# Cách đọc ổ đĩa định dạng FAT32

## Cấu trúc FAT 32

A diagram of a cluster

Description automatically generated

Gồm 2 phần:

* Vùng hệ thống:
  + Boot Sector: 512-byte đầu tiên.
  + Vùng FAT
* Vùng Data
  + RDET – Root Directory Entry Table
  + Dữ liệu tập tin/thư mục

## Bước 1: Đọc phân vùng

* Lấy tên ổ đĩa và cho chương trình đọc file hệ thống của ổ đĩa đó.
* Nếu không tồn tại thì thoát.

## Bước 2: Đọc Boot Sector

* Đọc 512-byte đầu tiên tương ứng với Boot Sector
* Và lưu cái thông số chính cần quan tâm:

|  |  |  |
| --- | --- | --- |
| **Offset**  **(Hex)** | **Số byte** | **Ý nghĩa** |
| 0x03 | 8 | OME\_ID: Nơi sản xuất-version (MSDOS5.0) |
| 0x0B | 2 | Số byte trên Sector |
| 0x0D | 1 | SC: số secter trên cluster |
| 0x0E | 2 | SB: số sector thuộc vùng Bootsector |
| 0x10 | 1 | NF: số bảng FAT |
| 0x20 | 4 | Sv: kích thước volume |
| 0x24 | 4 | SF: kích thước mỗi bảng FAT |
| 0x2C | 4 | Cluster bắt đầu của RDET |
| 0x52 | 8 | Loại FAT (chuỗi “FAT32”) |

### Các thông số tính toán được:

* 1. Sector bắt đầu của bảng FAT = SB
  2. Sector bắt đầu của RDET: SRDET = SB + SF \* NF
  3. Sector bắt đầu vùng Data = SRDET + SRDET \* kích thước RDET (= 0 trong FAT32)
  4. i = SB + SF \* NF +SRDET + (k – 2) \* SC
     + i: sector thứ i trên phân vùng
     + k: cluster thứ k trẹn vùng Data
     + SRDET = 0 trong FAT32

## Bước 3: Đọc bảng FAT

Cấu trúc bảng FAT trong FAT 32:

* Thứ tự lưu trữ: Little endian
* Kích thước mỗi phần tử: 4 bytes

A table with text and numbers

Description automatically generated

Thuật toán đọc bảng FAT

* Bước 1: Nhảy đến vị trí bắt đầu của bảng FAT theo [công thức (Các thông số tính toán được - 1.)](#_Các_thông_số).
* Bước 2: Tạo một mảng để lưu kết quả.
* Bước 3: Đọc 4 bytes tiếp theo và chuyển thành giá trị (số nguyên).
* Bước 4: Nếu giá trị nằm thuộc vùng **IN USE** trong bảng trạng thái thì lưu vào mảng kết quả và đi lại bước 3.
* Bước 5: Kết thúc thuật toán và lưu mảng kết quả vào hệ thống.

### Truy xuất chuỗi cluster:

* 1. Lấy cluster bắt đầu i tra vào bảng FAT
  2. Cập nhật i + 1 = FAT[i] nếu i ≠ EOF
  3. Khi i = EOF các cluster đã thăm trước đó là chuỗi cluster.

## Bước 4: RDET – Root Directory Entry Table

### **Cấu trúc RDET trong FAT 32**

* Gồm các Entry kích thước 32 bytes chứa thông tin về tập tin/ thư mục
* Có 2 loại Entry:
  1. Entry chính: chứa metadata của tập tin
  2. Entry phụ: chứa tên của tập tin
* Các Entry phụ đánh số từ 1 và nằm ngay phía trên Entry chính mà nó lưu tên

|  |
| --- |
| Entry phụ N |
| … |
| Entry phụ 2 |
| Entry phụ 1 |
| Entry chính |

### **Entry chính**

Cấu trúc Entry chính – Các vùng quan trọng

|  |  |  |
| --- | --- | --- |
| **Offset**  **(Hex)** | **Số byte** | **Ý nghĩa** |
| 0x00 | 8 | Tên – mã ASCII |
| 0x08 | 3 | Tên mở rộng – mã ASCII |
| 0x0B | 1 | Thuộc tính |
| 0x14 | 2 | Cluster bắt đầu – Word cao |
| 0x1A | 2 | Cluster bắt đầu – Word thấp |
| 0x1C | 4 | Kích thước |

Byte tại vị trí 0x00: Trạng thái của Entry

* 0xE5: Đã xóa
* 0x00: Trống
* 0x05: Thay thế tên có kí tự bắt đầu là 0xE5

Tên file gồm 11-bytes gồm: 8-bytes tên chính + 3-byte phần mở rộng (8.3 format name) gọi là SFN (Short File Name)

* Tất cả kí tự của SFN được lưu ở dạng in hoa
* Dấu “.” chia cắt 2 phần bị xóa
* Các kí tự dấu cách (0x20h) được thêm vào nếu SFN không đủ format 8.3
* Các kí tự được phép sử dụng:

0～9 A～Z ! # $ % & ' ( ) - @ ^ \_ ` { } ~

* Các kí tự không được phép:
  1. \ / : \* ? " < > |
  2. Tên chỉ có phần mở rộng (“.ext”, “.txt”,…)
* Các tên vượt quá format 8.3 sẽ bị rút ngắn:
  1. Tên chính: rút thành tối đa 6 kí tự + ~N
  2. Phần mở rộng: rút ngắn thành 3 kí tự đầu
* Ví dụ về tên

|  |  |  |
| --- | --- | --- |
| **Tên thư mục** | **Tên được lưu** | **Lý do** |
| Hello.txt | HELLO TXT | Lắp đầy format 8.3 bằng dấu cách |
| LongFileName.docx | LONGFI~1DOC | Tên vi phạm 8.3 format |
| VeryLongText.txt | VERYLO~1TXT | Tên vi phạm 8.3 format |
| VeryLongFolder.txt | VERYLO~2TXT | Tên rút ngắn bị trùng |
| .ext | EXT~1 | Tên vi phạm 8.3 format |
| a+b=c | A\_B\_C~1 | Tên vi phạm 8.3 format |

### Thuộc tính (0.0.A.D.V.S.H.R)

* Tại offset 0x0B đọc 1-byte và tra bảng trạng thái sau:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **0** | **Archive** | **Directory** | **Volume label** | **System** | **Hidden** | **Read-only** |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

* Nếu bit nào bật 1 thì Entry có trạng thái đó.
* Đặc biệt: 0x0F dấu hiệu của entry phụ.

Cluster bắt đầu.

* Gồm 4-byte ghép từ 2-byte Word cao và 2-byte Word thấp
* Ở mỗi phần đọc 2-byte và sắp sếp theo kiểu little endian
* Đặt 2-byte Word cao trước và 2-byte Word thấp sau -> 4-byte thể hiện cluster bắt đầu
* Để truy xuất chuỗi cluster chứa thông tin thư mục: đưa cluster bắt đầu vào công thức ([Truy xuất chuỗi cluster](#_Truy_xuất_chuỗi)).

### **Entry phụ - VFAT (Virtual FAT)**

* Chứa tên phần Entry chính không hỗ trợ.
* Lưu dưới dạng UTF-16
* Nhận dạng: tại offset 0x0Bh: 0x0Fh.

Cấu trúc Entry phụ

|  |  |  |
| --- | --- | --- |
| **Offset**  **(Hex)** | **Số byte** | **Ý nghĩa** |
| 0x00 | 1 | Thứ tự entry (bắt đầu từ 1) |
| 0x01 | 10 | 5 ký tự UniCode – UTF16 |
| 0x0B | 1 | Dấu hiện nhận biết (0x0F) |
| 0x0E | 12 | 6 ký tự tiếp theo |
| 0x1C | 4 | 2 ký tự tiếp theo |

* Tối đa 255 kí tự
* Mỗi Entry phụ lưu được 13 ký tự unicode nếu hơn 13 ký tự tiếp tục tạo thêm Entry phụ
* Tối đa 20 Entry phụ
* Tên lưu đúng như hiển thị
* Hiển thị được các kí tự (+, [,], =,…) mà format 8.3 không hỗ trợ

## Bước 5 : Đọc RDET và tạo cây thư mục

Cấu trúc cây thư mục :

* Mỗi phần tử là CFolder gồm có :
  1. Tên, trạng thái, chuỗi cluster, kích thước
  2. subItem: mảng chứa thư mục con là các Cfolder

Thuật toán tạo cây thư mục

Bước 1: Lấy offset bắt đầu của RDET từ công thức ([Các thông số tính toán được-2.](#_Các_thông_số)).

* startOffset = SRDET \* bytePerSector

Tạo biến Cfolder “root” để thể hiện cây thư muc:

* Tên: tên phân vùng (A~Z)
* Trạng thái: Directory
* Chuỗi Cluster: Cluster bắt đầu của RDET
* Kích thước: 0
* subItem: rỗng

Bước 2: Từ offset bắt đầu và đưa vào hàm sau để có được thông tin về số Entry chính và Entry phụ:

**function** numberOfFile (interger *offset*) : **return** interger array

Entry *entry*

*vfat* ← 0;

*result* ← array of interger

**while not** entry.name[0] is 0 **do**

readEntryFromOffset(*entry*, *offset*)

*offset* ← *offset* + sizeof(*entry*)

**if** *entry*.attr is 0x0Fh

increase *vfat* by 1

**else**

add vfat to *result*

vfat ← 0

**return** *result*

Bước 3: Dùng mảng *result* trả về, tạo mảng Cfolder có kích thước bằng kích thước mảng *result*:

Bước 3.1: với mỗi phần tử trong *result* đọc 32-byte theo cấu trúc Entry phụ và chuyển hóa thành tên rồi lưu vào mảng tên dài và đánh dấu cờ có Entry phụ.

Bước 3.2: Đọc 32-byte kế tiếp và lưu theo cấu trúc Entry chính.

Bước 3.3: Nếu có Entry phụ thì thay thế tên của Entry chính thành Entry phụ.

Bước 3.4: Từ dãy binary chuyển hóa thành các trạng thái theo công thức ([Thuộc tính (0.0.A.D.V.S.H.R)](#_Thuộc_tính_(0.0.A.D.V.S.H.R))):

Bước 3.5: Từ cluster bắt đầu của Entry chính dò bảng FAT và tạo nên mảng các cluster liên tiếp theo công thức ([Truy xuất chuỗi cluster](#_Truy_xuất_chuỗi)).

Bước 3.5: Tạo Cfolder với các tham số: tên, trạng thái, mảng cluster liên tiếp, kích thước và đẩy vào mảng CFolder.

Bước 4: Cập nhật mảng subItem của root thành mảng Cfolder

Bước 5: Với mỗi phần tử trong subItem

Bước 5.1: Nếu trạng thái là thư mục quay lại bước 2 với các thông số sau:

* 1. root: phần tử hiện tại
  2. offset: cluster đầu tiên trong mảng cluster \* sectorPerCluster \* bytePerSector.

Bước 5.2: Nếu không phải thư mục thì không làm gì

## Bước 6: Khai thác thông tin của thư mục

* Cây thư mục: lưu tên thư mục, trạng thái, cluster bắt đầu, kích thước.
* Nội dung thư mục/tập tin: Hiển thị metadata và:
  1. Thư mục: cây thư mục từ subItem
  2. Tập tin: đọc từng byte với mỗi cluster trong mảng cluster và in ra nội dung.