NINTENDO64
DiskDrive
Multi FileSystem Specification

1998/07/11 Version 1.00

INDEX

1. Introduction

This specification contains detail information of MultiFileSystem (MFS) which is a file system Nintendo recommends. If you create your own libraries to access the MFS format disk by referring this specification, it might have compatibility problems.

Therefore, this specification should be referred to use MFS libraries and make tools for 64DD.

2. Outline of MultFileSystem

2.1. Outline

The 64DD has two areas which are read-only ROM area and read-write RAM area. In MFS, its file system is separated for the ROM area and RAM area because of easiness of file management and usage efficiency of disk. MFS divide disk into five areas shown at Figure 2.1.

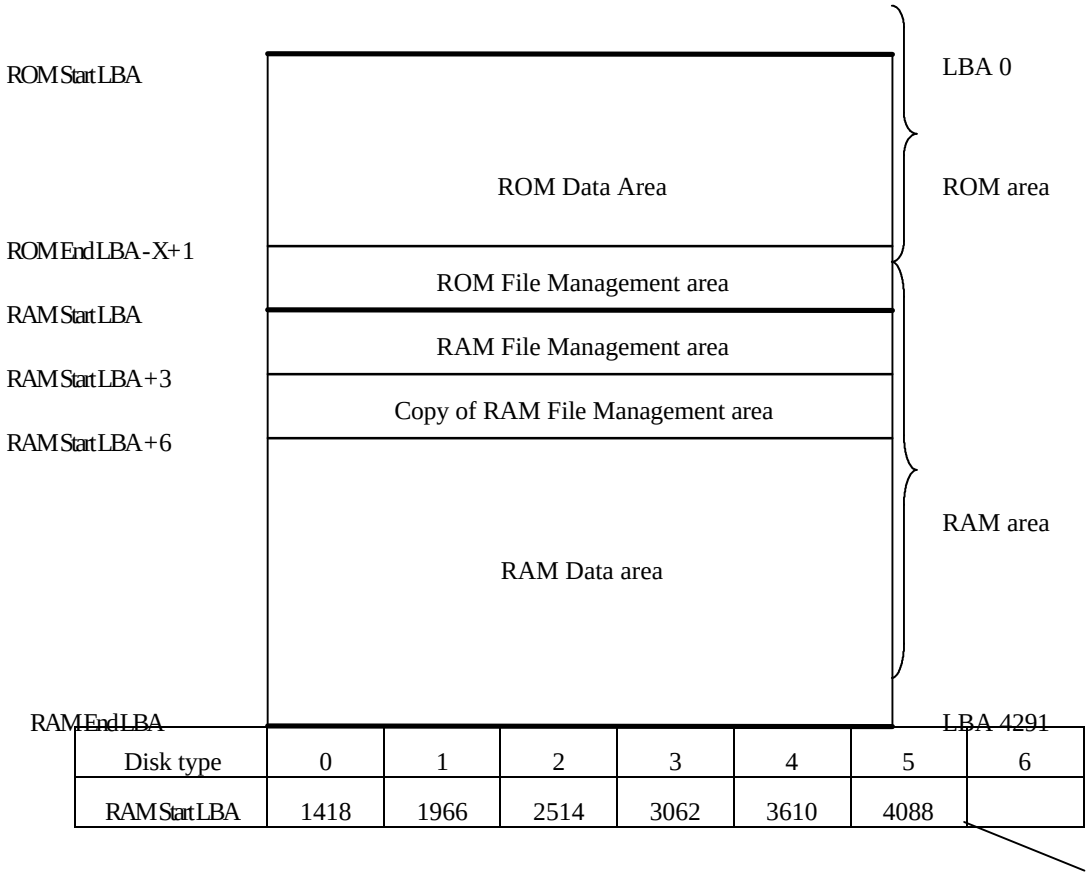


Figure 2.1: Disk block figure

In Figure 2.1, "X" is a number of blocks of ROM File Management area. The number of ROM File Management area is changeable and set depending on file numbers stored in the ROM Date area.

It is not necessary that both ROM area and RAM area need to be MFS format in a disk. Please note that there is only ROM area at disk type 6.

It is necessary to set 1 in the RAM area usage in the disk ID area, when MFS is ussed in RAM area.

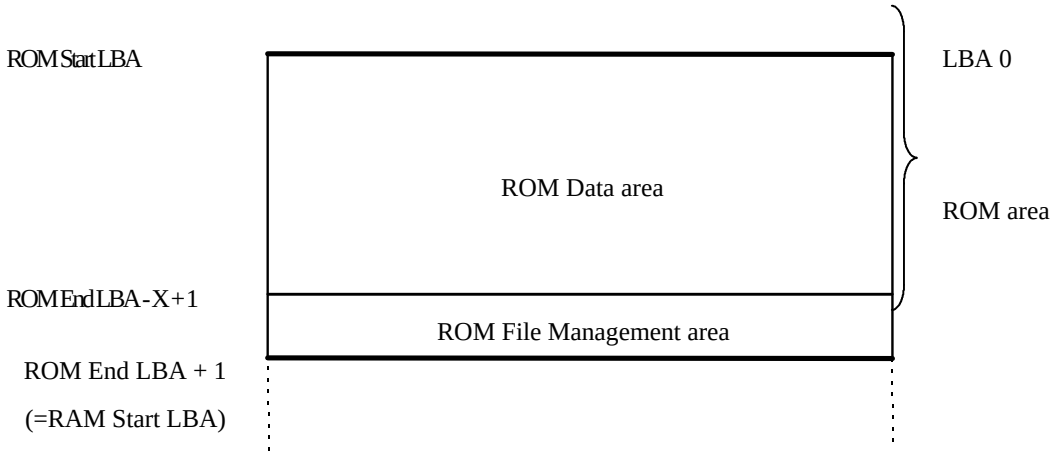
3. File System - ROM area

3.1. Outline of File System - ROM area

The file system in ROM area divides the ROM area into two blocks, which are "ROM File Management area" and "ROM Data area". (Figure 3.1)

In the "ROM File Management area", there are disk information and information which is needed to manage files. The block number in this is depending on numbers of stored files and directories. Since it is in ROM area, it cannot be changed by application.

In the "ROM Data area", there are actual files. It is possible to have mixed data which either MFS or Leo functions can access.



Disk type	0	1	2	3	4	5	6
ROM End LBA	1417	1965	2513	3061	3609	4087	4291

• X is number of blocks in the ROM File Management area

Figure 3.1:ROM area block figure

3.2. File Management area - ROM area

The ROM File Management area locates at the end of the ROM area. (Figure 3.1) There are disk and file information in this area to manage files.

The number of blocks in the ROM File Management area is changeable depending on number of files and directories. It should be decided when making disk image in ROM. Since it is in ROM area, it cannot be changed by game application.

In the ROM File Management area, there are "ID" and "Directory Entry". (Figure 3.2)

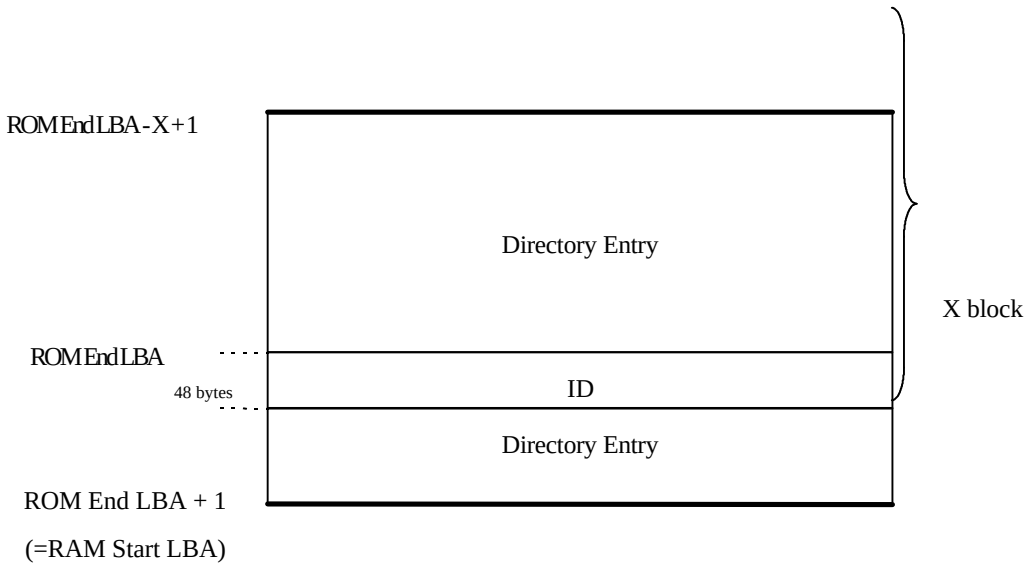


Figure 3.2: ROM File Management area block figure

3.3. ID - ROM area

The ID area for ROM locates at the end block in ROM area (Figure 3.2) and is 48 bytes from the top. It contains disk discrimination and information. The libraries detect whether or not ROM area is MFS format by checking this discrimination.

The disk information contains disk attribute, volume and etc. The size of the ID area is fixed 48 bytes. (Figure 3.3)

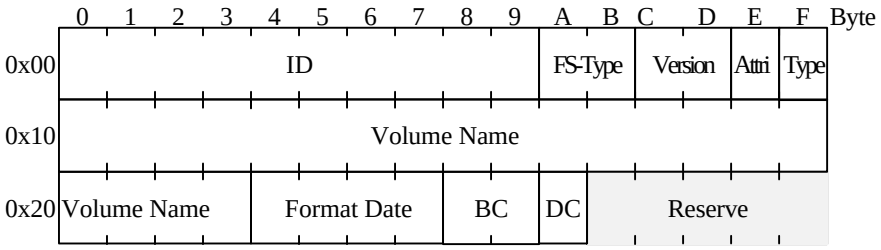


Figure 3.3: ID - ROM area

00 - 09 : ID (Disk discrimination)

"64dd-Multi" in ASCII code from top

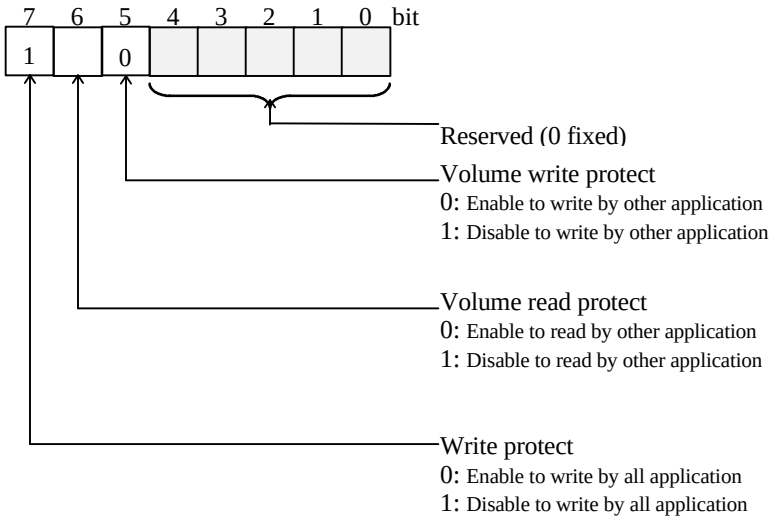
0A - 0B : FS-Type•i File system type•j

"02" in ASCII code for ROM area

0C - 0D : Version (File system version)

Only version 1 exists. "01" in ASCII code.

0E : Attr (Volume attribute)



- If "Volume write protect" is 1, disk can be written only when the company code and game code in the disk system area matches those specified in the libraries. Although this is invalid anyway in ROM area, "0" should be specified here.
- If "Volume read protect" is 1, disk can be read only when the company code and game code in the disk system area matches those specified in the libraries.
- If "Write Protect" is 1, it prohibits all applications from writing into disk. This bit should be 1 in the ROM area.

0F : Type (Disk type)

Disk type (0 to 6) in hexadecimal.

10 - 23 : Volume Name

Volume name in Shift JIS code. NULL (0x00) for unused area. (Refer to Appendix. A)

24 - 27 : Format Date

Date when disk is formatted. (Refer to chapter 5)

28 - 29 : BC (Number of blocks in management area)

Number of blocks used for the ROM File Management area.

2A : DC (Destination code)

0 for Japan, 1 for US in hexadecimal.

2B : Reserve

Reserved area. It should be filled with 0 in hexadecimal.

3.4. Directory Entry - ROM area

The Directory Entry is a table which contains name ,attribute and other information of files or directories. The size of the Directory Entry is depending on number of files and directories.

The Directory Entry is allocated from the end of the ID area and from big offset of block to small offset in the block. If the entries are more than 1 block, it will be allocated from the ROM End LBA to ROM Start LBA. The free entries in the block the last entry exists is filled with "0". (Figure 3.4)

The contents of the Directory Entry are partially different between files and directories. (Figure 3.5, 3.6) The directory entry 0 is a root directory so that it cannot be changed.

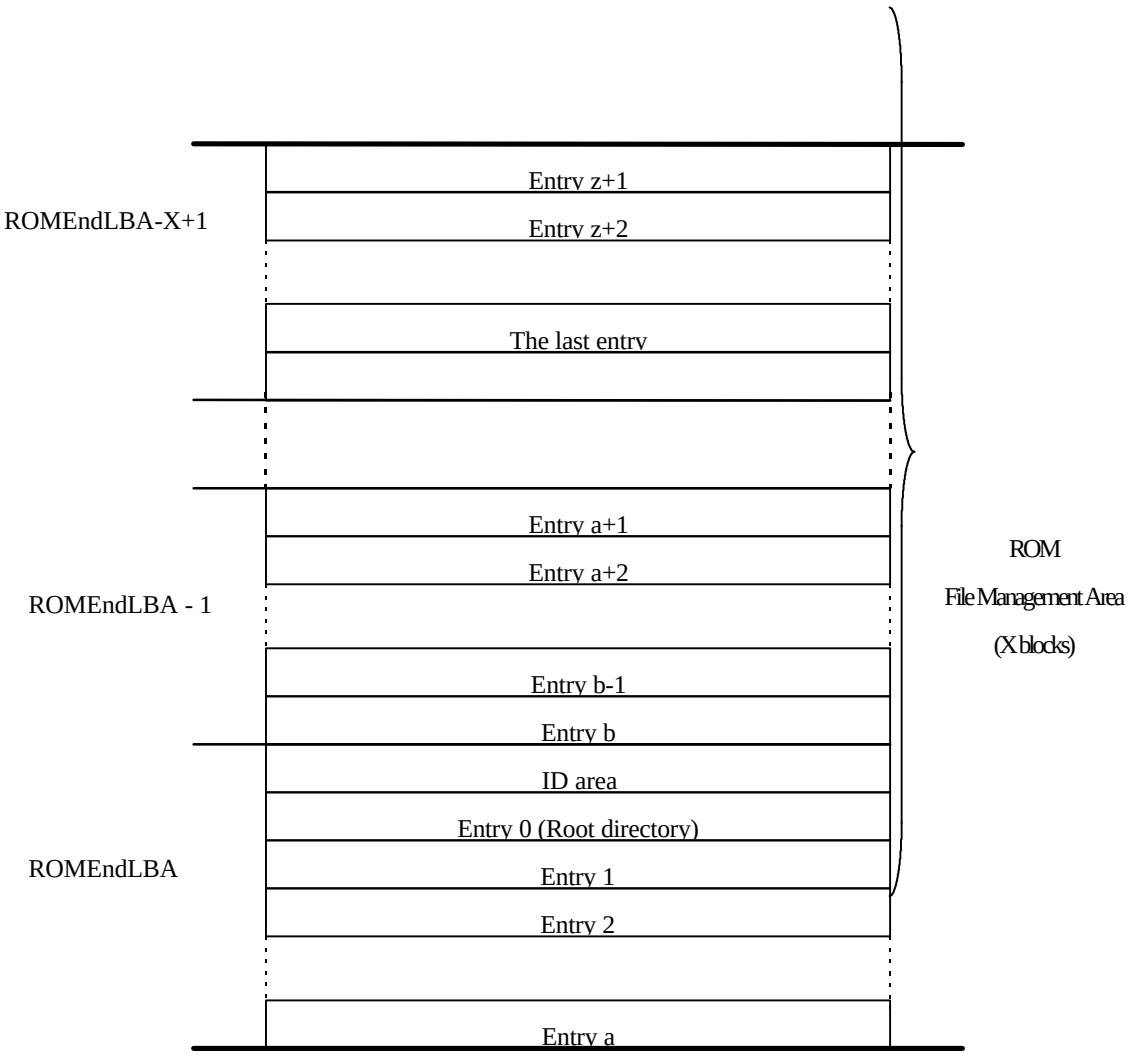
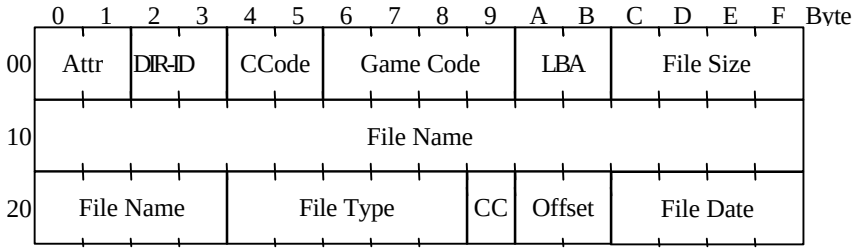
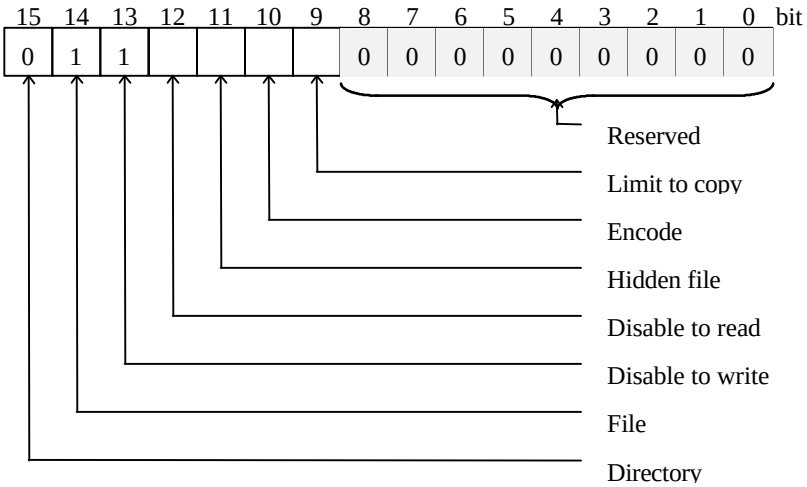


Figure 3.4: Directory Entry allocation in ROM area

Figure 3.5: Directory Entry (File) - ROM area



00 - 01 : Attr (Attribute)



"Limit to copy" - This bit enable/disable to limit number of times for copy. If it is 1, copy is allowed within number specified in CC(Copy Counter). This bit is only for RAM area so that this should be set with "0" in ROM area.

"Encode" - If this is "1", it is an encoded file.

"Hidden file" - This represents file is hidden for user. Hiding file is done by application.

"Disable to read" - If this is "1", file can be accessed only when the company code and game code in the file matches those specified in the libraries. If this is "0", there is no limitation.

"Disable to write" - If this is "1", file cannot be written, deleted and renamed. This bit is for RAM area so that this should be "1" for ROM area.

"File" - This represents the entry is a file. This should be 1 for file.

"Directory" - This represents the entry is a directory. This should be 0 for file.

Note: If no entry, these attributes should be 0.

02 - 03 : DIR-ID (Parent directory ID)

This is parent’s directory ID which a file belongs. The file is affected by an attribute which parent's directory exits one above hierarchy.

04 - 05 : CCode (Company code)

06 - 09 : Game Code (Game code)

0A - 0B : LBA (Start LBA)

Start LBA which contains data of file is stored.

0C - 0F : File Size

10 - 23 : File Name

Use the shift JIS code. The blank should be filled with NULL (0x00). (Refer to Appendix A)

24 - 28 : File Type (Extension)

Type of file should be specified in ASCII code.

29 : CC (Number of times can be copied)

When the ninth bit of the attribute is "1", this number is used for number of times to be copied. This should decrement by "1" every time it is copied. If this becomes 0, the copy should not be allowed any more. If the ninth bit is "0", there is no limitation.

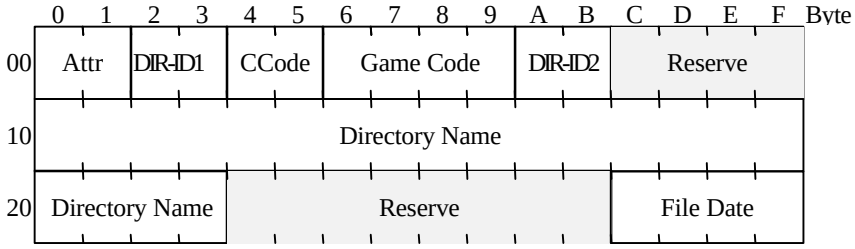
2A - 2B : Offset

Offset from the top of a block specified at the start LBA. Data of files is allocated from this offset. This offset should be a multiple of 8.

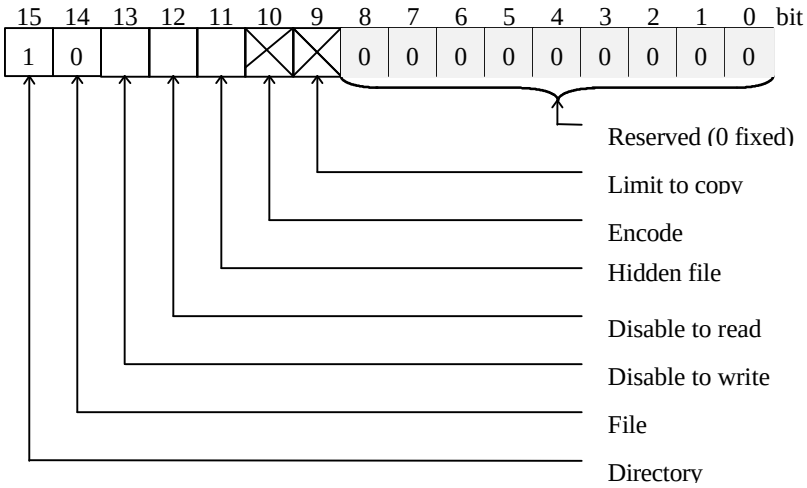
2C - 2F : File Date (Date)

Date when entry is created. (Refer to the fifth chapter)

Figure 3.6: Directory Entry in ROM area (Directory)



00 - 01 : Attr (Attribute)



"Limit to copy" - This bit enable/disable to limit number of times for copy. In case of directory, this is invalid so set "0".

"Encode" - If this is "1", it is an encoded file. In case of directory, this is invalid so set "0".

"Hidden file" - This represents file is hidden for user. Hiding file is done by application.

"Disable to read" - If this is "1", file can be accessed only when the company code and game code in the directory matches those specified in the libraries. If this is "0", there is no limitation.

"Disable to write" - If this is "1", file cannot be written, deleted and renamed. This bit is for RAM area so that this should be "1" for ROM area.

"File" - This represents the entry is a file. This should be 0 for directory.

"Directory" - This represents the entry is a directory. This should be 1 for directory.

Note: If no entry, these attributes should be 0.

02 - 03 : DIR-ID1 (Parent directory ID)

This is parent's directory ID which a file belongs. The directory ID is 0x0000 for a root

directory. The parent directory of the root directory is 0xFFFE.

The directory is affected by an attribute which parent's directory exits one above hierarchy.

04 - 05 : CCode (Company code)

06 - 09 : Game Code (Game code)

0A - 0B : DIR-ID2 (Directory ID)

Directory ID of entry itself.

0C - 0F : Reserve

Reserved area. This should be filled with "0".

10 - 23 : Directory Name

Use Shift JIS code. The blank should be filled with NULL(0x00). (Refer to Appendix. A)

24 - 2B : Reserve

Reserved area. This should be filled with "0".

2C - 2F : File Date

Date when a directory is created. (Refer to the fifth chapter)

3.5. Data area - ROM area

Data in ROM area should be stored from LBA0 [(ROM End LBA) - (Number of blocks of ROM File Management area)]. However, files which are not managed by MFS (In another word, files (data) are accessed directly application.) can be included.

File data is stored from "Offset" in the block specified "Start LBA" in "Directory Entry".

The "Offset" is a multiple of 8. Files should be stored in unit of a multiple of 8. If file size is not a multiple of 8, the blank at the end is filled with "0".

4. File System - RAM area

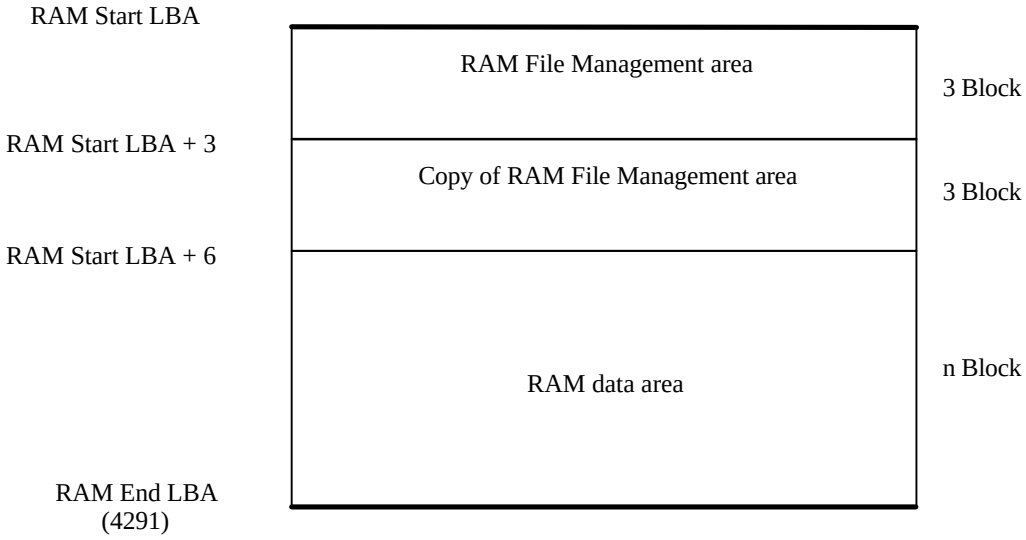
4.1. Outline of File System - RAM area

The file system in RAM area divides the RAM area roughly into three blocks, which are "RAM File Management area", "Copy of the RAM File Management area" and "RAM Data area". (Figure 4.1)

In the "RAM File Management area", there are disk information and information which needs to manage files. The number of block is 3 not depending on the disk type.

In the "Copy of the RAM File Management area", The same information in the "RAM File Management area" is stored. In case the "RAM File Management area" is destroyed, this is used to fix it. This is 3 blocks not depending on the disk type.

In the "RAM Data area", actual files are in here. The number of blocks is depending on the disk type.



Disk Type	0	1	2	3	4	5	6
RAM Start LBA	1418	1996	2514	3062	3610	4088	-
Data area n(Block)	2870	2322	1774	1226	678	200	-

Figure 4.1: Whole RAM area block fiture

4.2. File Management area - ROM area

The ROM File Management area locates at the top of the RAM area. There are disk and file information in this area to manage files.

The ROM File Management area is divided into three areas, which area "ID", "File Allocation Table (FAT)" and "Directory Entry". (Figure 4.2)

Although the block size of the file control area is fixed three blocks, the actual size of the file management area varies because the size of the block is depending on the disk type. Therefore, the size of "ID" and "FAT" is fixed, and the size of "Directory Entry" is depending on disk type.

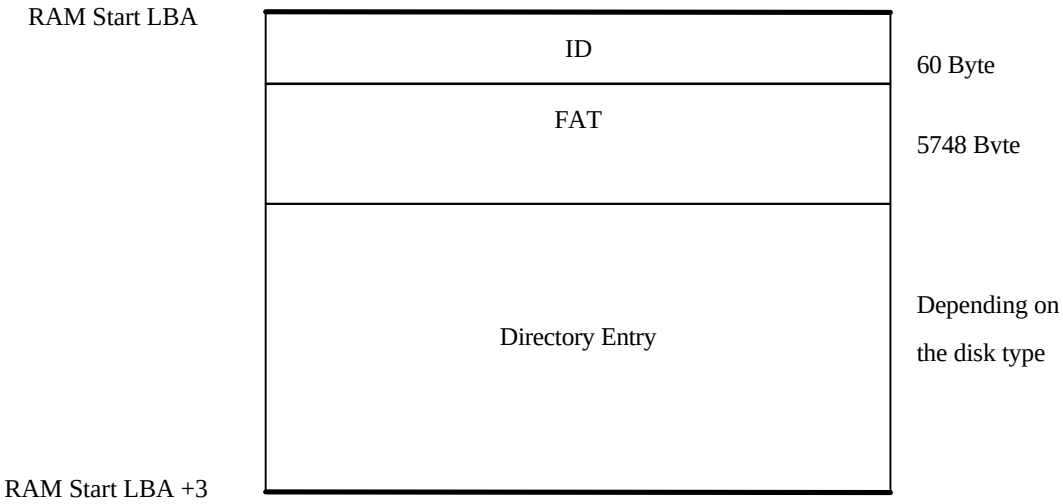


Figure 4.2: File Management area - RAM area

4.3. ID - RAM area

The ID area contains disk discrimination code and disk information.

The disk discrimination code contains ID to distinguish file system. It can be checked by this whether or not RAM area of disk is for MFS format.

The disk information contains disk attribute, volume and etc.

The size of the ID area is fixed 60 bytes. (Figure 4.3)

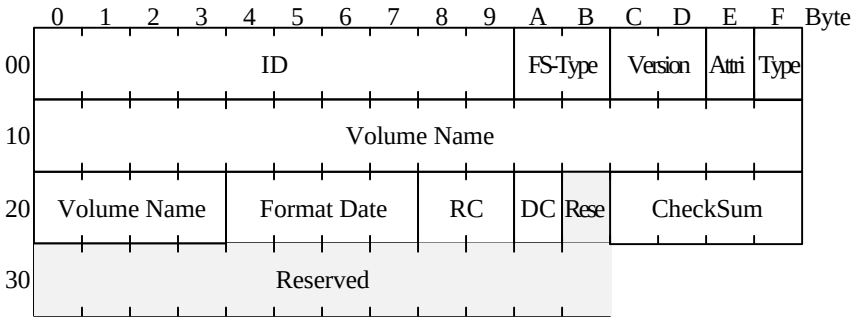


Figure 4.3: ID - RAM area

00 - 09 : ID (Disk discrimination)

"64dd-Multi" from top in ASCII code.

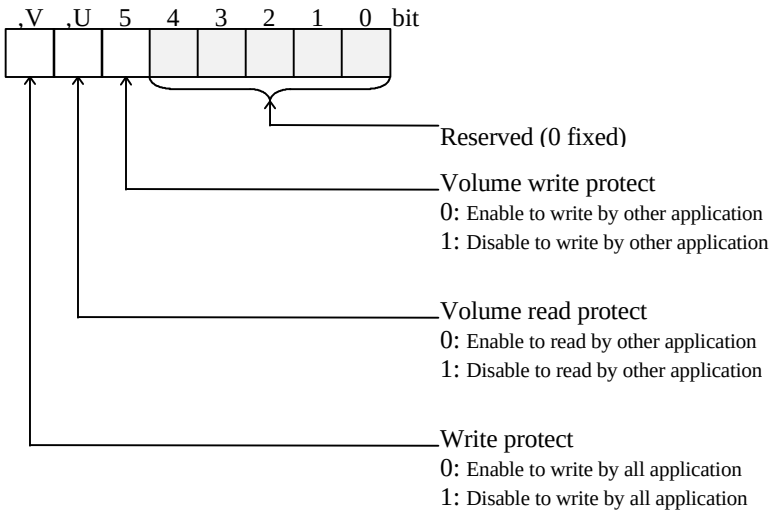
0A - 0B : FS-Type (Type of file system)

"01" in ASCII code in RAM area.

0C - 0D : Version (File system version)

Only version 1 exists. "01" in ASCII code.

0E : Attr (Volume attribute)



If "Volume write protect" is 1, disk can be written only when the company code and game code in the disk system area matches those specified in the libraries. Although this is invalid anyway in ROM area, "0" should be specified here.

If "Volume read protect" is 1, disk can be read only when the company code and game code in the disk system area matches those specified in the libraries.

If "Write Protect" is 1, it prohibits all applications from writing into disk.

0F : Type (Disk type)

Disk type (0 to 5) in hexadecimal.

10 - 23 : Volume Name

Volume name in Shift JIS code. NULL (0x00) for unused area. (Refer to Appendix. A)

24 - 27 : Format Date

Date when disk is formatted. (Refer to chapter 5)

28 - 29 : RC (Renewal Counter)

This increments every time the file management area is written. 0x0000 should be next after 0xFFFF. This can be used to check for change of disk contents and for initial value (seed) for random number.

2A : DC (Destination code)

0 for Japan, 1 for US in hexadecimal.

2B : Reserve

Reserved area. It should be filled with 0 in hexadecimal.

2C - 2F :CheckSum (Check sum)

The check sum of ID area should be stored. The check sum should be calculated every time values in the file management area are changed. Here is how to calculate:

1) The check sum area should be filled with "0".

2) The 4-byte exclusive OR should be applied from top of entire area of the file management area.

3) A value given by the exclusive OR should be stored in the check sum area.

At the production testing, the exclusive OR is applied to the file management area including the check sum area. If "0" is given, the file management area is correct.

30 - 3B :Reserve

Reserved area. It should be filled with 0 in hexadecimal.

4.4. File Allocation Table (FAT)

FAT is a table which contains link and block information of file blocks and its size is fixed 5748 bytes. Sixteen-bit unit is used for the FAT and a relative block address from top LBA in RAM is stored in FAT entry. Since data area starts from RAM Start LBA+6, a value stored in FAT entry should be more than 6.

Values stored in FAT entry will be one described in table 4.1. Top twelve bytes of FAT is FAT-ID area and it should be filled with "0".

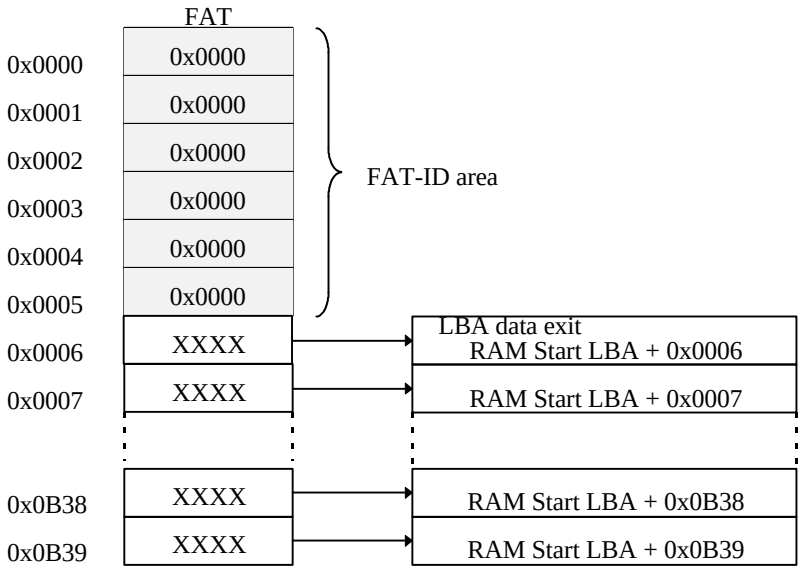


Figure 4.4: FAT

Table 4.1: Value in FAT

FAT Entry value	FAT contests
0x0000	Unused block
0x0001- 0x0005	Unused area
0x0006- 0x0B39	Represent next LBA
0x0B3A - 0x7FFF	Unused area
0x8000 - 0xFFFC	Reserved
0xFFFD	Out of the management
0xFFFE	Prohibited to use
0xFFFF	Last block of file

4.5. Directory Entry - RAM area

The Directory Entry is a table to manage files and directories and contains attributes, entry number of FAT, date and so on. (Figure 4.5) The Size of the Directory Entry is depending on the disk type, however, the size of a single Directory Entry is fixed 48 bytes. Therefore, numbers of the Directory Entry is also depending on the disk type. (Refer to 4.6)

The contents of the Directory Entry is partially different between files and directories. (Figure 2.5 for files, 2.6 for directories) The Directory Entry 0 is a root directory so that it cannot be changed.

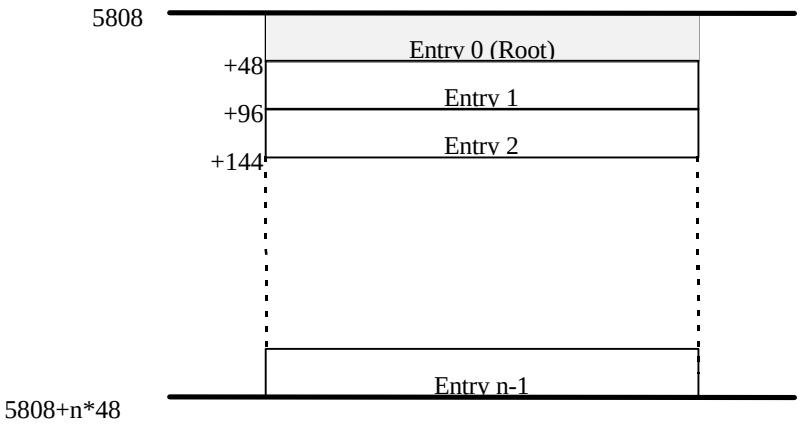
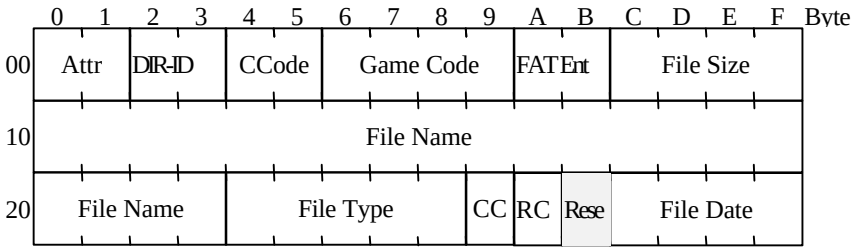
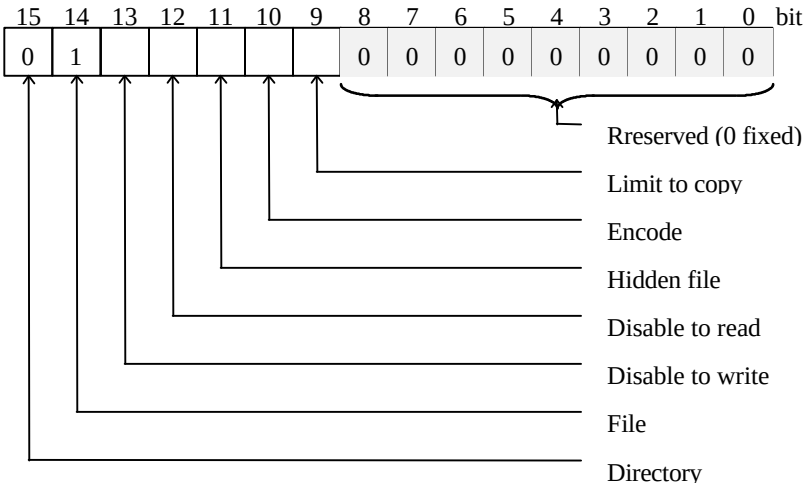


Figure 4.5: Directory Entry Table - RAM area

Figure 4.6: Directory Entry - RAM area (File)



00 - 01 : Attr (Attribute)



"Limit to copy" - This bit enable/disable to limit number of times for copy. If it is 1, copy is allowed within number specified in CC(Copy Counter).

"Encode" - If this is "1", it is an encoded file. This is not supported currently.

"Hidden file" - This represents file is hidden for user. Hiding file is done by application.

"Disable to read" - If this is "1", file can be accessed only when the company code and game code in the file matches those specified in the libraries. If this is "0", there is no limitation.

"Disable to write" - If this is "1", file cannot be written, deleted and renamed.

"File" - This represents the entry is a file. This should be 1 for file.

"Directory" - This represents the entry is a directory. This should be 0 for file.

Note: If no entry, these attributes should be 0.

02 - 03 : DIR-ID (Parent directory ID)

This is parent's directory ID which a file belongs.

The directory is affected by an attribute which parent's directory exits one above hierarchy.

04 - 05 : CCode (Company code)

06 - 09 : GameCode (Game code)

0A - 0B : FAT (FAT entry number)

FAT entry number

0C - 0F : File Size

10 - 23 : File Name

Use Shift JIS code. The blank should be filled with NULL(0x00). (Refer to Appendix. A)

24 - 28 : File Type (Extension)

Specify a type of file by ASCII code.

29 : CC (Number of times can be copied)

When the ninth bit of the attribute is "1", this number is used for number of times to be copied. This should decrement by "1" every time it is copied. If this becomes 0, the copy should not be allowed any more.

2A : RC (Renewal Counter)

This increments every time either the files are written, file name is changed or attributes are updated. This should be "0" for new files. This counter stops at "0xFF".

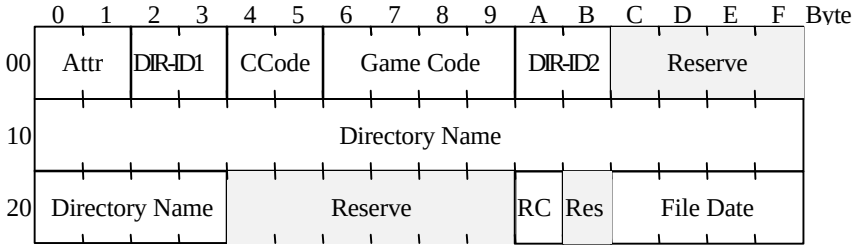
2B : Reserve

Reserved area. Reserved area. This should be filled with "0".

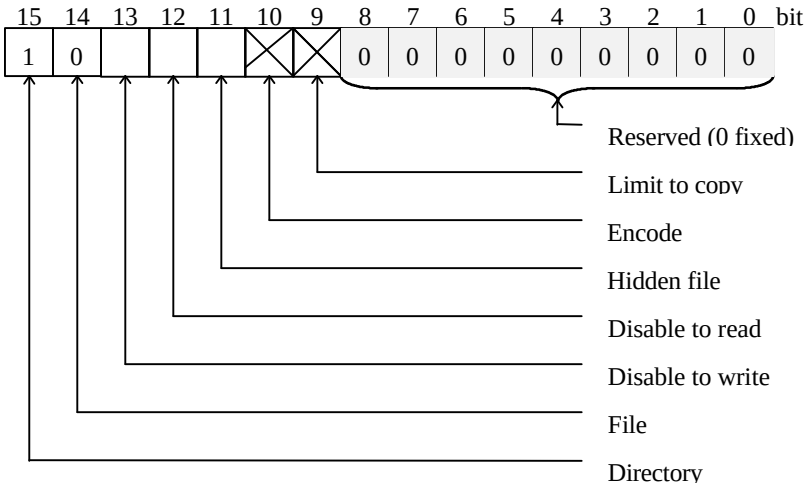
2C - 3F : File Date

Date when a file is created. (Refer to the fifth chapter)

Figure 4.7: Directory Entry - RAM area (Directory)



00 - 01 : Attr (Attribute)



"Limit to copy" - This bit enable/disable to limit number of times for copy. This is invalid for directories.

"Encode" - If this is "1", it is an encoded file. In case of directory, this is invalid so set "0".

"Hidden file" - This represents file is hidden for user. Hiding file is done by application.

"Disable to read" - If this is "1", file can be accessed only when the company code and game code in the directory matches those specified in the libraries. If this is "0", there is no limitation.

"Disable to write" - If this is "1", the directory cannot be written, deleted and renamed.

"File" - This represents the entry is a file. This should be 0 for directory.

"Directory" - This represents the entry is a directory. This should be 1 for directory.

Note: If no entry, these attributes should be 0.

02 - 03 : DIR-ID1 (Parent directory ID)

This is parent's directory ID in one above hierarchy from one a file belongs. The directory

ID is 0x0000 for a root directory. The parent directory of the root directory is 0xFFFE. The directory is affected by an attribute which parent's directory exits one above hierarchy.

04 - 05 : CCode (Company code)

06 - 09 : Game Code

0A - 0B : DIR-ID2 (Directory ID)

Directory ID of entry itself.

0C - 0F : Reserve

Reserved area. This should be filled with "0".

10 - 23 : Directory Name

Use Shift JIS code. The blank should be filled with NULL(0x00). (Refer to Appendix. A)

24 - 29 : Reserve

Reserved area. This should be filled with "0".

2A : RC (Renewal Counter)

This increments every time either the contests of entry or the files in the directory are updated. This counter stops at "0xFF".

2B : Reserve

Reserved area. This should be filled with "0".

2C - 3F : Date

Date when a directory is created. (Refer to the fifth chapter)

4.6. Number of files can be stored

The top block size of RAM area varies depending on the disk type so that the size of the RAM File Management area varies. The size for "ID" and "FAT" in it are fixed, so only the "Directory Entry" varies depending on the disk type.

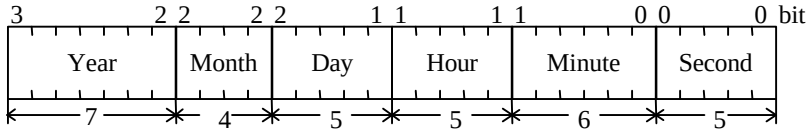
Therefore, the number of files can be stores varies depending on the disk type. (Table 4.2)

Table 4.2: Disk type and number of files

Disk Type	Number of files
0	899
1	814
2	729
3	644
4	559
5	474
6	

5. Date area

Date and time information is stored in the following format (Figure 5.1).



00 - 04 : Second (0 to 29)
Half of actual second

05 - 10 : Minute (0 to 59)

11 - 15 : Hour (0 to 23)

16 - 20 : Day (1 to 31)

21 - 24 : Month (1 to 12)

25 - 31 : Year (0 to 99)
"Year" is a relative value to 1996. To get an actual year, the value in "Year" should be added by 1996. Due to some limitation of 64DD libraries, the maximum year is 2095.

Figure 5.1: Date area

APPENDIX.A Shift JIS code in MultiFileSystem

The shift JIS code is developed by Microsoft and used in various platforms as internal code including Japanese OS for Macintosh and OS for Microsoft. The shift JIS is also called as either "MS Kanji" or "SJIS" which Shift-JIS is shorten. In 64DD, the shift JIS is used to access Kanji characters in the internal ROM as character code for name of volume and file.

The shift JIS has two modes which are one byte per one character mode and two bytes per one character. In one byte per one character mode, ASCII/JIS roman characters and 8-bit half size Katakana are available.

If value for a character is within either 129 to 159 or 224 to 239 in decimal (0x81 to 0x9F or 0xE0 to 0xEF in hexadecimal), it becomes two bytes per one character mode so that the value is considered as first byte out of two bytes. The second byte should be within either 64 to 126 or 128 to 252 in decimal (0x40 to 0x7E or 0x80 to 0xFC in hexadecimal).

However, font data in the internal 64DD ROM is only first to 47th division of JIS X 0208-1990 set which are non-Kanji characters and JIS first level Kanji characters. It means that characters can be used for file name and etc. are limited. For detail, please refer to "64DD programming manual - Appendix B Font data in the internal ROM".

Character codes can be used in MultiFileSystem are described in table A.1. Differences from the shift JIS code are that fist byte is within 0x81 to 0x98 and 0x20 which is control code as space is included within byte area of ASCII/JIS roman characters. However, '/', '.' and ':' are reserved symbols so that it shall not be used in either file or directory name

Table A.1 : Character codes in MultiFileSystem

	Decimal	Hexadecimal
2-byte character		
First byte area	129-152	81-98
Second byte area	64-126,128-252	40-7E,80-FC
Half size Katakana byte area	161-223	A1-DF
ASCII/JIS Roman byte area*	32-126	20-7E

*0x2E,0x2F,0x3A ('/', '.', ':') are reserved.

APPENDIX.B Update History

Version 1.00

Official release

Version 0.91

Change "Volume protect" of attribute in the Volume ID area to "Volume read protect" and "Volume write protect".

Version 0.9

Add file system in ROM area.

Change the description of the file system in RAM area.

Add area for copy in the File Management area.

Delete the reserved area.

Change FAT area because of adding the file management area.

Version 0.8

In ID area, "Country code" has been changed to "Destination code".

In the explanation of file name for directory entry, an explanation of a symbol for directory distinction has been added.

In the Appendix. A, the explanation of a symbol for directory distinction has been added.

In 2.6, the start of year has been changed from 1997 to 1996.

In 2.4, condition for free area in directory entry has been changed.

The contents in 2.4 has been changed.

The explanation in 2.3 has been changed.

In 2.6, the bit for second has been changed from 6 to 5, and bit for year has increased to 7.

Version 0.7

"Renewal Counter" has been added in ID area and Directory Entry.

Format to save date area in ID area and Directory Entry has been changed.

Area out of control has been added in FAT Entry.

"2.6 Date area" has been changed.

"APPENDIX. A Shift JIS code in MultiFileSystem" has been added.

"APPENDIX. B History of update" has been added.