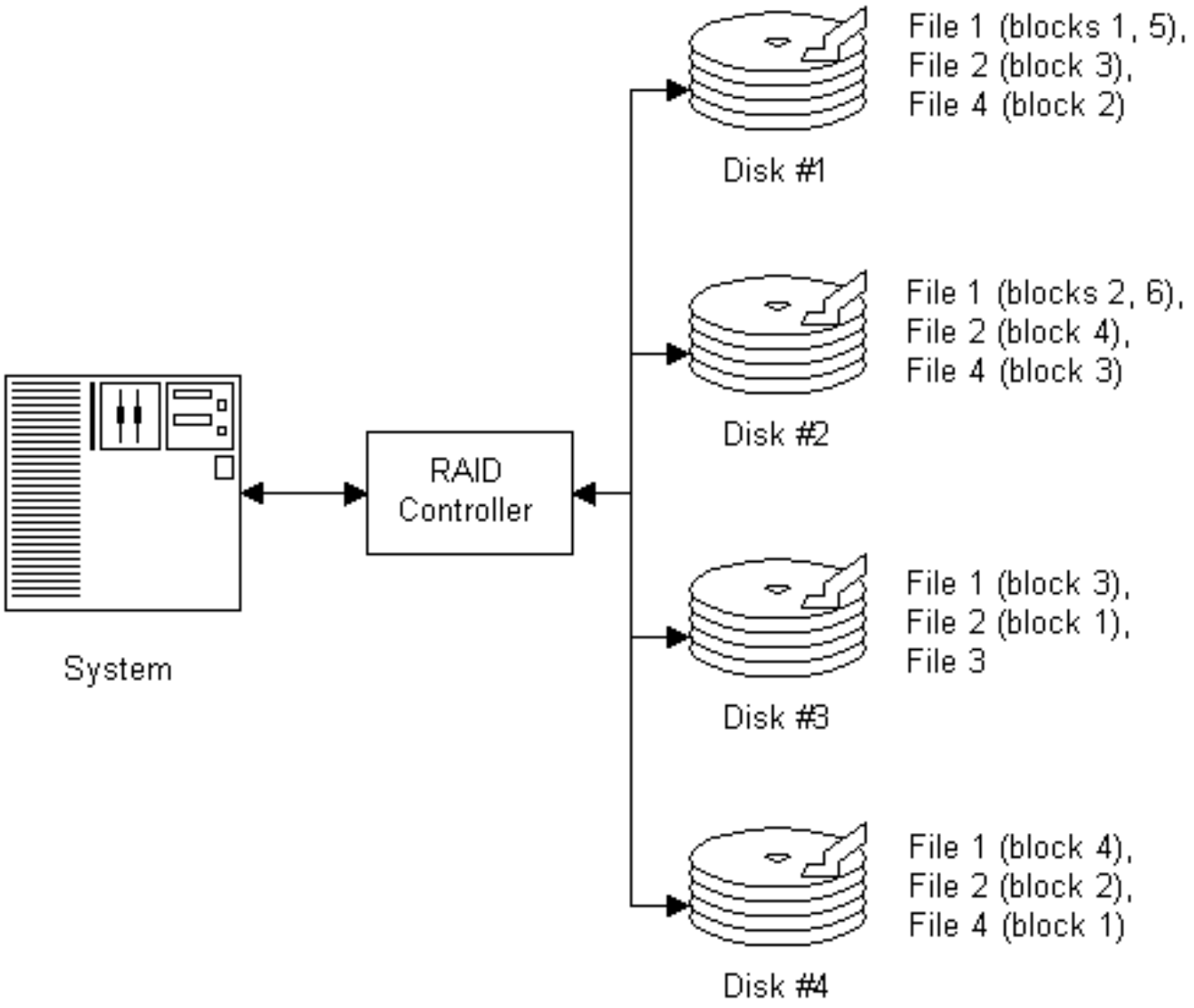


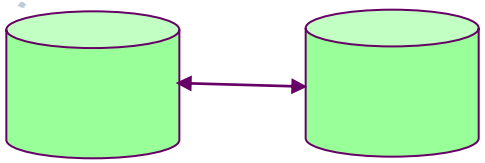
Figure 1: Definition of a Striping Unit. This figure shows the mapping of logical data to the disks for a striping unit of two sectors. The numbers in the figure are logical sectors; the circled two sectors constitute one stripe unit.

Stripping

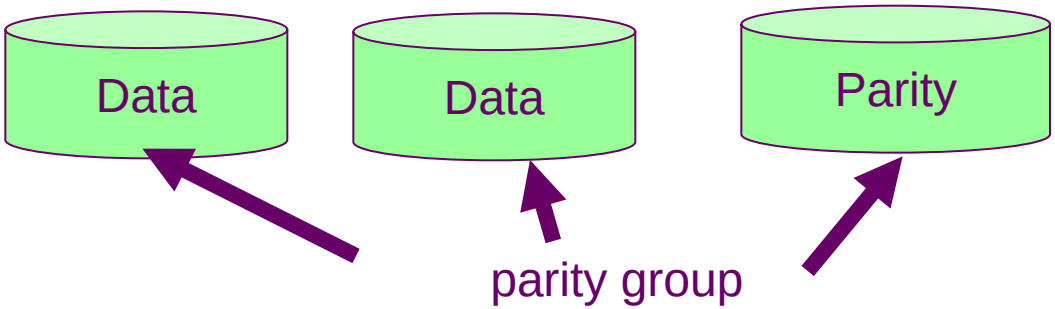

File 1 (blocks 1, 5),
File 2 (block 3),
File 4 (block 2)
Disk #1
File 1 (blocks 2, 6),
File 2 (block 4),
File 4 (block 3)
Disk #2
RAID
Controller
System
File 1 (block 3),
File 2 (block 1),
File 3
Disk #3
File 1 (block 4),
File 2 (block 2),
File 4 (block 1)
Disk #4

Mirroring v Parity

- Mirroring

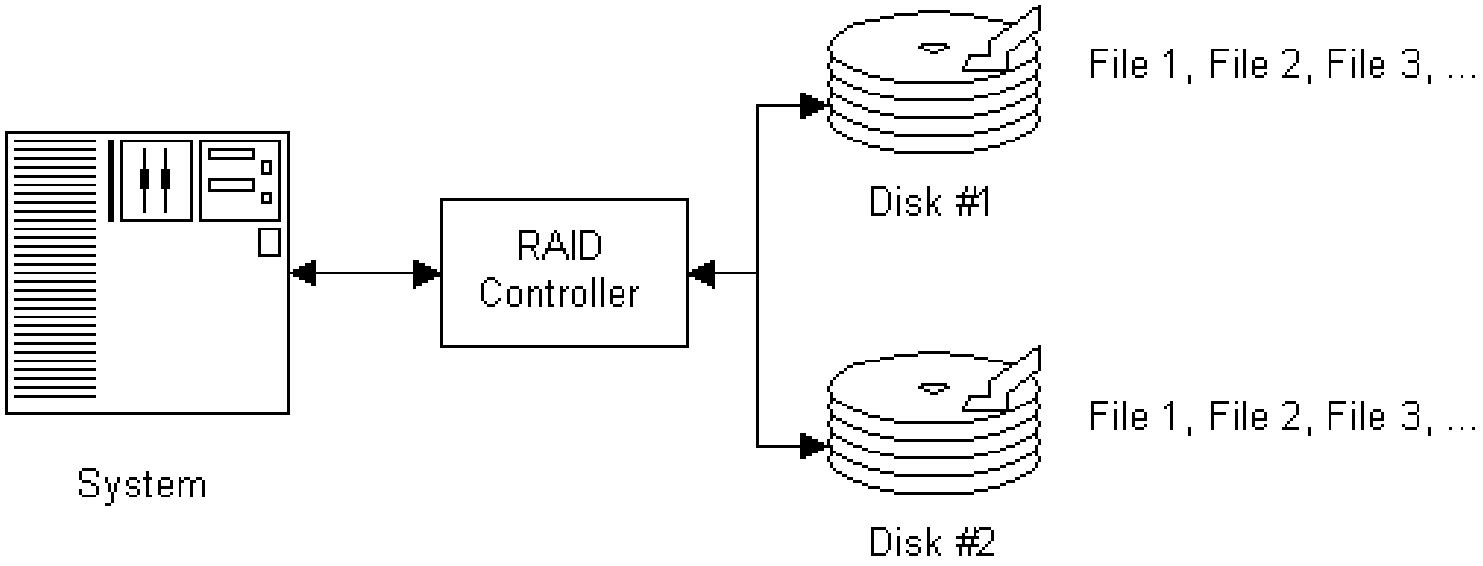


- Parity



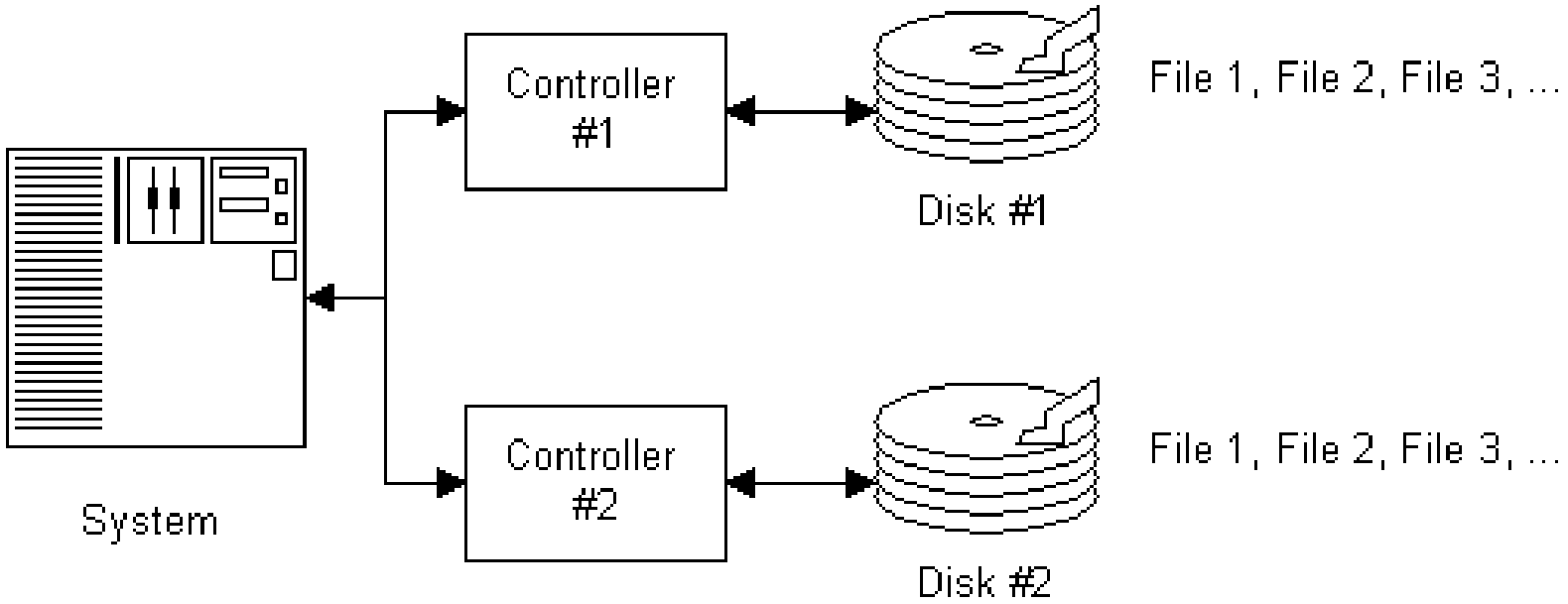
Mirroring

- Block diagram of a RAID mirroring configuration. The RAID controller duplicates the same information onto each of two hard disks.
- RAID controller is represented as a "logical black box"
- **RAID controller** can be implemented in **software**, or several **different types of hardware:**
  - **Integrated controller**
  - **bus-based add-in card**
  - **stand-alone RAID hardware**



Duplexing

- Duplexing goes one step beyond mirroring,
- it also **duplicates RAID Controller**
- There are hardware RAID duplexing solutions but usually only on very expensive external RAID boxes.)



Parity

- **In general**
  - N data + 1parity information
  - XOR operation ⊕, best for this purpose
  - Bit level, byte level, block level, stripe level
- **Bit level**
- A⊕B = B⊕A , (A⊕B) ⊕ A =B, (A⊕B) ⊕ B =A
- **Parity calculation**
- DP=D1 ⊕ D2 ⊕ ….Di ⊕ …. ⊕ Dn

- **New parity**
- DPnew= DP_old ⊕ Di = D1 ⊕ D2 ⊕ ….Di ⊕ …. ⊕ Dn
- DP include all previous data information

- **Failed data reconstruction**
- Di (failed) = D1 ⊕ D2 ⊕ ….**Dp** ⊕ …. ⊕ Dn
- No Order

Mirroring v Parity

- **Advantage** of parity:
- **Space overhead (storage efficiency)**
  - Mirroring 50%, parity 100/N

- The chief **disadvantages** parity:
- **1. complexity:**
  - all those parity bytes have to be computed--millions of them per second!--and that takes computing power.
  - This means a hardware controller that performs these calculations is required for high performance—
  - if you do *software* RAID with striping and parity the system CPU will be dragged down doing all these computations.
- **2. Time to recovery**
  - Also, while you can recover from a lost drive under parity, the missing data all has to be rebuilt, which has its own complications;
  - recovering from a lost mirrored drive is comparatively simple.

RAID Performance Issues

- Key to performance increases under RAID is **parallelism**
- **Read performances**
  - read = reading of data, only
- **Write performances**
  - write = writing of data + writing of redundant information

Performance

- **Random Read Performance**:
  - **Large number of small read requests**
  - How the RAID level performs on random access reads of files in the array.
  - Typically, this is most important for **transactional environments** with **smallish files**, especially ones with a high ratio of reads to writes.

- **Random Write Performance**:
  - **Large number of small write requests**
  - How the RAID level performs when writing small files in random places on the array.
  - Again, this is most relevant to **transaction-processing environments**, however, it is even more important to applications where a large number of writes are done, because write performance is much worse than read performance for many popular RAID levels.

Performance

- **Sequential Read Performance**:
  - **small or medium number** of **large read requests**
  - performance of the RAID level when reading large files sequentially from the array.
  - This is of greatest concern in applications where there are **many more reads than writes**, for example, a **server containing many large graphics files**.

- **Sequential Write Performance**:
  - **small or medium number** of **large write requests**
  - The RAID level's general performance when writing large files.
  - This is sometimes less important than sequential read performance, but is critical for situations where large files are written often,
  - such as **video or audio editing**.

RAID classes

- Mirroring
- Striping Without Parity
- Striping With Parity
- Combined systems (Nested systems)

Mirroring

- **Read** performance under mirroring is **far superior** to **write performance**.

- Let's suppose you are mirroring two drives under RAID 1. Every piece of data is duplicated, stored on both drives. This means that every byte of data stored must be written to both drives, making write performance under RAID 1 actually a bit slower than just using a single disk; even if it were as fast as a single disk, both drives are tied up during the write.

- But when you go to read back the data? There's absolutely no reason to access both drives; the controller, if intelligently programmed, will only ask one of the drives for the data--the other drive can be used to **satisfy a different request**.
- This makes **RAID significantly faster** than a **single drive for reads**, under **most conditions**.

Striping Without Parity

- **has about equal** read and write performance
- (or more accurately, roughly the same ratio of read to write performance that a single hard disk would have.)

- The **reason** is that the "**chopping up**" of the **data**
- **without parity** calculation
- means
- you must access the same number of drives for reads
- as you do for writes.

Striping With Parity

- **Striping With Parity:** As with mirroring, write performance when striping with parity (RAID levels 3 through 6) is worse than read performance,
- but unlike mirroring, the "hit" taken on a write when doing striping with parity is much more significant.

- Here's how the different accesses fare:
  - **For reads**, striping with parity **can** actually **be faster** than **striping without parity**.
  - The parity information is not needed on reads, and this makes the array behave during reads in a way similar to a RAID 0 array, except that the data is spread across **one extra drive**, slightly improving parallelism.

  - **For sequential writes**, there is the dual overhead of parity calculations as well as having to write to an additional disk to store the parity **information. This makes sequential writes slower than** striping without parity.

Striping With Parity

- **The biggest discrepancy under this technique is between random reads and random writes**. Random reads that only require parts of a stripe from one or two disks can be processed in parallel with other random reads that only need parts of stripes on different disks. In theory, random writes would be the same, except for one problem: every time you change any block in a stripe, **you have to recalculate the parity for that stripe, which requires *two* writes *plus* reading back all the other pieces of the stripe**!

  - Consider a RAID 5 array made from five disks, and a particular stripe across those disks that happens to have data on drives #3, #4, #5 and #1, and its parity block on drive #2. You want to do a small "random write" that changes just the block in this stripe on drive #3. Without the parity, the controller could just write to drive #3 and it would be done. With parity though, the change to drive #3 affects the parity information for the entire stripe. So this single write turns into a read of drives #4, #5 and #1, a parity calculation, and then a write to drive #3 (the data) and drive #2 (the newly-recalculated parity information). This is why striping with parity stinks for random write performance. (This is also why RAID 5 implementations in software are not recommended if you are interested in performance.)

Striping With Parity

- Another hit to write performance comes from the **dedicated parity drive** used in certain striping with parity implementations (in particular, RAID levels 3 and 4).
- Since only one drive contains parity information, ***every* write must write to this drive**, turning it into a performance bottleneck.
- Under implementations with distributed parity, like RAID 5, all drives contain data and parity information, so there is no single bottleneck drive; the overheads mentioned just above still apply though.

Single RAID Levels

- There are **8 "regular" RAID levels**, which are used to varying degrees in the "real world" today.

- A few levels, especially **RAID 0, RAID 1 and RAID 5**, are **extremely popular**, while a couple are rarely if ever seen in modern systems.

- For each level, I provide a comprehensive discussion of its attributes and characteristics in the areas of capacity, performance, fault tolerance, cost and more.

Normal, Degraded state and Rebuilding

- **Normal operation state**. All disk are in normal operated state.

- **Degrade state** (one or few drives were failed)
  - When an array enters a degraded state, performance is **reduced for two main reasons.**
  - The first is that one of the drives is no longer available, and the array must compensate for this loss of hardware. In a two-drive mirrored array, you are left with an "array of one drive", and therefore, performance becomes the same as it would be for a single drive. In a **striped array with parity**, **performance is degraded due to the loss** of a drive and the need to regenerate its lost information from the parity data, on the fly, as data is read back from the array.
- **Rebuild state**
  - The second reason for degraded operation after a drive failure is that after the toasted drive is replaced, the data that was removed from the array with its departure **must be regenerated on the new disk**. This process is called **rebuilding.**

Base RAID Levels

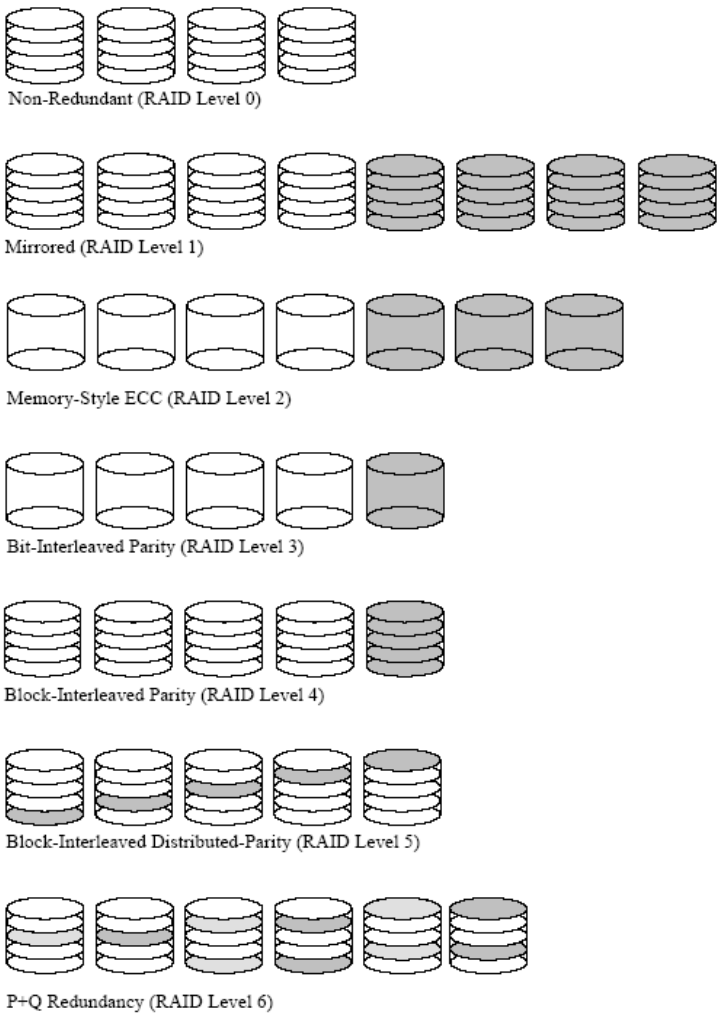
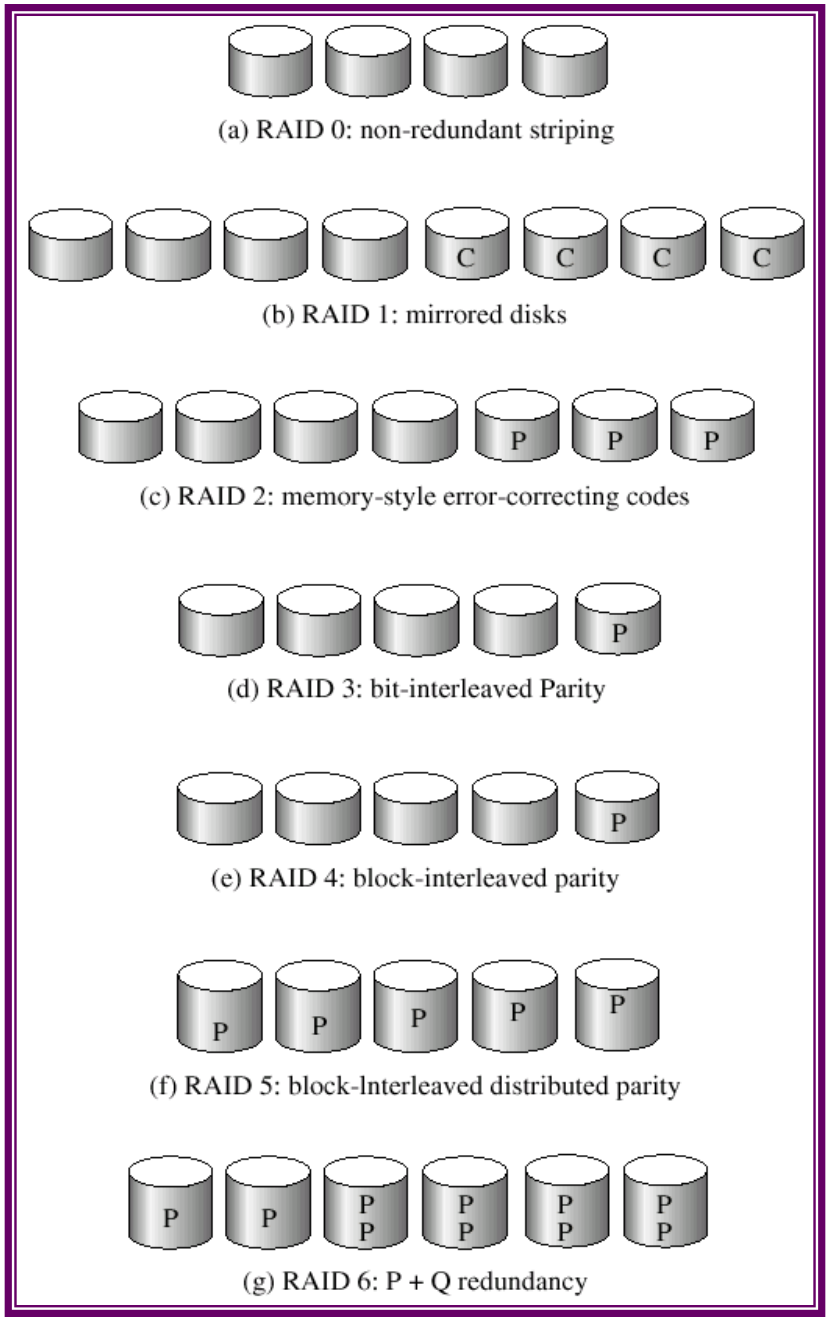


Figure 3:RAID Levels 0 Through 6. All RAID levels are illustrated at a user capacity of four disks. Disks with multiple platters indicate block-level striping while disks without multiple platters indicate bit-level striping. The shaded platters represent redundant information.



(a) RAID 0: non-redundant striping

(b) RAID 1: mirrored disks

(c) RAID 2: memory-style error-correcting codes

(d) RAID 3: bit-interleaved Parity

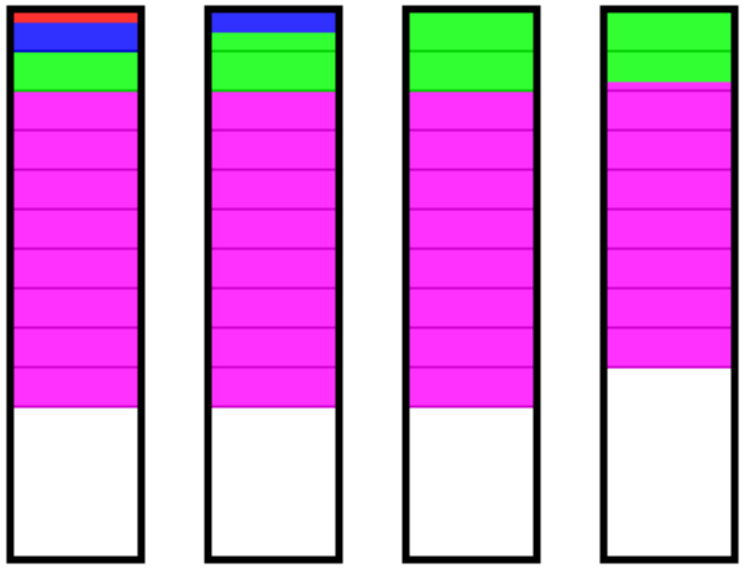
(e) RAID 4: block-interleaved parity

(f) RAID 5: block-Interleaved distributed parity

(g) RAID 6: P + Q redundancy

RAID Level 0

- **Common Name(s):** RAID 0. **Striping without parity**
- **Description:** Files are broken into stripes of a size dictated by the user-defined stripe size of the array, and stripes are sent to each disk in the array.
- Giving up redundancy allows this RAID level the best overall performance characteristics of the single RAID levels, especially for its cost.



Non-Redundant (RAID Level 0)

- **lowest cost** of any RAID organization
- does not employ redundancy **at all.**
- **best write performance**
  - since it never needs to update redundant information.
- Surprisingly, it does not have the best read performance.
  - Redundancy schemes that duplicate data, such as mirroring, can perform better on reads by
  - selectively scheduling requests on the disk with the shortest expected seek and rotational delays
- **Without redundancy, any single disk failure will result in data-loss.**
- Non-redundant disk arrays are widely used in supercomputing environments where performance and capacity, rather than reliability, are the primary concerns.

Optimal stripe unit- RAID-0

- RAID-0
- **Stripe unit**=

$$\sqrt{\frac{PX(L-1)Z}{N}}$$

- where
  - P is the average disk positioning time,
  - X is the average disk transfer rate,
  - L is the concurrency,
  - Z is the request size, and
  - N is the array size in disks

- **MTTF(RAID-0) = MTTF(disk) / N**

RAID 0 Features

- **Controller Requirements:** Supported by all hardware controllers, both SCSI and IDE/ATA, and also most software RAID solutions.
- **Hard Disk Requirements:** Minimum of two hard disks
- **Array Capacity:** Size of Smallest Drive * Number of Drives.
- **Storage Efficiency:** 100% if identical drives are used.
- **Fault Tolerance:** None. Failure of any drive results in loss of all data, short of specialized data recovery.
- **Availability:** Lowest of any RAID level. Lack of fault tolerance means no rapid recovery from failures. Failure of any drive results in array being lost and immediate downtime until array can be rebuilt and data restored from backup.
- **Degradation and Rebuilding:** Not applicable.

RAID 0 Features

- **Random Read Performance:** Very good;
  - better if using larger stripe sizes if the controller supports **independent reads** to different disks in the array.
- **Random Write Performance:** Very good;
  - again, best if using a larger stripe size and a controller supporting **independent writes**.
- **Sequential Read Performance:** Very good to excellent.
- **Sequential Write Performance:** Very good.

RAID 0 Features

- **Cost:** Lowest of all RAID levels.

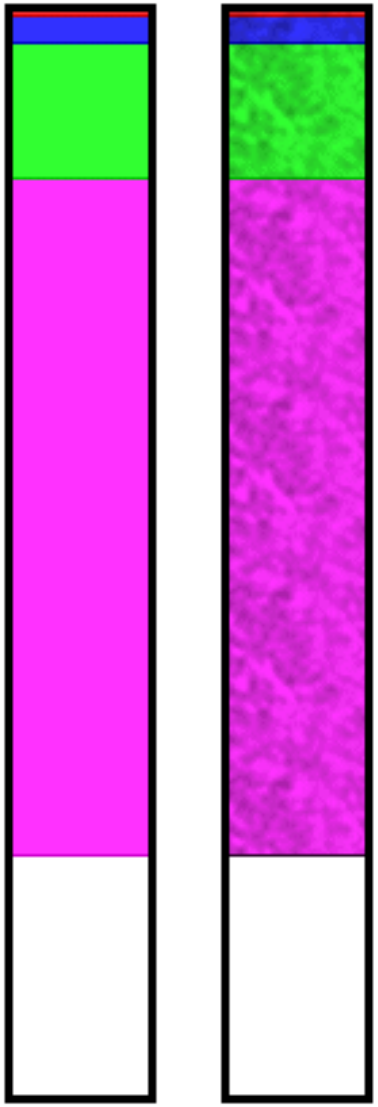
- **Special Considerations:** Using a RAID 0 array without backing up any changes made to its data at least daily is a loud statement that that data is not important to you.

- **Recommended Uses:** Non-critical data (or data that changes infrequently and is backed up regularly) requiring high speed, particularly write speed, and low cost of implementation.
  - audio and video streaming and editing
  - web servers
  - graphic design
  - high-end gaming or hobbyist systems
  - temporary or "scratch" disks on larger machines

RAID Level 1

- **Common Name(s):** RAID 1; RAID 1 with Duplexing.
- **Technique(s) Used:** **Mirroring or Duplexing**
- **Description:** RAID 1 is usually implemented as mirroring; a drive has its data duplicated on two different drives using either a hardware RAID controller or software (generally via the operating system). If either drive fails, the other continues to function as a single drive until the failed drive is replaced. Conceptually simple, RAID 1 is popular for those who require fault tolerance and don't need top-notch read performance. A variant of RAID 1 is duplexing, which duplicates the controller card as well as the drive, providing tolerance against failures of either a drive or a controller. It is much less commonly seen than straight mirroring.

RAID-1 (four file illustration)



Mirrored (RAID Level 1)

- uses twice as many disks as a non-redundant disk array
- writing = 2 x write cycle
- slowest for writes (twice =100%) but ???
- When data is read, it can be retrieved from the disk with the shorter queuing, seek and rotational delays.
- If a disk fails, the other copy is used to service requests.
- Mirroring is frequently used **in database** where availability and transaction rate are more important than storage efficiency

RAID 1 Features

- **Controller Requirements:** Supported by all hardware controllers, both SCSI and IDE/ATA, and also most software RAID solutions.
- **Hard Disk Requirements:** Exactly two hard disks. Any type may be used but they should ideally be identical.
- **Array Capacity:** Size of Smaller Drive.
- **Storage Efficiency:** 50% if drives of the **same size** are used, otherwise (Size of Smaller Drive / (Size of Smaller Drive + Size of Larger Drive))
- **Fault Tolerance:** Very good; duplexing even better.
- **Availability:** Very good. Most RAID controllers, even low-end ones, will support hot sparing and automatic rebuilding of RAID 1 arrays.
- **Degradation and Rebuilding:** Slight degradation of read performance; write performance will actually improve. Rebuilding is relatively **fast**.

RAID 1 Features

- **Random Read Performance:** Good. Better than a single drive but worse than many other RAID levels.
- **Random Write Performance:** Good. Worse than a single drive, but **better than many other RAID levels**. :^)
- **Sequential Read Performance:** Fair; about the same as a single drive.
- **Sequential Write Performance:** Good; again, better than many other RAID levels.

RAID 1 Features

- **Cost:** Relatively high due to redundant drives; lowest storage efficiency of the single RAID levels. Duplexing is still more expensive due to redundant controllers. On the other hand, no expensive controller is required, and large consumer-grade drives are rather inexpensive these days, making RAID 1 a viable choice for an individual system.
- **Special Considerations:** RAID 1 arrays are limited to the size of the drives used in the array. Multiple RAID 1 arrays can be set up if additional storage is required, but RAID 1+0 begins to look more attractive in that circumstance. Performance may be reduced if implemented using software instead of a hardware controller; duplexing may require software RAID and thus may show lower performance than mirroring.
- **Recommended Uses:** Applications requiring high fault tolerance at a low cost, without heavy emphasis on large amounts of storage capacity or top performance. Especially useful in situations where the perception is that having a duplicated set of data is more secure than using parity. For this reason, RAID 1 is popular for accounting and other financial data. It is also commonly used for small database systems, enterprise servers, and for individual users requiring fault tolerance with a minimum of hassle and cost (since redundancy using parity generally requires more expensive hardware.)

RAID Level 2

- **Common Name(s):** RAID 2.
- **Technique(s) Used:** **Bit-level striping with Hamming code ECC**.
- **Description:** Level 2 is the "black sheep" of the RAID family, because it is the only RAID level that does not use one or more of the "standard" techniques of mirroring, striping and/or parity. RAID 2 uses something *similar* to striping with parity, but not the same as what is used by RAID levels 3 to 7. It is implemented by splitting data at the *bit* level and spreading it over a number of data disks and a number of redundancy disks. The redundant bits are calculated using Hamming codes, a form of error correcting code (ECC). Each time something is to be written to the array these codes are calculated and written along side the data to dedicated ECC disks; when the data is read back these ECC codes are read as well to confirm that no errors have occurred since the data was written. If a single-bit error occurs, it can be corrected "on the fly". If this sounds similar to the way that ECC is used *within* hard disks today, that's for a good reason: it's pretty much exactly the same. It's also the same concept used for ECC protection of system memory.

RAID Level 2

- Level 2 is the only RAID level of the ones defined by the original Berkeley document that is not used today, for a variety of reasons. It is expensive and often requires many drives--see below for some surprisingly large numbers.
- The controller required was complex, specialized and expensive. The performance of RAID 2 is also rather substandard in transactional environments due to the bit-level striping. But most of all, level 2 was obviated by the use of ECC within a hard disk; essentially, much of what RAID 2 provides you now get for "free" within each hard disk, with other RAID levels providing protection above and *beyond* ECC.

- Due to its cost and complexity, level 2 never really "caught on". Therefore, much of the information below is based upon theoretical analysis, not empirical evidence.

Memory-Style ECC (RAID Level 2)

- Memory systems have provided recovery from failed components with much less cost than mirroring by using Hamming codes.
- Hamming codes contain parity for distinct overlapping subsets of components.
- In one version of this scheme, **4 data disks require 3 redundant disks**, one less than mirroring.
- Since the number of redundant disks is proportional to the log of the total number of disks in the system, storage efficiency increases as the number of data disks increases.

- If a single component fails, several of the parity components will have inconsistent values, and the failed component is the one held in common by each incorrect subset.
- The lost information is recovered by reading the other components in a subset, including the parity component, and setting the missing bit to 0 or 1 to create the proper parity value for that subset.
- Thus, multiple redundant disks are needed to identify the failed disk, but only one is needed to recover the lost information.

RAID 2 Features

- **Controller Requirements:** Specialized controller hardware required.
- **Hard Disk Requirements:** Depends on exact implementation, but a typical setup required **10 data disks and 4 ECC disks** for a total of **14**, or **32 data disks and 7 ECC disks for a total of 39**! The disks were spindle-synchronized to run in tandem.
- **Array Capacity:** Depends on exact implementation but would be rather large if built today using modern drives.
- **Storage Efficiency:** Depends on the number of data and ECC disks; for the 10+4 configuration, about 71%; for the 32+7 setup, about 82%.
- **Fault Tolerance:** Only fair; for all the redundant drives included, you don't get much tolerance: only one drive can fail in this setup and be recoverable "on the fly".
- **Availability:** Very good, due to "on the fly" error correction.
- **Degradation and Rebuilding:** In theory, there would be little degradation due to failure of a single drive.

RAID 2 Features

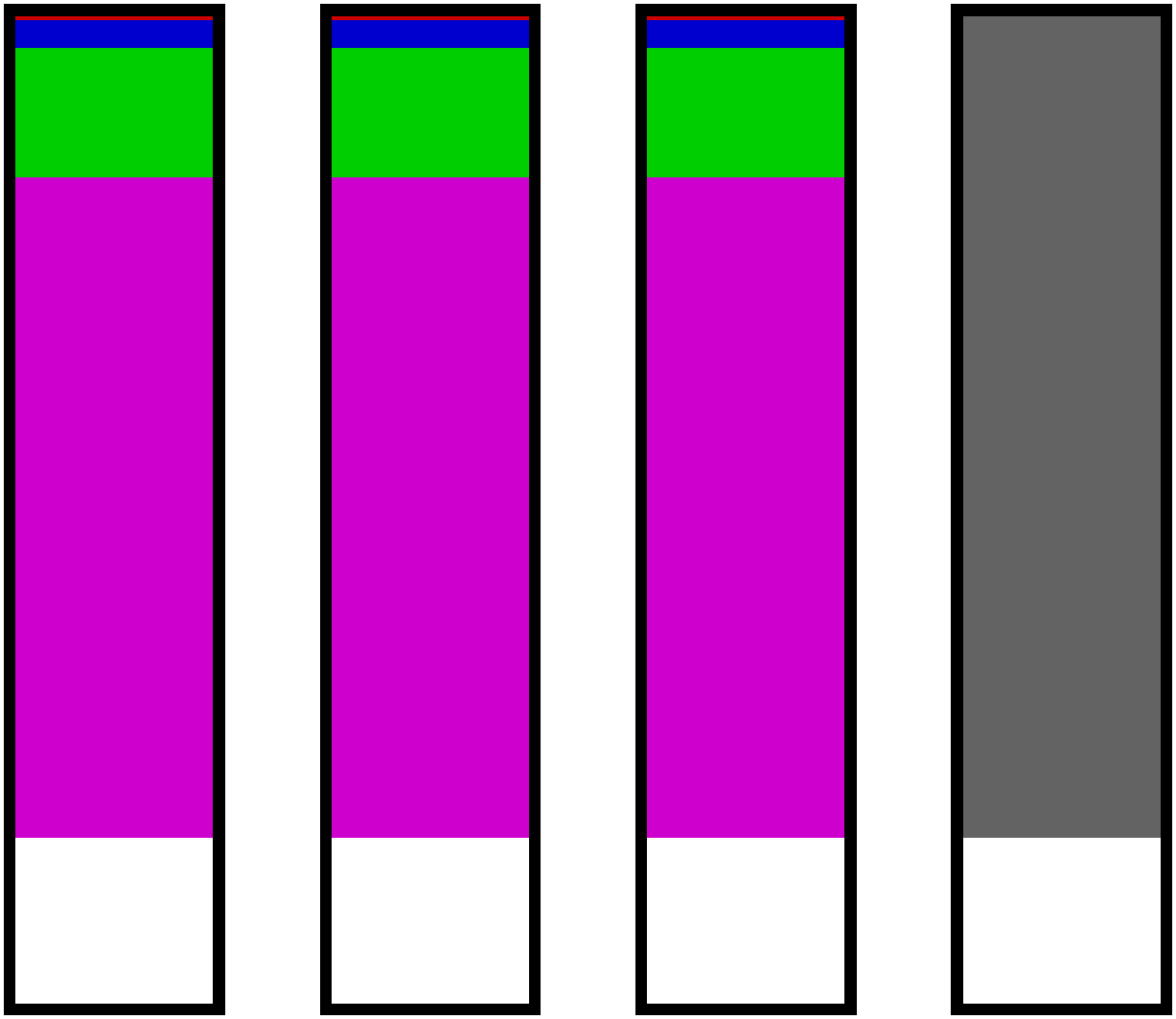
- **Random Read Performance:** Fair. Bit-level striping makes multiple accesses impossible.
- **Random Write Performance:** Poor, due to bit-level striping and ECC calculation overhead.
- **Sequential Read Performance:** Very good, due to parallelism of many drives.
- **Sequential Write Performance:** Fair to good.
- **Cost:** Very expensive.
- **Special Considerations:** Not used in modern systems.
- **Recommended Uses:** Not used in modern systems.

RAID Level 3

- **Common Name(s):** RAID 3. (Watch out for some companies that say their products implement RAID 3 when they are really RAID 4.)
- **Technique(s) Used:** **Byte-level striping with dedicated parity.**

- **Description:** Under RAID 3, data is striped across multiple disks at a byte level; the exact number of bytes sent in each stripe varies but is typically under 1024. The parity information is sent to a dedicated parity disk, but the failure of any disk in the array can be tolerated
- The dedicated parity disk *does* generally serve as a performance bottleneck, especially for random writes, because it must be accessed any time anything is sent to the array;
- RAID 3 differs from RAID 4 only in the size of the stripes sent to the various disks.

RAID3 example (four disks and four files)



Bit-Interleaved Parity (RAID Level 3)

- One can improve upon memory-style ECC disk arrays by noting that, unlike memory component failures, disk controllers can easily identify which disk has failed.
- Thus, one can use a single parity disk rather than a set of parity disks to recover lost information.
- Stripe unit < 512 bytes (1byte or 1 bit)
  - **each read request accesses all data disks**
  - **each write request accesses all data disks and the parity disk.**

- **only one request can be serviced at a time**.
- Because the parity disk contains only parity and no data, the parity disk cannot participate on reads, resulting in slightly lower read performance than for redundancy schemes that distribute the parity and data over all disks.
- Bit-interleaved, parity disk arrays are frequently used in applications that require high bandwidth but not high I/O rates.

RAID 3 features

- **Controller Requirements:** Generally requires a medium-to-high-end hardware RAID card.
- **Hard Disk Requirements:** Minimum of three standard hard disks; maximum set by controller. Should be of identical size and type.
- **Array Capacity:** (Size of Smallest Drive) * (Number of Drives - 1)
- **Storage Efficiency:** If all drives are the same size, ((Number of Drives - 1) / Number of Drives).
- **Fault Tolerance:** Good. Can tolerate loss of one drive.
- **Availability:** Very good. Hot sparing and automatic rebuild are usually supported by controllers that implement RAID 3.
- **Degradation and Rebuilding:** Relatively little degrading of **performance** if a drive fails. Rebuilds can take many hours.

RAID 3 features

- **Random Read Performance: Good, but not great**, due to byte-level striping.
- **Random Write Performance:** Poor, due to byte-level striping, parity calculation overhead, and the bottleneck of the dedicated parity drive.
- **Sequential Read Performance:** Very good.
- **Sequential Write Performance:** Fair to good.

RAID 3 features

- **Cost:** Moderate. A hardware controller is usually required, as well as at least three drives.

- **Special Considerations:** Not as popular as many of the other commonly-implemented RAID levels. For transactional environments, RAID 5 is usually a better choice.

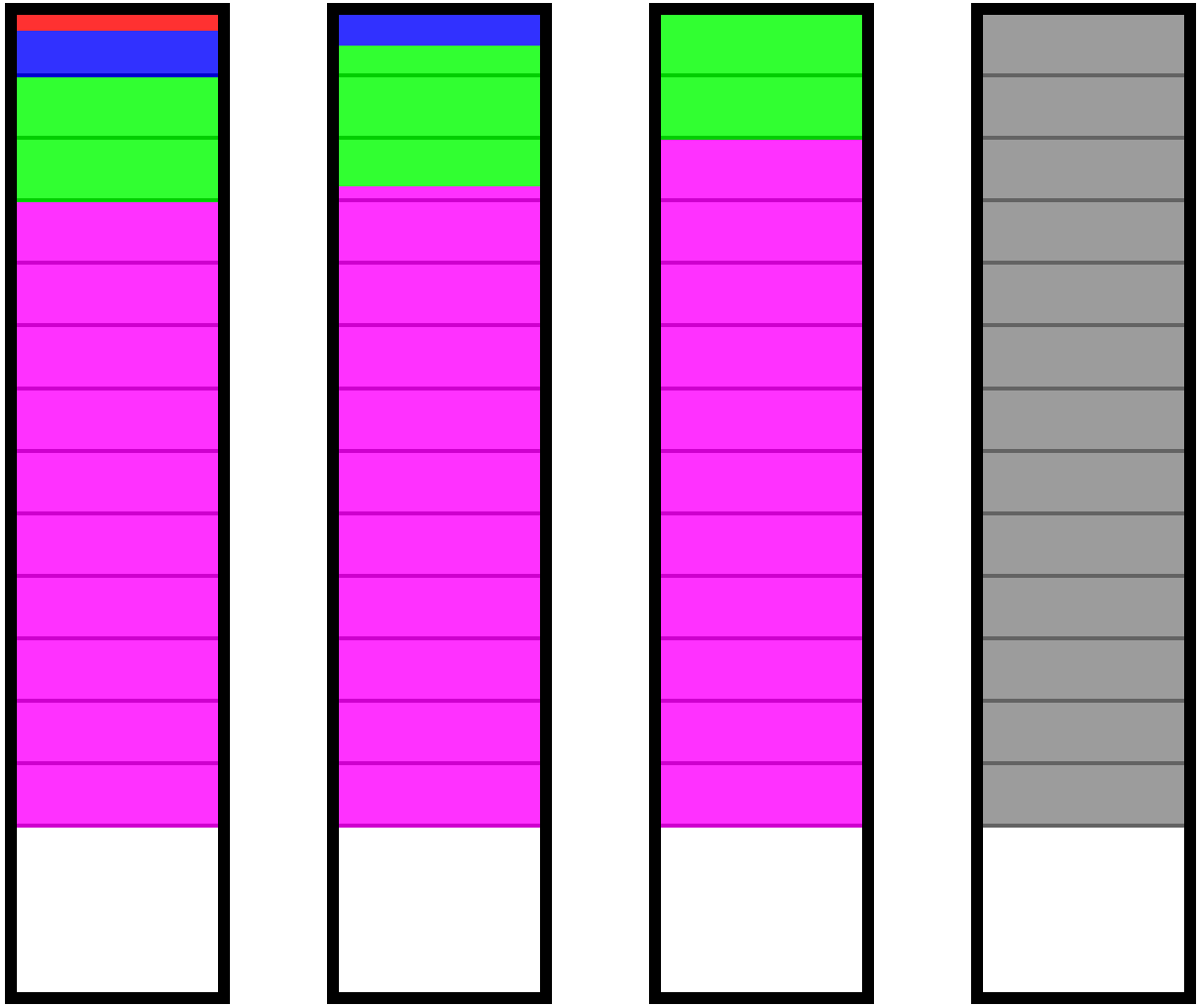
- **Recommended Uses:** Applications working with large files that require high transfer performance with redundancy, especially serving or editing large files: multimedia, publishing and so on. RAID 3 is often used for the same sorts of applications that would typically see the use of RAID 0, where the lack of fault tolerance of RAID 0 makes it unacceptable.

RAID 4

- **Common Name(s):** RAID 4 (sometimes called RAID 3 by the confused).
- **Technique(s) Used:** **Block-level striping with dedicated parity**.

- **Description:** RAID 4 improves performance by striping data across many disks in blocks, and provides fault tolerance through a dedicated parity disk. This makes it in some ways the "middle sibling" in a family of close relatives, RAID levels 3, 4 and 5. It is like RAID 3 except that it uses blocks instead of bytes for striping, and like RAID 5 except that it uses dedicated parity instead of distributed parity.
  - Going from byte to block striping improves random access performance compared to RAID 3,
  - but the **dedicated parity disk remains a bottleneck**, especially for random write performance.
  - Fault tolerance, format efficiency and many other attributes are the same as for RAID 3 and RAID 5.

RAID4 example (four disks and four files)



Block-Interleaved Parity (RAID Level 4)

- striping unit = N disk blocks.
- Read requests smaller than the striping unit access only a single data disk.

- Write requests must update the requested data blocks and must also compute and update the parity block.
- For large writes that touch blocks on all disks, parity is easily computed by exclusive-or'ing the new data for each disk.
- For small write requests that update only one data disk, parity is computed by noting how the new data differs from the old data and applying those differences to the parity block.
- **Small write** requests thus require **4 disk I/Os:**
  - one to write the new data,
  - two to read the old data and old parity for computing the new parity, and
  - one to write the new parity.
  - This is referred to as a **read-modify-write** procedure.

- Because a block-inter-leaved, parity disk array has only one parity disk, which must be updated on all write operations, the parity disk can easily become a bottleneck. Because of this limitation, the block-interleaved distributed-parity disk array is universally preferred over the block-interleaved, parity disk array.

RAID 4 Features

- **Controller Requirements:** Generally requires a medium-to-high-end hardware RAID card.
- **Hard Disk Requirements:** Minimum of three standard hard disks; maximum set by controller. Should be of identical size and type.
- **Array Capacity:** (Size of Smallest Drive) * (Number of Drives - 1).
- **Storage Efficiency:** If all drives are the same size, ((Number of Drives - 1) / Number of Drives).
- **Fault Tolerance:** Good. Can tolerate loss of one drive.
- **Availability:** Very good. Hot sparing and automatic rebuild are usually supported..
- **Degradation and Rebuilding:** Moderate degrading if a drive fails; potentially lengthy rebuilds.

RAID 4 Features

- **Random Read Performance:** Very good.
- **Random Write Performance:** **Poor** to fair, due to **parity calculation overhead** and the **bottleneck of the dedicated parity drive**.
- **Sequential Read Performance:** Good to very good.
- **Sequential Write Performance:** Fair to good.

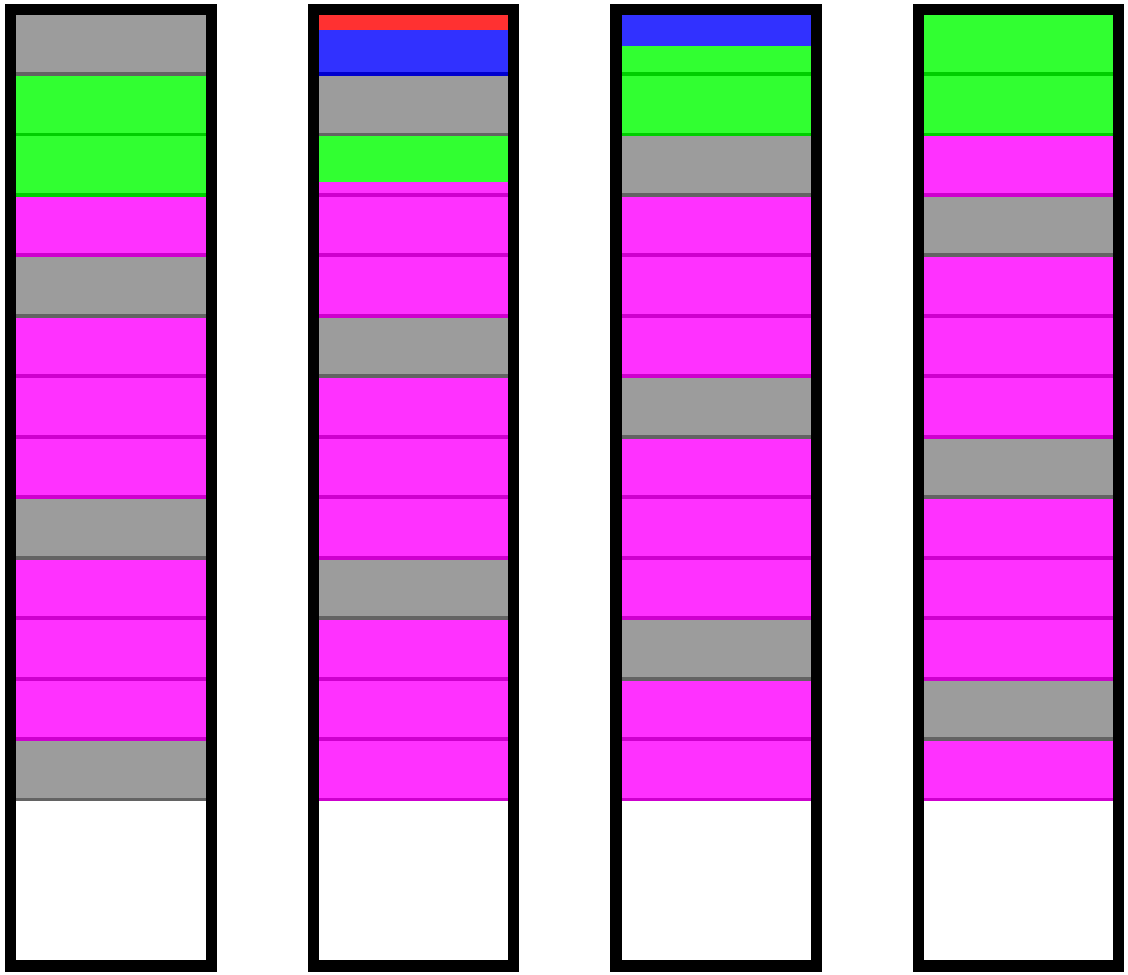
RAID 4 Features

- **Cost:** Moderate. A hardware controller is usually required, as well as at least three drives.
- **Special Considerations:** Performance will depend to some extent upon the stripe size chosen.
- **Recommended Uses:** Jack of all trades and master of none, RAID 4 is not as commonly used as RAID 3 and RAID 5, because it is in some ways a "compromise" between them that doesn't have a target market as well defined as either of those two levels. It is sometimes used by applications commonly seen using RAID 3 or RAID 5, running the gamut from databases and enterprise planning systems to serving large multimedia files.

RAID 5

- **Common Name(s):** RAID 5.
- **Technique(s) Used:** **Block-level striping with distributed parity**.
- **Description:** One of the most popular RAID levels, RAID 5 stripes both data and parity information across three or more drives. It is similar to RAID 4 except that it **exchanges the dedicated parity drive for a distributed parity algorithm, writing data and parity blocks across all the drives in the array**. This removes the "bottleneck" that the dedicated parity drive represents, improving write performance slightly and allowing somewhat better parallelism in a multiple-transaction environment, though the overhead necessary in dealing with the parity continues to bog down writes. Fault tolerance is maintained by ensuring that the parity information for any given block of data is placed on a drive separate from those used to store the data itself. The performance of a RAID 5 array can be "adjusted" by trying different stripe sizes until one is found that is well-matched to the application being used.

RAID5 example (four disks and four files)



Block-Interleaved Distributed-Parity (RAID Level 5)

- distributed-parity disk array eliminates the parity disk bottleneck
- **Data ->on all disks**
- **Parity ->on all disks**
- all disks participate in servicing read operations
- RAID 5 is the **best for:**
  - small read
  - large read
  - large write performance of any redundant disk array.
- Small write requests are somewhat inefficient compared with redundancy schemes such as mirroring however, due to the need to perform read-modify-write operations to update parity.
- This is the major performance weakness of RAID level 5 disk arrays and has been the subject of intensive research.

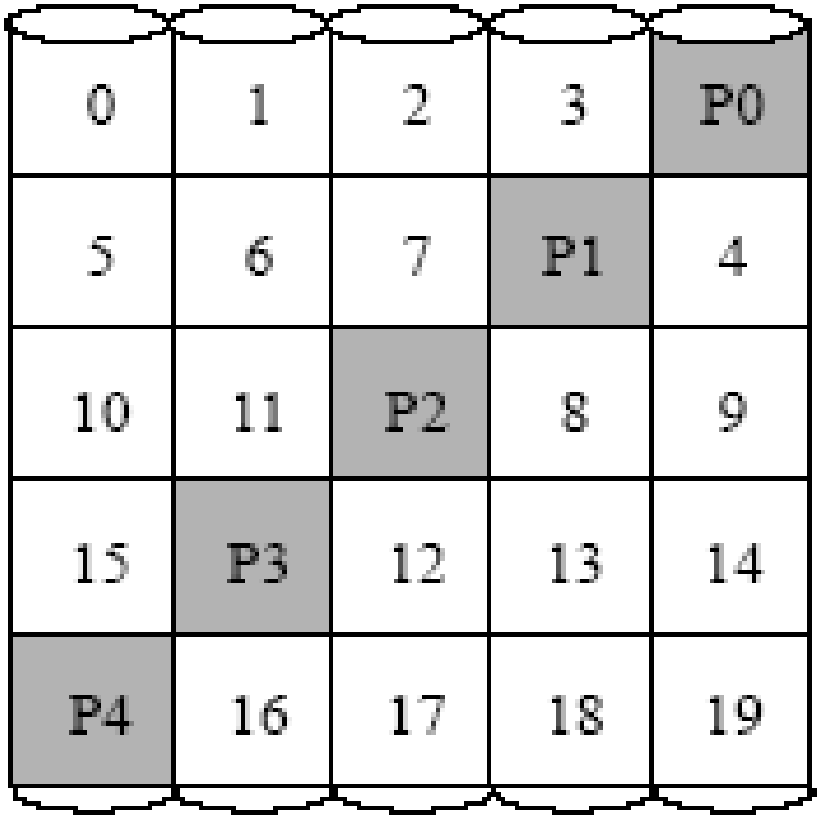


Figure 4: RAID level 5 Left-Symmetric Parity Placement. Each square corresponds to a stripe unit. Each column of squares corresponds to a disk. P0 computes the parity over stripe units 0, 1, 2 and 3; P1 computes parity over stripe units 4, 5, 6 and 7; etc. Lee [Lee91b] shows that the left-symmetric parity distribution has the best performance. Only the minimum repeating pattern is shown.

RAID-5

- **stripe_unit=**
  - 0.5K+1/4 * average positioning time * data transfer rate * (concurrency-1))
- **stripe_unit=**
  - 0.5 * average positioning time * data transfer rate

$$\frac{MTTF(disk)^2}{N \times (G-1) \times MTTR(disk)}$$

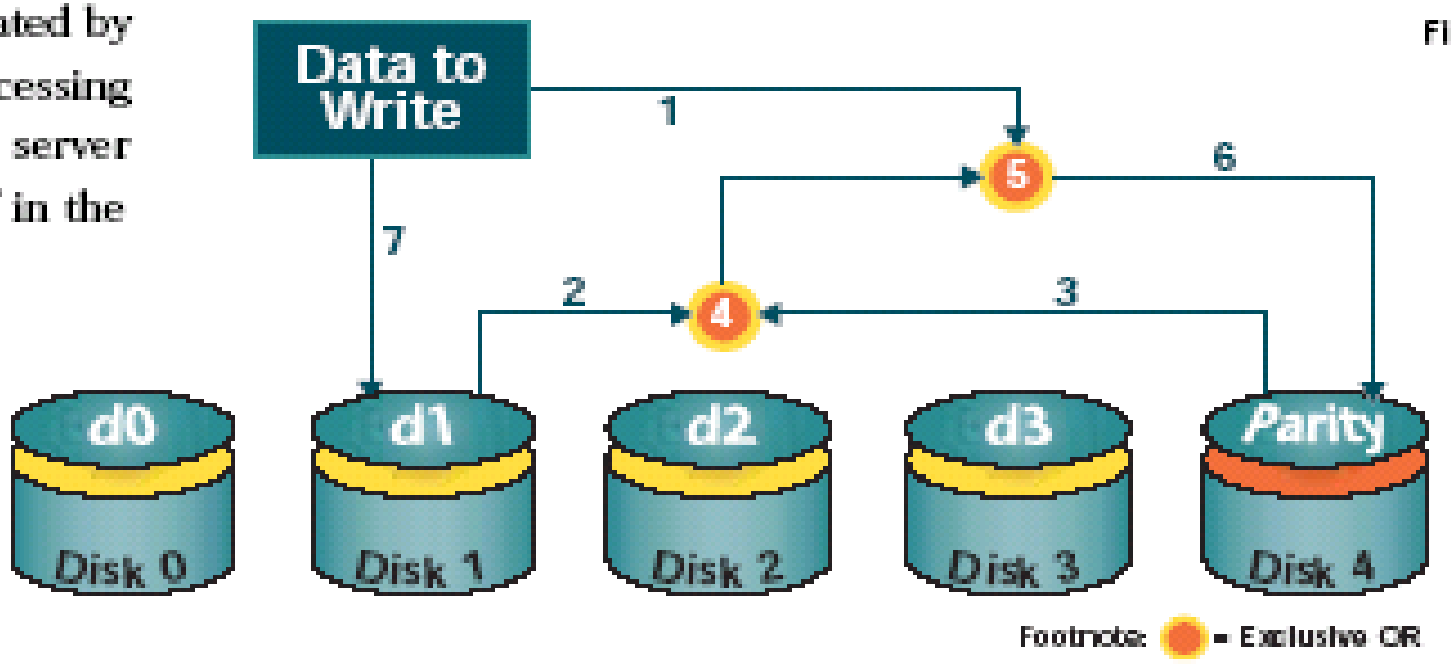
- **MTTF(RAID-5) = $MTTF^2$(disk) / MTTR(disk)xN(G-1)**
  - N is a total number of disks
  - G is a parity group
- **MTTF(RAID-1) = $MTTF^2$(disk) / 2xMTTR(disk)**
  - N = 2
  - G = 2

RAID5 small-write performance problem

- Parity logging
- RAID caching

RAID5 small-write performance problem

den created by
ion processing
in both server
al itself in the
nance.
for
oft-
ntly
ally
usi-
nts.
tions typically
apabilities for
ower or other
rm effectively
nory is used to
letion, there is
e solution. This
dditional fault



Figure

RAID 5 WRITE SEQUENCE:

1) Get new data to be written to disk
2) Read old block contents (data to replace) into internal buffer
3) Read old block's corresponding parity into internal buffer
4) Remove target block's contribution to parity
5) Compute new parity using XOR from (1) and (4)
6) Write new parity to disk
7) Write new data to disk
8) Signify I/O completion via interrupt (not shown)

RAID 5 Features

- **Controller Requirements:** Requires a moderately high-end card for hardware RAID; supported by some operating systems for software RAID, but at a substantial performance penalty.
- **Hard Disk Requirements:** Minimum of three standard hard disks; maximum set by controller. Should be of identical size and type.
- **Array Capacity:** (Size of Smallest Drive) * (Number of Drives - 1).
- **Storage Efficiency:** If all drives are the same size, ((Number of Drives - 1) / Number of Drives).
- **Fault Tolerance:** Good. Can tolerate loss of one drive.
- **Availability:** Good to very good. Hot sparing and automatic rebuild are usually featured on hardware RAID controllers supporting RAID 5 (software RAID 5 will require down-time).
- **Degradation and Rebuilding:** Due to **distributed parity**, **degradation** can be substantial after a failure and during rebuilding.

RAID 5 Features

- **Random Read Performance:** Very good to excellent; generally better for larger stripe sizes. Can be better than RAID 0 since the data is distributed over one additional drive, and the parity information is not required during normal reads.
- **Random Write Performance:** Only fair, due to parity overhead; this is improved over RAID 3 and RAID 4 due to eliminating the dedicated parity drive, but the overhead is still substantial.
- **Sequential Read Performance:** Good to very good; generally better for smaller stripe sizes.
- **Sequential Write Performance:** Fair to good.

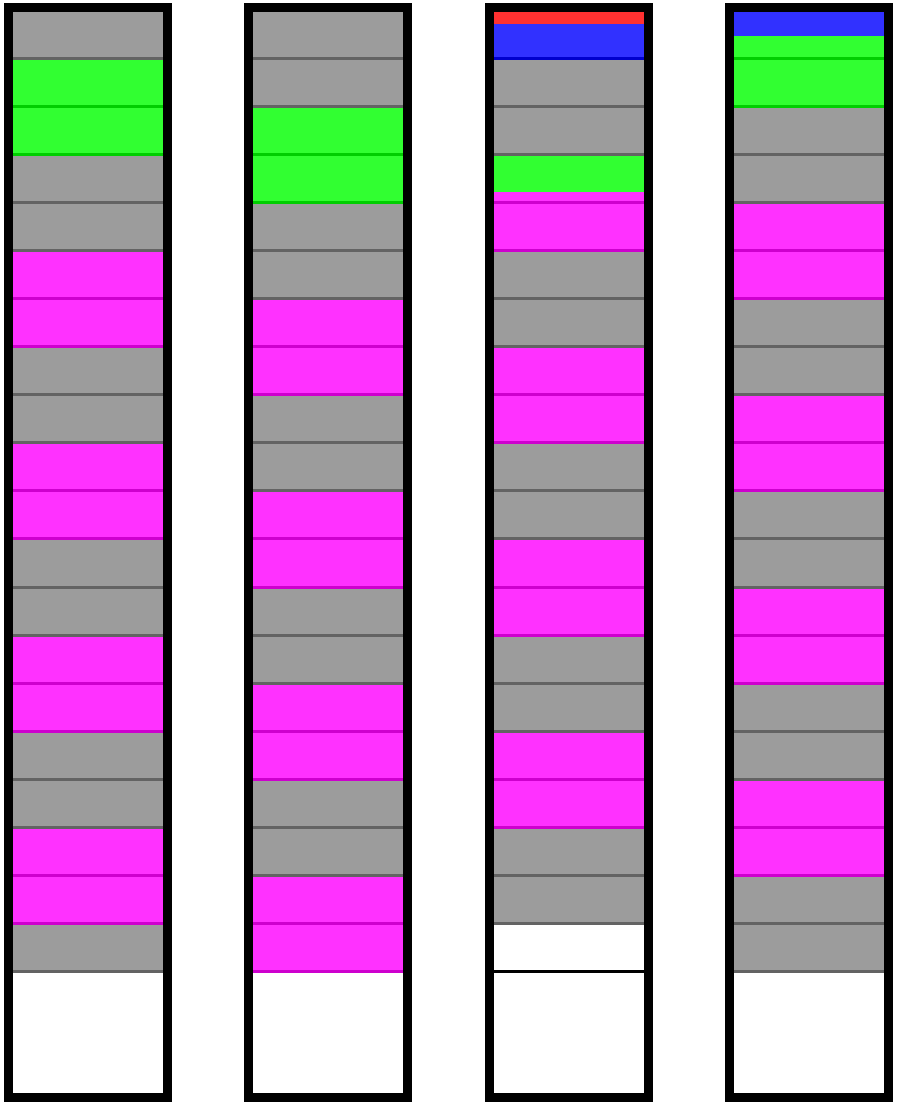
RAID 5 Features

- **Cost:** Moderate, but often less than that of RAID 3 or RAID 4 due to its greater popularity, and especially if software RAID is used.
- **Special Considerations:** Due to the amount of parity calculating required, software RAID 5 can *seriously* slow down a system. Performance will depend to some extent upon the stripe size chosen.
- **Recommended Uses:** RAID 5 is seen by many as the ideal combination of good performance, good fault tolerance and high capacity and storage efficiency. It is best suited for transaction processing and is often used for "general purpose" service, as well as for relational database applications, enterprise resource planning and other business systems. For write-intensive applications, RAID 1 or RAID 1+0 are probably better choices (albeit higher in terms of hardware cost), as the performance of RAID 5 will begin to substantially decrease in a write-heavy environment.

RAID Level 6

- **Common Name(s):** RAID 6. Some companies use the term "RAID 6" to refer to proprietary extensions of RAID 5; these are not discussed here.
- **Technique(s) Used:** **Block-level striping with *dual* distributed parity.**
- **Description:** RAID 6 can be thought of as "RAID 5, but more". It stripes blocks of data and parity across an array of drives like RAID 5, except that it calculates *two* sets of parity information for each parcel of data. The goal of this duplication is solely to improve fault tolerance; RAID 6 can handle the failure of any two drives in the array while other single RAID levels can handle at most one fault. Performance-wise, RAID 6 is generally slightly worse than RAID 5 in terms of writes due to the added overhead of more parity calculations, but may be slightly faster in random reads due to spreading of data over one more disk. As with RAID levels 4 and 5, performance can be adjusted by experimenting with different stripe sizes.

RAID6 example (four disks and four files)



P+Q Redundancy (RAID Level 6)

- Parity is a redundancy code capable of correcting any *singl*e, self-identifying failure.
- As larger disk arrays are considered, multiple failures are possible and stronger codes are needed.
- Thus, applications with more stringent reliability requirements require stronger error-correcting codes.
- One such scheme, called *P+Q redundancy*, uses Reed-Solomon codes to protect against up to two disk failures using the bare minimum of two redundant disks.
- The P+Q redundant disk arrays are structurally very similar to the block-interleaved distributed-parity disk arrays and operate in much the same manner.
- In particular, P+Q redundant disk arrays also perform small write operations using a **read-modify-write procedure**, except that instead of four disk accesses per write requests, P+Q redundant disk arrays require **6 disk accesses** due to the need to update both the ‘P’ and ‘Q’ information.

RAID6

- **Controller Requirements:** Requires a specialized (usually meaning expensive) hardware controller.
- **Hard Disk Requirements:** Minimum of four hard disks; maximum set by controller. Should be of identical size and type.
- **Array Capacity:** (Size of Smallest Drive) * (Number of Drives - 2).
- **Storage Efficiency:** If all drives are the same size, ((Number of Drives - 2) / Number of Drives).
- **Fault Tolerance:** Very good to excellent. Can tolerate the simultaneous loss of any two drives in the array.
- **Availability:** Excellent.
- **Degradation and Rebuilding:** Due to the complexity of dual distributed parity, degradation can be substantial after a failure and during rebuilding. Dual redundancy may allow rebuilding to be delayed to avoid performance hit.

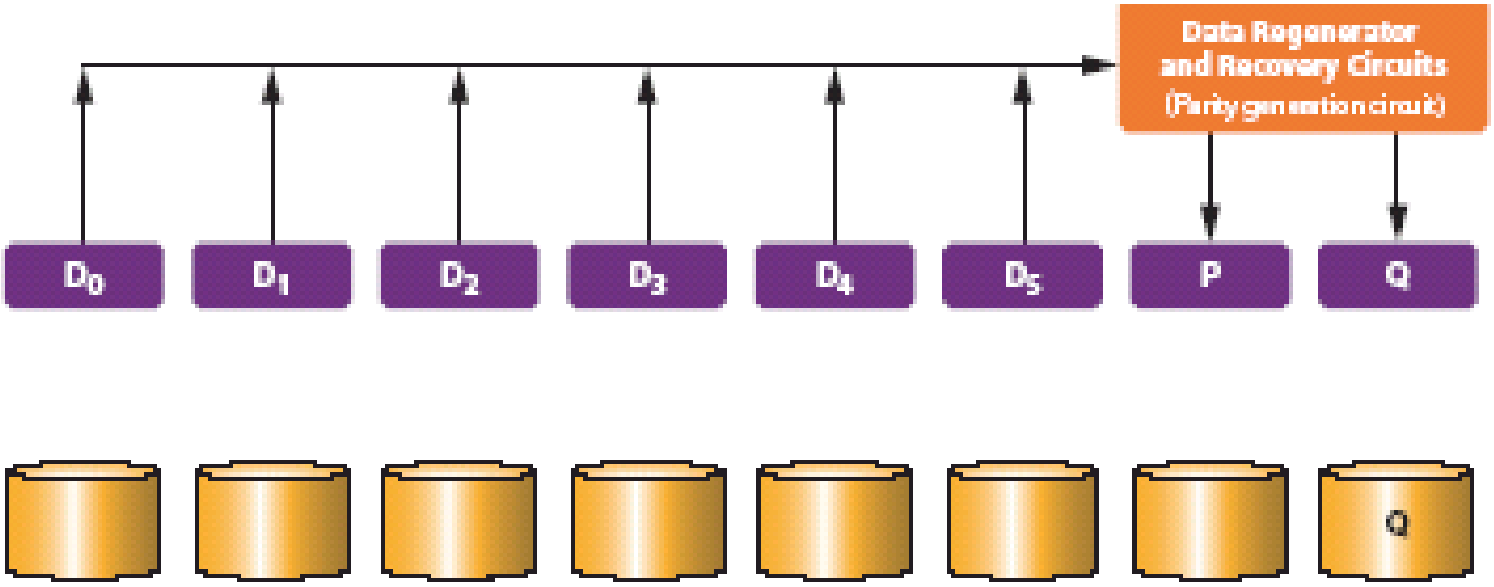
RAID6

- **Random Read Performance:** Very good to excellent; generally better for larger stripe sizes.
- **Random Write Performance:** Poor, due to dual parity overhead and complexity.
- **Sequential Read Performance:** Good to very good; generally better for smaller stripe sizes.
- **Sequential Write Performance:** Fair.

RAID6

- **Cost:** High.
- **Special Considerations:** Requires special implementation; not widely available.
- **Recommended Uses:** In theory, RAID 6 is ideally suited to the same sorts of applications as RAID 5, but in situations where additional fault tolerance is required. In practice, RAID 6 has never really caught on because few companies are willing to pay for the extra cost to insure against a relatively rare event--it's unusual for two drives to fail simultaneously (unless something happens that takes out the entire array, in which case RAID 6 won't help anyway). On the lower end of the RAID 5 market, the rise of hot swapping and automatic rebuild features for RAID 5 have made RAID 6 even less desirable, since with these advanced features a RAID 5 array can recover from a single drive failure in a matter of hours (where without them, RAID 5 would require downtime for rebuilding, giving RAID 6 a substantial advantage.) On the higher end of the RAID 5 market, RAID 6 usually loses out to multiple RAID solutions such as RAID 10 that provide some degree of multiple-drive fault tolerance while offering improved performance as well.

Formula for Generating Parity Data in RAID-6



RAID-6 (6D+2P) configuration creates parity based on striped data segments, and stores this information in the disk drives of the RAID group.

Formula for generating parity data (P and Q):

P: Exclusive OR is used

$P = D_0 \oplus D_1 \oplus D_2 \oplus D_3 \oplus D_4 \oplus D_5$

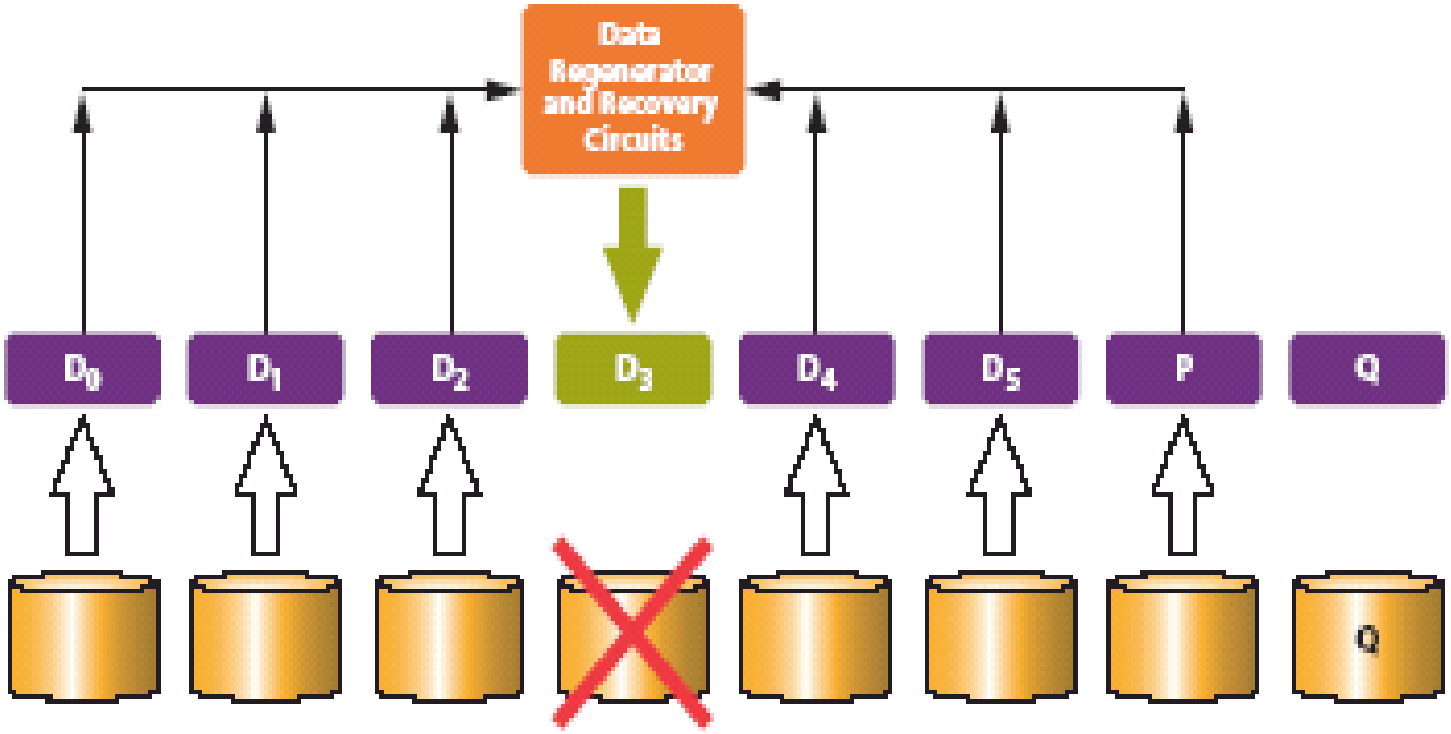
Q: Coefficients for locations and exclusive OR are used

$Q = A_0 * D_0 \oplus A_1 * D_1 \oplus A_2 * D_2 \oplus A_3 * D_3 \oplus A_4 * D_4 \oplus A_5 * D_5$

$D_0 - D_5$: Data

$A_0 - A_5$: Coefficients for locations

Recovering Data from a Failed Disk Drive



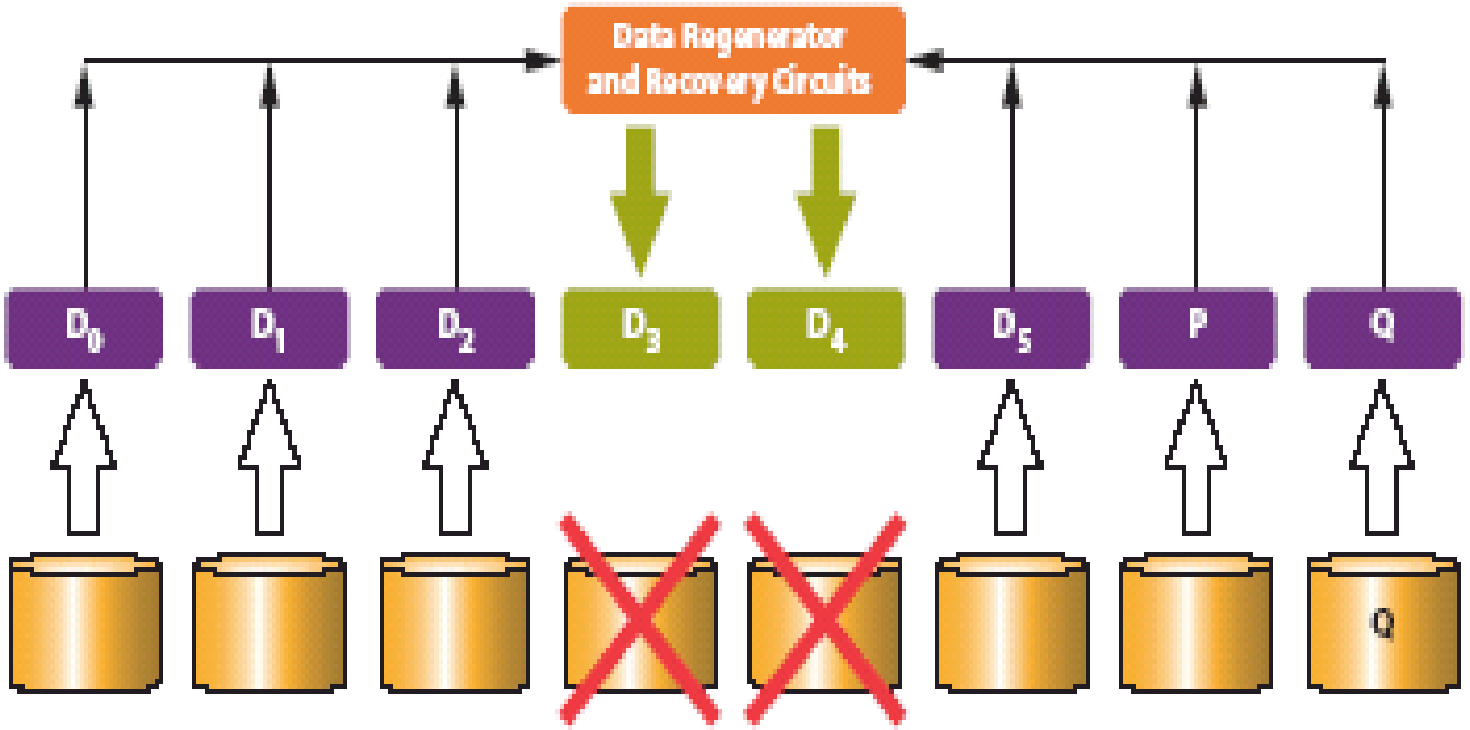
In this case, a disk drive storing data D3 fails and is recovered under RAID-6.

Formula:

$D_3 = D_0 \oplus D_1 \oplus D_2 \oplus D_4 \oplus D_5 \oplus P$

D_3 is calculated by solving formula.

Recovering data from two failed disk drives



In this case, a disk drive storing data D3 and D4 is recovered under RAID-6.

Formula 1 and formula 2:

$D_0 \oplus D_1 \oplus D_2 \oplus D_3 \oplus D_4 \oplus D_5 = P$

$A_0 D_0 \oplus A_1 D_1 \oplus A_2 D_2 \oplus A_3 D_3 \oplus A_4 D_4 \oplus A_5 D_5 = Q$

D3 and D4 are calculated by solving the simultaneous equations 1 and 2.

RAID 7

- **Common Name(s):** RAID 7.
- **Technique(s) Used:** **Asynchronous, cached striping with dedicated parity**.
- **Description:** Unlike the other RAID levels, RAID 7 isn't an open industry standard; it is really a trademarked marketing term of **Storage Computer Corporation**, used to describe their proprietary RAID design. (I debated giving it a page alongside the other RAID levels, but since it *is* used in the market, it deserves to be explained; that said, information about it appears to be limited.) RAID 7 is based on concepts used in RAID levels 3 and 4, but greatly enhanced to address some of the limitations of those levels. Of particular note is the inclusion of a **great deal of cache** arranged into multiple levels, and a **specialized real-time processor** for managing the array asynchronously. This hardware support--especially the cache--allow the array to handle many simultaneous operations, greatly improving performance of all sorts while maintaining fault tolerance. In particular, RAID 7 offers much improved random read and write performance over RAID 3 or RAID 4 because the dependence on the dedicated parity disk is greatly reduced through the added hardware. The increased performance of RAID 7 of course comes at a cost. This is an expensive solution, made and supported by only one company.

RAID 7

- **Controller Requirements:** Requires a specialized, expensive, proprietary controller.
- **Hard Disk Requirements:** Depends on implementation.
- **Array Capacity:** Depends on implementation.
- **Storage Efficiency:** Depends on implementation.
- **Fault Tolerance:** Very good.
- **Availability:** Excellent, due to use of multiple hot spares.
- **Degradation and Rebuilding:** Better than many RAID levels due to hardware support for parity calculation operations and multiple cache levels.

RAID 7

- **Random Read Performance:** Very good to excellent. The extra cache can often supply the results of the read without needing to access the array drives.
- **Random Write Performance:** Very good; substantially better than other single RAID levels doing striping with parity.
- **Sequential Read Performance:** Very good to excellent.
- **Sequential Write Performance:** Very good.

RAID 7

- **Cost:** Very high.
- **Special Considerations:** RAID 7 is a proprietary product of a single company; if it is of interest then you should contact Storage Computer Corporation for more details on the specifics of implementing it. All the caching creates potential vulnerabilities in the event of power failure, making the use of one or more UPS units mandatory.
- **Recommended Uses:** Specialized high-end applications requiring absolutely top performance and willing to live with the limitations of a proprietary, expensive solution. For most users, a multiple RAID level solution like RAID 1+0 will probably yield comparable performance improvements over single RAID levels, at lower cost.