

Figure 5-18. (a) Physical geometry of a disk with two zones. (b) A possible virtual geometry for this disk.

sector. A possible virtual geometry for the physical disk of Fig. 5-18(a) is shown in Fig. 5-18(b). In both cases the disk has 192 sectors, only the published arrangement is different than the real one.

For Pentium-based computers, the maximum values for these three parameters are often (65535, 16, and 63), due to the need to be backward compatible with the limitations of the original IBM PC. On this machine, 16-, 4-, and 6-bit fields were used to specify these numbers, with cylinders and sectors numbered starting at 1 and heads numbered starting at 0. With these parameters and 512 bytes per sector, the largest possible disk is 31.5 GB. To get around this limit, many disks now support a system called **logical block addressing**, in which disk sectors are just numbered consecutively starting at 0, without regard to the disk geometry.

RAID

CPU performance has been increasing exponentially over the past decade, roughly doubling every 18 months. Not so with disk performance. In the 1970s, average seek times on minicomputer disks were 50 to 100 msec. Now seek times are slightly under 10 msec. In most technical industries (say, automobiles or aviation), a factor of 5 to 10 performance improvement in two decades would be major news, but in the computer industry it is an embarrassment. Thus the gap between CPU performance and disk performance has become much larger over time.

As we have seen, parallel processing is being used more and more to speed up CPU performance. It has occurred to various people over the years that parallel

I/O might be a good idea too. In their 1988 paper, Patterson et al. suggested six specific disk organizations that could be used to improve disk performance, reliability, or both (Patterson et al., 1988). These ideas were quickly adopted by industry and have led to a new class of I/O device called a **RAID**. Patterson et al. defined RAID as **Redundant Array of Inexpensive Disks**, but industry redefined the I to be "Independent" rather than "Inexpensive" (maybe so they could use expensive disks?). Since a villain was also needed (as in RISC versus CISC, also due to Patterson), the bad guy here was the **SLED (Single Large Expensive Disk)**.

The basic idea behind a RAID is to install a box full of disks next to the computer, typically a large server, replace the disk controller card with a RAID controller, copy the data over to the RAID, and then continue normal operation. In other words, a RAID should look like a SLED to the operating system but have better performance and better reliability. Since SCSI disks have good performance, low price, and the ability to have up to 7 drives on a single controller (15 for wide SCSI), it is natural that most RAIDs consist of a RAID SCSI controller plus a box of SCSI disks that appear to the operating system as a single large disk. In this way, no software changes are required to use the RAID, a big selling point for many system administrators.

In addition to appearing like a single disk to the software, all RAIDs have the property that the data are distributed over the drives, to allow parallel operation. Several different schemes for doing this were defined by Patterson et al., and they are now known as RAID level 0 through RAID level 5. In addition, there are a few other minor levels that we will not discuss. The term "level" is something of a misnomer since there is no hierarchy involved; there are simply six different organizations possible.

RAID level 0 is illustrated in Fig. 5-19(a). It consists of viewing the virtual single disk simulated by the RAID as being divided up into strips of k sectors each, with sectors 0 to $k - 1$ being strip 0, sectors k to $2k - 1$ as strip 1, and so on. For $k = 1$, each strip is a sector; for $k = 2$ a strip is two sectors, etc. The RAID level 0 organization writes consecutive strips over the drives in round-robin fashion, as depicted in Fig. 5-19(a) for a RAID with four disk drives. Distributing data over multiple drives like this is called **striping**. For example, if the software issues a command to read a data block consisting of four consecutive strips starting at a strip boundary, the RAID controller will break this command up into four separate commands, one for each of the four disks, and have them operate in parallel. Thus we have parallel I/O without the software knowing about it.

RAID level 0 works best with large requests, the bigger the better. If a request is larger than the number of drives times the strip size, some drives will get multiple requests, so that when they finish the first request they start the second one. It is up to the controller to split the request up and feed the proper commands to the proper disks in the right sequence and then assemble the results in memory correctly. Performance is excellent and the implementation is straightforward.

RAID level 0 works worst with operating systems that habitually ask for data one sector at a time. The results will be correct, but there is no parallelism and hence no performance gain. Another disadvantage of this organization is that the reliability is potentially worse than having a SLED. If a RAID consists of four disks, each with a mean time to failure of 20,000 hours, about once every 5000 hours a drive will fail and all the data will be completely lost. A SLED with a mean time to failure of 20,000 hours would be four times more reliable. Because no redundancy is present in this design, it is not really a true RAID.

The next option, RAID level 1, shown in Fig. 5-19(b), is a true RAID. It duplicates all the disks, so there are four primary disks and four backup disks. On a write, every strip is written twice. On a read, either copy can be used, distributing the load over more drives. Consequently, write performance is no better than for a single drive, but read performance can be up to twice as good. Fault tolerance is excellent: if a drive crashes, the copy is simply used instead. Recovery consists of simply installing a new drive and copying the entire backup drive to it.

Unlike levels 0 and 1, which work with strips of sectors, RAID level 2 works on a word basis, possibly even a byte basis. Imagine splitting each byte of the single virtual disk into a pair of 4-bit nibbles, then adding a Hamming code to each one to form a 7-bit word, of which bits 1, 2, and 4 were parity bits. Further imagine that the seven drives of Fig. 5-19(c) were synchronized in terms of arm position and rotational position. Then it would be possible to write the 7-bit Hamming coded word over the seven drives, one bit per drive.

The Thinking Machines' CM-2 computer used this scheme, taking 32-bit data words and adding 6 parity bits to form a 38-bit Hamming word, plus an extra bit for word parity, and spread each word over 39 disk drives. The total throughput was immense, because in one sector time it could write 32 sectors worth of data. Also, losing one drive did not cause problems, because loss of a drive amounted to losing 1 bit in each 39-bit word read, something the Hamming code could handle on the fly.

On the down side, this scheme requires all the drives to be rotationally synchronized, and it only makes sense with a substantial number of drives (even with 32 data drives and 6 parity drives, the overhead is 19 percent). It also asks a lot of the controller, since it must do a Hamming checksum every bit time.

RAID level 3 is a simplified version of RAID level 2. It is illustrated in Fig. 5-19(d). Here a single parity bit is computed for each data word and written to a parity drive. As in RAID level 2, the drives must be exactly synchronized, since individual data words are spread over multiple drives.

At first thought, it might appear that a single parity bit gives only error detection, not error correction. For the case of random undetected errors, this observation is true. However, for the case of a drive crashing, it provides full 1-bit error correction since the position of the bad bit is known. If a drive crashes, the controller just pretends that all its bits are 0s. If a word has a parity error, the bit from the dead drive must have been a 1, so it is corrected. Although both RAID levels

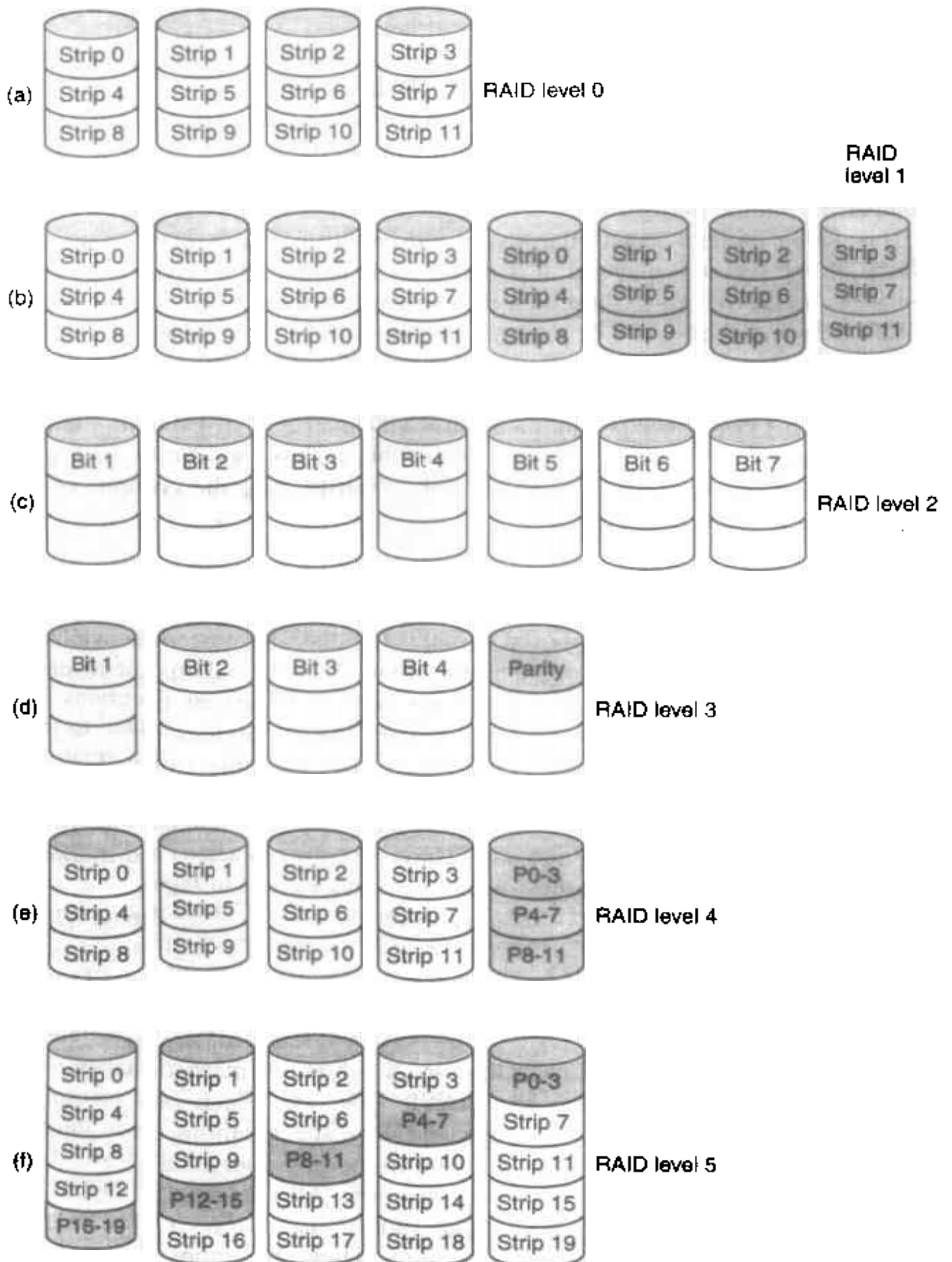


Figure 5-19. RAID levels 0 through 5. Backup and parity drives are shown shaded.

2 and 3 offer very high data rates, the number of separate I/O requests per second they can handle is no better than for a single drive.

RAID levels 4 and 5 work with strips again, not individual words with parity, and do not require synchronized drives. RAID level 4 [see Fig. 5-19(e)] is like RAID level 0, with a strip-for-strip parity written onto an extra drive. For example, if each strip is k bytes long, all the strips are EXCLUSIVE ORed together, resulting in a parity strip k bytes long. If a drive crashes, the lost bytes can be recomputed from the parity drive.

This design protects against the loss of a drive but performs poorly for small updates. If one sector is changed, it is necessary to read all the drives in order to recalculate the parity, which must then be rewritten. Alternatively, it can read the old user data and the old parity data and recompute the new parity from them. Even with this optimization, a small update requires two reads and two writes.

As a consequence of the heavy load on the parity drive, it may become a bottleneck. This bottleneck is eliminated in RAID level 5 by distributing the parity bits uniformly over all the drives, round robin fashion, as shown in Fig. 5-19(f). However, in the event of a drive crash, reconstructing the contents of the failed drive is a complex process.

CD-ROMs

In recent years, optical (as opposed to magnetic) disks have become available. They have much higher recording densities than conventional magnetic disks. Optical disks were originally developed for recording television programs, but they can be put to more esthetic use as computer storage devices. Due to their potentially enormous capacity, optical disks have been the subject of a great deal of research and have gone through an incredibly rapid evolution.

First-generation optical disks were invented by the Dutch electronics conglomerate Philips for holding movies. They were 30 cm across and marketed under the name LaserVision, but they did not catch on, except in Japan.

In 1980, Philips, together with Sony, developed the CD (Compact Disc), which rapidly replaced the 33 1/3-rpm vinyl record for music (except among connoisseurs, who still prefer vinyl). The precise technical details for the CD were published in an official International Standard (IS 10149), popularly called the **Red Book**, due to the color of its cover. (International Standards are issued by the International Organization for Standardization, which is the international counterpart of national standards groups like ANSI, DIN, etc. Each one has an IS number.) The point of publishing the disk and drive specifications as an International Standard is to allow CDs from different music publishers and players from different electronics manufacturers to work together. All CDs are 120 mm across and 1.2 mm thick, with a 15-mm hole in the middle. The audio CD was the first successful mass market digital storage medium. They are supposed to last 100 years. Please check back in 2080 for an update on how well the first batch did.