Multiple (Nested) RAID Levels

- The **single RAID levels** have distinct advantages and disadvantages, which is why most of them are used in various parts of the market to address different application requirements. It wasn't long after RAID began to be implemented that engineers looked at these RAID levels and began to wonder if it might be possible to get some of the advantages of more than one RAID level by designing arrays that use a combination of techniques. These RAID levels are called variously ***multiple*, *nested*, or *multi*-RAID levels**. They are also sometimes called ***two-dimensional***, in reference to the two-dimensional schematics that are used to represent the application of two RAID levels to a set of disks, as you shall see.

- **Multiple RAID** levels are most commonly **used** to **improve performance**, and they do this well. Nested RAID levels typically provide better performance characteristics than either of the single RAID levels that comprise them. The most commonly combined level is RAID 0, which is often mixed with redundant RAID levels such as 1, 3 or 5 to provide fault tolerance while exploiting the performance advantages of RAID 0. There is never a "free lunch", and so with multiple RAID levels what you pay is a cost in complexity: many drives are required, management and maintenance are more involved, and for some implementations a high-end RAID controller is required.

Multiple (Nested) RAID Levels

- **Not all combinations** of RAID levels exist (which is good, because I'd get really bored of describing them all! :^)) Typically, the most popular multiple RAID levels are those that combine single RAID levels that complement each other with different strengths and weaknesses. Making a multiple RAID array **marrying RAID 4** to **RAID 5** wouldn't be the **best idea**, since they are so similar to begin with.
- In this section I take a look at some of the more common multiple RAID levels. Note that some of the multiple RAID levels discussed here are frequently used, but others are rarely implemented. In particular, for completeness I describe both the "X+Y" and "Y+X" configurations of each multiple level, when in some cases only one or the other is commonly made into product. For example, I know that RAID 50 (5+0) is an option in commercial RAID controllers, but I am not sure if anyone makes a RAID 05 solution. There may also be other combinations of RAID levels that I am not aware of.

RAID X+Y vs. RAID Y+X

- A multiple RAID level is generally created by taking a number of disks and **dividing** them into **sets**.
  - **Within each set** a **single RAID level** is applied to form a number of arrays. **(X)**
  - Then, the **second RAID level** is applied to the arrays to create a higher-level array. This is why these are sometimes called *nested* arrays. **(Y)**
- Since there are two levels, there are two ways they can be combined.
- The **choice** of which level is applied first and which second has an impact on some important array characteristics.

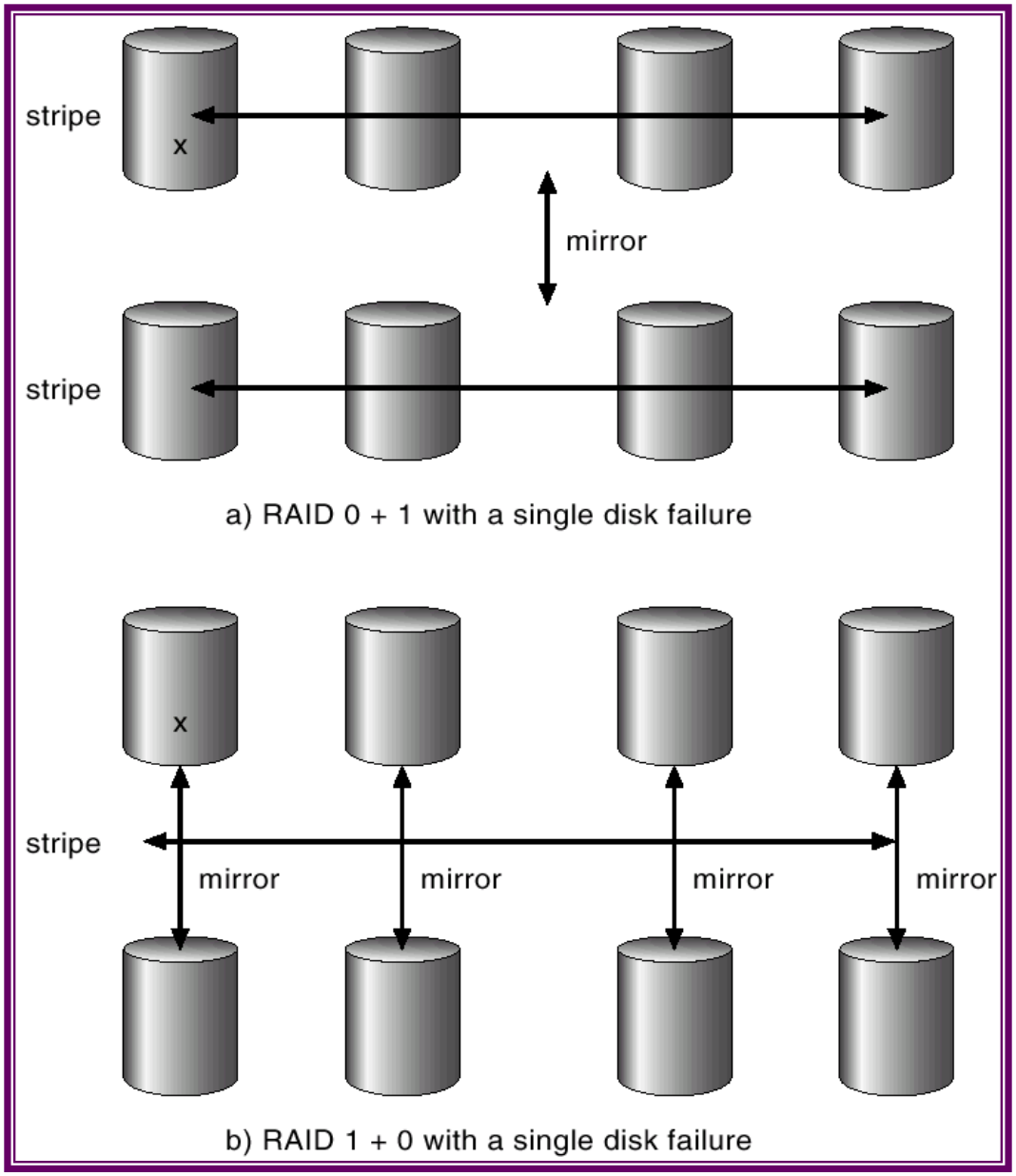
RAID 0+1 vs. RAID 1+0

- Let's take as an example multiple RAID employing RAID 0 and RAID 1 to create an array of ten disks. Much as we can define 10 to be 2*5 or 5*2, we can create our multiple RAID array two ways:

- **RAID 0, then RAID 1:** Divide the ten disks into 2 sets of 5. Turn each set into a RAID 0 array containing five disks, then mirror the two arrays. (Sometimes called a "mirror of stripes".)

- **RAID 1, then RAID 0:** Divide the ten disks into 5 sets of 2. Turn each set into a RAID 1 array, then stripe across the five mirrored sets. (A "stripe of mirrors").

RAID (0 + 1) and (1 + 0)



RAID X+Y v RAID Y+X

- In many respects, there *is* no difference between them:
- there is **no impact** on:
  - drive requirements
  - capacity
  - storage efficiency, and importantly
  - not much impact on performance.
- The **big difference** comes into play when we look at ***fault tolerance***. Most controllers implement multiple level RAID by forming a "super array" comprised of "sub-arrays" underneath it. In many cases the arrays that comprise the "super array"--often called *sets*--are considered to be logical "single units", which means that the controller only considers one of these "single units" to either be "up" or "down" as a whole. It will make use of redundancy features *within* a sub-array, but not *between* sub-arrays, even if the higher-level array means that drives in different sub-arrays will have the same data.
- That makes this sound much more complicated than it really is; it's much easier to explain with an example. Let's look at 10 drives and RAID 0+1 vs. RAID 1+0 again:

RAID X+Y v RAID Y+X

- **RAID 0+1:** We stripe together drives **1, 2, 3, 4 and 5** into RAID 0 **stripe set "A",** and drives **6, 7, 8, 9 and 10** into RAID 0 **stripe set "B".** We then mirror A and B using RAID 1. If one drive fails, say **drive #2**, then the entire stripe set "A" is lost, because RAID 0 has no redundancy; the RAID 0+1 array continues to chug along because the entire stripe set "B" is still functioning. However, at this point you are reduced to running what is in essence a straight RAID 0 array until drive #2 can be fixed. If in the meantime drive **#9** goes down, **you lose the entire array**.

- **RAID 1+0:** We mirror drives **1 and 2** to form RAID 1 mirror set "A"; **3 and 4** become "B"; **5 and 6** become "C"; **7 and 8** become "D"; and **9 and 10** become "E". We then do a RAID 0 stripe across sets A through E. If drive #2 fails now, only mirror set "A" is affected; it still has drive #1 so it is fine, and the RAID 1+0 array continues functioning. If while drive #2 is being replaced drive #9 fails, the array is fine, because drive #9 is in a different mirror pair from #2. Only two failures in the same mirror set will cause the array to fail, so in theory, five drives can fail--as long as they are all in different sets--and the array would still be fine.

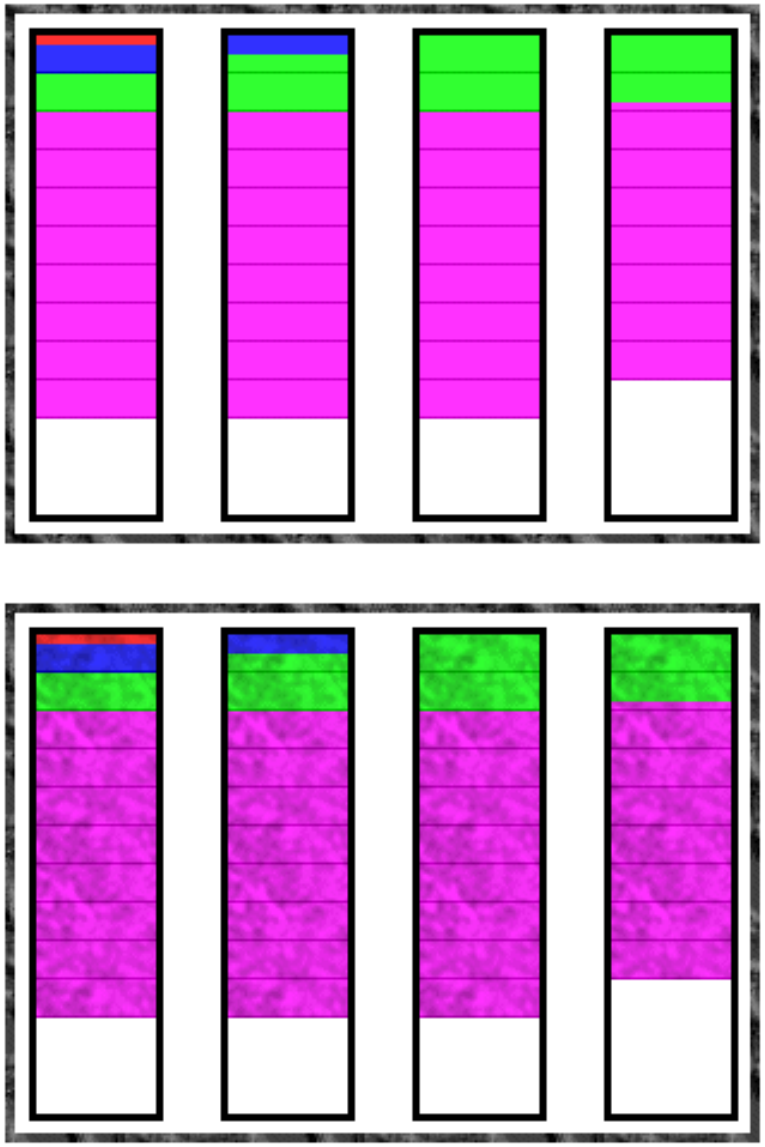
Fault tolerance and Rebuild

- Clearly, RAID 1+0 is more robust than RAID 0+1. Now, if the controller running RAID 0+1 **were smart**, when drive #2 failed it would continue striping to the other four drives in stripe set "A", and if drive #9 later failed it would "realize" that it could use drive #4 in its stead, since it should have the same data. This functionality would theoretically make RAID 0+1 just as fault-tolerant as RAID 1+0. Unfortunately, most controllers ***aren't* that smart**. It pays to ask specific questions about how a multiple RAID array implementation handles multiple drive failures, but in general, a controller won't swap drives between component sub-arrays unless the manufacturer of the controller specifically says it will.
- The same impact on fault tolerance applies to **rebuilding**. Consider again the example above. In RAID 0+1, if drive #2 fails, the data on **five hard disks will need to be rebuilt**, because the whole stripe set "A" will be wiped out. In RAID 1+0, **only drive #2** has to be rebuilt. Again here, the advantage is to RAID 1+0.

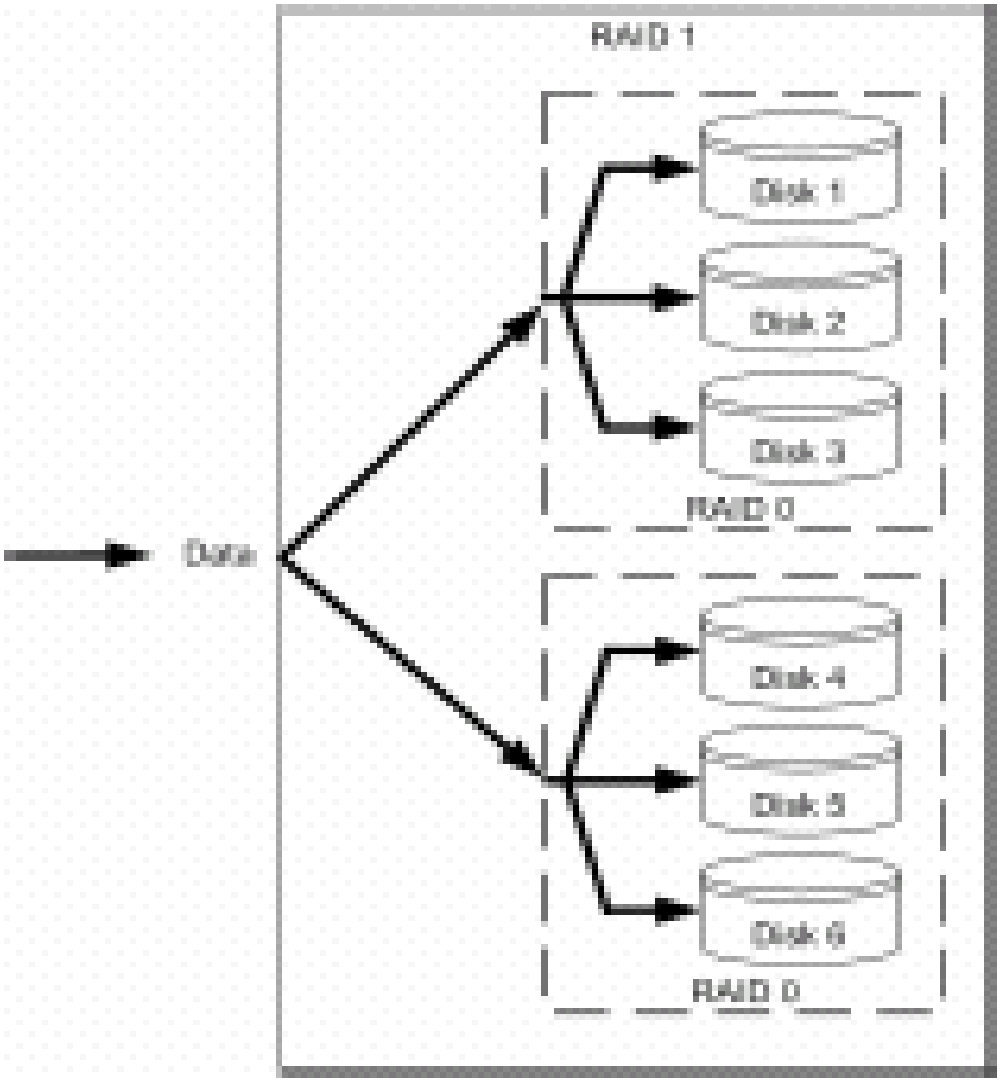
RAID Levels 0+1 (01) and 1+0 (10)

- **Common Name(s):** RAID 0+1, 01, 0/1, "mirrored stripes", "mirror of stripes"; RAID 1+0, 10, 1/0, "striped mirrors", "stripe of mirrors". Labels are often used incorrectly; verify the details of the implementation if the distinction between 0+1 and 1+0 is important to you.
- **Technique(s) Used:** **Mirroring** and **striping without parity**.
- **Description:** The most popular of the multiple RAID levels, RAID 01 and 10 combine the best features of striping and mirroring to yield large arrays with high performance in most uses and superior fault tolerance. RAID 01 is a mirrored configuration of two striped sets; RAID 10 is a stripe across a number of mirrored sets. RAID 10 and 01 have been increasing dramatically in popularity as hard disks become cheaper and the four-drive minimum is legitimately seen as much less of an obstacle. RAID 10 provides better fault tolerance and rebuild performance than RAID 01. Both array types provide very good to excellent overall performance by combining the speed of RAID 0 with the redundancy of RAID 1 without requiring parity calculations.

RAID Levels 0+1 (01) and 1+0 (10)



RAID 0+1



RAID 1+0

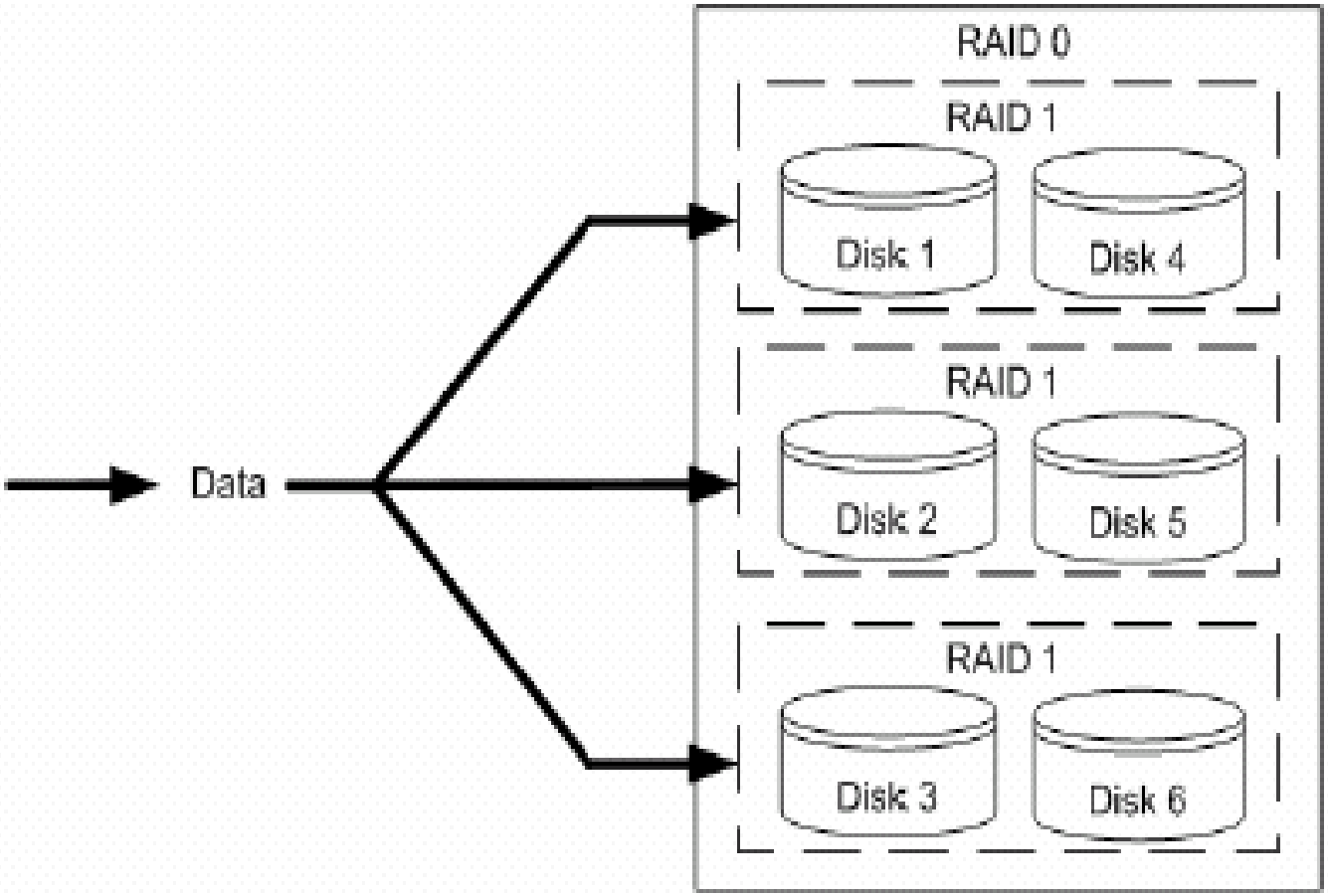


Figure 3.4: RAID 1+0 operation.

RAID 0+1; 1+0 Features

- **Controller Requirements:** Almost all hardware controllers will support one or the other of RAID 10 or RAID 01, but often not both. Even low-end cards will support this multiple level, usually RAID 01. High-end cards may support both 01 and 10.
- **Hard Disk Requirements:** An even number of hard disks with a minimum of four; maximum dependent on controller. All drives should be identical.
- **Array Capacity:** (Size of Smallest Drive) * (Number of Drives) / 2.
- **Storage Efficiency:** If all drives are the same size, 50%.
- **Fault Tolerance:** Very good for RAID 01; excellent for RAID 10.
- **Availability:** Very good for RAID 01; excellent for RAID 10.
- **Degradation and Rebuilding:** **Relatively little for RAID 10**; can be more substantial for RAID 01.

RAID 0+1; 1+0 Features

- **Random Read Performance:** Very good to excellent.
- **Random Write Performance:** Good to very good.
- **Sequential Read Performance:** Very good to excellent.
- **Sequential Write Performance:** Good to very good.

RAID 0+1; 1+0 Features

- **Cost:** Relatively high due to large number of drives required and low storage efficiency (50%).

- **Special Considerations:** Low storage efficiency limits potential array capacity.

- **Recommended Uses:** Applications requiring **both high performance and reliability and willing to sacrifice capacity to get them**. This includes enterprise servers, moderate-sized database systems and the like at the high end, but also individuals using larger IDE/ATA hard disks on the low end. Often used in place of RAID 1 or RAID 5 by those requiring higher performance; may be used instead of RAID 1 for applications requiring more capacity.

RAID Levels 0+3 (03 or 53) and 3+0 (30)

- **Common Name(s):** The most confusing naming of any of the RAID levels. :^) In an ideal world, this level would be named RAID 0+3 (or 03) or RAID 3+0 (30). Instead, the number *53* is often used in place of 03 for reasons I have never been able to determine, and worse, 53 is often actually implemented as 30, not 03. As always, verify the details of the implementation to be sure of what you have.

- **Technique(s) Used:** **Byte striping** with **dedicated parity** combined **with block striping**.

- **Description:** RAID 03 and 30 (though often called 53 for a reason that utterly escapes me) combine byte striping, parity and block striping to create large arrays that are conceptually difficult to understand. :^) RAID 03 is formed by putting into a RAID 3 array a number of striped RAID 0 arrays; RAID 30 is more common and is formed by striping across a number of RAID 3 sub-arrays. The combination of parity, small-block striping and large-block striping makes analyzing the theoretical performance of this level difficult. In general, it provides performance better than RAID 3 due to the addition of RAID 0 striping, but closer to RAID 3 than RAID 0 in overall speed, especially on writes. RAID 30 provides better fault tolerance and rebuild performance than RAID 03, but both depend on the "width" of the RAID 3 dimension of the drive relative to the RAID 0 dimension: the more parity drives, the lower capacity and storage efficiency, but the greater the fault tolerance. See the examples below for more explanation of this.
- Most of the characteristics of RAID 0+3 and 3+0 are similar to those of RAID 0+5 and 5+0. RAID 30 and 03 tend to be better for large files than RAID 50 and 05.

RAID 30/03/53/35 Features

- **Controller Requirements:** Generally requires a high-end hardware controller.
- **Hard Disk Requirements:** Number of drives must be able to be factored into two integers, one of which must be 2 or higher and the other 3 or higher (you can make a RAID 30 array from 10 drives but not 11). Minimum number of drives is six, with the maximum set by the controller.
- **Array Capacity: For RAID 03**: (Size of Smallest Drive) * (Number of Drives In Each RAID 0 Set) * (Number of RAID 0 Sets - 1). **For RAID 30**: (Size of Smallest Drive) * (Number of Drives In Each RAID 3 Set - 1) * (Number of RAID 3 Sets).
- For example, the capacity of a RAID 03 array made of 15 x 18 GB drives arranged as 3x 5-drive RAID 0 sets would be 18 GB * 5 * (3-1) = 180 GB. The capacity of a RAID 30 array made of 21 18 GB drives arranged as three seven-drive RAID 3 sets would be 18 GB * (7-1) * 3 = 324 GB. The same 21 drives arranged as seven three-drive RAID 3 sets would have a capacity of 18 GB * (3-1) * 7 = "only" 252 GB.
- **Storage Efficiency: For RAID 03**: ((Number of RAID 0 Sets - 1) / Number of RAID 0 Sets). **For RAID 30:** ((Number of Drives In Each RAID 3 Set - 1) / Number of Drives In Each RAID 3 Set).
- Taking the same examples as above, the 15-drive RAID 03 array would have a storage efficiency of (3-1)/3 = 67%. The first RAID 30 array, configured as three seven-drive RAID 3 sets, would have a storage efficiency of (7-1)/7 = 86%, while the other RAID 30 array would have a storage efficiency of, again, (3-1)/3 = 67%.

RAID 30/03/53/35 Features

- **Array Capacity:**
- **For RAID 03**:
- (Size of Smallest Drive) * (Number of Drives In Each RAID 0 Set) * (Number of RAID 0 Sets - 1).
- **For RAID 30**:
- (Size of Smallest Drive) * (Number of Drives In Each RAID 3 Set - 1) * (Number of RAID 3 Sets).

- For example, the capacity of a **RAID 03** array made of **15 x 18 GB drives** arranged as **3 sets** x **5-drive RAID 0**, would be
- 18 GB * 5 * (3-1) = 180 GB.

- The capacity of a RAID 30 array made of **21 x 18 GB** drives arranged as
- **3 sets** x **7-drive RAID 3** would be
- 18 GB * (7-1) * 3 = 324 GB.
- The same 21 drives arranged as **7 sets** x **3-drive RAID 3** sets would have a capacity of 18 GB * (3-1) * 7 = "only" 252 GB.

RAID 30/03/53/35 Features

- **Storage Efficiency:**
- **For RAID 03**:
- ((Number of RAID 0 Sets - 1) / Number of RAID 0 Sets).
- **For RAID 30:**
- ((Number of Drives In Each RAID 3 Set - 1) / Number of Drives In Each RAID 3 Set).

- Taking the same examples as above, the 15-drive RAID 03 array would have a storage efficiency of (3-1)/3 = 67%.

- The first RAID 30 array, configured as
- **3 sets** x **7-drive RAID 3**, would have a storage efficiency of (7-1)/7 = 86%,

- while the other RAID 30 array would have a storage efficiency of, again,
- (3-1)/3 = 67%.

RAID 30/03/53/35 Features

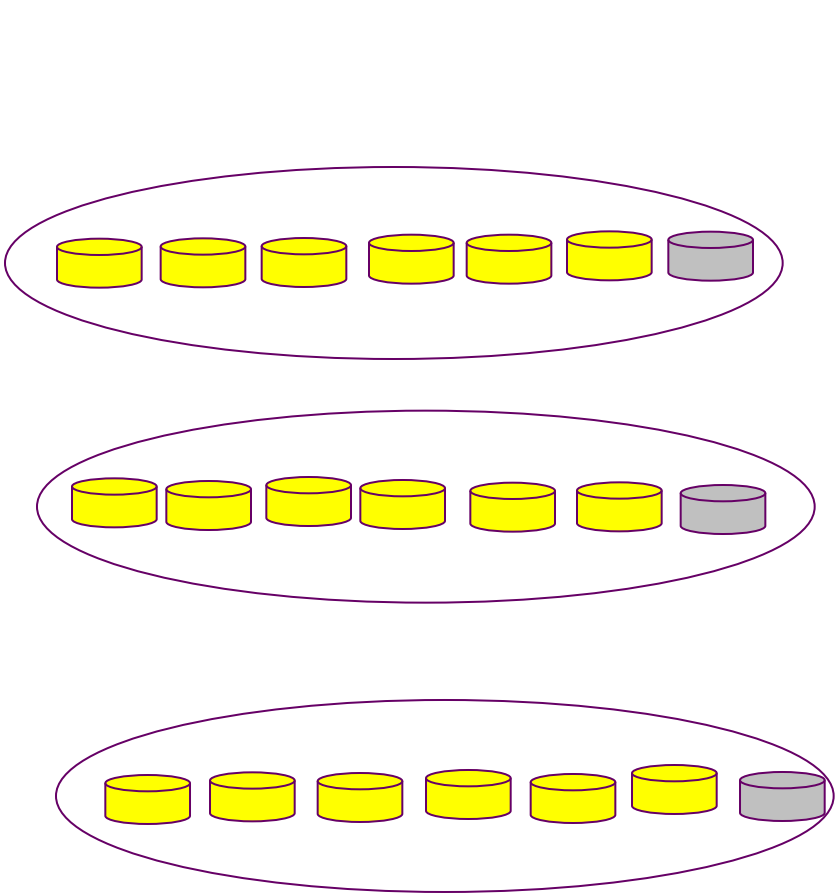
- **Fault Tolerance:** Good to very good, depending on whether it is RAID 03 or 30, and the number of parity drives relative to the total number. **RAID 30** will provide better fault tolerance than RAID 03.

- Consider the **2 different 21-drive RAID 30 arrays** mentioned above: the first one (3x 7-drive RAID 3 sets) has higher capacity and storage efficiency, but can only tolerate **three maximum potential drive failures**; the one with lower capacity and storage efficiency (7x3-drive RAID 3 sets) can handle as many as 7 , if they are in different RAID 3 sets. Of course few applications really require tolerance for seven independent drive failures! And of course, if those 21 drives were in a **RAID 03** array instead, failure of a second drive after one had failed and taken down one of the RAID 0 sub-arrays would crash the entire array.

- **Availability:** Very good to excellent.

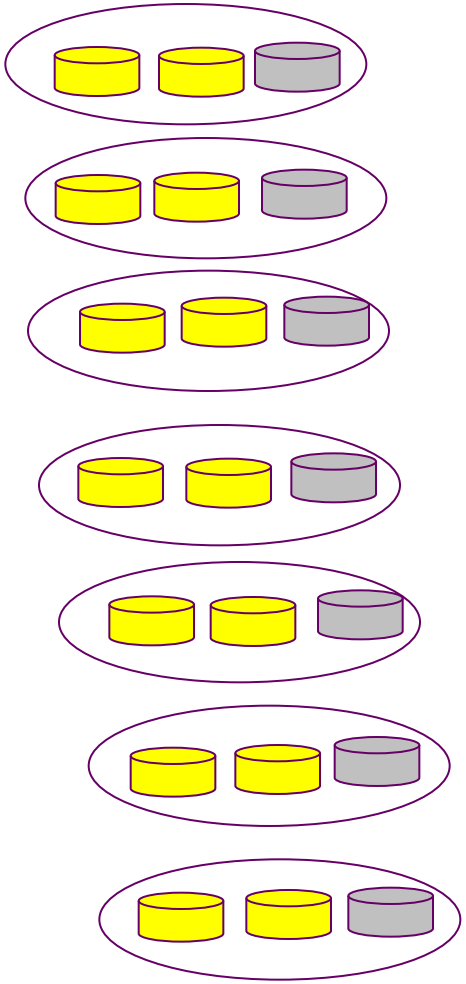
- **Degradation and Rebuilding:** Relatively **little** for RAID 30 (though more than RAID 10); can be **more substantial for RAID 03.**

3+0 (21 drives)



Best case: max 3 failures

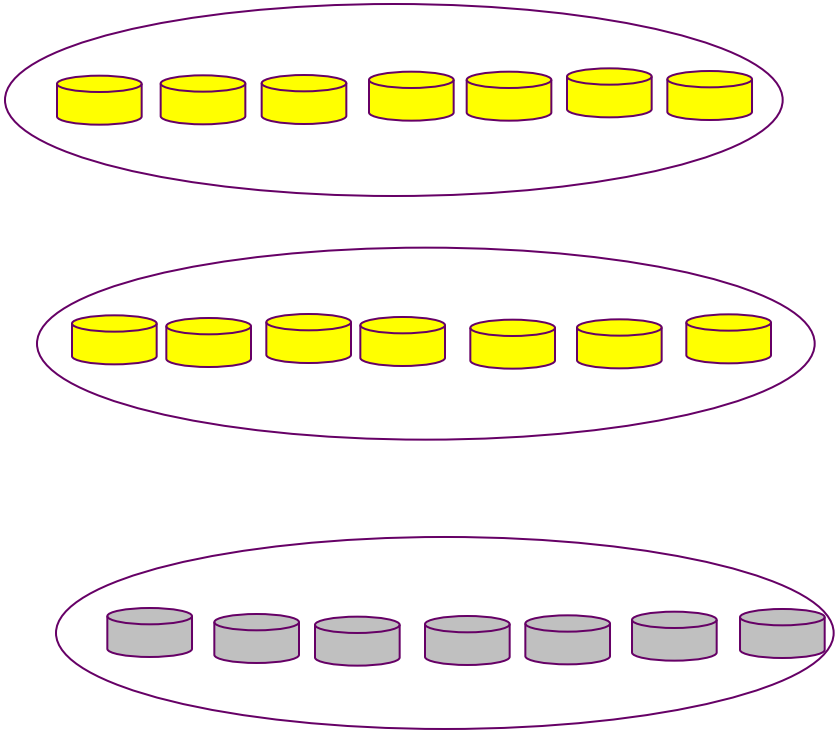
Worst case: 2 failures->fatal



Best case: max 7 failures

Worst case: 2 failures->fatal

0+3 (21 drives)



Best case: max 7 failures

Worst case: 2 failures->fatal

RAID 30/03/53/35 Features

- **Random Read Performance:** Very good, assuming RAID 0 stripe size is reasonably large.
- **Random Write Performance:** Fair.
- **Sequential Read Performance:** Very good to excellent.
- **Sequential Write Performance:** Good.

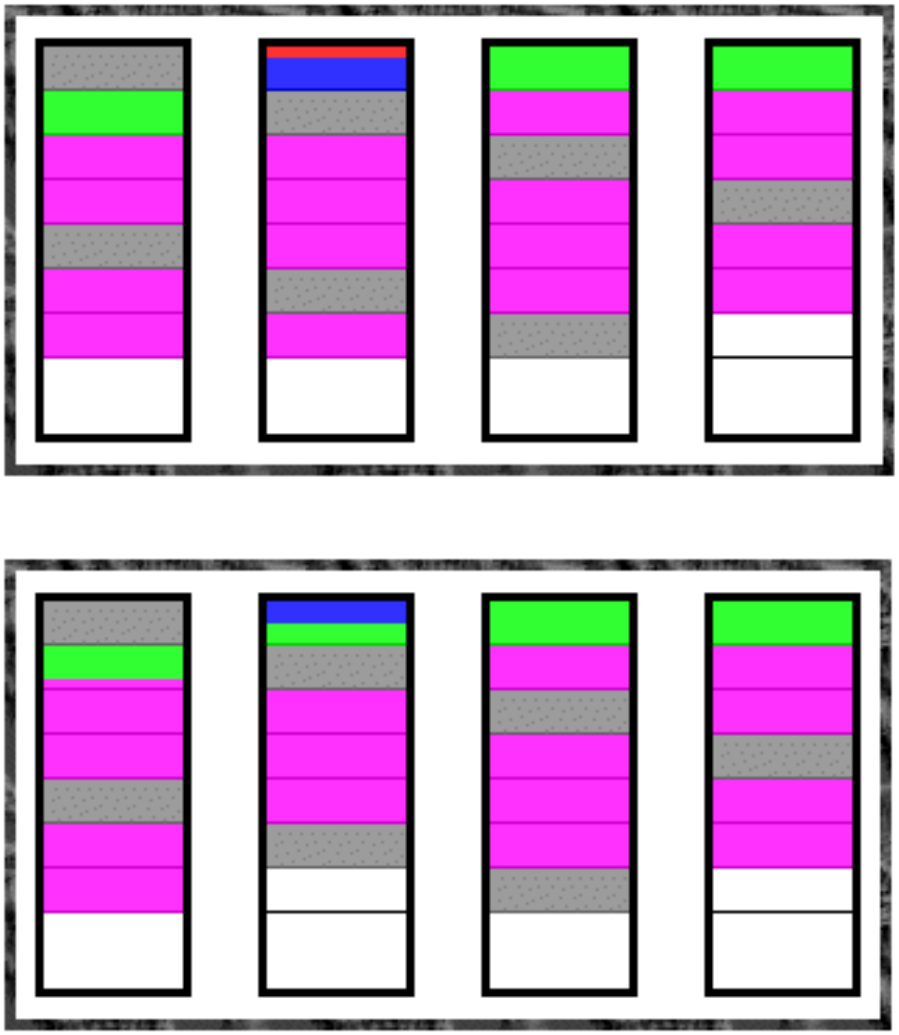
RAID 30/03/53/35 Features

- **Controller Requirements:** Generally requires a high-end hardware controller.
- **Cost:** Relatively high due to requirements for a hardware controller and a large number of drives; storage efficiency is better than RAID 10 however and no worse than any other RAID levels that include redundancy.
- **Special Considerations:** Complex and expensive to implement.
- **Recommended Uses:** Not as widely used as many other RAID levels. Applications include data that requires the **speed of RAID 0 with fault tolerance and high capacity**, such as **critical multimedia** data and **large database or file servers**. Sometimes used instead of RAID 3 to increase capacity as well as performance.

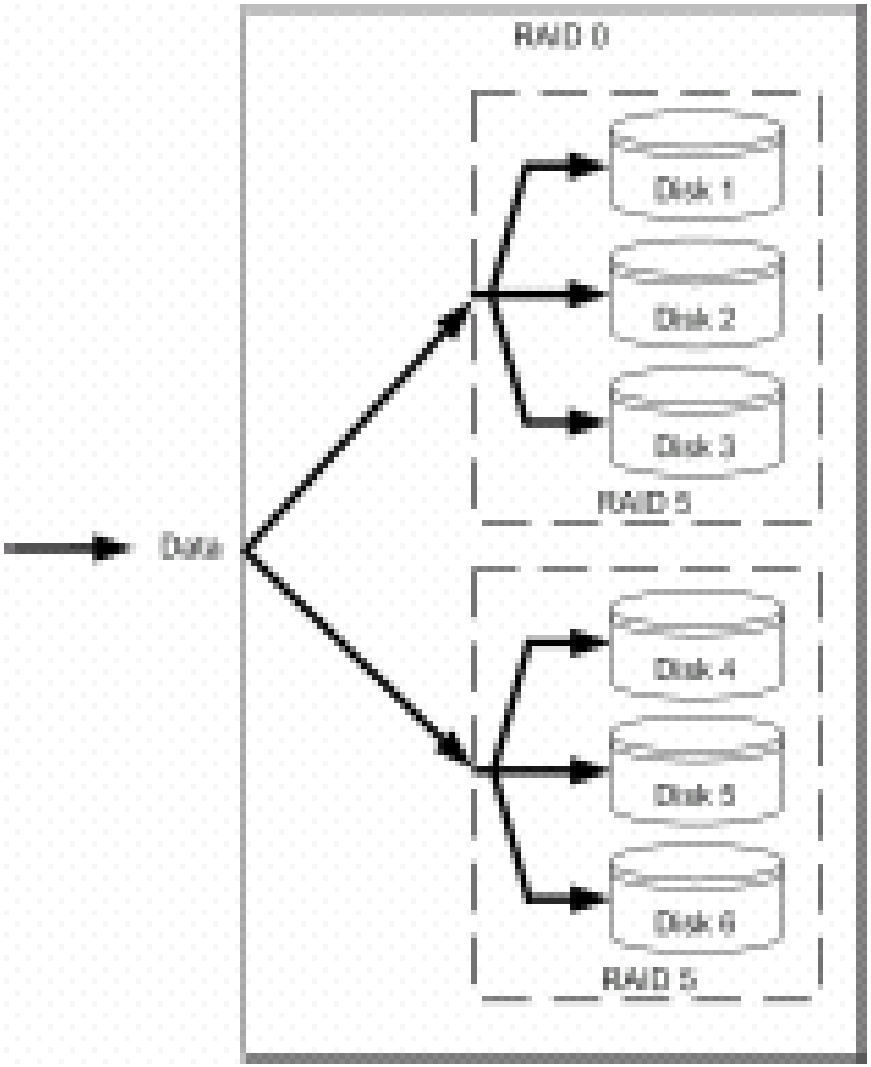
RAID Levels 0+5 (05) and 5+0 (50)

- **Common Name(s):** RAID 0+5 or 05; RAID 5+0 or 50. As with the other multiple RAID levels, verify the exact implementation instead of relying on the label.
- **Technique(s) Used:** **Block striping** with **distributed parity** combined with **block striping**.
- **Description:** RAID 05 and 50 form large arrays by combining the block striping and parity of RAID 5 with the straight block striping of RAID 0. RAID 05 is a RAID 5 array comprised of a number of striped RAID 0 arrays; it is less commonly seen than RAID 50, which is a RAID 0 array striped across RAID 5 elements. RAID 50 and 05 improve upon the performance of RAID 5 through the addition of RAID 0, particularly during writes. It also provides better fault tolerance than the single RAID level does, especially if configured as RAID 50.
- Most of the characteristics of RAID 05 and 50 are similar to those of RAID 03 and 30. RAID 50 and 05 tend to be preferable for transactional environments with smaller files than 03 and 30.

RAID Levels 0+5 (05) and 5+0 (50)



RAID 50



RAID 50/05 Features

- **Controller Requirements:** Generally requires a high-end hardware controller.
- **Hard Disk Requirements:** Number of drives must be able to be factored into two integers, one of which must be 2 or higher and the other 3 or higher (you can make a RAID 30 array from 6 drives but not 7). Minimum number of drives is six, with the maximum set by the controller.
- **Array Capacity:** Same as RAID 03 and 30. For RAID 05: (Size of Smallest Drive) * (Number of Drives In Each RAID 0 Set) * (Number of RAID 0 Sets - 1). For RAID 50: (Size of Smallest Drive) * (Number of Drives In Each RAID 5 Set - 1) * (Number of RAID 5 Sets).
- For example, the capacity of a RAID 05 array made of 15 18 GB drives arranged as three five-drive RAID 0 sets would be 18 GB * 5 * (3-1) = 180 GB. The capacity of a RAID 50 array made of 21 18 GB drives arranged as three seven-drive RAID 5 sets would be 18 GB * (7-1) * 3 = 324 GB. The same 21 drives arranged as seven three-drive RAID 5 sets would have a capacity of 18 GB * (3-1) * 7 = 252 GB.
- **Storage Efficiency:** Same as for RAID 03 and 30. For RAID 05: ((Number of RAID 0 Sets - 1) / Number of RAID 0 Sets). For RAID 50: ((Number of Drives In Each RAID 5 Set - 1) / Number of Drives In Each RAID 5 Set).
- Taking the same examples as above, the 15-drive RAID 05 array would have a storage efficiency of (3-1)/3 = 67%. The first RAID 50 array, configured as three seven-drive RAID 5 sets, would have a storage efficiency of (7-1)/7 = 86%, while the other RAID 50 array would have a storage efficiency of (3-1)/3 = 67%.

RAID 50/05 Features

- **Array Capacity:** Same as RAID 03 and 30.

- **For RAID 05:**
- (Size of Smallest Drive) * (Number of Drives In Each RAID 0 Set) * (Number of RAID 0 Sets - 1).

- **For RAID 50:**
- (Size of Smallest Drive) * (Number of Drives In Each RAID 5 Set - 1) * (Number of RAID 5 Sets).

- For example, the capacity of a RAID 05 array made of 15 x 18 GB drives arranged as **3 sets** x **5-drive RAID 0** sets would be 18 GB * 5 * (3-1) = 180 GB.

- The capacity of a RAID 50 array made of 21 x 18 GB drives arranged as
- **3 sets** x **7-drive RAID 5 sets** would be 18 GB * (7-1) * 3 = 324 GB.
- The same 21 drives arranged as **7sets** x **3-drive RAID 5** sets would have a capacity of 18 GB * (3-1) * 7 = 252 GB.

RAID 50/05 Features

- **Storage Efficiency:** Same as for RAID 03 and 30.
- **For RAID 05:**
- ((Number of RAID 0 Sets - 1) / Number of RAID 0 Sets).

- **For RAID 50:**
- ((Number of Drives In Each RAID 5 Set - 1) / Number of Drives In Each RAID 5 Set).

- Taking the same examples as above, the 15-drive RAID 05 array would have a storage efficiency of (3-1)/3 = 67%.

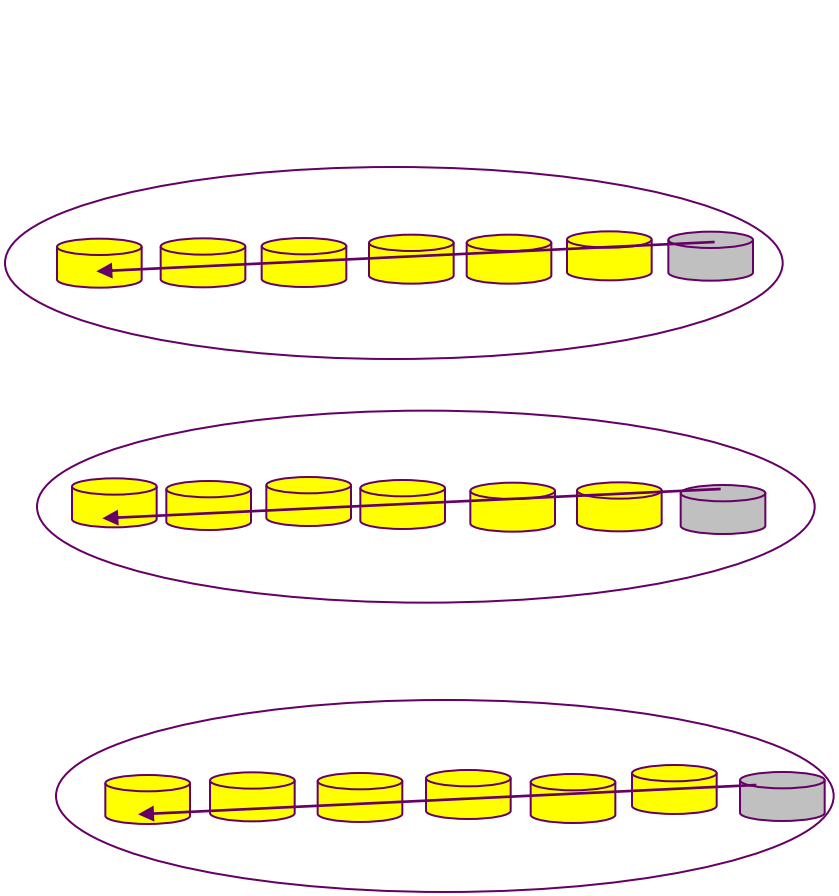
- The first RAID 50 array, configured as three seven-drive RAID 5 sets, would have a storage efficiency of (7-1)/7 = 86%,

- while the other RAID 50 array would have a storage efficiency of (3-1)/3 = 67%.

RAID 50/05 Features

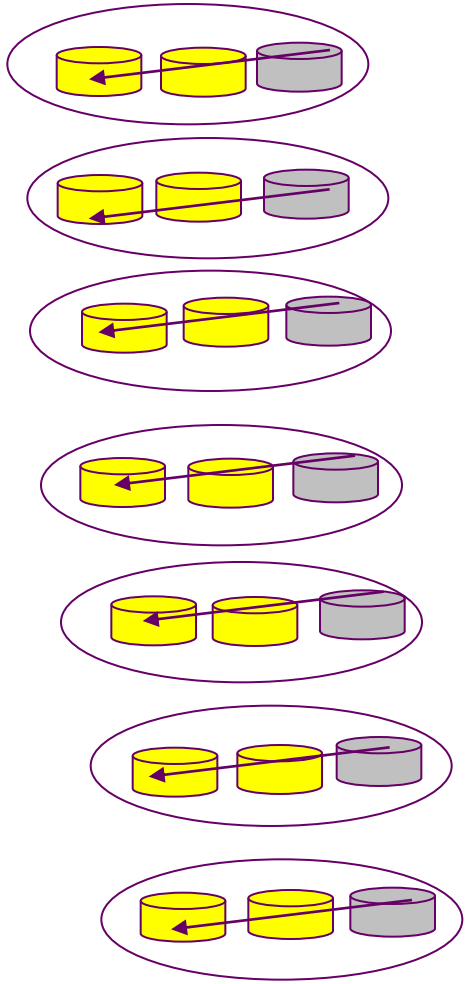
- **Fault Tolerance:** Same as for RAID 03 and 30. Good to very good, depending on whether it is RAID 05 or 50, and the number of parity drives relative to the total number. **RAID 50** will provide better fault tolerance than **RAID 05**.
- Consider the two different **21-drive RAID 50 arrays** mentioned above: the first one (3x 7-drive RAID 5 sets) has higher capacity and storage efficiency, but can only tolerate **3 maximum potential drive failures**; the one with lower capacity and storage efficiency (**7 x 3-drive RAID 5 sets**) can handle as many as **7** , if they are in different RAID 5 sets. Of course few applications really require tolerance for seven independent drive failures! And of course, if those 21 drives were in a **RAID 05** array instead, **failure** of a second drive after one had failed and taken down one of the RAID 0 sub-arrays would crash the entire array.
- **Availability:** Very good to excellent.
- **Degradation and Rebuilding:** Moderate for RAID 50; worse for RAID 05.
- **Random Read Performance:** Very good to excellent.

5+0 (21 drives)



Best case: max 3 failures

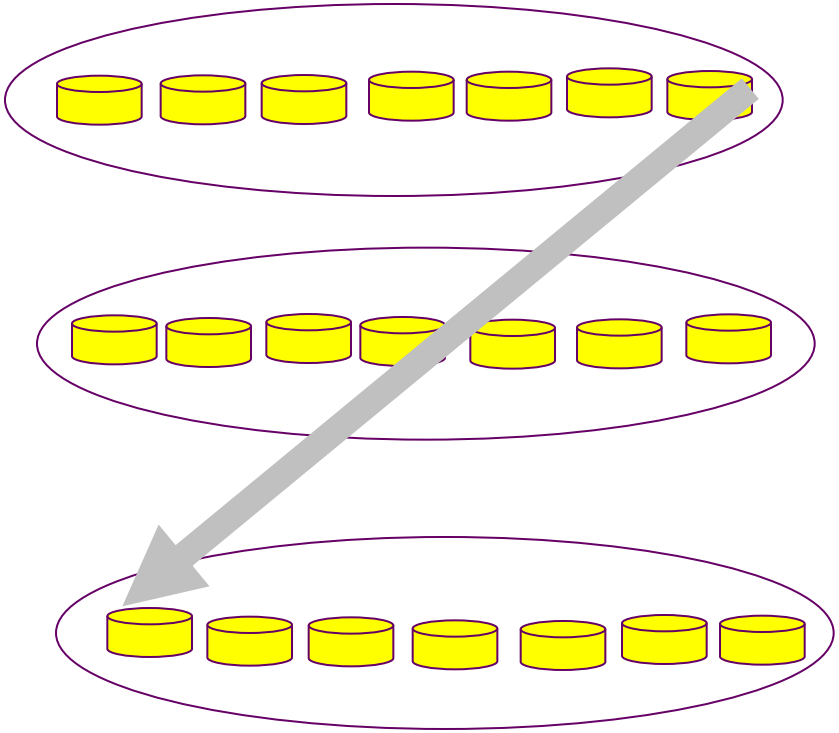
Worst case: 2 failures->fatal



Best case: max 7 failures

Worst case: 2 failures->fatal

0+5 (21 drives)



Best case: max 7 failures

Worst case: 2 failures->fatal

RAID 50/05 Features

- **Random Read Performance:** Very good to excellent.
- **Random Write Performance:** Good.
- **Sequential Read Performance:** Very good.
- **Sequential Write Performance:** Good.

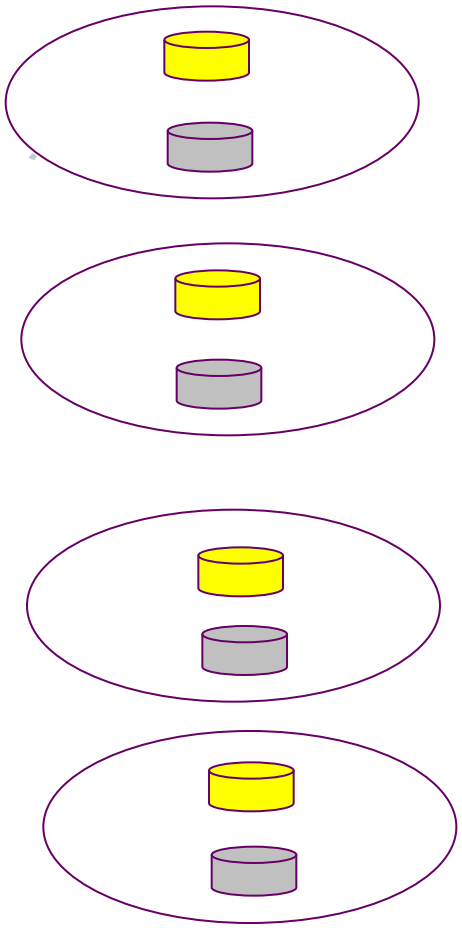
RAID 50/05 Features

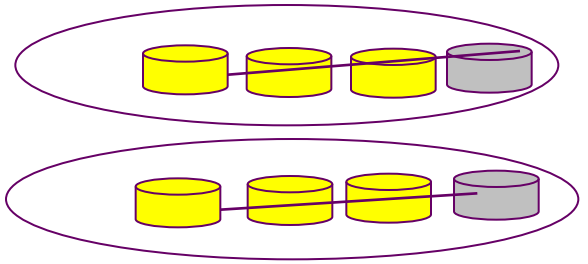
- **Controller Requirements:** Generally requires a high-end hardware controller.
- **Cost:** Relatively high due to requirements for a hardware controller and a large number of drives; storage efficiency is better than RAID 10 however and no worse than any other RAID levels that include redundancy.
- **Special Considerations:** Complex and expensive to implement.
- **Recommended Uses:** Applications that require high fault tolerance, capacity and **random positioning performance**. Not as widely used as many other RAID levels. Sometimes used instead of RAID 5 to increase capacity. Sometimes used for large databases.

RAID Levels 1+5 (15) and 5+1 (51)

- **Common Name(s):** RAID 1+5 or 15; RAID 5+1 or 51. "Common" is a bit of a stretch with this level, as it is less common than probably any other, so it's important to verify the details of each implementation.
- **Technique(s) Used:** **Mirroring** (or duplexing) combined with **block striping** with **distributed parity**.
- **Description:** RAID 1+5 and 5+1 might be sarcastically called "the RAID levels for the truly paranoid". :^) The only configurations that use both redundancy methods, mirroring and parity, this "belt and suspenders" technique is designed to maximize fault tolerance and availability, at the expense of just about everything else. A RAID 15 array is formed by creating a striped set with parity using multiple mirrored pairs as components; it is similar in concept to RAID 10 except that the striping is done with parity. Similarly, RAID 51 is created by mirroring entire RAID 5 arrays and is similar to RAID 01 except again that the sets are RAID 5 instead of RAID 0 and hence include parity protection. Performance for these arrays is good but not very high for the cost involved, nor relative to that of other multiple RAID levels.
- The fault tolerance of these RAID levels is truly amazing; an **8-drive RAID 15** array can tolerate the failure of **any 3 drives simultaneously**; an eight-drive **RAID 51** array can also handle 3 **and even as many as *5***, as long as at least one of the mirrored RAID 5 sets has no more than one failure! The price paid for this resiliency is complexity and cost of implementation, and very low storage efficiency.
- The RAID 1 component of this nested level may in fact use duplexing instead of mirroring to add even more fault tolerance.

1+5 (8 drives) or 5+1





Best case: max 5 failures

Any 3

any 3 drives simultaneously
Best case 5

RAID Levels 1+5 (15) and 5+1 (51) Features

- **Controller Requirements:** Requires at least a high-end controller; may in fact require multiple systems and/or specialized hardware or software. RAID 51 is sometimes implemented as a "hybrid" of hardware and software RAID, by doing software mirroring at the operating system level over a pair of RAID 5 controllers, thus implementing duplexing for even higher fault tolerance. In theory this could be done with Windows NT software mirroring and a pair of hardware RAID 5 cards, if you could set it all up to work together properly.
- **Hard Disk Requirements:** An even number of hard disks with a minimum of six; maximum dependent on controller. All drives should be identical.
- **Array Capacity: (Size of Smallest Drive) * ((Number of Drives / 2) - 1).** So an array with ten 18 GB drives would have a capacity of 18 GB * ((10/2) - 1) = just 72 GB.
- **Storage Efficiency:** Assuming all drives are the same size, ((Number of Drives / 2) - 1) / (Number of Drives). In the example above, efficiency is 40%. **This is the worst storage efficiency of any RAID level**; a six-drive RAID 15 or 51 array would have a storage efficiency of just 33%!
- **Fault Tolerance:** Excellent; by far the best of any level.
- **Availability:** Excellent.
- **Degradation and Rebuilding:** Can be substantial.

RAID Levels 1+5 (15) and 5+1 (51) Features

- **Random Read Performance:** Very good.
- **Random Write Performance:** Good.
- **Sequential Read Performance:** Very good.
- **Sequential Write Performance:** Good.

RAID Levels 1+5 (15) and 5+1 (51) Features

- **Cost:** Very high. An uncommon solution requiring a lot of storage devices for relatively low capacity, and possibly additional hardware or software.

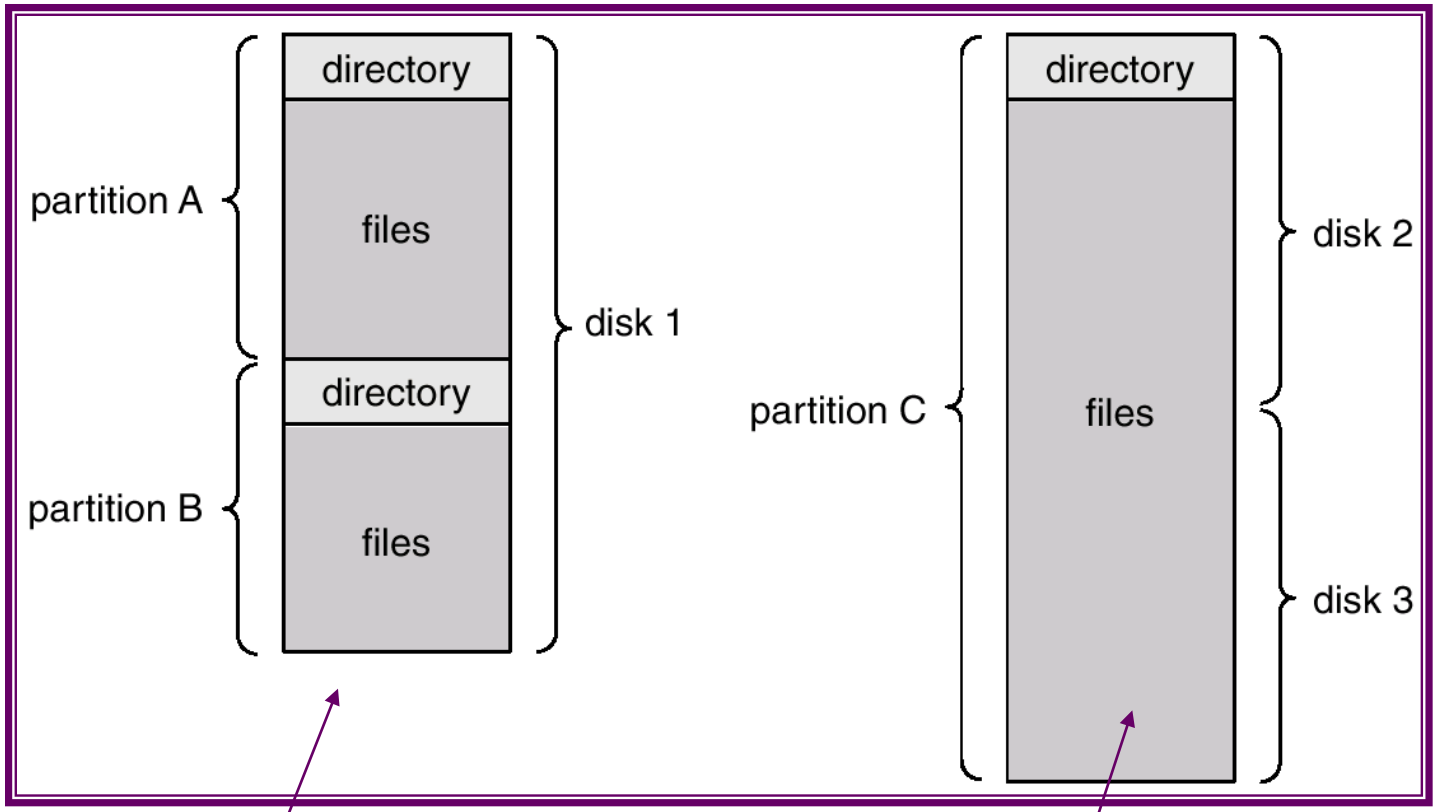
- **Special Considerations:** Complex and very expensive to implement.

- **Recommended Uses:** Critical applications requiring very high fault tolerance. In my opinion, if you get to the point of needing this much fault tolerance this badly, you should be looking beyond RAID to **remote mirroring, clustering or other redundant server setups**; **RAID 10 provides most of the benefits** with **better performance and lower cost**. Not widely implemented.

"Just A Bunch Of Disks" (JBOD)

- If you have some disks in a system that you decide not to configure into a RAID array, what do you do with them? Traditionally, they are left to act as independent drive volumes within the system, and that's how many people in fact use two, three or more drives in a PC. In some applications, however, it is desirable to be able to use all these disks as if they were one single volume. The proper term for this is *spanning*; the pseudo-cutesy term for it, clearly chosen to contrast against "redundant array of inexpensive disks", is *Just A Bunch Of Disks* or *JBOD*. How frightfully clever.
- JBOD isn't really RAID at all, but I discuss it here since it is sort of a "third cousin" of RAID...
- JBOD can be thought of as the opposite of partitioning: while partitioning chops single drives up into smaller logical volumes, **JBOD combines drives into larger logical volumes**. **It provides no fault tolerance**, nor does it **provide any improvements in performance compared** to the independent use of its constituent drives. (In fact, **it arguably hurts performance**, by making it more difficult to use the underlying drives concurrently, or to optimize different drives for different uses.)

A Typical File-system Organization



disk partitioning

JBOD=

Just Bunch of Disk

JBOD v RAID0

- When you look at it, JBOD doesn't really have a lot to recommend it. It still requires a controller card or software driver, which means that almost any system that can do JBOD can also do RAID 0, and RAID 0 has significant performance advantages over JBOD. Neither provide fault tolerance, so that's a wash. There are only **2 possible advantages of JBOD over RAID 0**:

- **Avoiding Drive Waste:** If you have a number of odd-sized drives, JBOD will let you combine them into a single unit without loss of any capacity; a **10 GB drive and 30 GB** would combine to make a 40 GB JBOD volume but only a 20 GB RAID 0 array. This may be an issue for those expanding an existing system, though with drives so cheap these days it's a relatively small advantage.

- **Easier Disaster Recovery:** If a disk in a RAID 0 volume dies, the data on every disk in the array is essentially destroyed because all the files are striped; if a drive in a JBOD set dies then it may be easier to recover the files on the other drives (but then again, it might not, depending on how the operating system manages the disks.) Considering that you should be doing regular backups regardless, and that even under JBOD recovery can be difficult, this too is a minor advantage.

- **Note:** Some companies use the term "spanning" when they really mean striping, so watch out for that!

RAID Configuration and Implementation

- RAID Controllers and Controller Features
- RAID Interfaces
- Multiple Channels and Throughput Issues
- RAID Hard Disk Drive Requirements
- RAID Management

RAID Controllers and Controller Features

Hardware RAID

- **Hardware RAID volumes** are set up using a hardware RAID controller card. This setup requires the disks in the array to be connected to the controller card.
- **Good hardware controllers** are in many ways **like miniature computers, incorporating dedicated processors that exceed the power** of **processors that ran entire PCs just** a **few years ago**.

Software RAID

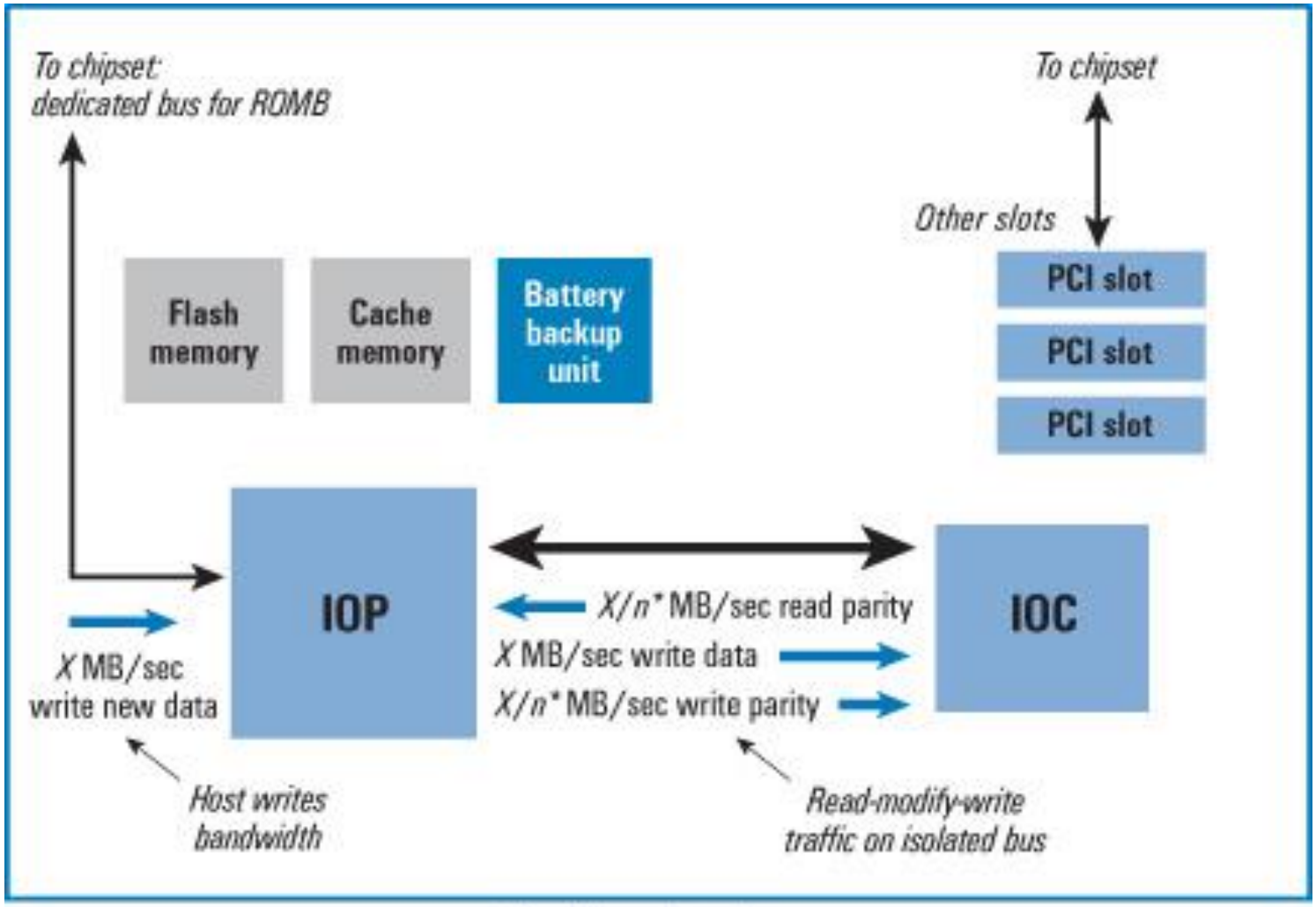
- With software RAID, disks are managed through either the **OS** or a **third-party application**.

Hardware RAID

- **Internal RAID Controller**
  - used in lower-end systems.
  - A specialized RAID controller is installed into the PC or server, and the array drives are connected to it.
    - **SCSI RAID**
    - **ATA RAID**
    - **SATA/SAS RAID**
    - **Motherboard ATA/SATA RAID**

- **External RAID Controller:**
  - used in higher-end design
  - RAID controller is removed to a separate box.
  - Interface to system
    - **FC**
    - **SCSI**

RAID controller: block diagram

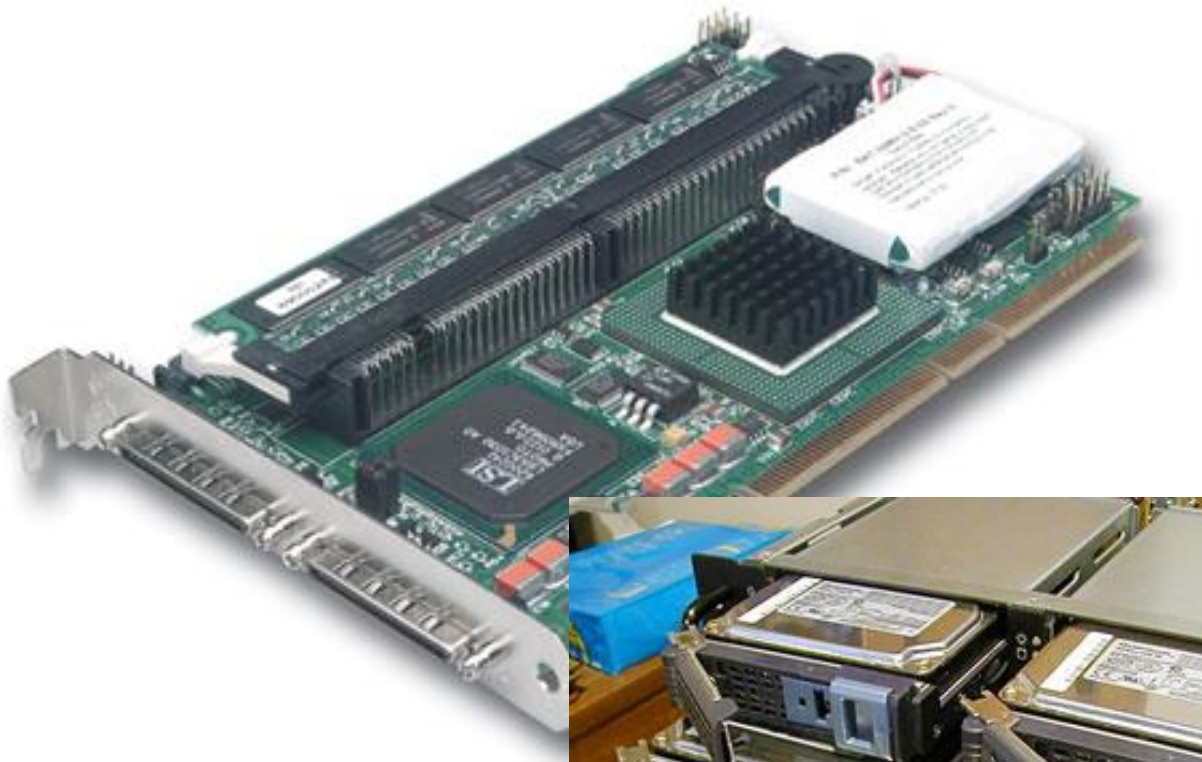


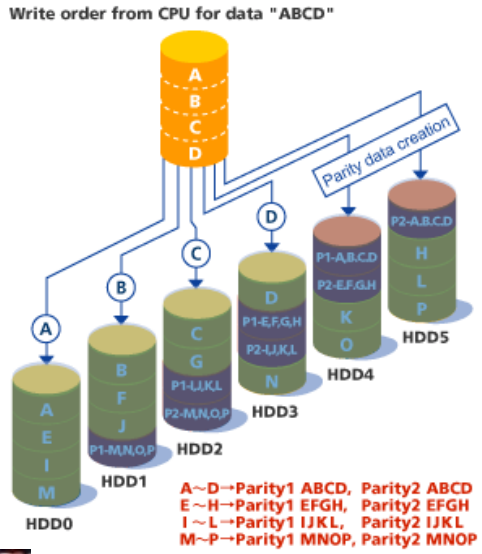
Motherboard top view
(several motherboard components not shown)

*n = number of disks in RAID array

RAID controllers

RAID controllers





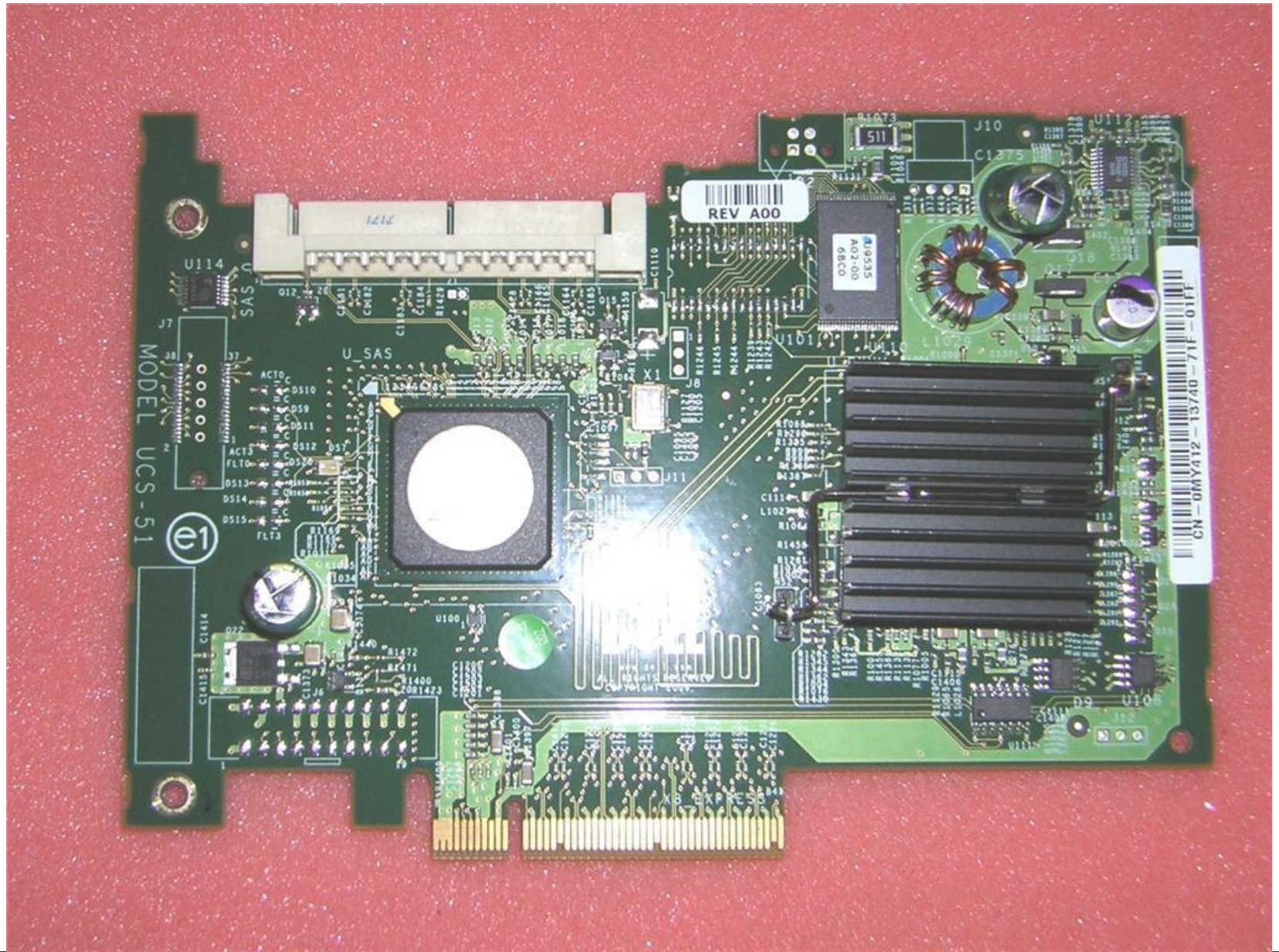
RAID adv

RAID controllers

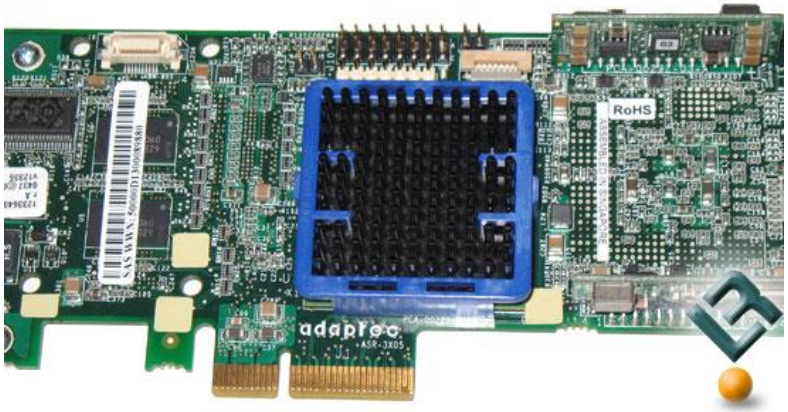
RAID controllers

RAID controllers

RAID controllers



RAID controllers



Software RAID

- Software RAID is just like hardware RAID, except that it **uses software instead of hardware.**

- All kidding aside, that pretty much *is* what software RAID is about. **Instead** of **using** a **dedicated hardware controller** to perform the various functions required to implement a RAID array, these functions are performed by the system processor using **special software routines**. Since array management is a low-level activity that must be performed "underneath" the other software that runs on the PC,

- **software RAID** usually is **implemented** at the **operating system level.**
  - Windows NT and Windows 2000,
  - as well as
  - most of the various flavors of UNIX
- support some RAID levels in software

OS RAID-sw support

- **Windows OSs** natively support software RAID, which can be configured using the Disk Management utility, which is a part of the **Computer Management Microsoft Management Console (MMC**).
- Using Disk Management, you can configure RAID 0, 1, and 5

- **With Linux OSs**, software RAID 0, 1, and 5 can be configured using the **Disk Druid tool** during a GUI installation of the OS.
- If the OS is already installed, you can use the **Raidtools** package to configure and manage software RAID.
- Although increasing performance and availability of disks through RAID is an important part of enterprise file serving, there are more elements of the data path that must be considered as well.

Hardware RAID v Software RAID

- **Software RAID benefits**

- **Cost:** If you are already running an operating system that supports software RAID, you have no additional costs for controller hardware; you may need to add more system memory to the system, however.

- **Simplicity:** You don't have to install, configure or manage a hardware RAID controller.

- **Duplexing:** Duplexed RAID 1 can sometimes be implemented in software RAID but *not* in hardware RAID, depending on the controller.

Hardware RAID v Software RAID

- **Hardtware RAID benefits**
- **Performance:** The best-known drawback of software RAID is that it provides lower overall system performance than hardware RAID. The reason is obvious: cycles are "stolen" from the CPU to manage the RAID array. In reality, this slowdown isn't *that* excessive for simple RAID levels like RAID 1, but it can be substantial, particularly with any RAID levels that involve striping with parity (like RAID 5).
- **Boot Volume Limitations:** Since the operating system has to be running to enable the array, this means the operating system cannot boot from the RAID array! This requires a separate, non-RAID partition to be created for the operating system, segmenting capacity, lowering performance further and slowing boot time.
- **Level Support:** Software RAID is usually limited to RAID levels 0, 1 and 5. More "interesting" RAID levels require hardware RAID (with the exception of duplexing, mentioned above.)
- **Advanced Feature Support:** Software RAID normally doesn't include support for advanced features like **hot spares** and drive swapping, which improve availability.

Hardware RAID v Software RAID

- **Operating System Compatibility Issues:** If you set up RAID using a particular operating system, only that operating system can generally access that array. If you use another operating system it will not be able to use the array. This creates problems with multiple-OS environments that hardware RAID avoids.

- **Software Compatibility Issues:** Some software utilities may have conflicts with software RAID arrays; for example, some partitioning and formatting utilities. Again, hardware RAID is more "transparent" and may avoid these problems.

- **Reliability Concerns:** Some RAID users avoid software RAID over concern **with potential bugs** that might compromise the integrity and reliability of the array. While hardware RAID controllers can certainly *also* have bugs, I think it's reasonable to believe that some operating systems are more likely to have these sorts of problems than a good-quality hardware RAID controller would.

Drive Swapping

- **Hot Swap:** A true hot swap is defined as one where the drive can be replaced while the rest of the system remains completely **uninterrupted**. This means the system carries on functioning, the **bus keeps transferring data**, and the **hardware change** is completely transparent.

- **Warn Swap:** In a so-called "warm swap", the power remains on to the hardware and the operating system continues to function, but all activity must be stopped on the bus to which the device is connected. This is worse than a hot swap, obviously, but clearly better than a cold one.

- **Cold Swap:** The system must be powered off before making the swap.

Hot Spare

- Another approach is through the use of ***hot spares***. Additional drives are attached to the controller and left in a **"standby"** mode. If a failure occurs, the controller can use the spare drive as a replacement for the bad drive. A very simple concept, and a feature that is supported by most RAID implementations, even many of the inexpensive hardware RAID cards and software RAID solutions. Typically, the only cost is "yet another" hard disk that you have to buy but can't use for storing data. :^)

- You may ask though: if I have hot swap capability, why do I need hot spares anyway? I can just replace a drive when it fails, right? That's true, but the main advantage that hot sparing has over hot swapping is that with a controller that supports hot sparing, the rebuild will be *automatic*. The controller detects that a drive has gone belly up, it disables it, and immediately rebuilds the data onto the hot spare. This is a tremendous advantage for anyone managing many arrays, or for systems that run unattended--do you really want to have to go **into the office at 4 am** on a rainy **Sunday to hot-swap** a drive for the benefit of your **overseas users**?

Hot Spare

- As features, **hot sparing** **and** **hot swapping** **are independent**: you can have one, or the other, or both.

- They will work together, and often are used in that way. However, sparing is particularly important if you *don't* have hot swap (or warm swap) capability.

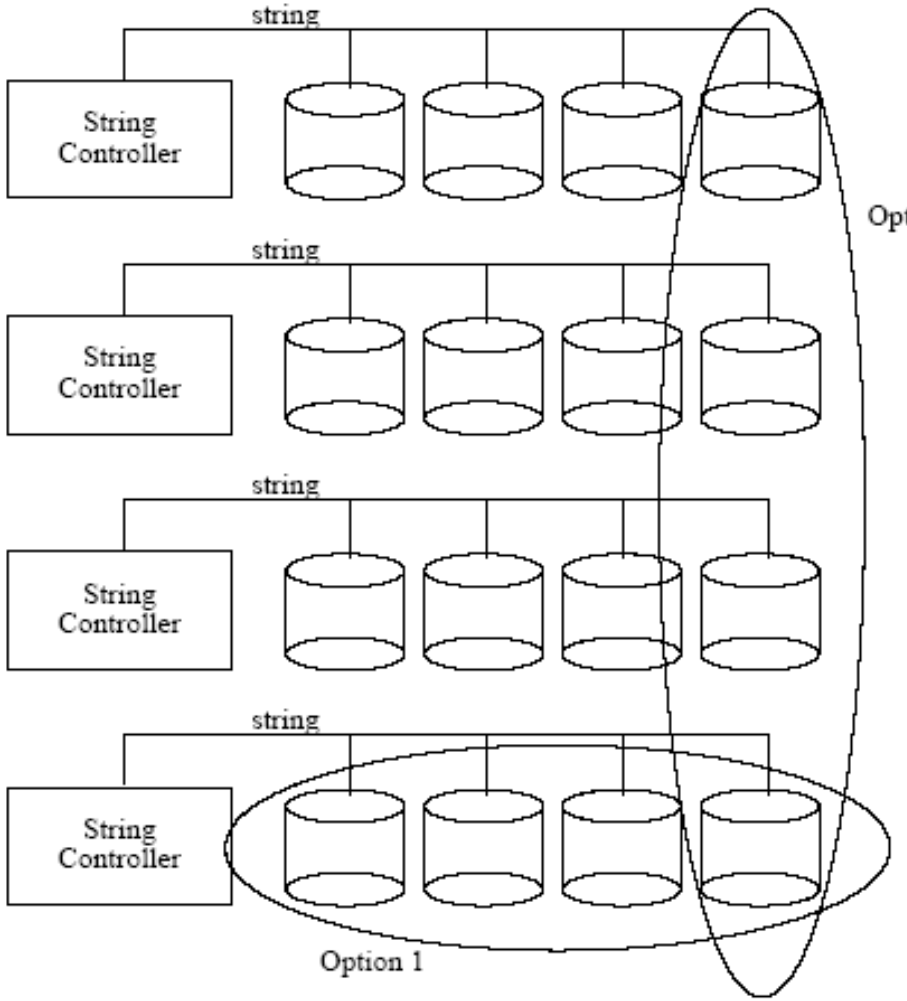
- The reason is that it will let you get the array back into normal operating mode quickly, delaying the time that you will have to shut down the system until when you want to do it.

- You of course lose the hot sparing capability in the meantime; when the failed drive is replaced, the new drive becomes the new hot spare.

RAID interfaces

- **PATA not-used** (no multi-tasking bus capability)
- SCSI yes, but **multiple channels RAID controller required**!!!
- But what about if we decide we want to create a **larger array**, say, an 8-drive array? Then we have a problem. Even if we use the *average* STR figure of those drives, 32 MB/s, we need 256 MB/s, far in excess of what Ultra160 can provide.
- **To avoid this problem**, higher-end SCSI RAID controllers provide support for **multiple channels**. Essentially, the RAID controller has not one SCSI bus with which to communicate with the drives in the array, but two or more. For example, some cards have 4 channels. Each of these is capable of handling 160 MB/s in theory, yielding a whopping theoretical bandwidth of 640 MB/s.

Multiple channels: Orthogonal RAID



better

Figure 7: Orthogonal RAID. This figure present two options of how to organize error-correction groups in the presence of shared resources, such as a string controller. Option 1 groups four disks on the same string into an error-correction group; Option 2 groups one disk from each string into a group. Option 2 is preferred over Option 1 because the failure of a string controller will only render one disk from each group inaccessible.

Advanced RAID Features

- Caching
- Floating parity
- Parity logging
- De-clustered parity

Caching

- write-back caching.
- built-in backup battery
- Parity caching

Caching in RAID

- Buffering and caching, minimize the performance degradations of small writes in a RAID level 5.

- **Delayed Write**
- 1) by giving future updates the opportunity to **overwrite previous updates**, thus **eliminating the need to write the first update**,

- 2) by lengthening the queue of requests seen by a disk scheduler and allowing more **efficient scheduling**

- RAID level 5 will still be four times worse than a RAID level 0.

- An extension of write buffering is to group sequential writes together. RAID caching include:
  - Data
  - parity

Parity Logging

- Stodolsky and Gibson propose an approach called **parity logging** to reduce the penalty of small writes in RAID level 5 disk arrays [Stodolsky93, Bhide92].
- Parity logging reduces the overhead for small writes **by delaying** the **read of the old parity and the write of the new parity**. Instead of **immediately updating** the **parity**, an **update image**, which is the difference between the old and new parity, is temporarily written to a **log**. Delaying the update allows the parity to be grouped together in large contiguous blocks that can be **updated more efficiently.**

- This delay takes place **in two parts**.
- First, the parity update image is stored temporarily in non-volatile memory. When this memory, which could be a few tens of KB, fills up, the parity update image is written to a log region on disk.
- When the log fills up, the parity update image is read into memory and added to the old parity. The resulting new parity is then written to disk.

Parity Logging

- Although this scheme transfers **more data to and from disk**, the transfers are in much larger units and are hence **more efficient**; **large sequential disk accesses** are an order of magnitude more efficient than small random accesses (Section 2.1).

- **Parity logging reduces the small write overhead** from four disk accesses to a little more than two disk accesses, the same overhead incurred by mirrored disk arrays. The costs of parity logging are the memory used for temporarily storing update images, the extra disk space used for the log of update images, and the additional memory used when applying the parity update image to the old parity.

- This technique can also be applied to the second copy of data in mirrored disk arrays to reduce the cost of writes in mirrored disk arrays from two to a little more than one disk access [Orji93].

Declustered Parity

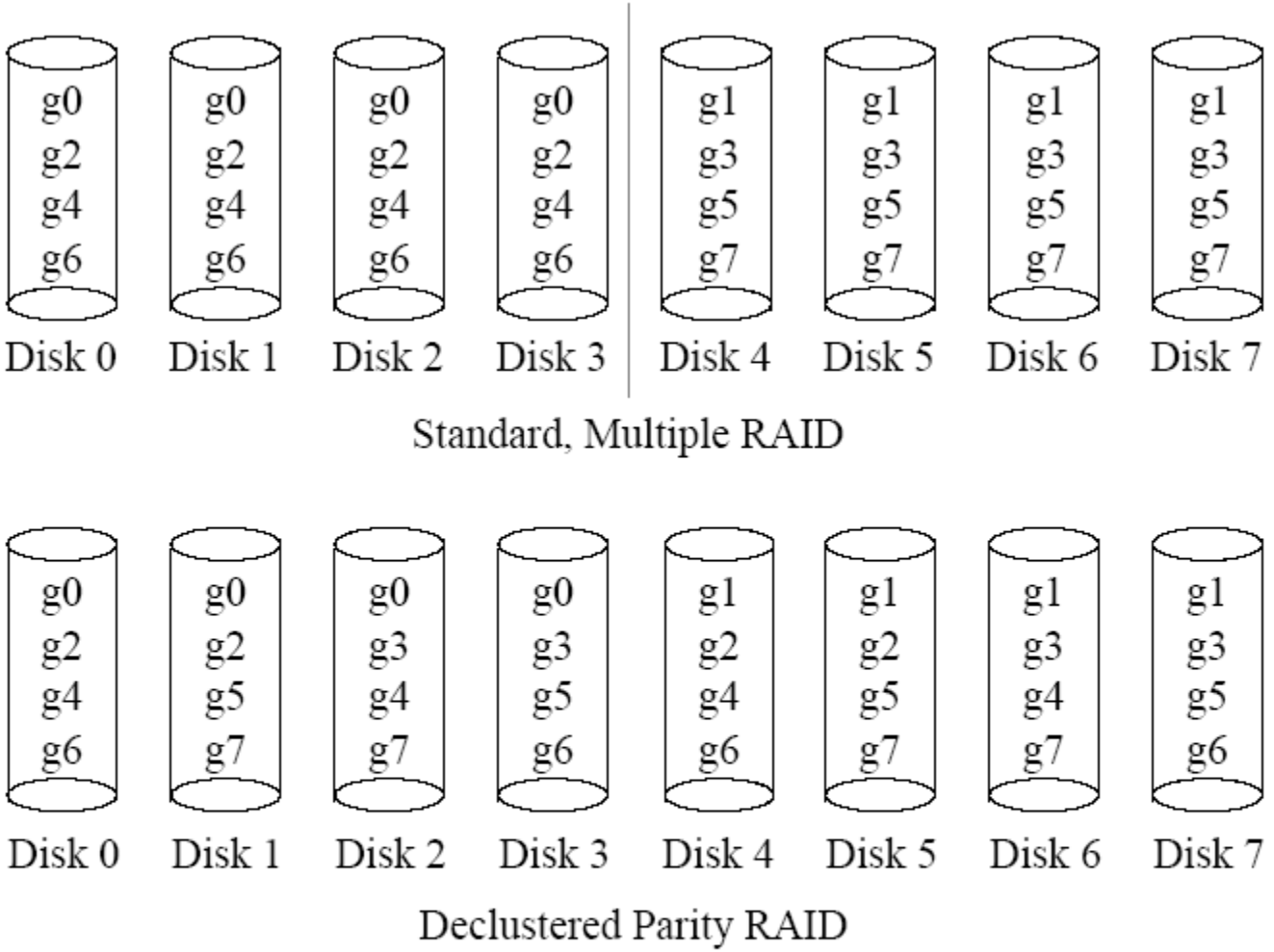


Figure 8: Standard Versus Declustered Parity RAID. This figure illustrates examples of standard and declustered parity RAID with eight disks and a parity group size of four. Identically labeled blocks belong to the same parity group. In the standard RAID organization, parity groups are composed of disks from one of two non-overlapping subsets of disks. In the declustered parity RAID, parity groups span many overlapping subsets of disks.