

Подскажи мне алгоритм и формулы для программы, которая будет форматировать дискеты и жесткие диски в файловых системах FAT12, FAT16, FAT32. Нужен порядок действий, какие параметры вычислять первым делом, какие после, на что обращать внимание.

🌀 Думал 8 секунд ▾

Хм, пользователь хочет алгоритм и формулы для низкоуровневого форматирования дисков в файловых системах FAT. Задача нетривиальная, требует понимания структуры FAT, расчёта параметров в зависимости от размера носителя и выбора правильной версии FAT.

Нужно разбить ответ на логические блоки: сначала общий алгоритм с ветвлением по типу FAT, затем детали для каждой файловой системы, включая формулы и ограничения. Важно подчеркнуть, что это низкоуровневая работа с диском, а не вызов стандартных утилит.

Стоит упомянуть ключевые параметры вроде размера кластера, количества зарезервированных секторов, размера FAT — они зависят от объёма диска. Для FAT32 добавить нюансы вроде зеркальной копии FAT и позиции корневого каталога. В конце хорошо бы дать практические советы по проверке расчётов.

Отличная задача! Вот подробный алгоритм и формулы для программы низкоуровневого форматирования в FAT. Это не просто вызов стандартных утилит, а именно создание структуры разделов "с нуля".

## Общий порядок действий (Алгоритм)

### 1. Определение параметров носителя и выбор FAT:

- Получить общий размер диска/раздела в секторах ( `total_sectors` ).
- Получить размер сектора (обычно 512, 1024, 2048, 4096 байт).  
`bytes_per_sector` .
- **Ключевое решение:** На основе `total_sectors * bytes_per_sector` выбрать тип FAT (12/16/32). Используй пороговые значения (см. ниже).

### 2. Вычисление основных геометрических параметров:

- Рассчитать `sectors_per_cluster` . Это степень двойки (1, 2, 4, 8, ... 64). Зависит от размера тома.
- Определить размер зарезервированной области ( `reserved_sectors` ), обычно 1 для дисков, больше для разделов. Для FAT32 минимум 32.

- Определить количество копий FAT ( `num_fats` ). Обычно 2.
- Рассчитать размер одной таблицы FAT ( `sectors_per_fat` ). **Самый сложный расчет**, зависит от всего.

### 3. Создание загрузочной записи (BPB - Bios Parameter Block):

- Заполнить структуру BPB (для FAT12/16) или BPB32 (для FAT32) вычисленными значениями.
- Для FAT32 дополнительно вычислить и заполнить структуру FSInfo.

### 4. Запись структур на диск:

- **Сектор 0:** Записать MBR с таблицей разделов (если форматируем весь диск, а не раздел).
- **Первый сектор раздела (Logical Sector 0):**
  - Записать BPB (и возможно, загрузочный код).
  - Для FAT32: записать FSInfo (обычно в сектор 1) и Backup Boot Sector (обычно в сектор 6).
- **Зарезервированная область:** Пропустить `reserved_sectors` .
- **Таблицы FAT:** Записать `num_fats` копий FAT, каждая размером `sectors_per_fat` .
  - Инициализировать первые кластеры:
    - `FAT[0] = 0xF?? | media_type` (например, `0xF8` для HDD).
    - `FAT[1] = 0xFFF / 0xFFFF / 0xFFFFFFFF` (конец цепочки).
- **Область данных:** Заполнить корневой каталог (для FAT12/16) нулями. Для FAT32 корневой каталог находится в кластере 2.

## Формулы и детали для каждой FAT

### 1. Выбор FAT (Пороговые значения)

- **FAT12:** До **4084** кластеров в области данных. (Макс. объем зависит от размера кластера, но обычно до ~32 МБ).
- **FAT16:** От **4085** до **65524** кластеров. (Обычно от ~32 МБ до ~2 ГБ с 32-килобайтными кластерами. Теоретически до 4 ГБ с 64К кластерами, но не поддерживается многими ОС).

- **FAT32:** От **65525** кластеров и более. (От ~512 МБ до 2 ТБ. Официально до 32 ГБ в Windows, но работает и больше).

**Важно:** Это кластеры, а не байты!

## 2. Ключевые вычисляемые параметры

**Общие переменные:**

- `total_sectors` - общее число секторов раздела.
- `bytes_per_sector` - 512, 1024, 2048, 4096.
- `sectors_per_cluster ( spc )` - степень двойки. Подбирается для оптимального количества кластеров.
- `reserved_sectors ( rsvd_sec_cnt )` - обычно 1 (FAT12/16) или 32 (FAT32).
- `num_fats` - почти всегда 2.
- `root_dir_entries ( root_ent_cnt )` - только для FAT12/16. Обычно 512 (для дискет 224). Для FAT32 = 0.
- `sectors_per_fat ( fat_sz32 для FAT32 )` - **самый важный и сложный параметр.**

**Алгоритм расчета `sectors_per_fat` (итеративный или прямой)**

**Для FAT12/16:**

1. Рассчитать размер корневого каталога в секторах:

$$\text{root\_dir\_sectors} = ((\text{root\_dir\_entries} * 32) + (\text{bytes\_per\_sector} - 1)) / \text{bytes\_per\_sector}$$

2. Рассчитать размер области данных в секторах:

$$\text{data\_sectors} = \text{total\_sectors} - (\text{reserved\_sectors} + (\text{num\_fats} * \text{fat\_sz}) + \text{root\_dir\_sectors})$$

Здесь `fat_sz` (искомое) пока неизвестно.

3. Рассчитать общее число кластеров:

$$\text{total\_clusters} = \text{data\_sectors} / \text{sectors\_per\_cluster}$$

4. Теперь можно найти нужный размер FAT:

- **Для FAT16:** Каждая запись FAT - 2 байта.

$$\text{fat\_sz} = (\text{total\_clusters} * 2 + (\text{bytes\_per\_sector} - 1)) / \text{bytes\_per\_sector}$$

- Для **FAT12**: Каждая запись - 1.5 байта.  

$$\text{fat\_sz} = (\text{total\_clusters} * 3 + 1) / (2 * \text{bytes\_per\_sector})$$
 (с округлением вверх).

5. Но `fat_sz` влияет на `data_sectors`. Это цикл. На практике используют итерацию или таблицы. Простой способ:

- Сначала грубо оцени  

$$\text{data\_sectors} = \text{total\_sectors} - \text{reserved\_sectors} - \text{root\_dir\_sectors}.$$
- Посчитай `total_clusters`, затем `fat_sz`.
- Уточни `data_sectors`, вычтя `num_fats * fat_sz`.
- Пересчитай `total_clusters`. Если изменился сильно - повтори.

### Для FAT32:

1. Формула проще, т.к. корневой каталог в области данных.

$$\text{data\_sectors} = \text{total\_sectors} - \text{reserved\_sectors} - (\text{num\_fats} * \text{fat\_sz32})$$

2.  $\text{total\_clusters} = \text{data\_sectors} / \text{sectors\_per\_cluster}$

3. Размер записи FAT32 - 4 байта. Поэтому:

$$\text{fat\_sz32} = (\text{total\_clusters} * 4 + (\text{bytes\_per\_sector} - 1)) / \text{bytes\_per\_sector}$$

4. Опять цикл. Стартовое приближение: `fat_sz32 = 0`. Повторять расчет, пока `fat_sz32` не стабилизируется (обычно 2-3 итерации).

## Структуры данных (Упрощенно)

Общая структура BPB (смещения для сектора 512 байт):

text

```

0x00: JMP инструкция (3 байта)
0x03: OEM Name (8 байт)
0x0B: bytes_per_sector (WORD)
0x0D: sectors_per_cluster (BYTE)
0x0E: reserved_sectors (WORD)
0x10: num_fats (BYTE)
0x11: root_ent_cnt (WORD) ; 0 для FAT32
0x13: total_sectors_16 (WORD) ; если 0, см. 0x20
0x15: media (BYTE) ; 0xF8 для HDD
0x16: fat_sz_16 (WORD) ; 0 для FAT32
0x18: sectors_per_track (WORD)

```

0x1A: num\_heads (WORD)  
0x1C: hidden\_sectors (DWORD) ; важно для разделов  
0x20: total\_sectors\_32 (DWORD)

### Дополнение для FAT12/16 (Extended BPB):

text

0x24: drive\_number (BYTE)  
0x25: reserved (BYTE)  
0x26: boot\_sig (BYTE) ; 0x29  
0x27: volume\_id (DWORD)  
0x2B: volume\_label (11 BYTE)  
0x36: file\_system\_type (8 BYTE) ; "FAT12 ", "FAT16 "  
0x3E: boot\_code (448 BYTE)  
0x1FE: signature (WORD) ; 0xAA55

### Дополнение для FAT32 (Extended BPB):

text

0x24: fat\_sz\_32 (DWORD) ; sectors\_per\_fat  
0x28: ext\_flags (WORD)  
0x2A: fs\_version (WORD) ; 0  
0x2C: root\_cluster (DWORD) ; обычно 2  
0x30: fs\_info\_sector (WORD) ; обычно 1  
0x32: backup\_boot\_sector (WORD) ; обычно 6  
0x34: reserved[12] (12 BYTE)  
... (поля drive\_number, boot\_sig, volume\_id, label, type аналогичны, но смещения другие)  
0x1FE: signature (WORD) ; 0xAA55

### Порядок записи для FAT32 (особое внимание!)

1. **Сектор 0:** Основная загрузочная запись (BPB32).
2. **Сектор 1:** Структура **FSInfo** (опционально, но рекомендуется).
3. **Сектор 6:** Резервная копия загрузочного сектора (Backup Boot Sector).  
Должна быть **идентична** сектору 0.
4. **Сектор 7 и далее:** Зарезервированная область.
5. **FAT1:** Начинается с сектора reserved\_sectors (обычно 32).

6. **FAT2:** Начинается с сектора `reserved_sectors + fat_sz32`.
7. **Область данных:** Начинается с сектора `reserved_sectors + num_fats * fat_sz32`.
  - **Кластер 2:** Является началом корневого каталога (значение `root_cluster` в BPB).

## На что обратить особое внимание

1. **Выравнивание:** Для производительности на современных SSD резервируй `reserved_sectors` так, чтобы FAT и данные начинались с границ больших блоков (например, 1 МБ).
2. **Размер кластера:** Большой кластер уменьшает размер FAT и ускоряет работу с большими файлами, но ведет к потере места на мелких файлах. Выбирай по таблицам Microsoft.
3. **FAT32 и большие диски:** При размерах >32 ГБ многие ОС (старые Windows) откажутся монтировать том. Это ограничение на уровне ОС, а не файловой системы.
4. **hidden\_sectors:** Критически важный параметр при форматировании **раздела**, а не всего диска. Это смещение раздела от начала диска в секторах. Если установить неверно, диск не загрузится или данные будут потеряны.
5. **Зеркалирование FAT:** В FAT32 есть флаг `ext_flags`, который может отключать зеркалирование FAT (копию). По умолчанию зеркалирование включено, и FAT1 и FAT2 должны быть идентичны.
6. **Тестирование:** Всегда тестируй алгоритм на виртуальных дисках (файлах-образах) перед работой с реальными носителями. Используй `hexdump` или дисковые редакторы для проверки структур.

**Рекомендация:** Изучи исходный код утилит `mkfs.fat` (из пакета `dosfstools` в Linux) или `fatformat`. Там много тонкостей с учетом множества крайних случаев.

Продолжить с DeepSeek