On FAT Size Calculation

Bernd Böckmann

February 16, 2025

1 Introduction

In the following a closed formula is derived to give a good upper bound for the sector count needed to store file allocation tables. Let this sector count be $sfat$. Further, let ts be the total sector count of the drive or partition holding the filesystem, bps the size of a single sector in bytes, cls the number of data clusters, spc the number of sectors per cluster, res the number of reserved sectors, and $nfat$ the number of allocation tables. When calculating $sfat$ for a FAT16 or FAT12 filesystem, res should include the sector count of the root directory.

2 Formula derivation

The number of clusters adheres to the following formula:

$$cls = \left\lfloor \frac{ts - res - nfat \cdot sfat}{spc} \right\rfloor \tag{1}$$

Let for the moment ignore the fact that $sfat$ must be a whole number. As the first two clusters of every FAT system are reserved, the size of a file allocation table is calculated as the rational number

$$sfat = \frac{(cls + 2) \cdot ft}{bps} \tag{2}$$

where ft is the size of a single FAT entry in bytes. It is $ft = 1.5$ for FAT12, $ft = 2$ for FAT16, and $ft = 4$ for FAT32. Inserting cls in into the last formula gives

$$\begin{aligned} sfat &= \frac{\left(\left\lfloor \frac{ts-res-nfat\cdot sfat}{spc} \right\rfloor + 2\right) \cdot ft}{bps} \\ &= \frac{\left\lfloor \frac{ft(ts-res-nfat\cdot sfat+2\cdot spc)}{spc} \right\rfloor}{bps} \qquad (3) \\ &\leq \frac{\frac{ft(ts-res-nfat\cdot sfat+2\cdot spc)}{spc}}{bps} \qquad (4) \end{aligned}$$

Re-arranging (4) gives

$$sfat \cdot bps \cdot spc \leq ft(ts - res - nfat \cdot sfat + 2 \cdot spc)$$

and

$$sfat \cdot (bps \cdot spc + ft \cdot nfat) \leq ft(ts - res + 2 \cdot spc)$$

Division by $bps \cdot spc + ft \cdot nfat$ yields

$$\begin{aligned} sfat &\leq \frac{ft(ts - res + 2 \cdot spc)}{bps \cdot spc + ft \cdot nfat} \\ &\leq \left\lceil \frac{ft(ts - res + 2 \cdot spc)}{bps \cdot spc + ft \cdot nfat} \right\rceil \qquad (5) \end{aligned}$$

The last formula introduces the rounding up to the next integer number, as we can only allocate a multiple of whole sectors. When performing integer arithmetic and the modulo operation is available, we can add +1 to result if the remainder of the division is not zero. Otherweise, we can rewrite (5) as

$$\begin{aligned} sfat &\leq \frac{ft(ts - res + 2 \cdot spc) + bps \cdot spc + ft \cdot nfat - 1}{bps \cdot spc + ft \cdot nfat} \\ &= \frac{ft(ts - res + 2 \cdot spc) - 1}{bps \cdot spc + ft \cdot nfat} + 1 \qquad (6) \end{aligned}$$

Formula (6) provides an upper bound for FAT size calculation, so it is guaranteed to not return a value smaller than the real one. However, it has some problems with overflow behaviour and the fact that $ft = 1.5$ for FAT12, which is bad when doing integer arithmetic. Formula (6) may be simplified (and slightly rounded up) by

$$sfat \leq \frac{ft(ts - res + 2 \cdot spc)}{bps \cdot spc + ft \cdot nfat} + 1$$

and with $2 \cdot ft$ being an integer number

$$sfat \leq \frac{ts - res + 2 \cdot spc}{2 \cdot bps \cdot spc/2ft + nfat} + 1 \tag{7}$$

The last formula may still overflow on $ts - res + 2 \cdot spc$ if ts reaches the end of the number range. If we assume $nfat = 2$, we can develop (7) into

$$sfat \leq \frac{(ts - res + 1)/2 + spc}{bps \cdot spc/2ft + 1} + 1 \tag{8}$$

This formula should be overflow safe in the case that ts reaches the end of the number range. The term $bps \cdot spc/2ft$ might be problematic on 16-bit arithmetic. It can be re-arranged to $bps/2ft \cdot spc$ at the cost of a slightly more inaccurate result.

3 A formula for n · 512-byte sectors

Lets assume $bps = n \cdot 512$, with $n = 1$ for 512-byte sectors and $n = 8$ for 4096-byte sectors. Let further assume that we have two file allocables $nfat = 2$. Formula (8) can then be used to calculate $sfat_{12}$ for FAT12, $sfat_{16}$ for FAT16 and $sfat_{32}$ for FAT32 as follows using integer arithmetic:

$$sfat_{12} \leq \frac{(ts - res + 1)/2 + spc}{n \cdot 170 \cdot spc + 1} + 1 \tag{9}$$

$$sfat_{16} \leq \frac{(ts - res + 1)/2 + spc}{n \cdot 128 \cdot spc + 1} + 1 \tag{10}$$

$$sfat_{32} \leq \frac{(ts - res + 1)/2 + spc}{n \cdot 64 \cdot spc + 1} + 1 \tag{11}$$

The formula for FAT12 is slightly more inaccurate than the others, because the value $512/3 = 170,6667$ is rounded down to 170 in order to round the whole formula up. We can also express as a single formula

$$sfat \leq \frac{(ts - res + 1)/2 + spc}{n \cdot t \cdot spc + 1} + 1 \tag{12}$$

with $t = 170$ for FAT12, $t = 128$ for FAT16 and $t = 64$ for FAT32, and $n = bps/512$.

4 Getting exact numbers

As the above formulas give an upper bound for the number of sectors required for the FAT, what can be done to get exact numbers to not waste any sectors?

Starting from an initial value for $sfat$, we can iterate a few times (usually not more than one iteration is needed), and reduce $sfat$ by one until the following condition holds:

$$(ts - res - nfat \cdot sfat)/spc > sfat \cdot bps/ft - 2 \tag{13}$$

If the left term is larger than the right term, the FAT is not large enough to store the number of clusters needed. We have to take into account the two reserved clusters. This is what the term -2 is for. So the algorithm is:

```
$sfat \leftarrow$ calculate initial value representing upper bound
**while** $(ts - res - nfat \cdot sfat)/spc \leq sfat \cdot bps/ft - 2$ **do**
    $sfat \leftarrow sfat - 1$
**end while**
$sfat \leftarrow sfat + 1$
```