DISC ORGANIZATION

The disc organization described below applies not only to user libraries and files, but also to the system components which are part of the SYSTEM library. They are considered as belonging to a special user, with the user identification SYSTEM.

Disc Space Allocation

The space on disc is divided into granules, track areas which are eight sectors long, each sector consisting of 200 words. The number of granules per disc depends on the type of disc used. All space allocation on disc takes place on the basis of these granules.

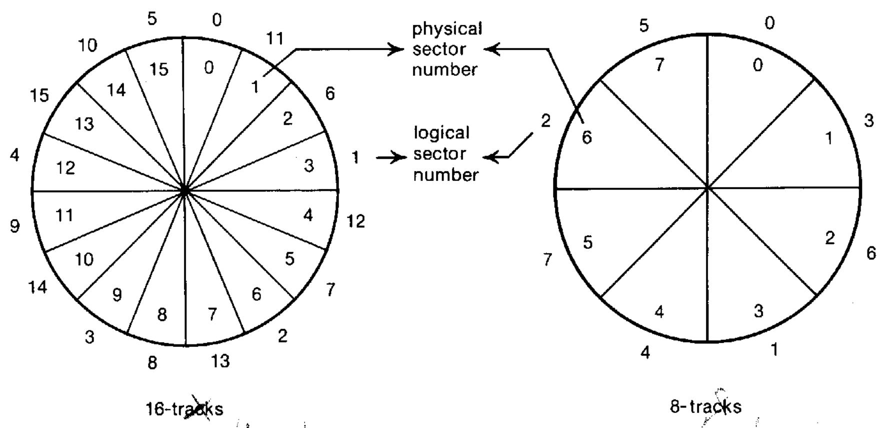
When he writes a file onto disc the user need not reserve disc space himself, nor does he need to know the expected length of the file, because disc space is allocated by the monitor. The monitor allocates as many granules as necessary to the file which is being written. These granules are chained and attached to the file code assigned to that file. Allocation is done one granule at a time, so a file always occupies an integer number of granules and one granule can not be shared by several files. (See also Data Management, in Part 2.) On each disc the system maintains an allocation table which enables the system to know which granules are still available for allocation.

At the start of a session, allocation starts from the first granule available and new granules, if any, are added in ascending order (higher sector addresses). No backward search is done to take into account any granules which may have been deallocated again, e.g. after deleting a file. On the other hand, a specific command provides the possibility to start allocation at the first available granule (SCR: Scratch control command). The allocation table stored on the disc is updated only when a file is made permanent (KPF control command) and when a file is deleted (DEL control command).

It will be clear that the granules allocated to a file will not necessarily be consecutive. Therefore an allocation table is also maintained for each file and attached to it. This table contains the addresses of the granules allocated to the file. It is 200 words long, thus limiting the file length to a maximum of 320k words (200 x 8 sectors). See File Structure.

Allocation is done for temporary files only when they are written. So, a file which has been made permanent (KPF control command) cannot be extended directly. Updating a sequential file is done in the normal manner by copying it through the Update processor. Updating a permanent file in random access can only be done directly if no extra granule is required. (See Data Management.)

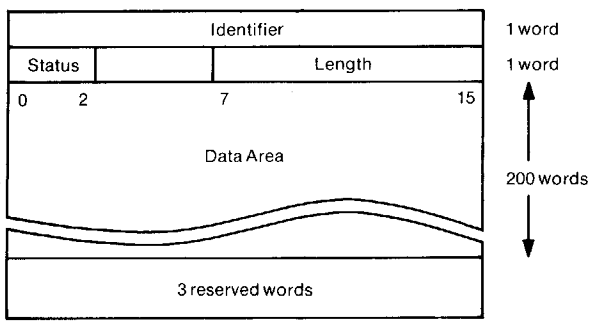
The sectors on a disc track are arranged in a certain logical order to provide for the most efficient sector handling. For this purpose, the sectors are not handled according to their physical numbering sequence, but according to the logical numbers assigned to the sectors. This is shown in the following illustration:



The disc sectors are always, except in one case, handled according to their logical numbers, e.g. all Data Management operations take place on this basis and when a disc dump is made, the sectors are dumped in their **logical** order. Only with disc error messages, the sector number indicated is the **physical** sector number.

Sector Format

The layout of a sector on disc is as follows:



where:

- *identifier* is a sector identifier written at disc initialization time by a utility program called Premark.
- the second word is filled by the system only for files written in sequential mode. In other cases this word remains unused.

Status:	bit 0 = 1:	this sector has been deleted (see File Structure: Object Files).
	bit 1 = 1:	an EOS (End-Of-Segment) record has been written in this sector. This record is the last one in the sector, for the first record following an EOS always starts at a new sector.

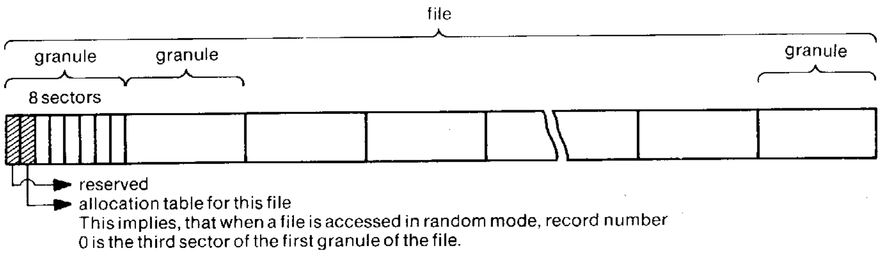
bit 2 = 1: An EOF (End-Of-File) record has been written in this sector. An EOF record requires a full sector without any other records in it. So when an EOF is written for a file, first the current sector buffer is written, if necessary, and then an additional sector for the EOF record.

Length (bits 7-15): length of the data part in this sector in characters (up to 400).

- *data area:* contains data written in either sequential or random access mode (see Data Management).

File Structure

A file is always stored on an integer number of granules (not necessarily adjacent), which means that two or more files can not share the same granule. The structure of a file is as follows:


file
granule
granule
granule
8 sectors
reserved
allocation table for this file
This implies, that when a file is accessed in random mode, record number 0 is the third sector of the first granule of the file.

There are four types of files:

- *Source files*
  Source files are sequential files input in source language or after an update of the source language. These files are used as input to one of the language processors.

- *Object Files*
  An object file is a sequential file with one record per object cluster. Each object module is followed by an EOS record. A new object module starts at a new sector. The final object module in an object file is followed by both an EOS and an EOF record.
  As an object module is not a file, it need not necessarily be stored on an integer number of granules. Therefore deletion of an object module need not result in deallocation of a granule. However, in such cases a flag is set in the second word of every sector of the deleted module (See Sector Format: Status). When an object file is read sequentially, the Data Management routines will automatically skip any deleted sectors.

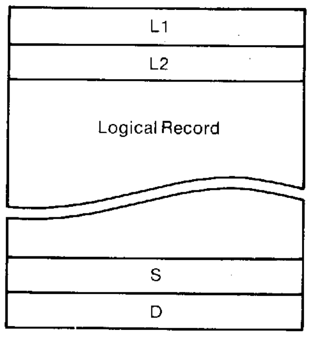
- *Load Files*
  A load file is accessed in random mode. Each full sector of such a file contains 188 code words and 12 control words for relocation bits (see Linkage Editor).
  The first four words of a load file contain the following information:

  - start address of the load module
  - number of sectors in the load module
    memory length required
  - address of entry points table (for debugging functions).

- *Undefined Files*
  This type comprises all other files, such as data files.

These files may be organized and accessed in one of two ways:
- A file is **sequential** when the only relation between the different records is their sequence. When such a file is created, the records must be presented in the same order in which they must be written onto the disc. To access the file, it must be scanned sequentially until the desired record is found.
  The logical sequence in a sequential file is identical to the physical sequence of the records in the file.
- When the **direct** access method is chosen, the records within a file may be organized in any manner and accessing a record may be done at random, by specifying a relative sector number from 0 to 1597. This is possible because with this method a logical record is equivalent to a physical record on the disc: one sector. When a direct access file is created, the records may be delivered in any order. The system will create a granule table and allocate granules to the records (sectors) that are delivered. Each granule address is noted in the table, and when the user wants to read a particular record, he specifies the relative sector number, upon which the system will be able to find it with the aid of this granule table. Thus, such a direct access file may be considered a keyed file, where the relative number of the record (=sector) is the key.

Record Structure
The logical record structure in a **sequential file** is as follows:



where:

- L1:record length in characters (S and D included). This is the actual length on the disc, i.e. any trailing blanks have been removed by Data Management when the record was written.
- L2: initial record length in characters. This is the length requested in the I/O request when the record was written.
- S: relative sector address of first word of record (for backspace handling).
- D: displacement in characters in the sector S of the beginning of the record.
- Logical record: data part, up to 1600 words (one granule) long, trailing blanks removed.

A sequential file is always closed with an EOF record.

In a **random file** a logical record equals a physical record: one sector on disc. The first word of a sector contains a cylinder identification, the rest is available for data, unless the format of the records is to be compatible with that of records in a sequential file, in which case it must comply with the record structure illustrated above.

For further details see Data Management in Part 2 of this book.

CATALOGUE AND LIBRARY STRUCTURE

On a disc, the Catalogue contains one entry for each user identification declared on that disc. Each of these entries contains a pointer to a library, one for each user in the Catalogue. Each user library consists of a directory and a library body.

Catalogue Structure

The first granule on a disc contains a catalogue of the users of this disc. Its layout is as follows:
- Sector 0: Volume label and disc allocation table (see Premark: Appendix D).
- Sector 1: IPL (Initial Program Loader).
- Sectors 2 to 7: Catalogue.

The Catalogue consists of entries occupying 8 words each; each entry relates to a user who has been declared for this disc (Declare User control command: DCU). An entry has the following format:

USER IDENTIFICATION				RESERVED		POINTER	RESERVED
0	1	2	3	4	5	6	7

- words 0 to 3 contain the user identification, as declared in the DCU command
- words 4 and 5 are not used
- word 6 contains a pointer to the user directory; it is the disc sector address of the granule containing the user library directory.
- word 7 is reserved.

If the user identification is SYSTEM, the value of the pointer in word 6 is 8, because the system directory always occupies the second granule on the disc.
The Catalogue may contain up to 150 entries (6 sectors, 25 entries each).
When an entry has been deleted, the first word is filled with /0000.
The last entry in the catalogue is followed by a word containing /FFFF.

Directory Structure
Each user is provided with his own directory and library. The directory occupies one granule and contains the names of and pointers to the user's files. The granule containing the user directory may be located anywhere on the disc, except when the user is SYSTEM. In this case it is the second granule on the disc.
Each entry in a directory consists of 8 words:

FILE NAME	TYPE	RESERVED	POINTER	RESERVED
0 1 2	3	4 5	6	7

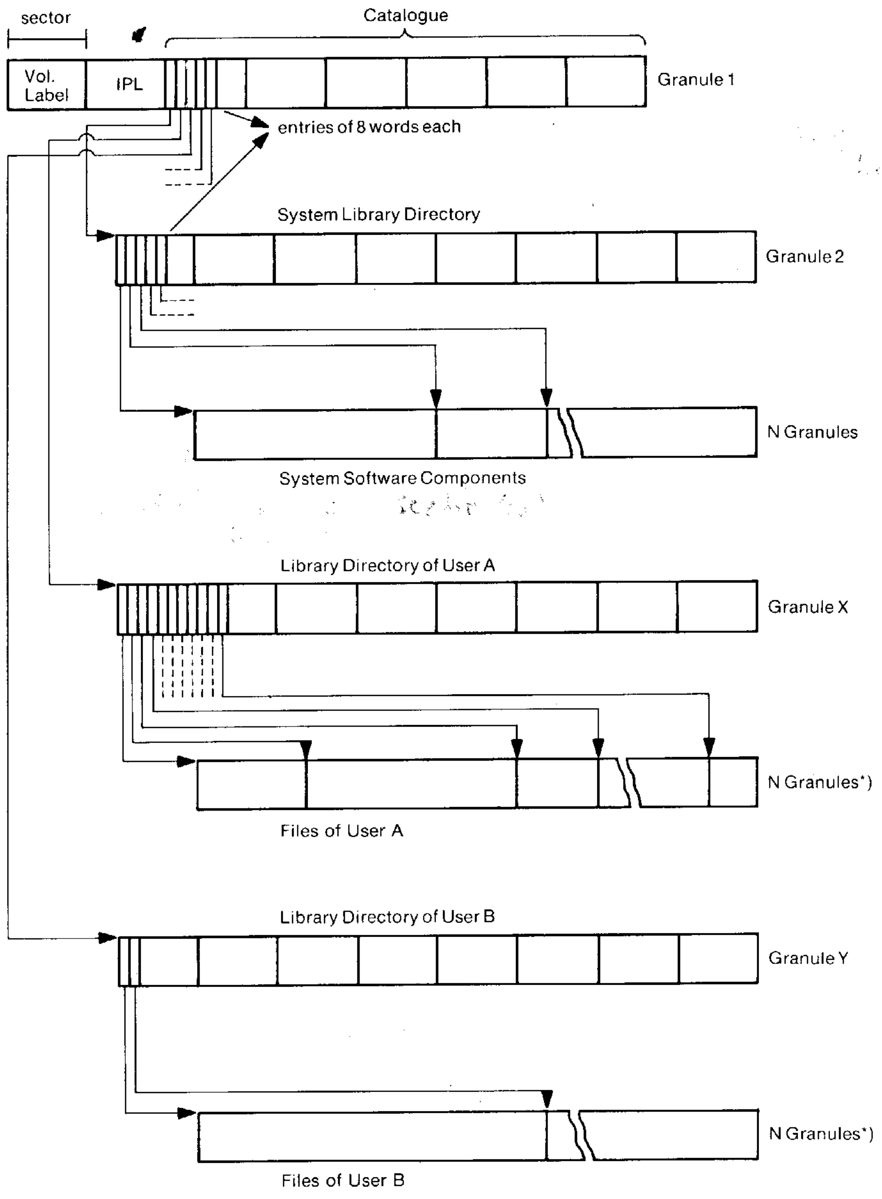
- words 0, 1 and 2 contain the file name
- word 3 contains the file type, which may be one of the following:
  . for source files: SC
  . for object files: OB
  . for load files: LM
  . for undefined files: UF
- word 6 is the file pointer; it contains the disc sector address of the first granule of the file
- words 4, 5 and 7 are reserved.

A user directory may contain up to 200 entries (8 sectors of 25 entries each).
Each entry points to a granule table (GRANTB), containing the successive addresses of the granules allocated to the file represented by the entry in the directory. GRANTB may contain up to 200 granule addresses.
When an entry has been deleted, the first word is filled with the value /0000.
The last entry in a directory is followed by a word containing the value /FFFF.

The granule for the user directory is allocated at the time when a new user is declared to the system (DCU command) and entered in the Catalogue.

On the following page a drawing is shown of the DOM library structure.

[illegible]



*) These granules are not necessarily adjacent.

Example of the Structure of Catalogue and Libraries

SYSTEM DISC STRUCTURE

The System Disc contains all the system software, i.e. the monitor, the system processors and the system object library. These software components are on the disc in the same manner as the files of the different users are, i.e. in a user library coupled to a user indentification and a directory. For the system software the user identification SYSTEM is used. It is necessary that the first file in the library of the 'user' SYSTEM is the monitor and that all the granules it occupies are adjacent. The format of the monitor is the same as that of load modules and it may be catalogued under name and type.
The other SYSTEM software components must be present in the directory of this first user identified by SYSTEM, but the granules they occupy need not necessarily be adjacent. Thus the first granules of the System Disc are occupied as follows:

Granule 0:	Sector 0: Volume Label
	Sector 1: IPL=Initial Program Loader
	Sectors 2 to 7: Volume Catalogue
Granule 1:	Directory of the first user, i.e. SYSTEM

Granules 2 to n consecutively contain the Monitor.
The other software components use granules randomly.
Any granules remaining on the System Disc can be used by other users for their files.

DISK PREMARK

PREMRK is a stand-alone program to be used for formatting a disc pack before it will actually be used. It divides the disc into sectors and writes identifiers in them and it checks for bad tracks.

PREMRK is in absolute format, so it is loaded by bootstrap, the disc unit being in the READY state. After it has been loaded it starts to type out the following questions on the operator's typewriter and the user can type in his answers:

NBR OF CYLINDERS =
Type in 4 decimal characters, specifying the number of cylinders on the disc, followed by LF CR

NBR OF TRACKS =
Type in 4 decimal characters, giving the number of tracks per cylinder, followed by LF CR

NBR OF SECTORS/TRACK =
Type in 4 decimal characters, specifying the number of sectors per track, followed by LF CR

DISK TYPE =
Type in 2 characters, specifying the type of disc unit, and the type of channel used, as follows:

1st character:	X:	moving-head disc (X1210)
	C:	X1215
	F:	fixed-head disc
2nd character:	M:	Multiplex channel
	D:	DMA

then type in LF CR.

DISK UNIT PHYSICAL ADDRESS =
Type in two hexadecimal characters, followed by LF CR

LABEL =
Type in 8 characters, giving the volume label, followed by LF CR

DATE =
Type in 6 decimal characters, specifying the date, followed by LF CR

PACK NBR =
Type in 3 characters to specify the pack number, followed by LF CR

SYSTEM USERID =
Type in the identification of the system generated, followed by LF CR.

RUN AGAIN?:
Type in NO if no other discs are to be formatted, or OK if another disc must be done. In the second case, the above procedure is repeated.

Example:

```
INITIALISATION OF PREMRK
NBR. OF CYLINDERS = 0203
NBR. OF TRACKS = 0002
NBR. OF SECTORS/TRACK = 0016
DISK TYPE: XM
DISK UNIT PHYSICAL ADDRESS = 02
LABEL = EXAMPLE
DATE = 12/02/74
PACK NBR. = 001
SYSTEM USERID = SAG
- WRITING THE IDENTIFIERS
- CHECKING THE IDENTIFIERS
- END OF CHECK
- NBR. OF BAD GRANULES = 0000

RUN AGAIN?: NO
END OF PREMRK
```

The user replies are given in boldface.

Error Messages:
BAD GRANULE ZERO (granule zero of the pack is bad and therefore this pack is not usable)
NO SYSTEM USER POSSIBLE (granule one of the pack is bad and therefore no system catalogue can be opened).

CASSETTE TAPE PREMARK (LRC cassettes only)

PREMRK is a stand-alone program to be used for formatting a cassette tape before it will actually be used.
It writes a label at the start of the tape, according to the user's specifications, followed by a tape mark.
PREMRK is in absolute format, so it is loaded by bootstrap, the cassette tape unit being in the Ready state.
Cassettes introduced under BOM or DOM will have to be premarked, otherwise system hang-up will follow.

After PREMRK has been loaded, it starts to type out the following questions and messages, to which the user must type in his answers:
- SERIAL NUMBER?
  Type in a number of up to 6 characters.
- SECURITY CODE?

Type in a one-digit hexadecimal character.
- OWNER NAME?
Type in a character string of up to 39 characters to identify the owner.
- CASSETTE TAPE ADDRESS?
Type in the physical device address.

Now PREMRK starts writing and, after it is finished, types out:
STATUS=XXXX (see appendix C)
RUN AGAIN?
to which the user can answer with YES or NO, depending on whether he wamts another cassette premarked or not (Note: For each side of the cassette, a separate PREMRK run is required.). At the end, it types: END OF PREMRK.

Appendix C Control Unit Status Word Configuration

Bit	Description	Control unit ASR	CR	DF	DM	LP	PTP	PTR	CASS Tape	PLOT	MT
0	–										
1	has become ready			x	x				x		x
2	rewinding										x
3	tape mark has been read								x		x
4	no data (CRC)								x		
5	on cylinder			x	x						
	beginning of tape (CRC)								x		
6	seek error				x						
	write unable								x		x
7	A or B side								x		
8	Device Address			x					x		x
9	Device Address			x	x				x		x
10	EOT							x	x		x
	tape low						x				
11	Program error			x	x				x	x	x
12	incorrect length		x	x	x				x		x
	Y limit overpass									x	
13	Parity error								x		x
	data fault		x	x	x						
14	throughput error	x	x	x	x			x	x		x
15	not operable (only significant bit for TST)	x	x	x	x	x	x	x	x	x	x

DM = moving-head disc
DF = fixed-head disc