

LẬP TRÌNH XỬ LÝ ĐĨA&FILE

- CƠ BẢN VỀ LƯU TRỮ TRÊN ĐĨA TỬ.
- MỘT ỨNG DỤNG HIỂN THỊ SECTOR
- MỘT ỨNG DỤNG HIỂN THỊ CLUSTER.
- CÁC CHỨC NĂNG VỀ FILE Ở MỨC HỆ THỐNG.
- QUẢN LÝ ĐĨA VÀ THƯ MỤC.
- TRUY XUẤT ĐĨA VỚI INT 13H CỦA ROMBIOS
- BÀI TẬP
- GIỚI THIỆU FILE VÀ LẬP TRÌNH XỬ LÝ FILE

CƠ BẢN VỀ LƯU TRỮ TRÊN ĐĨA TỪ

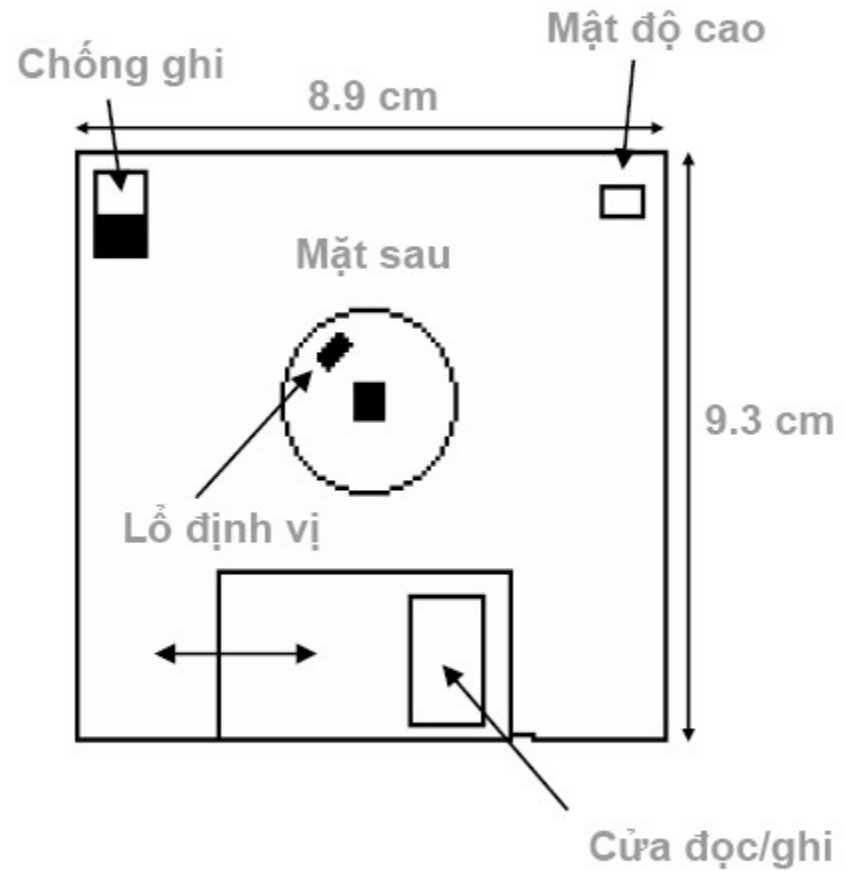
Ngôn ngữ ASM vượt trội hơn các ngôn ngữ khác về khả năng xử lý đĩa.

Ta xem xét việc lưu trữ thông tin trên đĩa theo 2 mức độ : mức phần cứng/BIOS và mức phần mềm/DOS.

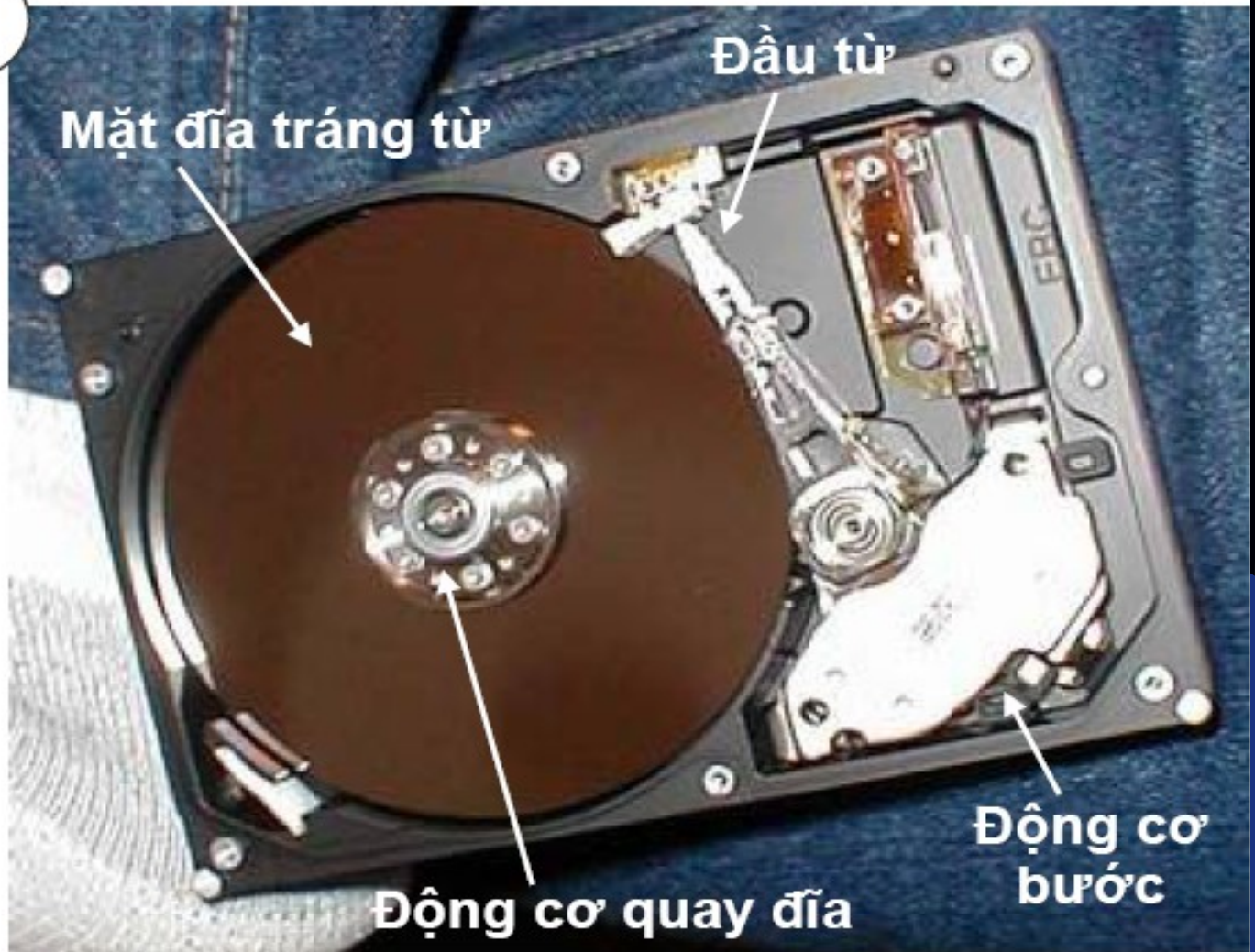
- mức phần cứng : lưu trữ thông tin liên quan đến cách dữ liệu được lưu trữ 1 cách vật lý như thế nào trên đĩa từ?

- mức phần mềm : việc lưu trữ được quản lý bởi tiện ích quản lý File của HĐH DOS.

Đĩa mềm 3.5"



Đĩa cứng



CÁC ĐẶC TÍNH LUẬN LÝ & VẬT LÝ CỦA ĐĨA TỪ

Ở mức vật lý : đĩa được tổ chức thành các Tracks, Cylinders, Sectors.

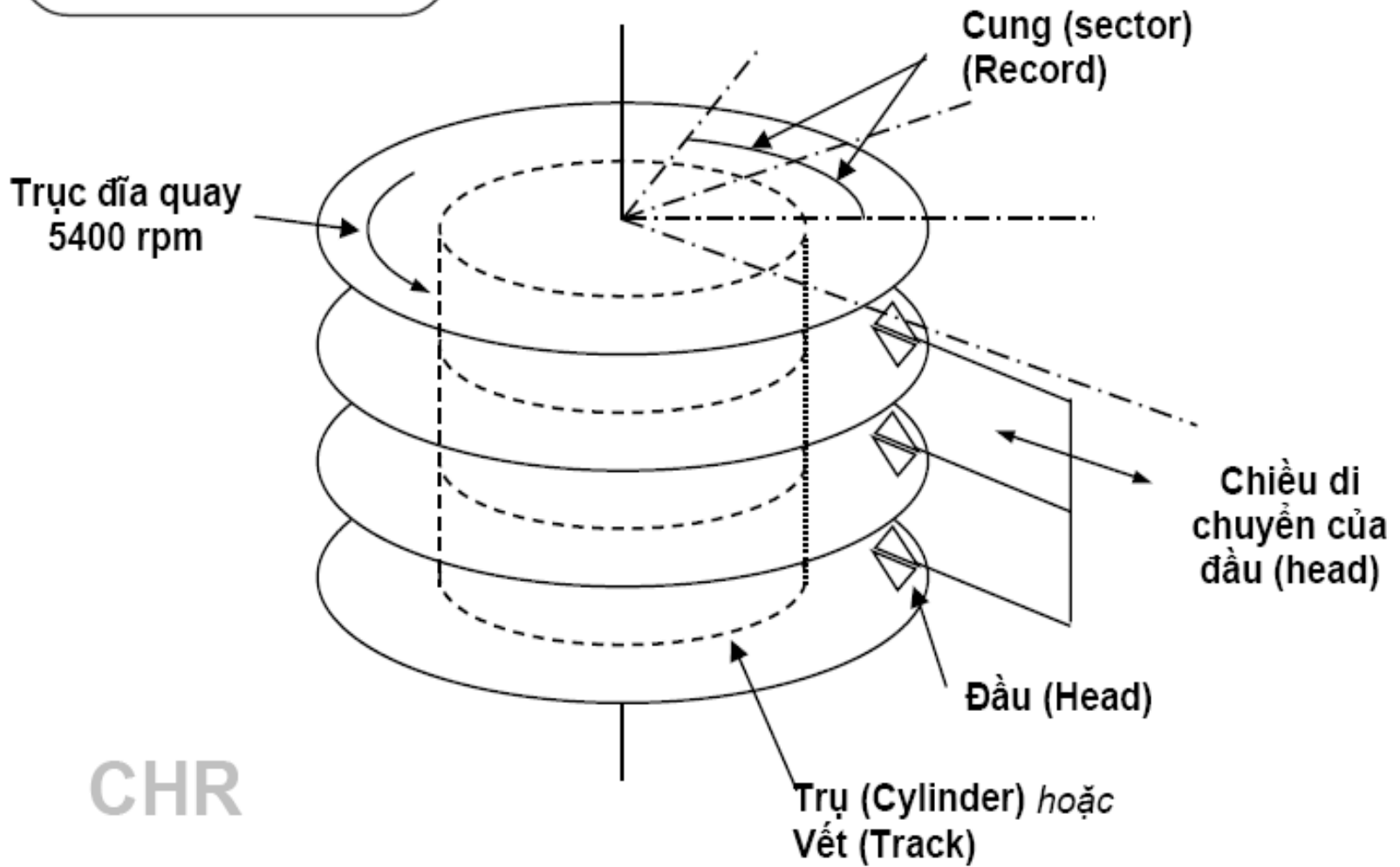
→ Khả năng lưu trữ của đĩa được mô tả bằng 3 thông số :

C (cylinder number)

H (Head side)

R (sector number)

Phân chia đĩa vật lý



CÁC KHÁI NIỆM TRACK, CYLINDER, SECTOR

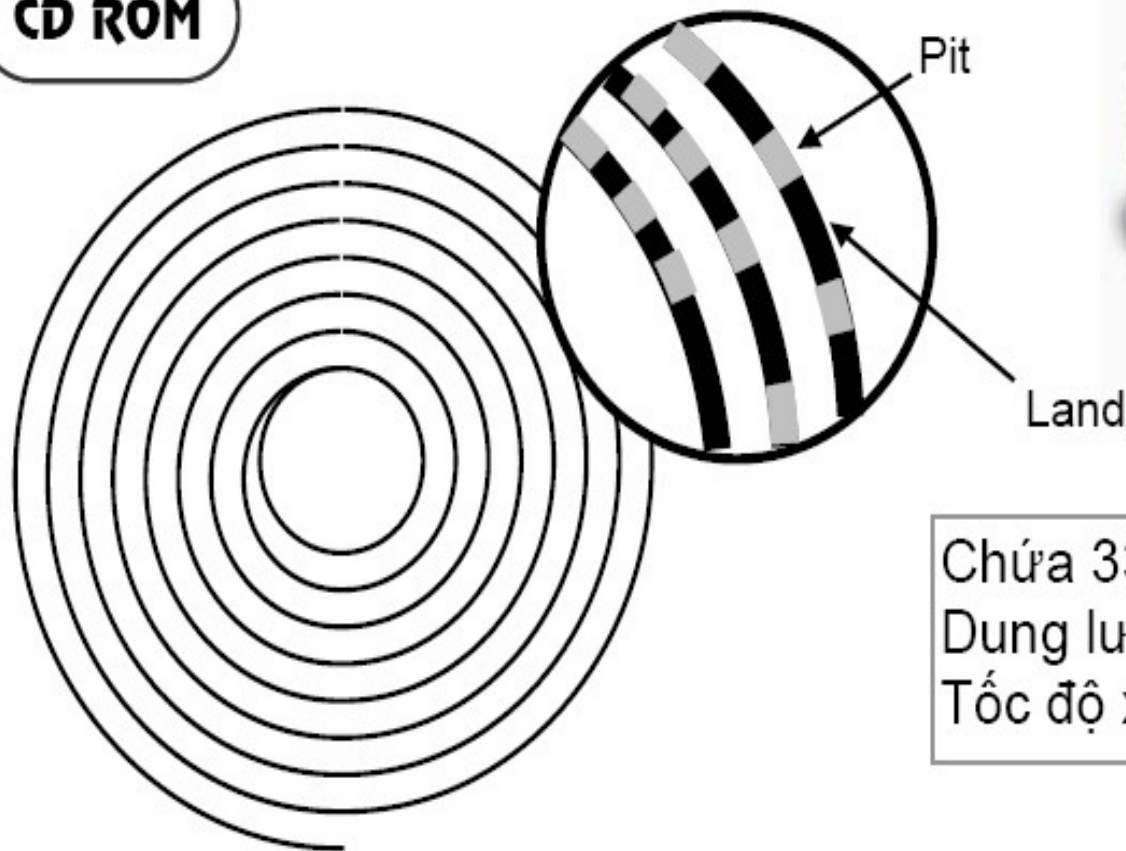
Tracks : là các vòng tròn đồng tâm được tạo ra trên bề mặt đĩa.

Cylinder : tập các tracks cùng bán kính trên 1 chông đĩa. Mặt đĩa có bao nhiêu track thì sẽ có bấy nhiêu Cylinder.

Sector : là 1 đoạn của track (cung tròn) có khả năng lưu trữ 512 bytes dữ liệu.

Các sector được đánh số bắt đầu từ 1 trên mỗi track → trên 1 đĩa tồn tại nhiều sector cùng số hiệu.

CD ROM



Chứa 330.000 khối dữ liệu.
Dung lượng 650 MB / 74 min
Tốc độ x1 = 153.60 KByte/s

Thông tin ghi theo rãnh (track) hình xoắn ốc
Dùng tia laser đục lỗ 1 μm trên rãnh gọi là Pit.
Phần không bị đục lỗ trên rãnh gọi là Land.

Ở mức luận lý : đĩa được tổ chức thành các Clusters, các files mà DOS sẽ dùng để cấp phát vùng lưu trữ cho dữ liệu cần lưu trữ.

Cluster : là 1 nhóm gồm 2,4,6 các sector kề nhau. Đó chính là đơn vị cấp phát vùng lưu trữ cho dữ liệu (file). Các cluster được đánh số bắt đầu từ 0.

Nếu dữ liệu cần lưu trữ chỉ 1 byte thì hệ điều hành cũng cấp phát 1 cluster.

số bytes/cluster hay sector/cluster tùy thuộc vào từng loại đĩa.

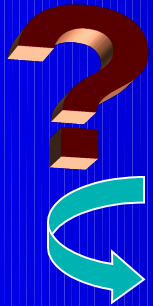
TƯƠNG QUAN GIỮA SECTOR VẬT LÝ VÀ SECTOR LOGIC TRÊN ĐĨA MỀM

MẶT ĐĨA	TRACK	SECTOR	SECTOR LOGIC	THÔNG TIN
0	0	1	0	BOOT RECORD
0	0	2-5	1-4	FAT
0	0	6-9	5-8	Thư mục gốc
1	0	1-3	9-11	Thư mục gốc
1	0	4-9	12-17	Dữ liệu
0	1	1-9	18-26	Dữ liệu

BAD SECTOR



Trên bề mặt đĩa có thể tồn tại các sector mà HĐH không thể ghi dữ liệu vào đó hoặc không thể đọc dữ liệu từ đó. Các sector này gọi là Bad Sector.



Làm sao biết sector nào là bad sector

Kiểm tra giá trị của các phần tử (entry) trong bảng FAT, phần tử nào chứa giá trị (F)FF7H thì cluster tương ứng bị Bad

BẢNG FAT FILE ALLOCATION TABLE

DOS quản lý các File nhờ vào 1 bảng gọi là bảng FAT.

Trong bảng FAT có ghi cluster bắt đầu của File này ở đâu ? Và đĩa còn bao nhiêu Clusters trống chưa cấp phát.

tổ chức luận lý của đĩa được mô tả như hình sau :



Thí dụ về bảng FAT

Đĩa mềm 3.5”” 360K thì :

Sector 0 : boot sector

Sector 1-4 : bảng FAT

Sector 5 – 11 : thư mục gốc

Sector 12-719 : vùng chứa data

BOOT RECORD

- Còn được gọi là Boot Sector. Ổ đĩa cứng gọi là Master boot, là Sector đầu tiên khi đĩa được format.
- chứa 1 chương trình nhỏ cho biết dạng lưu trữ trên đĩa và tên hệ thống MT, kiểm tra xem có các file hệ thống IO.SYS, MSDOS.SYS, COMMAND.COM hay không ?
- nếu có thì nạp chúng vào bộ nhớ (gọi là chương trình khởi động của HĐH)

BOOT RECORD (tt)

- Tọa độ vật lý :

C=0, H=0, R =1 (C0H0R1) tức ở tại sector đầu tiên của track đầu tiên, mặt trên của đĩa đầu tiên trong ổ đĩa cứng.

- Trong Master boot có chứa bảng PARTITION TABLE cho biết tầm địa chỉ vật lý (dung lượng) của ổ đĩa luận lý.

Master boot không thuộc Partition nào

BOOT RECORD (tt)

- BOOT RECORD được ROM BIOS nạp vào địa chỉ 0000:7C00H.
- Nếu máy không bị Virus thì lệnh đầu tiên của chương trình BOOT là JMP 7C3EH, nghĩa là nhảy đến chương trình nạp mồi.
- chương trình nạp mồi (Bootstrap Loader) nạp thành phần cốt lõi của DOS lên RAM trong quá trình khởi động MT.

THÔNG TIN TRONG MASTER BOOT

BYTEĐẦU	SỐ BYTES	THÔNG TIN
00H	3	chỉ thị nhảy về nơi chứa CT nạp mỗi
03H	8	Tên nhà sản xuất và hệ điều hành
0BH	2	Bytes/sector
0DH	1	Sector/block (mỗi block ≥ 1 sector)
0EH	2	Số lượng Sectors không dùng đến kể từ sector 0.
10h	1	Số lượng bảng FAT

THÔNG TIN TRONG MASTER BOOT

BYTE ĐẦU SỐ BYTES

THÔNG TIN

11H	2	Số Entry của thư mục gốc ổ đĩa.
13H	2	Tổng số sector của ổ đĩa logic này.
15H	1	Byte mô tả
16H	2	Số sector cho 1 bảng FAT
18H	2	Số Sectors trong 1 track.
1AH	2	Số lượng đầu đọc
1CH	4	Số lượng sector ẩn
20H	4	Tổng số sectors

THÔNG TIN TRONG MASTER BOOT

BYTE ĐẦU SỐ BYTES

THÔNG TIN

3EH		Bootstrap
....		
1BEH	64	PARTITION TABLE
.....		
1FEH	1	Giá trị 55H
1FFH	1	Giá trị 0AAH

THÔNG TIN TRONG MASTER BOOT

Từ thông tin trong bảng FORMAT, ta tính được địa chỉ của bảng FAT1, FAT2, Thư mục gốc ổ đĩa, địa chỉ bắt đầu của vùng dữ liệu.

BẢNG FAT

- Bảng chứa các danh sách liên kết các clusters. Mỗi danh sách trong bảng cho DOS biết rằng các clusters nào đã cấp phát, các clusters nào chưa dùng.
 - tùy theo ổ đĩa có thể có 1 hay 2 bảng FAT, bảng FAT2 để dự phòng.
- có 2 loại bảng FAT :
 - bảng có Entry 12 bit cho đĩa mềm.
 - bảng có Entry 16 bit cho đĩa cứng.

PARTITION TABLE

64 Bytes của Partition table được chia làm 4, mỗi phần 16 bytes mô tả cho 1 partition các thông tin sau :

Bytes	Mô tả
00H	active flag (=0 Non bootable =80H Bootable)
01H	starting head – Nơi bắt đầu Partition
02H	starting cylinder

PARTION TABLE

Bằng FDISK của HĐH ta có thể chia không gian lưu trữ của đĩa cứng thành các phần khác nhau gọi là Partition.

DOS cho phép tạo ra 3 loại Partition :

Primary Dos, Extended Dos và None Dos

Ta có thể cài đặt các HĐH khác nhau lên các Partition khác nhau.

PARTITION TABLE

03H **starting sector**

04H **partition type :**

0 Non Dos

1 cho đĩa nhỏ 12 bit FAT Entry

4 cho đĩa lớn 16 bit FAT Entry

5 Extended Dos

05H **Ending nơi kết thúc Partition**

06H **Ending Cylinder**

07H **Ending Sector**

08H, 0BH **Starting sector for partition**

0Ch,0FH **Partition length in sectors**

Một số thí dụ

kiểm tra Partition Active

- đọc sector đầu tiên của đĩa cứng lưu vào biến.
- kiểm tra offset 00 của 4 phần tử Partition trong Partition Table

```
MOV CX, 4
```

```
MOV SI, 1BEH
```

```
PACTIVE :
```

```
MOV AL, MBOOT [SI]
```

```
CMP AL, 80H
```

```
JE ACTIVE
```

```
ADD SI, 16
```

```
LOOP PACTIVE
```

```
NO_ACTIVE :
```

```
.....
```

```
ACTIVE : .....
```

Một số thí dụ

Đọc nội dung của BootSector ghi vào biến dem

- đọc sector đầu tiên của đĩa cứng lưu vào buffer.
- tìm partition active (phần tử trong bảng partition có offset 80h)
- đọc byte tại offset 01h và word tại offset 02h của phần tử partition tương ứng ở trên (head, sector, cylinder) để xác định số hiệu bắt đầu của partition active → boot sector của đĩa cứng.
- đọc nội dung của sector đọc được ở trên lưu vào buffer.

Một số thí dụ

ACTIVE :

MOV AX, 0201H ; đọc 1 sector

MOV CX, WORD PTR MBOOT [SI+2] ; sector cylinder

MOV DH, BYTE PTR MBOOT[SI+1] ; head

MOV DL, 80H ; đĩa cứng

MOV ES, CS ; trở về đầu vùng buffer lưu

LEA BX, BUFFER

INT 13H

THƯ MỤC GỐC (ROOT DIRECTORY)

- Là danh sách tất cả các Files đã có trên đĩa, các thư mục cấp 1 đã có.
- Mỗi phần tử (32 bytes) trong bảng thư mục sẽ chứa thông tin về tên file hoặc là thư mục, kích thước, thuộc tính, cluster bắt đầu của file này hoặc cluster bắt đầu của thư mục thứ cấp (thư mục con).

mỗi bảng thư mục chứa tối đa 112 entry, mỗi entry là 32 bytes.

THƯ MỤC GỐC (ROOT DIRECTORY)

Offset	Nội dung	Kích thước
00H	tên chính của File	8 bytes
08H	phần mở rộng của tên file	3 bytes
0BH	thuộc tính của File	1 byte
0CH	dự trữ	10 bytes
16H	giờ thay đổi thông tin cuối cùng	2 bytes
18H	ngày thay đổi thông tin cuối cùng	2 bytes
1Ah	cluster đầu tiên của File	2 bytes
1CH	Kích thước File	4bytes

BYTE THUỘC TÍNH

x	x	a	d	v	s	h	r
---	---	---	---	---	---	---	---

x : không sử dụng

a : thuộc tính lưu trữ (Archive)

d : thuộc tính thư mục con (Sub – Directory)

v : thuộc tính nhãn đĩa (Volume)

s : thuộc tính hệ thống (System)

h : thuộc tính ẩn (Hidden)

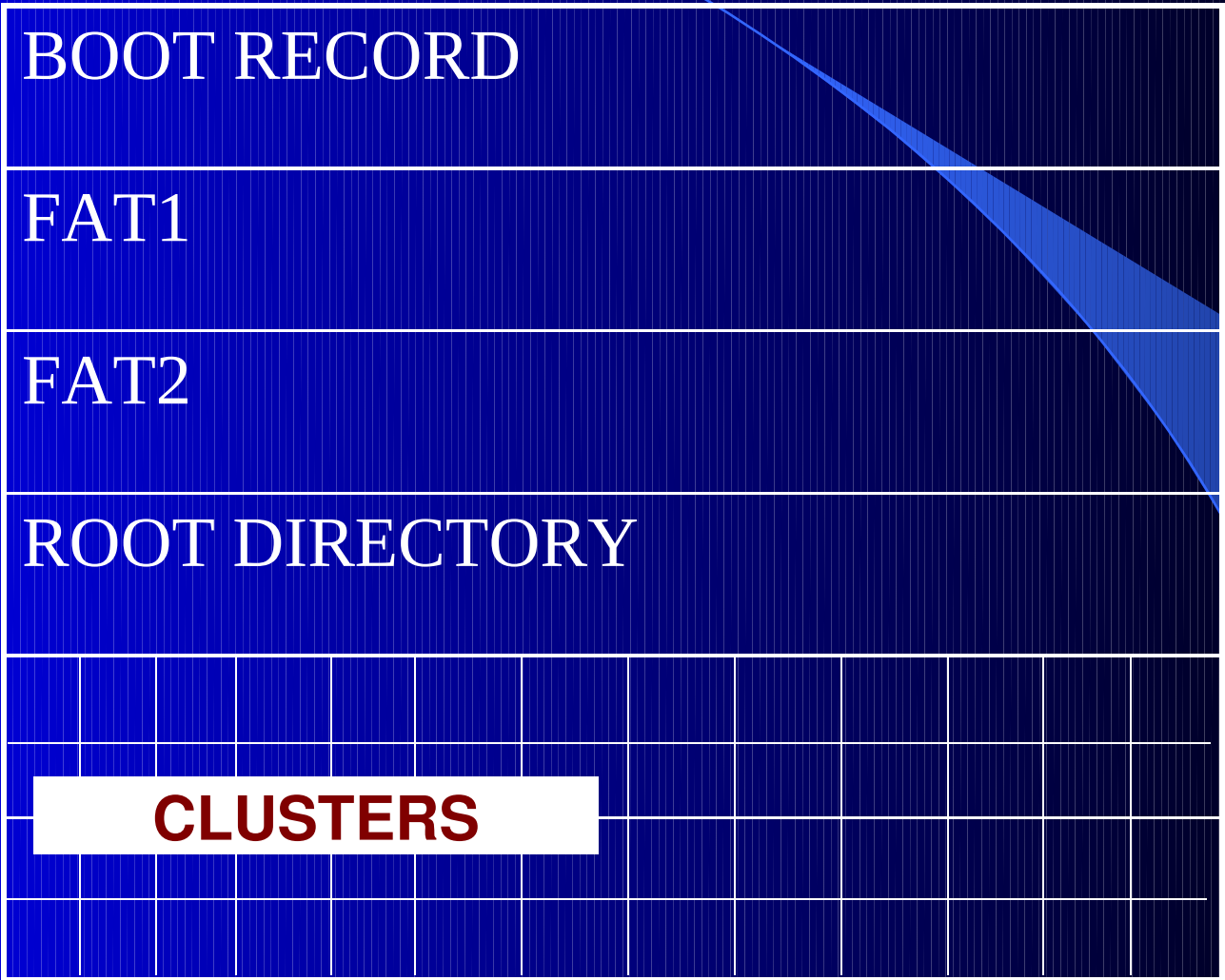
r : thuộc tính chỉ đọc (Read Only)

VÙNG LƯU TRỮ

- là vùng dành cho việc lưu trữ dữ liệu.
- như vậy việc lưu trữ dữ liệu trên đĩa có cấu trúc là 1 danh sách liên kết mà bảng thư mục gốc là đầu của danh sách liên kết.
- đầu mỗi cluster luôn luôn chứa địa chỉ của cluster sau nó cho biết phần còn lại của file là cluster nào. Nếu giá trị này là 0 thì cluster này là cluster cuối cùng.

SỰ PHÂN VÙNG TRÊN ĐĨA

SYSTEM
AREA



DATA
AREA

CÁC LOẠI ĐĨA

Disk Type	sides	track per side	sectors per track	total sectors	total bytes	cluster size
360K	2	40	9	720	1,024	368,640
720K	2	80	9	1,440	512	737,280
1.2MB	2	80	15	2,400	512	1,228,800
1.4MB	2	80	18	2,880	512	1,474,560
32MB	6	614	17	62,610	2,048	32,056,832

TÍNH DUNG LƯỢNG ĐĨA

Công thức tính dung lượng đĩa :

Dung lượng đĩa (bytes) = số byte/1 sector * số sector/1 track * số track/ 1 mặt đĩa * số mặt đĩa.

MỘT SỐ HÀM THAO TÁC VỚI FILE VÀ ĐĨA INT 21H

HÀM 36H INT 21H :

Lấy số bytes còn trống trên đĩa

Input :

AH = 36H DL = 063 đĩa (0 : mặc định, 1 ổ A

Output :

Có lỗi AX = 0FFFFH

Không lỗi : AX = số sector / cluster

BX = số cluster còn trống

DX = tổng số cluster trên đĩa

CX = số bytes/cluster



BÀI TẬP

Viết chương trình tạo thư mục với yêu cầu tên thư mục (có thể bao gồm tên ổ đĩa, đường dẫn và tên thư mục) được nhập từ bàn phím, cho phép sửa sai khi gõ nhầm tên thư mục.

Viết chương trình ghi dữ liệu vào file với yêu cầu :

- Tên file nhập từ bàn phím
- Dữ liệu ghi vào file cũng gõ từ bàn phím và kết thúc việc nhập bằng phím CTRL+Z

Viết chương trình gộp nội dung 1 file vào cuối 1 file khác.

LAÁP TRÌNH XÖÙ LYÙ FILE

**GIÖI THIỆU FILE
CÁC HÀM CHỨC NĂNG XỬ LÝ FILE
CỦA INT 21H CỦA DOS**

GIỚI THIỆU FILE

- Trong quản lý File, Dos vay mượn khái niệm Handle trong HÃH Unix để truy xuất File và thiết lập.



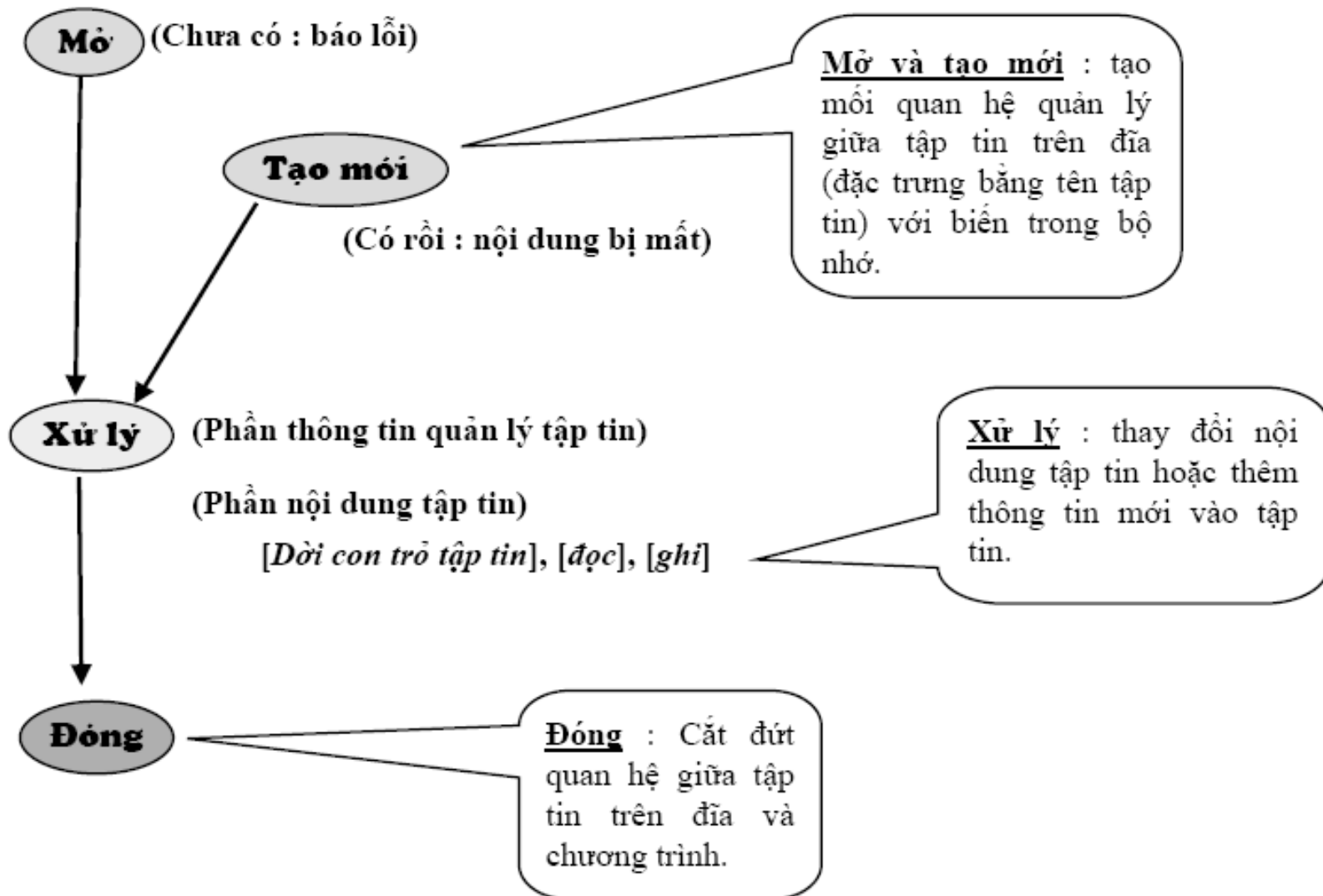
- Handle là 1 số 16 bits để Dos sử dụng để nhận biết File nào mở hoặc 1 thiết lập trong hệ thống.

GIÒÙI THIEÄU FILE

- Coù 5 Handle thieát bò chuaån ñöôïc Dos nhaän daïng.

Handle	Thieát bò
0	Keyboard, standard input
1	Console, standard output
2	Error output thieát bò xuaát loãi – maøn hình
3	Auxiliary device asynchronous
4	Printer

CÁC THAO TÁC XỬ LÝ FILE



CAÙC CHÖÙC NAÊNG CÔ BAÛN VEÀ XÖÙ LYÙ FILE CUÛA INT 21H

Chöùc naên

CAÙC CHÖÙC NAÊNG
NAÏY PHAÛI ÑÖA
VAØO AH

Taùc vui

- 3Ch Taio File môùi
- 3Dh Môu File ñaõ cou ñeã xuaát/nhaäp/vöøa nhaäp vöøa xuaát
- 3Eh Ñöùng theu File
- 3Fh Ñoïc töø File hay ñoïc töø thieát bò 1 soá bytes ñönh tröôùc
- 40h Ghi vaøo File hay ñoïc töø thieát bò 1 soá bytes ñönh tröôùc
- 42h di chuyeån con tröôùc File tröôùc khi ñoïc/ ghi

CHÖÙC NAÊNG TAÏO FILE

3Ch

CREATE FILE FUNCTION 3Ch

Chöùc naêng : Môu 1 File môùi ñeà ñoïc ghi. Neáu file ñaõ coù thì file cuõ seõ bò xoà.

AH = 3Ch

DS:DX ñeà chæ cuûa teân File muoán môu (ASCIIZ String)

CX = thuoäc tính File

(0 normal 1 ReadOnly 2 Hidden 4 System)

Xuaát : không loãi CF =0 AX = File Handle Coù loãi CF =1.

Maõ loãi trong AX (3,4,5).

3 Ch
CREATE FILE FUNCTION
3 Ch

Ex :

CREATE_FILE :

MOV AH, 3CH

MOV DX, OFFSET NEWFILE

MOV CX, 0

INT 21H

JC DISPLAY_ERROR

MOV NEWFILEHANDLE, AX

...

NEWFILE DB ' FILE1.DOC ',0

NEWFILEHANDLE DW ?

CHÖÜC NAÊNG TAÏO FILE 3 Ch

CREATE FILE FUNCTION 3 Ch

Ex :

CHÖÜC NAÊNG 3Ch COÙ 1 KHUYEÁT ÑIEÂM LAØ NEÁU COÙ 1 FILE CUØNG TEÂN(CUØNG ÑÖÔØNG DAÃN) ÑAÕ TOÀN TAÏI THÌ FILE CUÕ SEÕ BÒ XOÙA.

ÑEÃ BAÛO VEÃ FILE, COÙ 2 CAÙCH :

C1 : MÔU FILE BAÈNG CHÖÜC NAÊNG 3Dh, NEÁU FILE CHÖA COÙ THÌ TRAÛ VEÀ LOÃI SOÁ 2 (FILE NOT FOUND) → YEÂN TAÂM MÔU FILE MÔUI.

C2 : DUØNG CHÖÜC NAÊNG 5Bh MÔU FILE COÙ KIEÂM TRA TEÂN FILE NAØY ÑAÕ COÙ CHÖA.

CHỖC NAÊNG 5Bh TAÏO FILE MÔÙI COÙ KIEÂM TRA

ÑIEÀU KIEÃN : GIOÁNG CHỖC NAÊNG 3Ch

NEÁU FILE NAỖY ÑAỖ COÙ THÌ KHOÃNG MÔÙ FILE MÔÙI MAỖ TRAÙ
VEÀ LOÃI 50h

CREATE_FILE :

MOV AH,5BH

MOV DX, OFFSET FILENAME

MOV CX, 0

INT 21H

JC ERROR

....

FILENAME DB 'FILE1.DOC' , 0

CAÙC LOÃI KHI MÔU FILE

MAÕ LOÃI

DIEÃN GIAÛI

- 2 FILE NOT FOUND KHOÃNG TÌM THẢY FILE, COÙ THEÃ ÑÖÔNG DAÃN KHOÃNG ÑUỀNG HOẶC TEÃN FILE MÔ TAÛ KHOÃNG HỒP LỄ.
- 3 PATH NOT FOUND ÑÖÔNG DAÃN KHOÃNG COÙ.
- 4 TOO MANY OPEN FILES COÙ THEÃ DO LỄNH PATH XX TRONG CONFIG.SYS QUẦ NHỮ KHOÃNG CHO PHEÙP MÔ NHIEÀU FILE.
- 5 ACCESS DENIED TỖ CHỎI TRUY XUAÁT. COÙ THEÃ TA MUỐN XỎU FILE ÑANG MÔ, HAY FILE NAỖY COÙ THUỐC TÍNH CHẶ ÑỔC.

CH Mã truy nhập không hợp lệ.

FH Ổ đĩa không hợp lệ

05/02/11

LAP TRINH XU LY DIA TU

47

10h Đang tìm cách xóa thư mục hiện thời

CAÙC LOÃI KHI MÔÙ FILE

MAÕ LOÃI

DIEÃN GIAÙI

11H Không cung thiết bị

12H Không tìm được thêm File nào

CHÒUC NAÊNG MÔU FILE ÑAÕ COÙ 3DH Int 21h

OPEN FILE

ÑIEÀU KIEÄN :

AH = 3DH DS:DX ÑÒA CHÆ TEÂN FILE

AL = MODE

0: INPUT (MÔU CHÆ ÑOÏC)

1 : OUTPUT (MÔU ÑEẢ GHI)

2 : INPUT OUTPUT (MÔU VÖØA ÑOÏC VÖØA GHI)

XUAÁT :

KHOÂNG LOÃI CF = 0 AX = FILE HANDLE

COÙ LOÃI CF = 1 AX ← mã lỗi (2,4,512)



MỞ FILE HÀM 3CH INT 21H

- Trước khi sử dụng 1 file, ta phải mở nó.
- Để tạo 1 file mới hay ghi lại 1 file cũ, ta sử dụng tên file và thuộc tính của File.
- → DOS trả về thẻ file

MỞ FILE HÀM 3CH INT 21H

AH = 3CH

DS:DX địa chỉ của chuỗi ASCII

(chuỗi tên File kết thúc bằng byte 0)

CL = thuộc tính File

Nếu thành công, AX = thẻ File

**Nếu CF được set thì có lỗi, mã lỗi chứa trong AX
(lỗi 3,4,5)**

**Viết code mở 1 File mới với thuộc tính chỉ đọc,
tên File là FILE1**

Fname DB 'FILE1',0

FHANDLE DW ?

MOV AX,@DATA

MOV DS,AX

MOV AH,3CH

MOV CL,1

LEA DX,FNAME

INT 21H

MOV FHANDLE, AX

JC OPEN_ERROR

.....

CHÖÜC NAÊNG MÔU FILE ÑAÖ COÙ SÃN

HÃM 3DH INT 21H

OPEN FILE

AH = 3DH

DS:DX = địa chỉ của chuỗi ASCII
(chuỗi tên File kết thúc bằng byte 0)

AL = mã truy cập

0 : mở để đọc

1 : mở để ghi

2 : mở để đọc và ghi

→ Thành công, AX = Fhandle

→ Có lỗi. Mã lỗi chứa trong AX (2,4,5,12)

CHÖÜC NAÊNG MÔU FILE ÑAÕ COÙ SÃN

HÃM 3DH INT 21H

OPEN FILE

```
MOV AH, 3DH
MOV AL, 0
MOV DX, OFFSET FILENAME
INT 21H
JC DISPLAY_ERROR
MOV INFILEHANDLE, AX
.....
INFILE DB ' D:\FILE1.DOC', 0
INFILEHANDLE DW ?
```

CHỖC NAÊNG 3EH ÑOÙNG FILE

ÑIEÀU KIEÄN :

AH = 3EH BX = FILE HANDLE CAÀN ÑOÙNG

XUAÁT :

KHOÂNG LOÃI CF = 0 COÙ LOÃI CF = 1

EX :

MOV AH, 3EH

MOV BX, INFILEHANDLE

INT 21H

JC DISPLAY_ERROR

.....

INFILE DB 'D:\FIEL1.DOC', 0

INFILEHANDLE DW ?

LOÃI SOÁ 6 : INVALID HANDLE

**FILE HANDLE TRONG BX
KHOÂNG PHAÛI LAØ THEÙ FILE
CUÙA FILE ÑAÕ MÔÙ.**

CHÖÜC NAÊNG 3FH ÑOÏC FILE

ÑOÏC 1 SOÁ BYTES TÖØ FILE LÖU VAØO BOÄ NHÖU

ÑIEÄU KIEÄN :

AH = 3FH BX = FILE HANDLE , CX = SOÁ BYTES CAÀN ÑOÏC

DS:DX : ÑÒA CHÆ BOÄ ÑEÄM.

XUAÁT :

AX = SOÁ BYTES ÑOÏC ÑÖÖÏC, NEÁU AX = 0 HAY AX < CX FILE ÑAÕ KEÁT THUÙC.

NEÁU CÖØ CF ÑÖÖÏC LAÄP → COÙ LOÃI, MAÕ LOÃI CHÖÜA TRONG AX(5,6)

CHÖÙC NAÊNG 3FH ÑÖÏC FILE

EX : ÑÖÏC 1 SECTOR 512 BYTES TÖØ FILE

.DATA

HANDLE DW ?

**BUFFER DB 512
DUP(?)**

MOV AX, @DATA

MOV DS, AX

MOV AH, 3FH

MOV CX, 512

MOV BX, HANDLE

MOV CX, 512

INT 21H

JC READ_ERROR

**NEÁU CAÀN ÑÖÏC HEÁT CAÙC SECTOR
CHO ÑEÁN HEÁT FILE → EOF**

CMP AX, CX

JL EXIT

JMP READ_LOOP

CHỖC NAÊNG 40H GHI FILE

GHI 1 SOÁ BYTES LÊÂN FILE HAY THIẾT BÒ

INPUT :

AH =40H BX = THEÛ FILE CX = SOÁ BYTES CAÀN GHI

DS:DX : ÑÒA CHÆ VUÔNG ÑEÄM.

OUTPUT :

AX : SOÁ BYTES GHI ÑÖÖIC, NEÁU AX<CX, COÙ LOÃI (ÑÓA ÑAÀY).NEÁU CF ÑÖÖIC LAÁP → COÙ LOÃI, MAÕ LOÃI TRONG AX (5,6).

HAØM 40H CUÕNG COÙ THEÁ DUÕNG ÑEÁ ÑÒA DÖÖ LIEÄU RA MAØN HÌNH

CON TROÛ FILE

- DUØNG ÑEẢ ÑÒNH VÒ TRONG FILE.
- KHI FILE ÑÖÖIC MÔU, CON TROÛ FILE NAÈM ÔU ÑAÀU FILE.
- SAU MOÃI THAO TAUC ÑOIC, CON TROÛ FILE SEÕ DI CHUYEẢN ÑEẢN BYTE KEÁ.
- SAU KHI GHI 1 FILE MÔU CON TROÛ CHÆ ÑEẢN CUOÁI FILE (EOF).
- ÑEẢ DI CHUYEẢN CON TROÛ FILE HAØM 42H

MINH HỌA LẬP TRÌNH FILE

Viết chương trình cho phép User gõ vào tên File (có thể có kèm theo tên ổ đĩa, thư mục chứa file), chương trình sẽ đọc và hiển thị nội dung File ra màn hình.



DÒCH CHUYỂN CONTROL FILE HẠM 42H INT 21H

AH = 42H AL = PHƯƠNG THỨC TRUY NHẬP

0 DÒCH CHUYỂN TỔNG SỐ VỊ NÀO FILE.

**1 DÒCH CHUYỂN TỔNG SỐ VỊ VÀ TRÍ HIỆN THỜI
CUA CONTROL.**

2 DÒCH CHUYỂN TỔNG SỐ VỊ CUỐI FILE.

BX = THEU FILE.

CX : DX SỐ BYTES CẦN DÒCH CHUYỂN.

OUTPUT :

**DX:AX : VÀ TRÍ MÙI CUA CONTROL FILE TÍNH BẰNG BYTE
TỔ NÀO FILE.**

NEÁU CF =1 MÃ LÃI TRONG AX (1, 6).

DÒCH CHUYỂN CONTROL FILE HÀM 42H INT 21H

CX : DX CHỖA SỐA BYTES NẾA DI CHUYỂN CONTROL. NẾU LAØ SỐA DÖÔNG → CHUYỂN VÈ CUỐI FILE.

NEÁU LAØ SỐA AÂM → CHUYỂN VÈ NÀU FILE.

DI CHUYỂN CONTROL FILE NẾAN CUỐI FILE VAØ XÀUC NÖNH KÍCH THÖÖC FILE

MOV AH, 42H ; DI CHUYỂN CONTROL FILE

MOV BX, HANDLE ; LAÁY THEU FILE

XOR DX, DX

XOR CX, CX ; DÖCH CHUYỂN 0 BYTE

MOV AL, 2 ; TÍNH TÖØ CUỐI FILE

INT 21H ; CHUYỂN CONTROL NẾAN CUỐI FILE, DX:AX KÍCH THÖÖC FILE

JC MOVE_ERROR

03/02/11

LAP TRÌNH XU LY DIA T62

THAY ÑOÀI THUOÁC TÍNH FILE HAØM 43H INT 21H

INPUT :

AH = 43H DS :DX = ÑÒA CHÆ CHUOÀI ASCII STRING

AL = 0 ÑÈẢ LAÁY THUOÁC TÍNH FILE AL =1 ÑÈẢ THAY ÑOÀI THUOÁC TÍNH FILE, CX = THUOÁC TÍNH FILE MÔÙI (NEÁU AL =1)

OUTPUT :

NEÁU THAØNH COÂNG, CX = THUOÁC TÍNH HIEÄN THÔØI

NEÁU CF ÑÖÖIC LAÁP → COÙ LOÃI, MAÕ LOÃI TRONG AX (2,3,5).

Ex : thay ñoài thuoäc tính File thaønh hidden file

MOV AH, 43H	; Haøm laáy / ñoài thuoäc tính File
MOV AL, 1	; tuyø chöïn thay ñoài thuoäc tính
LEA DX, FILENAME	; laáy tên file keá câu ñöông ðaãn.
MOV CX, 1	I; thuoäc tính Hideen
INT 21H	; ñoài thuoäc tính
JC ATT_ERROR	; thoàùt neáu coù loãi, maõ loãi trong AX

LẬP TRÌNH FILE

1. **Viết chương trình chép một file nguồn đến một file đích trong đó thay chữ thường bằng chữ hoa.**
2. **Viết chương trình đọc 2 file và hiển thị chúng bên cạnh nhau trên màn hình. Chú ý có chức năng dừng từng trang màn hình nếu file quá dài.**
3. **Viết chương trình ghép nội dung 1 file vào cuối 1 file khác đã có.**
4. **Viết chương trình tạo 1 thư mục, tên thư mục được gõ từ bàn phím (tên thư mục có thể bao gồm tên ổ đĩa, đường dẫn).**