File - Multics design proposals

TO: Distribution

FROM: Janice Phillipps

DATE: June 20, 1973

SUBJECT: Proposed Multics ANSI Tape DIM

Revision II

The attached pages consist of a design document for a Multics ANSI tape facility prefaced by three lists: a summary of features to be included in the initial version of the DIM, a summary of features to be included in a later version of the DIM, and a list of features which will be omitted from the DIM.

The objective of the ANSI tape effort essentially is to install a DIM that can read and write ANSI standard tapes containing ANSI standard labels.

The initial support planned for the ANSI tape DIM will include items from the list below.

Initial restrictions:

- 9-track tape only.
- For an input tape, all required ANSI volume and file labels plus the ANSI HDR2 file label must be present for each file on the tape.
- All labels and data on the tape must be recorded in the same character code. In later versions of the DIM this restriction will be lifted.
- Tapes will be read and written only in D-format (variable length) records in the initial version of the DIM. Blocking and unblocking of variable length unspanned records will be inplemented initially. This restriction on spanned records will be lifted in later versions of the DIM.
- Single volume files and multi-file volumes will be all that the initial version of the DIM will handle. Multi-volume files and automatic volume switching will come in later versions.

Features of the initial DIM:

- Automatic volume mounting and verification.
- Automatic file verification and positioning.
- Character code conversion within 8-bit types.
- Automatic label creation and label processing.
- File creation and processing.

The later versions of the ANSI tape DIM will include items from the following list.

- Full support for 9-track EBCDIC (IBM labeled) tapes. The DIM will be able to read and write 9-track tapes created by OS/360 or OS/370. This will include handling V-format spanned records (see Figure 4).

- Support for DOS/360 (IBM labeled) tapes. The DIM will be able to read and write tapes created by DOS/360.

- Support for binary mode tapes. The DIM will be able to handle binary mode, non-ASCII tapes.

- Information supplied by the user pertaining to file unblocking will be accepted in the attach call (third argument) and passed to the DIM giving the user an alternative to using the information supplied in the HDR2 file-header label.

- A routine will be created to parse the third argument to ios_$attach, selecting the tape qualifiers.

- Undefined length records -- U-format, will be an acceptable record format.

- Fixed length records -- F-format, blocked and unblocked will be supported by the DIM as well as the variabel length (D-format) records and variable length blocks.

- Spanned records recorded in ASCII -- S-format, will be an acceptable record format.

- Support for the detachment disposal options.

- Modification of the interface buffering to handle record I/O.

- An additional order call will be created for dispatching user file labels.

- A feature to erase tapes will be added to the DIM.

- Possible support for 7-track ANSI tapes will be added, depending on the hardware available.

- Multi-volume files will be supported.

- Automatic volume switching for multi-volume files will be supported.

There are no plans for the ANSI tape DIM to support any of the following list.

- BCD tapes. The user who wants to work with BCD tape will have to use the Multics non-standard tape DIM, nstd_.

- User volume labels. The ANSI tape DIM will pass over user volume labels without reading or writing them.

- Certain ANSI optional system header and trailer labels. The ANSI taoe DIM will pass over, without reading or writing, the system header labels HDR3 through HDR9 and system trailer labels EOV3 through EOV9 and EOF3 through EOF9.

- Access control (accessibility) as a feature on labels. Tape access will be regulated by the access control list of a tape's associated BVDS segment (formerly known as the TRDS) only.

- Accessibility restriction at the file level. Again, due to the selected inplementation of access control, accessibility will be restricted only at the volume level, not at the file level.

- Label format consisiting of upper case only. Labels will consist of full ASCII without a conversion to upper case only.

- Mixed character codes within files or labels. The ANSI DIM will not recognize more than one character code within any given file or label.

TABLE OF CONTENTS

TO: Robert Daley

FROM: Janice Phillipps

DATE: January 19, 1973

SUBJECT: Design for a Multics ANSI Tape DIM

There has been a growing need within our Multics user community for a standardized magnetic tape handling package which would generally function to facilitate information interchange both intermurally and intramurally. This memorandum is a design proposal for just such a package: a Multics I/O system outer module which will create and process magnetic tapes in accordance with the American National Standards Institute specifications for information interchange.

Several terms defined in the ANSI standard and used in this memo will be described first for the sake of clarity. The term "volume" is used to be synonymous with a reel of magnetic tape. A "record" means a collection of related data items which an operating system treats as a unit. The term "file" is used to refer to a collection of logically related records which are treated as a unit. A "block" refers to a group of contiguous characters read from or written onto tape as a unit. Each block may contain one or more record. A "label" refers to a block at the beginning of a file or volume which serves to identify or delimit or both identify and delimit a file or volume.

ANSI specified tapes are recorded as one or more ASCII files surrounded by interchange standard labels. ANSI tapes can be likened to IBM standard labeled tapes in three main features: 1) each file on both IBM and ANSI tapes is preceded and followed by at least one operating system label, 2) the files and labels on both IBM and ANSI tapes are separated by a tapemark, and 3) the end-of-information position on an ANSI volume and an IBM volume is indicated by two consecutive tapemarks.

The ANSI interchange standard uses labels to identify and to structure files and it specifies a structure for blocks containing the records making up a file.

The initial objective for this proposed ANSI tape DIM is to establish and implement a small but valid set of ANSI information interchange tape specifications: to establish a DIM that would be able to read and write ANSI standard tapes containing ANSI standard labels. At least initially, this DIM will support only those tapes which strictly comply to the ANSI specifications for interchange — namely, 9-track tapes containing all required ANSI operating system labels plus the HDR2 label, and ASCII files with density of 800 characters per inch recorded in odd parity, non-return-to-zero (NRZI) mode.

The initial version of the ANSI tape DIM will be restricted to handling single-volume files and multi-file volumes. Only variable length unspanned records in D-format will be implemented. By the final version of the DIM, volume switching will accommodate single-file-multi-volume and multi-file-multi-volume tapes with any of the following record formats: fixed length records--blocked or unblocked (F-format or FB-format), variable length records--spanned or unspanned, blocked or unblocked (VBS-format, VB-format and V-format in EBCDIC, D-format and DB-format in ASCII), ASCII spanned records (S-format), and undefined length records (U-format). The spanning feature allows the use of variable length logical records that are larger than the physical block size, thus packing variable length records into a fixed block size.

As later versions of the DIM are developed, a set of features slightly larger than the ANSI required set of specifications for information interchange will be made available to Multics users. This extended set of features will include support for 7-track ANSI labeled ASCII tapes and 9-track IBM standard labeled (EBCDIC) tapes.

Implementing the feature of 7-track ASCII tape processing in the DIM will be largely a matter of accomodating an additonal frame size and an additional buffering scheme for reading and writing. As IBM standard file labels and ANSI standard file are only slightly different, the implementation of processing IBM standard labeled tapes will be essentially a matter of the character conversion between EBCDIC and 9-bit ASCII. It is not planned to have the ANSI DIM support IBM BCD tapes. Multics users wishing to process BCD tapes will be referred to using nstd_ --the Multics non-standard tape DIM.

II. American National Standard Labels

This section will consider American National Standard labels as defined by the American National Standards Institute (ANSI) in the _American National Standard Magnetic Tape Labels for Information Interchange_, ANS X3.27-1969. The organization, format and content of these ANSI labels will be discussed in connection with the presentation of the proposed Multics tape DIM.

Types of ANSI Labels

An ANSI tape label can be identified by its first four characters. The first three characters, the label identifiers, establish the label type and the fourth character, the label number, indicates the relative position of the label within its label type group.

There are four types of ANSI tape labels for information interchange -- beginning-of-volume labels, beginning-of-file labels, end-of-file labels, and end-of-volume labels. Each of these four label types can be created in one of two categories: as a user label or as an operating system label. These four label types in both label categories are to be discussed in detail below as the ANSI required, ANSI prohibited, and ANSI optional aspects of interchange labeling are considered.

According to ANSI specifications, an operating system volume header label must appear as the first record of an ASCII interchange tape volume and may not appear anywhere else on the tape; each file must be surrounded by at least one file-header label and at least one file-trailer label also created by the operating system. For multi-volume files, an end-of-volume label created by the operating system is required to be at the end of each reel to be continued, to signal volume switching. All user created file labels (containing user-specified information on a file) are unrestricted in number and are optional in the ANSI specifications.

Single File, Single Volume

VOL1 HDR1 HDR2 *--File A--* EOF1 EOF2 **

Single File, Multi-Volume

VOL1 HDR1 HDR2 *--First part of File A--* EOV1 EOV2 **

VOL1 HDR1 HDR2 *--Last part of File A-- *EOF1 EOV2 **

Multi-File, Single Volume

VOL1 HDR1 HDR2 *--File A--* EOF1 EOF2 * HDR1 HDR2 *--File B --* EOF1 EOF2 **

Multi-File, Multi-Volume

VOL1 HDR1 HDR2 *--File A--* EOF1 EOF2 * HDR1 HDR2 * --First part of File B--* EOV1 EOV2 **

VOL1 HDR1 HDR2 *--Continuation of File B--* EOV1 EOV2 **

VOL1 HDR1 HDR2 *--Last part of File B--* EOF1 EOF2 * HDR1 HDR2 *--File C--* EOF1 EOF2 **

Figure 1: Four Cases of ANSI Tape Volume Organization

Volume Organization With ANSI Labels

One of the features of the ANSI labeling scheme is that it provides a basis for tape volume organization. As indicated in Figure 1, there are four basic volume layouts possible with ANSI standard tapes. Each of the four cases will be considered in terms of volume format, using Figure 1 as the basis of discussion.

Single File, Single Volume: The system volume-header label (VOL1) is followed by the system first file-header label and system second file-header label (HDR1 and HDR2). The tape data, File A, is preceded and followed by a tape mark (the asterisk represents a tape mark). The two system file-trailer labels are identified as EOF1 and EOF2. Two tape marks follow the EOF trailer label group to indicate that File A is the last data on the volume and it is not continued onto another volume. Tapes which have ANSI optional user volume-header labels (UVL1-UVL9: contain user specified information about the volume) following the system volume-header label (VOL1), will be acceptable on input to the Multics ANSI DIM, but these user labels will not be created on output for there is no way for the Multics user to communicate directly with the DIM in order to create these volume lables.

Tapes with system file-header labels HDR3-HDR9, with system file-trailer labels EOF3-EOF9 or EOV3-EOV9 will be accepted on input to the DIM but these labels will not be written on output. In the course of label processing, any of these optional file labels will be copied to a place where the user can access them with an order call. It is felt that these optional user volume labels and these optional system file-header and trailer labels are neither vital nor popular enough in information interchange to warrant implementation in the Multics DIM, as even the IBM ANSI implementation does not support them.

Also acceptable for input to the Multics DIM are tapes on which the system second file-header label and system second file-trailer label have been omitted, provided that the user has supplied enough information in the attach calls to enable unblocking of the files on the tapes. On output, the DIM will write only the label patterns shown in Figure 1.

Single File, Multi-Volume: The last volume of an ANSI volume set is organized the same as a single ANSI volume is organized. For other volumes in the set, however, system file-trailer labels are EOV1 and EOV2 (indicating volume switching ahead) instead of EOF1 and EOF2 as they are in the single file, single volume case; the file-trailer label group is followed by two tape marks indicating that there is no more information on the current volume. The system file-header and file-trailer labels are EOV1 and EOV2 (indicating volume switching ahead) instead of EOF1 and EOF2 as in the single file, single volume case; the file-trailer label group is followed by two tape marks indicating that there is no more information on the current volume. The system file-header and file-trailer labels (and user file labels, if present) are repeated on each volume, each volume having its own volume-header label.

Multi-File, Single Volume: A system volume-header label (VOL1) marks the beginning of tape information. Each file of the volume is preceded by a system header label group (HDR1 and HDR2) and a tape mark, and is followed by a tape mark and a system trailer label group (EOF1 and EOF2). Each system trailer label group is followed by a tape mark; the trailer group of the last file on the volume is followed by two tape marks.

Multi-File, Multi-Volume: This case arises when more than one volume is used to contain multiple file data. The last volume is organized the same as the multi-file, single-volume case. On the other volumes in the volume set, the system file-trailer labels are identified as EOV1 and EOV2 instead of EOF1 and EOF2; the last file's file-trailer label group is followed by two tape marks. Each tape volume has a separate system volume-header label.

ANSI Label Processing

The ANSI tape label standard does not make any specific requirements for the processing of tape labels, however, at least two assumptions regarding the implementation of label processing underlie the ANSI standard. One of these assumptions is that the user is given the option for his convenience, of using or not using user file labels. Therefore, rather than including user file label processing in with the implementation of operating system file label processing, the Multics ANSI DIM will only be instrumental in dispatching user labels--leaving creating and processing user file labels up to the user himself. In outline, to read or to write a user file label on a given tape, the user must set a switch in the attach call to alert the DIM to look out for user file label dispatching, and then the user must issue an order call to pass the I/O switch a pointer to the user file label on output and issue an order call to receive a pointer to the user file label from the I/O switch on input. The user label will be translated into 9-bit internal code (ASCII) when the I/O switch passes-receives the pointer to the user file label.

The second of the above mentioned label processing implementation assumptions made by the ANSI standard, is that the operating system is permitted for its convenience, to ignore on input the contents of any field which appears in any label. In the DIM's label processing implementation, any information contained in tape labels which is not immediately required by the operating system will be ignored. For example, this occurs in the case of volume and file access control. The accessibility field in both the VOL1 label and the HDR1 labels which the standard provides to indicate any restrictions on who may have access to information of the tape, will be ignored by the Multics DIM in favor of using the more comprehensive Multics access control list on the tape's associated BVDS (formerly the TRDS). Another example of omitting fields of a label on input as a standard implementation allowance for the DIM is the case of multi-volume file processing. The DIM will only process file sequence number and file identifier fields in the HDR1 label on the first volume of a multi-volume file (mounted for read), on the assumption that the file which continues on to the next volume of the volume set is the same file that was started on the preceding volume of the volume set.

The label processing routines of the Multics ANSI DIM will expect labels to be a fixed length of 80 characters. Systems involved in interchange with the Multics system that are forced by hardware to read and write larger than 80 character records must keep in mind that the Multics facility will truncate labels in excess of 80 characters, only processing the first 80 characters.

The main functions performed by the ANSI DIM label processing routines are six. 1) Volume labels on tapes mounted for read are checked to verify that the correct volume has been mounted and that the user has permission to access the information on the volume. 2) Labels on input tapes are identified as being recorded in one of four acceptable character codes and then are translated and stored as 9-bit internal code (ASCII); labels to be created on output tapes are translated from 9-bit internal code into the user specified character code. These translations are to be done by a table lookup method. 3) File labels are checked for correct file verification. 4) Prior to volume switching -- in the case of a multi-volume file -- the existing volume label and the existing file header and file trailer labels are checked on output tapes to verify that the requested volume of the volume set is indeed the volume that is mounted and that the requested file is the file that is attached. This is done to prevent the destruction of existing data by erroneous overwriting. 5) New volume and new file labels are created and written for output tapes. 6) Information needed to do file unblocking will be sought in the active Stream Data Block (SDB) should HDR2 labels be missing on a tape mounted for read.

Processing the volume-header label

The ANSI DIM will call the hardcore tape DCM for channel and device assignment, using the new ring 1 attachment code for the tape mounting, mounting check and later, for the tape dismounting functions (see MSB-69, "User Tapes").

Volume label processing will be done in ring 1. The "VOL1" label will be created automatically when an ANSI tape is first mounted for use. On subsequent use of the tape, a call to the tape mounting procedure and a call to a character conversion routine, will check the first four characters of the first record of an already recorded

ANSI tape for the string "VOL1" in one of the four implementation recognized character codes. This VOL1 label identifies the tape giving its volume serial number. The serial number on the volume label will be checked against the number specified in the mount request (reel id) to verify that the requested volume is cleared satisfactorily, a switch is set in the Tape Communications Segment data base (tcs) to indicate that volume mounting has been verified.

Next, volume access is checked by scanning the access control list of the tape's BVDS. (As mentioned below, the VOL1 label's accessibility field is ignored). On input tapes, the tape owner field in the volume label will not be checked although the first fourteen characters of the user's name (personid) will be filled in this field on output. The last field on the VOL1 label indicates the level of ANSI label standardization on the volume. A "1" in this field signifies that all labels and data on the tape conform to the requirements of the standard; an ASCII space character in this label standard level field, signifies a deviation from the fully supported ANSI interchange standard and that the format and contents of the labels and data on the tape require the agreement of the interchange parties.

For the initial version of the ANSI DIM, the label standard version field will be used in the following way. If the version field contains a "3", indicating fully standard level at version 3, the DIM will expect labels and data in 8-bit ASCII (may be IBM ASCII) and will expect all the ANSI required labels to be present. If the level field contains an ASCII space, the required ANSI labels will be expected but any of the four character codes acceptable for labels and data will be processed.

File Labels

The creation and processing of operating system file labels will be done in ring 4. When an ANSI file is to be written onto tape, the DIM will automatically create and write the HDR1 and HDR2 file-header labels and write either the EOF1 and EOF2 or the EOV1 and EOV2 file-trailer labels by using the information in the SDB and issuing calls to compose the labels. No effort will be made initially to convert labels to upper case only; rather, full ASCII or EBCDIC character codes will be used until later versions of the DIM.

Except for a few aspects of file label processing which are closely linked to the process of file attachment and which will be covered later on in this memo, this section will continue the discussion of ANSI file labels and will outline the DIM's implementation of file label processing. In reading this section, it will be helpful to refer to Appendix A of this memo -- the declarations of the file labels that can be read and produced by the Multics ANSI tape DIM.

The HDR1 label contains operating system and device-dependent data relating to a file and the HDR2 label contains additional characteristics of the file. While processing an input tape, the DIM will expect file labels to be recorded in the same character code as the volume label was recorded in; an abnormal termination results if discrepant character codes are found during label processing. For output tapes, the character codes of labels on a given tape will be consistant. Generation number and generation version number of a recorded file are ANSI optional HDR1 information and will be skipped by the label processing in version one of the Multics DIM. In later versions of the DIM, if the user supplies a generation number and generation version number, they will be included in the HDR1 label and included as well in the file identifier as "xxxxxxxxxGnnnnVnn". Expiration date will be read by the DIM on input but will not be enforced. On output, the expiration date will be "ƀ00000" -- signifying that the file may be overwritten at any future time. File accessibility will be inherted from the volume's accessibility and therefore will not be checked at the file level (ring 4).

The HDR1 system_code will be checked on input tapes and will be filled in on output tapes; only if the code is "multics_astd_" will the DIM process the HDR2 label for file character code information because this label information is kept in the section reserved for the use of the system which recorded the file and it is structured uniquely.

The information contained in the file-trailer labels is essectially a duplicate of that contained in the file-header labels. This is an ANSI interchange design to facilitate reading tapes backwards. The only label fields that contain different information are the label identifier field HDR vs. EOV or EOF, and the block count. The block count is maintained in the EOV1 or EOF1 label and is verified at either an end-of-data or an end-of-volume condition. The HDR1 label merely initializes the block count to zero.

The HDR2 label and the EOV2 or EOF2 label contain the logical record format, the logical record length and the block size of an interchange file. The record format of an interchange file is indicated either by the character F, for fixed length records, by the character D, for variable length records, by the character U, for undefined length records, or by the character S, for ASCII spanned records. All but D-format will be later options provided by the Multics DIM.

ANSI D-format records cannot be spanned from one block to the next. ANSI S-format records are used when ASCII spanned records are desired. The length of a logical record is expressed as a number of characters but the exact interpretation of this number depends on the associated record format. In the case of F-format, the logical record length informs the DIM of the number of characters in a logical record; however, in a D-format record, the logical record length includes the length of the record control word -- which defines the record length. Moreover, in the D-format case, the record length describes the maximum logical record size to be expected by the DIM.

The block size of an ASCII file can be either fixed length, variable length, or undefined length -- F, D, or U. Just as with the logical record length, the block size for D-format records indicates a maximum block length.

The EOV2 label will contain a special item in the operating system reserved section; this will be the identity of the next volume in a multi-volume set. Preceding a volume switch in multi-volume processing, the volume id of the next volume in the set can be extracted from the EOV2 label and checked against the volume id in the attach call. For single volume files this field will contain ASCII spaces.

File Label Processing

When a volume has been mounted and verified or when it has been rewound, file label processing begins with initializing a current file position counter in the SDB. Only at these times can it be certain that the label processing will start at the first file on the volume. A file requested by number might be located on a mounted volume which was left positioned at a file other than the first one; in this case, the current file position counter is not initialized but is compared to the file sequence number of the current HDR1 label. This will determine the magnitude and direction in which to space the tape towards the requested file. Then each time an HDR1 label is read, the current file position counter is incremented, positively or negatively depending upon the relative location of the file on the volume. When a file has been requested by name, the current file location counter is first checked to see if the tape is positioned at the beginning of the volume. If the tape is not positioned at the first file on the volume, the file label chain, which contains information on all the files which have been processed so far on the current volume, is referenced for the name of the file requested. If an entry for the requested file is found in the volume's file label chain, the associated file sequence number is extracted from the label chain entry and processing continues as described above. If however, the requested file name is not found in the file label chain search, the tape is then spaced along to the next HDR1 label until the requested file is found. If no HDR1 label is found for the requested file, abnormal termination results.

Aside from locating and identifying label types, label processing consists of converting each label to 9-bit internal code, copying the label information as well as the label itself into the SDB, and setting a switch to indicate what type of label has just been read. The procedure is just the reverse for processing output files.

This will determine the magnitude and direction in which to space the tape towards the requested file. Then each time an HDR1 label is read, the current file position counter is incremented, positively or negatively depending upon the relative location of the file on the volume. When a file has been requested by name, the current file location counter is first checked to see if the tape is positioned at the beginning of the volume. If the tape is not positioned at the first file on the volume, the file label chain, which contains information on all the files which have been processed so far on the current volume, is referenced for the name of the file requested. If an entry for the requested file is found in the volume's file label chain, the associated file sequence number is extracted from the chain entry and processing continues as described above. If, however the requested file name is not found in the file label chain search, the tape is then spaced along to the next and the next HDR1 label until the requested file is found. If no HDR1 label is found for the requested file, abnormal termination results.

Aside from locating and identifying label types, label processing consists of converting each label to 9-bit internal code, copying the label information as well as the label itself into the SDB, and setting a switch to indicate what type of label has just been read. The procedure is just the reverse for creating output files.

After the first file-header label has been processed on an input tape, the DIM anticipates finding a tape mark. If a tape mark is not the next item on the tape, an HDR2 label is assumed. The DIM will be able to obtain file unblocking information such as a logical record format, logical record length and block size in one of two ways - from the Stream Data Block (SDB) enter via attach call or from the second file-header label (HDR2). Information which the user has specified in the attach call, will have been examined and stashed away in the SDB. The DIM will refer to this information prior to processing the HDR2 label. The information the user supply in the attach call will take precedence over information in the HDR2 label. This feature provides the user with some flexibility in processing his tapes as it allows erroneously written labels and label-specified blocking restrictions to be temporarily overridden.

Logical record size and block size are taken from the HDR2 label and stored in the SDB as integers. At this point, HDR2 label processing is completed unless the system code in the HDR1 label was "multics_astd_". If the file was created by the Multics DIM, one last item can be checked in the HDR2 label: the character code in which the file was recorded. Although the character code in which a file is recorded is independent of the character code in which labels are recorded, if no special file character code is specified by the user-- either in the attach call or in the HDR2 label, the character code of the labels will be assumed for the data.

The next item of consideration in the course of label processing is user labels. When the HDR2 label has been processed and user labels have not been switched in for processing at attach time, the tape is spaced forward past the tape mark which signifies the end of the header label group.

A file is read or written and an-end-of-file condition is raised, triggering the processing of file-trailer labels. Block count verification is the most important part of the EOF1 and EOF2 label processing. The SDB's file block counter is compared to the contents of the block count field in the EOF1 label to verify that all blocks of a given file have been read. On output, the current value of the file block counter is recorded when the EOF1 label is created.

The processing of the EOV1 and EOV2 trailer labels is closely tied in with the implementation of volume switching. This will be discussed in the next section - attachment.

III. Attachment

When a user with an ANSI tape wants to attach to the I/O system, the attachment will be made to a file on the ANSI volume, not to the volume *per se*. Information required by the I/O system about the file to be attached and the volume on which it resides, will be supplied by the user in the third argument to ios_$attach. For example,

```
call ios_$attach (stream_name, "astd_", "3:file A:
013107, NORING", mode, status);
```

The contents of this third argument to ios_$attach will be all the information necessary to open a file, and in addition, any information that is needed to do file unblocking should either the user wish to override the information in the HDR2 label or should the HDR2 label be missing from the label group. The third argument of the attach call will be parsed by the DIM for the required information to open a file. Delimited by a colon, the first item in the third argument string will be the file sequence number of the file to be attached. The second item in the argument string will be the file name.

For the initial version of the DIM, the user must specify the file name and the file number in every attach call he makes, or abnormal termination will result. The later versions of the DIM will recognize an asterisk in the file sequence number field to mean that the user hasn't specified the exact file sequence number - i.e. Asterisk matches any file number. When the asterisk is used in the file sequence number field, the file name must be specified if the attachment is to write, but the file name may be omitted in the call if the attachment is to read. This design allows the to read the last file on the tape without knowing the file name; however, ANSI specifications require that a file name be given before a file is written on a tape, so an attachment for write cannot be made to a file on an ANSI volume without the user's specifying a file name. This decision was made to avoid having the DIM generate unique file names.

Another file reference feature that later versions of the DIM will recognize will be the identifier "END" appearing in the file sequence number field of the attach call. This qualifier will refer to the last file of the requested volume. If attachment is to be made for read, the last file on the volume will be read. If the attachment is for write, a file will be appended at the end of the tape. The same restrictions for file name specification apply with this feature. The next item of information in the third argument string is the reel id (all reel ids in the case of a multi-volume file). After this point, the tape qualifiers, if any are supplied. In the initial version of the DIM, tape qualifiers will not be processed. All blocking and unblocking information will come from HDR2 labels.

The tape qualifiers — delimited by a comma, — would include any of the following: ring, noring, block size, logical record length — maximum logical record size in the case of variable record length, record format, the character code in which the file is or is to be recorded, user-label switch (indicating that tape is to have user labels), and a multi-volume switch to signal a multi-volume file. In this way, the user specified information needed to access an ANSI tape file can be passed from the user on to the DIM.

An alternate proposal and perhaps a future improvement for handling file opening information in a call to ios_$attach would depend upon a modification to the file system whereby "link-like-things" could be placed in say the user's working directory. Then the third argument to ios_$attach could be the name of the "link" and the "link" could provide the file sequence number, the file name, reel id and, if any, qualifers - probably in the above mentioned format of

"3: file A: 013107, NORING"

There are several procedure operational steps that span the point at which the call to ios_$attach is made and the point when the volume is mounted and positioned at the requested file, ready to read or write data. If the DIM's initial check for interchange specified density (800 characters per inch), or volume label processing check for the correct volume, or the mount check for the write permission ring, or the access control check on the volume's BVDS fails, the attachment will not be made; but if these checks were successful, at this point the tape will be up and tape processing returns to ring 4 to locate and position to the requested file.

File-header label processing begins with reading the HDR1 labels. The file name and file sequence number are contained in this label. Consider Figure 2 below.

This chart assumes a single volume file or a multi-file volume: volume switching is not included here. The chart also assumes that the volume in reference has not been previously mounted in the user's current process. When a user's volume is first referenced, a threaded list of file label information is build up by the DIM as the header labels are read. (See Appendix C for the declaration of the file label chain.)

Each entry in the file label chain corresponds to one processed file-header label group for the current volume. The chain entry contains the file name, the file sequence number, the file section number — tells the current volume number of a multi-volume file, a switch to indicate the last file on a multi-file volume (called last_file_switch), the translated (into 9-bit ascii) header labels — HDR1 and HDR2, and a pointer to the next entry in the file label chain.

The file label chain is set up to make subsequent file references on the mounted volume more efficient. This chain can be referenced for redundant file name check before writing a file on a volume or its information, including the list of file-header labels, could be written into the BVDS and accessed through the file system by the user at a later time.

In the initial version of the DIM, the bulk of the information to open a file will be taken from the file-header labels HDR1 and HDR2. The attachment call will have to specify a file name and a file sequence number, but all blocking and unblocking information will be taken from HDR2 until later versions of the DIM.

Label processing during attachment procedings in the initial version of the DIM will consist of searching the file label chain for the file name specified, or if a label chain does not yet exist, the processing consists of reading file-header label groups, building the label chain as the reading is done. If there is any entry for the file on the label chain, the DIM then advances the tape to the beginning of that file. If the file is found by reading the labels, the tape is advanced past the label groups and positioned at the beginning of the date. Before the attachment is completed, the information needed to open the file is gathered from the HDR2 label and filled into the Stream Data Block (SDB). Figure 2 elaborates the file reference features that will be recognized by later versions of the DIM. If an asterisk appears in the file sequence-number field of the attach call, the DIM will search the label chain for the specified file name or label groups will be read for a file name match only. If the qualifier "END" is specified in the file sequence field, the DIM will advance the tape to the end of the volume and either back up to read the last file on the volume or prepare to append a file at the end of the volume, depending on the attachment mode specified. By specifying "END" to access the last file on a volume, label chain building is avoided.

	Attach File for Read	Attach File for Write
file sequence number is an asterisk (asterisk matches any number)	action: Read each file-header label looking for the specified file name until runoff end of tape: two tapemarks mean file is not on the tape. Error is returned in status string.	action: Search for file name as in attach for read. If file name is not found, return information in status. If the file is found but it is not the last file on the tape, rewrite will cause the destruction of sequential information on the tape.
file sequence number is "END" or an integer	action: If the identifier "END" is specified, skip to last file on tape and read, if the number specified is greater than the largest file number on the tape, return error in status string. Otherwise skip to the specified file number and check the HDR1 label for file name and file sequence number.	action: If "END" is in the file number field, skip to the end of the tape and append the file. Otherwise skip to the specified file number. If file is not the last one on the tape, rewrite will destroy sequential data on the tape. If the file number is greater than one more than the last file number on the tape, return error in the status string; else append the file with the given file number.

Figure 2: Implementation of File Attachment

If a file is requested for write and the specified file is not the last file on the tape, the rewrite will destroy the remaining data on the tape. A warning stating this condition will be returned in the status string. If the file number specified in the write attach call is greater than one more than the number of the last file on the tape, the tape will be moved to the end of the reel and an error will be returned in the status string, but if the number is the next sequential file number, the file will be written at the end of the tape.

At this point in the discussion of attachment, the problem of duplicate file names on tape should be considered. Before writing out a file on tape, it is possible to rewind the tape, pace through each HDR1 label checking against the name of the file to be written, for a possible duplicate file name. This extra checking would prevent mishap due to name duplication, however it would also be an expensive feature to have in terms of overhead. It is planned to leave the burden of possible file name duplication on the user. As far as the DIM is concerned any file name which the user specifies is a unique file name on that tape. Further, the first file of the requested file name appearing on the tape will be the file attached by the DIM.

For a given file to be attached for read, the processing of file-header labels probibly could continue through up to nine labels (HDR1 - HDR9), although only the first file-header label (HDR1) is absolutely required by the ANSI standard to be on the tape. All versions of this DIM will support full information processing of only HDR1 and HDR2 file-header labels, but in later versions of the DIM, the additional file-header labels will be read and put somewhere where the user can retrieve them with an order call.

In the I/O system each stream attachment creates an SDB, however for the ANSI implementation, rather than a one per-tape-volume data base, the SDB will be a one per-tape-volume-set data base, to cover the case of multi-volume files. Contained within the SDB is information on the one and only one currently open file, and information on the one and only one currently active volume.

Attaching a Single Volume File

The volume on which the requested file resides is mounted and verified. A threaded list called the volume label chain, is established with the current volume's serial number. An SDB is created and initialized with the multi-volume switch turned off. The volume_id pointer (for the SDB) set to point to the currently mounted volume's serial number, and the pointer to the next entry on the volume chain is set to null. The stream "filename" is attached when the file is found on the volume, and the information specified in either the HDR2 label or the attach call is filled into the SDB.

Attaching a Multi-Volume File for Read

If a user wants to request a file which is multi-volume, he must specify the volume serial numbers of all of the volumes containing the file he wants. This is a DIM restriction: all volumes for a multi-volume read attachment must be mounted at the onset of file processing. The DIM will take this volume set information from the attach call, see to it that the specified volumes are mounted, attach the stream "filename" to each of the volumes in the volume set and then set up an SDB with the multi-volume flag on. (One SDB is created for the entire volume set.) The threaded list of volume serial numbers of each volume in the volume set will be created and maintained via the SDB, and a volume list pointer will be set to point to the next needed volume in the volume set (see Appendix C and Appendix D). The SDB activity flag will be turned on and the SDB volume_id pointer will be set to point to the current volume's serial number. By the time a read attachment is finished for a requested multi-volume file, all of the information needed to proceed from one volume of the file to the next is contained in the SDB, and all volumes of the requested file will have been mounted.

Volume Switching

Volume switching in the DIM is to be done automatically — such that it is transparent to the user. The DIM sets up a volume counter which is incremented when volume switching is signaled by the appearance of an EOV1 label. This volume counter is compared to the number of volumes requested in the attach call to provide an index of processing completion. When volume switching has been signaled, the volume_id pointer in the SDB is updated to point to the next volume serial number in the set; a check is made to make sure that the new volume is mounted and verified and then the volume list pointer is updated to point to the yet next needed volume in the volume set.

Writing a Multi-Volume File

If in the course of creating a file, the end of the tape is reached and a second volume is needed, a second volume automatically is put up (if it is not already mounted), and the information in the SDB is updated to record this multi-volume change condition. File-trailer labels on the first volume of the new volume set are written but although processing of this volume is currently completed, the tape will remain mounted until the file is closed. Volume and file-header labels for the new volume are written by the DIM.

If an already recorded multi-volume file is requested for a write attachment, all volumes of the set must be specified in the attach call, just as is required by the DIM for read attachment. But in the course of rewriting the file, still another volume should be needed, it will be automatically set up to handle the remainder of the opened file.

The method which the DIM uses to check the HDR1 label after volume switching will differ for read and write attachments. If the situation happens to be the reading of a new volume of a continued multi-volume file, the file sequence number, accessibility, file section number, etc. of the HDR1 label is not important -- only the file name need be checked by the DIM. It is assummed that a to-be-continued multi-volume file continues at the beginning of the next volume in the volume set. The continued file must have the same filename on both (all) volumes or processing will be abnormally terminated.

The HDR2 label is not processed by the DIM in the situation where a new volume of a volume set has just been mounted for read. The file characteristics that were established when attachment was made to the first volume must apply to all subsequent volumes of that file -- especially if label information was to be overridden by the information supplied in the attach call.

Attachments for write involving multi-volume files require more extensive HDR1 and HDR2 label processing. Existing volume and file header labels must be carefully checked before rewriting so that data will not be lost due to mistaken idenity of file or volume resulting in the erroneous overwriting of tape labels.

Finally in the discussion of implementing multi-volume file processing, the block count of a multi-volume file will not be carried from one volume to the next but rather is initialized and verified anew for each volume of the volume set.

Attaching a File from a Multi-file, Single Volume Tape

There is to be a major restriction of the ANSI DIM with respect to multi-file volumes. There cannot be more than one file on a given volume attached at any one time. If a user has a volume mounted and he tries to attach a file which happens to be on that volume, an error will be returned from ring 1 -- error_table_$redundant_mount. If this happens the DIM will look through the threaded list of SDBs to find which SDB refers to the same volume as the user has requested in his latest attach call. If the appropiate SDB is not found, or if the SDB is found but is not inactive (activity flag indicating a file on the mounted tape is currently open), an error is returned and the attachment fails.

The steps which the DIM will follow in attaching a file on a multi-file volume are the same as the steps that were described in the single file volume situation -- up to the time when the file is to be located on the volume. In the multi-file volume case, the file label chain is searched for the specified file name, or each file-header (HDR1) label is read.

The multi-file single volume feature requires special restrictions on detaching. If normal detach (not a multi-file volume) occurs, the tape is taken down and the SDB is deleted. But if a special detach is signalled for a file on a multi-file volume, the tape is not taken down. The SDB for the file will have the activity flag turned off and then that SDB will be threaded onto a list of SDBS associated with the mounted volume, in order to facilitate later file accessing by the given user.

File Attachment From a Multi-File, Multi-Volume Set

This attachment case is a combination of the multi-volume file and multi-file volume situations described above. A file is requested which resides on say three volumes, but it is not the only file which resides on those three volumes. The DIM requires that the volume serial numbers of all three volumes be specified in the attach call. Just as for the multi-volume file attachment, the DIM will see to it that all volumes of the specified set are mounted and that the file is verified as being on the volumes. If the file is verified, the DIM will examine the SDB chain for two things: an activity flag on in an existing SDB and an existing inactive SDB for the requested file. If there is a file with an active SDB, the DIM cleans it up and force closed that file. If an existing inactive SDB for the currently requested file is found, it is made active and any required updating of the SDB as indicated by the attach call, will be done by the DIM. If no inactive SDB is found on the SDB chain, a new SDB will be created and all necessary file and volume information discussed above will be filled into it. The stream "filename" will be attached to all three volumes of the set.

When the user finishes with the file under discussion, and he wishes to detach, the same special detaching restrictions as for multi-volume files apply - namely, the SDB activity flag be set to inactive, the SDB be threaded on the list of SDBs, and the volumes of the file to be detached not be taken down.

If then the user requests to attach another file on the just referenced volume set, the redundant-mount condition from ring 1 will be raised. Again the same steps for attachment initialization are taken by the DIM; the SDB chain is checked for an already open file and a possible inactive SDB for the requested file. Either by reactivation or by creation, the SDB is established and the stream "filename" is attached to the volume set on which the requested file resides.

Attachment Modes

Only read and write may be specified as modes in I/O system calls to attach and to changemode.

IV. Character Codes

ANSI files and labels may be read or written in independent character codes, however once a character code has been

Files and Labels

external		internal
9-bit ASCII	-----	9-bit ASCII
8-bit ASCII	-----	9-bit ASCII
8-bit IBM ASCII	-----	9-bit ASCII
8-bit EBCDIC	-----	9-bit ASCII

Files Only

binary	-----	9-bit ASCII

Figure 3: Character Codes for Files and Labels.

established for labels, the remainder of the labels on the volume set must also be recorded in that specified character code. Although different files on a given volume may be recorded in different character codes if so indicated by the user, mixed modes within files or within labels on a volume-set will not be supported. If no special character code has been indicated for a given file, the DIM will expect the file to be recorded in the same character code as the labels on the volume.

The initial version of the Multics ANSI tape DIM will read and write files of 9-bit ASCII, 8-bit ASCII, 8-bit IBM ASCII, or 8-bit EBCDIC character codes. In addition, a later version of the DIM will read and write files in binary mode if it is so indicated by the user. Figure 3 itemizes these character conversions made available by the ANSI tape DIM.

When files are unblocked on input, they are converted to 9-bit internal code by table lookup from 8-bit or 9-bit ASCII or 8-bit elocdic and then right adjusted in 9-bit fields. On output the conversion procedure is reversed.

V. Reading and Writing

Reading and writing of ANSI tapes on Multics will be done in stream mode with calls to ios_$read and ios_$write where the number of elements which the user specifies will be the number of elements read or written. The DIM will allow the user to forward space a file or a record and to backward space a file or a record. With later extensions to the I/O switch, a true record mode attachment could be supported by the DIM.

Buffer positioning for reading or writing will be handled by the I/O system seek calls. A user will be able to forward space or backspace a file or a record by issuing the appropriate call to ios_$seek. Similarly, issuing calls to ios_$tell will report the current buffer position before or after a read or write operation.

VI. Permitted I/O System Calls

The following I/O system calls will be recognized by the ANSI tape DIM. The first five entries will be implemented in the initial version of the DIM, the last five entries will be implemented in the second version of the DIM.

attach	
detach	
order	version 1
read	
write	
changemode	
seek	
tell	version 2
setsize	
getsize	

VII. Buffering

The DIM's main concern with buffering will be for the blocking and unblocking of files. One implementation goal for the DIM is to produce buffered I/O capacity where possible - but the first couple of versions of the DIM will perform just synchronous I/O. This will contribute to more rapid debugging and will allow attention to be devoted to implementing full blocking capability - including record spanning, before tackling the implementation of buffered I/O.

Blocking

Table I shows a list of the blocking formats that the final version of the DIM will accommodate. The terms V, VB and VBS are used as well as D, DB, and S to clarify which IBM tapes (in IBM terminology) will be acceptable to the DIM.

Before a user can read a file on a given input tape, the DIM will need to have information such as block size, blocking format, logical record length, logical record format and character code of the file. This information will be available in the SDB. It is recalled from the discussion of attachment, that this unblocking information is filled into the SDB either from information derived from HDR2 label processing (initial version), or from information supplied in the attach call.

RECFM	READ lrecl : nelem			WRITE lrecl : nelem		
	<	=	>	<	=	>
D	1	1	2	3	4	5
DB	1	1	2	3	4	5
V	1	1	2	3	4	5
VB	1	1	2	3	4	5
U	1	1	2	3	4	5
F	1	1	2	3	4	5A
FB	1	1	2	3	4	5A
VBS	1	1	2	3A	4A	5
S	1	1	2	3A	4A	5

Figure 4: Blocking Implementation. Legend for Record Formats and Legend for Interpretations follow (pp 42 - 43).

Table I

Legend for Record Formats (RECFM):

D ---- Variable length records recorded in ASCII with length in first word of the record (Record Descriptor Word).

DB --- Variable length records recorded in ASCII blocked in variable length blocks with the length of the block in the first word of each block (Block Descriptor Word or Block Prefix Word).

V ---- Variable length records recorded in EBCDIC. The length of the record is in the first word of the record (Record Descriptor Word, recorded in binary).

VB --- Variable length records recorded in EBCDIC blocked in variable length blocks. The length of the block is in the first word of each block (Block Descriptor Word, in binary).

U ---- Undefined length records recorded in ASCII or EBCDIC. An optional Block Prefix Word containing the length of the record (in ASCII as OS doesn't check length of U-format records) may appear at beginning of record.

F ---- Fixed length records recorded in ASCII or EBCDIC (possible padding).

FB --- Fixed length records in fixed length blocks (possible padding).

VBS -- Variable length records recorded in EBCDIC which span the block size. The logical record length is greater than the block size, so each block can contain one or more "segments" of one or more logical records. Each record "segment" begins with a Segment Descriptor Word which tells the length of the "segment" and whether more is to come or not.

S ---- ASCII spanned records of variable size (block prefix not required).

Note: The difference between S records, and VBS records is essentially that the record length and spanning information is in ASCII for S records and in binary for VBS records.

Table II

Legend for Interpretations:

1 -- The user asks for some number of elements which is less than or equal to the logical record length. The DIM's action is to put as much of one logical record into the user's buffer as nelem will allow. Nelemt is set equal to nelem. If nelem is less than the logical record length, a warning is returned in the status telling the user that less than a full logical record was given.

2 -- The user asks for some number of elements greater than the logical record size. The DIM's action is to fetch one logical record. nelemt is set equal to nelem.

3 -- The user has tried to write out a number of elements greater than one logical record. This is an error.

3A -- If the user has tried to write a number of elements greater than the current block size. The DIM will span the record as long as the user has set the spanning flag at attach time. The nelem is the logical record "segment" size and the DIM sets nelemt equal to nelem.

4 -- The user tries to write out a number of elements equal to one logical record. This is fine. The DIM sets nelemt equal to nelem.

4A -- The user has specified some number of elements which equals the logical record length. The DIM fills in one block and keeps filling in blocks until the logical record is exhausted. The final block is not written right away. The DIM waits for more input from the user so that logical records may be packed into blocks. The final block will be written out at detach time. Here, logical record length is provided by the DIM as a convenience to the user, so that he will not create a tape with logical record size greater than the interchange installation can handle. The ANSI limitation on logical record length is 999 reels of tape: few installations can handle that.

5 -- The number of elements the user specifies to write is less than the logical record length. The nelem elements are written and nelemt is set equal to nelem.

5A -- In the case of fixed length records and blocked fixed length records, the DIM will pad to fill out the logical record length and/or the block size to the designated length. In unblocked records, the record is padded if need be, with blanks (ASCII or EBCDIC spaces). In the blocked records, the block is padded with the ANSI standard padding character -- the circumflex.

When the blocking information has been made available to the DIM and the user issues a call to ios_$read referencing a given number of elements (nelem) in the current file, the DIM looks to its unblocking buffers for the next logical record of the current file. If the buffers are empty, a call is made to read in the next block from the tape and unblocking is done. When the input records are unblocked, they are translated into 9-bit ASCII and the length of the logical record is compared with the nelem argument to see if an error will arise. (Refer to Figure 4 and Table II.) Then the translated record is put into the user's buffer and the return character count (nelemt)is set.

Since the Multics I/O system presently only handles stream oriented I/O, a file in the DIM's unpacked buffers will be dealt with as a stream of characters which the user can access by supplying a length and an offset of his choice. At a later time when record oriented I/O becomes an I/O system option, the notion of unpacked records will be built into the ANSI DIM.

Each time a block has been read by the DCM, the DIM updates a current block counter maintained in the SDB. When it appears that a complete file has been read in, the contents of this block counter is checked against the block count from the file's first trailer label to verify that all blocks of the file have been read. Abnormal termination results from a block count discrepancy.

For output tapes — when the user issues a call to ios_$write to do blocked output — the DIM begins to fill its blocking buffers. The initial version will have two buffers, each "block size" in length. As a buffer is full, the DCM is called to put out the block onto tape.

All output blocks from this DIM will be compatible with IBM system produced blocks in that Block Description Words and Block Prefixes will be present where they are present in IBM produced blocks and absent otherwise.

When a call to detach has been made, the output file is assumed completed. The final block is written out and the block count for the file entered in the first file-trailer label.

The ANSI specified acceptable range of block sizes for interchange tapes is a minimum of 18 characters and a maximum of 2048 characters. The Multics DIM will support these outer limits, although larger block sizes will be handled upon agreement of interchange parties.

VIII. Order Calls

At least two types of order calls will be needed in the ANSI tape facility. It will be necessary to have an order entry in the DIM which will look at the information in the SDB and from that compose system tape labels. The following requests will be recognized by the system label composing order entry:

VOL1, HDR1, HDR2, EOV1, EOV2, EOF1, and EOF2.

The second type of order call that will be implemented will be to read and write user file labels. This entry will recognize two requests: read and write. The read request will return a pointer to the string containing the user label read from the tape. The write request will expect to be accompanied by a pointer to the user composed file label.

IX. Detachment

As was pointed out in the section on attachment, the typical way to close a file--to detach a stream, is by issuing a call to ios_$detach which results in the DIM making a call in to ring 1 to rewind and take down the tape. The drive is then freed and the tape blocking structure is also freed; the SDB is terminated and the stream is detached. It was noted in the previous discussion of multi-volume file attachments, that it is not desirable to always have to unload or even rewind a volume for a file that is to be detached. A more efficient design feature would be to allow a volume to remain mounted for the duration of the user's process, even after a file on the volume has been opened and then closed. The IBM user has the disposition parameter in the DD statement to specify the nature of data set termination. The Multics user will have the third argument to ios_$detach, known as the disposal argument, to specify the extent to which a device is to be terminated.

The keywords acceptable to specify Multics ANSI stream termination have been taken from the IBM disposition keywords, with every effort made to maintain the same meanings.

Specifying the File Disposal

There are four modes that can be specified on terminating an ANSI stream. They enable the user to

- rewind and unload a volume------------------DELETE
- rewind but not unload a volume--------------KEEP
- leave the volume mounted and positioned at the logical end of the file just processed--LEAVE
- position at logical beginning of the file just processed------------------------------REREAD

The DELETE disposal deletes the file's SDB and frees the tape drive when the file is closed. All system maintained information about the file is released.

The KEEP disposal does not unload the volume when a file is closed but it positions the volume at its logical beginning (rewinds). The REREAD disposal positions the tape at the logical beginning of the file just read or written. The volume remains mounted each time a user accesses and closes a file in it, as long as KEEP, LEAVE or REREAD are specified as disposals. When the user finishes with the last file that he wants to access on the given volume, he specifies that the tape be taken down by using DELETE as the disposal for closing the file. If the volume is not taken down before the user's process is terminated, it will be unloaded at process termination. At this time the information maintained in the inactive SDB will be lost.

The LEAVE disposal does not rewind or unload a volume, but specifies that the disposal functions are to be passed over. The tape is positioned at the logical end of the last file to be processed (read or written). The SDB is maintained as inactive. Upon process termination, the information in the SDB made inactive with the LEAVE disposal, will be copied into the BVDS.

X. Error Handling

The only error reporting the DIM will perform will be to return status codes; it will not print out error messages. Standard Multics error codes or tape hardware status will be returned in the first word of the status string--an error being indicated by the non-zero contents of the first status word. Hardware statuses are recognized as negative integers; a list follows in Table III.

The standard Multics error codes already installed that will be returned by the DIM are included in Table IV. Proposed Multics Standard error codes follow in Table V.

Table III

	Major Status	Substatus
Peripheral Subsystem Ready	000000	
Write Protected		000XX1
Positioned on Leader		000X1X
Nine Track Tape Unit		0001XX
Device Busy	000001	
Device Attention	000010	
Write Inhibit		0XXXX1
No Such Tape Unit		0XXX1X
Tape Unit in Standby		0XX1XX
Tape Unit Check		0X1XXX
Blank Tape on Write		01XXXX
Device Data Alert	000011	
Transfer Timing Alert		000001
Blank Tape on Read		XXXX1X
Transmission Parity Alert		XXX1XX
Lateral Parity Alert		XX1XXX
Longitudinal Parity Alert		X1XXXX
End of Tape (EOT) Mark		1XXXXX
Bit During Erase		XXXX11
End of File	000100	
EOF Mark (Seven Track)		001111
EOF Mark (Nine Track)		011100
Data Alert Condition		111111
Single Character Record		XXXXXX
Command Reject	000101	
Invalid Operation Code		XXXXX1
Invalid Device Code		XXXX1X
Parity on I/O		XXX1XX
Positioned on Leader		XX1XXX
Read After Write		X1XXXX
Nine Track Error		1XXXXX
Program Load Termination	000111	
Peripheral Subsystem Busy	001000	

Table IV

error_table_$-noarg
error_table_$not_attached
error_table_$ionmat
error_table_$badmod (attach)
error_table_$invalid_write
error_table_$invalid_read
error_table_$no_device
error_table_$undefined_order_request

Installed status codes returned by the DIM.

Table V

error_table_$not_ANSI-interchange_tape

error_table_$bad_device_control_character

error_table_$bad_ANSI_volume_label

error_table_no_blocking_info

error_table_$invalid_character_code

error_table_$incorrect_ANSI_filename

error_table_$incorrect_ANSI_file_seq_number

error_table_$file_already_open

error_table_$file_not_found

error_table_$record_length_exceeded

error_table_$block_length_exceed

error_table_$ANSI_block_too_short

error_table_$invalid_tape_access

error_table_$unsupported_record_format

error_table_$ANSI_filename_missing

error_table_$ANSI_filenumber_missing

error_table_$end_reel_on ANSI_HDR

Appendix A

ANSI Label Declarations

```
/* "VOL1" -- ANSI required interchange tape label:
volume header label.  This label is required to be the
first block in the volume, and cannot appear any other
place in the volume  */


dcl 1 vol1_label based,
        2 label_identifier char (3),            /* must be "VOL" */
        2 label_number char (1),                /* must be "1" */
        2 volume_identifier char (6),           /*  volume serial no. */
        2 accessibility char (1),               /* will record as a space */
        2 reserved1 char (26),                  /* recorded as spaces */
        2 owner_identification char (14),       /* user name: personid */
        2 reserved2 char (28),                  /* recorded as spaces */
        2 label_standard_version char (1);      /* "3" means labels and data
                                                   formats on this volume conform
                                                   to ANSI standard version 3 */
```

```
/*  "HDR1" -- ANSI required interchange tape label:
    first file-header label.  Every file must be preceded by
    the first file-header label.  Thsi label contains system
    information and device-dependent data relating to the file. */


dcl 1 hdr1_label based,
      2 label_identifier char (3),              /* must be "HDR" */
      2 label_number char (1),                  /* must be "1" */
      2 file_identifier char (17),              /* file name */
      2 file-set_identifier char (6),           /* vol serial number
                                                   of tape containing file */
      2 file_section_number char (4),           /* rank of vol in multi-
                                                   volume group */
      2 file_sequence_number char (4),          /* rel. position within
                                                   multi-file group */
      2 generation_number char (4),             /* (optional) */
      2 generation_version_number char (2),     /* (optional) */
      2 creation_date char (6),                 /* bYYDDD */
      2 expiration_date char (6),               /* bYYDDD */
      2 accessibility char (1),                 /* security status-- ignored */
      2 block_count char (6),                   /* must be "000000" in HDR1 */
      2 system_code char (13),                  /* identifies system that
                                                   created file */
      2 reserved char (7);                      /* recorded as spaces */
```

```
/*  "HDR2" -- ANSI optional interchange tape label: second
    file-header label.  */


dcl 1 hdr2_label based,                   /* second file-header label */
      2 label_identifier char (3),        /* must be "HDR" */
      2 label_number char (1),            /* must be "2" */
      2 record_format char (1),           /* "F", "D", "U", or "S" */
      2 block_length char (5),            /* 18 to 2048 = chrs in blk */
      2 record_length char (5),           /* logical record length (max) */
      2 reserved_for_system_use,          /* unique to each system */
        3 density char (1),               /* "2" = 9-trk, 800 CPI NRZI */
        3 file_position char (1),         /* "0" = no vol switch occurred */
                                          /* "1" = switch has occurred */
        3 job_step_id char .(17),         /* recorded as spaces */
        3 recording_technique char (2), /* spaces = 9-trk 800 CPI */
        3 control_characters char (1),  /* "A" = ASCII,"b" = no ctl chrs */
        3 buffer_alignment_block char (1),
                                          /* recorded as a space */
        3 block_attribute char (1)        /* "B" = blocked, "b" = unblocked*/
        3 file_character_code char .)2), /*"EB", "IA", "9A", "8A", "BY" */
        3 pad1 char (9),                  /* for alignment .*/
      2 buffer_offset char ( 2),          /* (optional) lth of blk prefix */
      2 reserved1 char ( 28);             /* recorded as spaces */
```

/* "EOF1" -- ANSI required interchange tape label:
first end-of-file-trailer label. The last block of every
file must be followed by an end-of-file-trailer label. */

```
dcl 1 eof1_label based,                        /* 1st trailer label */
      2 label_identifier char (3),             /* must be "EOF" */
      2 label_number char (1),                 /* must be "1" */
      2 file_identifier char (17),             /* file name */
      2 file-set_identifier char (6),          /* vol serial number
                                                  of tape containing file */
      2 file_section_number char (4),          /* rank of vol in multi-
                                                  volume group */
      2 file_sequence_number char (4),         /* rel. position within
                                                  multi-file group */
      2 generation_number char (4),            /* (optional) */
      2 generation_version_number char (2),    /* (optional) */
      2 creation_date char (6),                /* ƀYYDDD */
      2 expiration_date char (6),              /* ƀYYDDD */
      2 accessibility char (1),                /* security status-- ignored */
      2 block_count char (6),                  /* must be "000000" in HDR1 */
      2 system_code char (13),                 /* identifies system that
                                                  created file */
      2 reserved char (7);                     /* recorded as spaces */
```

/* "EOF2" -- ANSI optional interchange tape label: second end-of-file-trailer label. The label identifier here is "EOF" and the label number is "2". The remaining fields contain the same information as the corresponding fields of the second file header label (all contents are optional). */

```
dcl eof2_label based,                     /* second trailer label */
      2 label_identifier char (3),        /* must be "EOF" */
      2 label_number char (1),            /* must be "2" */
      2 record_format char (1),           /* "F", "D", "U", or "S" */
      2 block_length char (5),            /* 18 to 2048 = chrs in blk */
      2 record_length char (5),           /* logical record length (max) */
      2 reserved_for_system_use,          /* unique to each system */
        3 density char (1),               /* "2" = 9-trk, 800 CPI NRZI */
        3 file_position char (1),         /* "0" = no vol switch occurred */
                                          /* "1" = switch has occurred */

        3 job_step_id char (17),          /* recorded as spaces */
        3 recording_technique char (2),   /* spaces = 9-trk 800 CPI */
        3 control_characters char (1),    /* "A" = ASCII,"b" = no ctl chrs */
        3 buffer_alignment_block char (1),
                                          /* recorded as a space */

        3 block_attribute char (1)        /* "B" = blocked, "b" = unblocked*/
        3 file_character_code char (2),   /*"EB", "IA", "9A", "8A", "BY" */
        3 pad1 char (9),                  /* for alignment */
      2 buffer_offset char ( 2),          /* (optional) lth of blk prefix */
      2 reserved1 char ( 28);             /* recorded as spaces */
```

/* "EOV1" -- ANSI required interchange tape label: first end of volume-trailer label. Whenever a volume ends within a file, the last block of the file in that volume is to be followed by an end-of-volume-trailer label. */

```
dcl 1 eov1_label based,                          /* trailer label */
      2 label_identifier char (3),               /* must be "HDR" */
      2 label_number char (1),                   /* must be "1" */
      2 file_identifier char (17),               /* file name */
      2 file-set_identifier char (6),            /* vol serial number
                                                    of tape containing file */
      2 file_section_number char (4),            /* rank of vol in multi-
                                                    volume group */
      2 file_sequence_number char (4),           /* rel. position within
                                                    multi-file group */
      2 generation_number char (4),              /* (optional) */
      2 generation_version_number char (2),      /* (optional) */
      2 creation_date char (6),                  /* ƀYYDDD */
      2 expiration_date char (6),                /* ƀYYDDD */
      2 accessibility char (1),                  /* security status-- ignored */
      2 block_count char (6),                    /* must be "000000" in HDR1 */
      2 system_code char (13),                   /* identifies system that
                                                    created file */
      2 reserved char (7);                       /* recorded as spaces */
```

/* "EOV2" -- ANSI optional interchange tape label: second end-of-volume-trailer label. Whenever a volume ends within a file, the last block of the file in that volume is to be followed by an end of volume lable, an EOV1; EOV2 label optionally follows an EOV1. */

```
dcl eov2_label based,                        /* second trailer label */
      2 label_identifier char (3),           /* must be "EOV" */
      2 label_number char (1),               /* must be "2" */
      2 record_format char (1),              /* "F", "D", "U", or "S" */
      2 block_length char (5),               /* 18 to 2048 = chrs in blk */
      2 record_length char (5),              /* logical record length (max) */
      2 reserved_for_system_use,             /* unique to each system */
        3 density char (1),                  /* "2" = 9-trk, 800 CPI NRZI */
        3 file_position char (1),            /* "0" = no vol switch occurred */
                                             /* "1" = switch has occurred */
        3 job_step_id char (17),             /* recorded as spaces */
        3 recording_technique char (2),      /* spaces = 9-trk 800 CPI */
        3 control_characters char (1),       /* "A" = ASCII,"b" = no ctl chrs */
        3 buffer_alignment_block char (1),
                                             /* recorded as a space */
        3 block_attribute char (1)           /* "B" = blocked, "b" = unblocked*/
        3 next_vol_id char (6),              /* reel id of next vol in set */
        3 pad1 char (5),                     /* for alignment */
      2 buffer_offset char ( 2),             /* (optional) lth of blk prefix */
      2 reserved1 char ( 28);                /* recorded as spaces */
```

Appendix B

Blocking Information

```
/*  Blocking Information  */

dcl 1 ans_block aligned based,
      2 tsegp ptr,                      /* to DCM-DSM tseg */
      2 cc bit (3),                     /* character code used */
      2 spanned bit (1),                /* "1" if spanned records */
      2 pad bit (32),                   /* alignment */
      2 blksize fixed bin,              /* length of block in chrs */
      2 lrecl fixed bin,                /* lrecl in chrs */
      2 blkp fixed bin,                 /* length of block prefix */
      2 pad_chrs,                       /* circumflex */
        3 pad1 char (1),
        3 pad3 char (3),
      2 bp (2) ptr,                     /* buffer pointers */
      2 buff_offset fixed bin,          /* current buffer offset */
      2 blngth fixed bin,               /* current blocking buffer lnth */
      2 blkct fixed bin;                /* current block count */
```

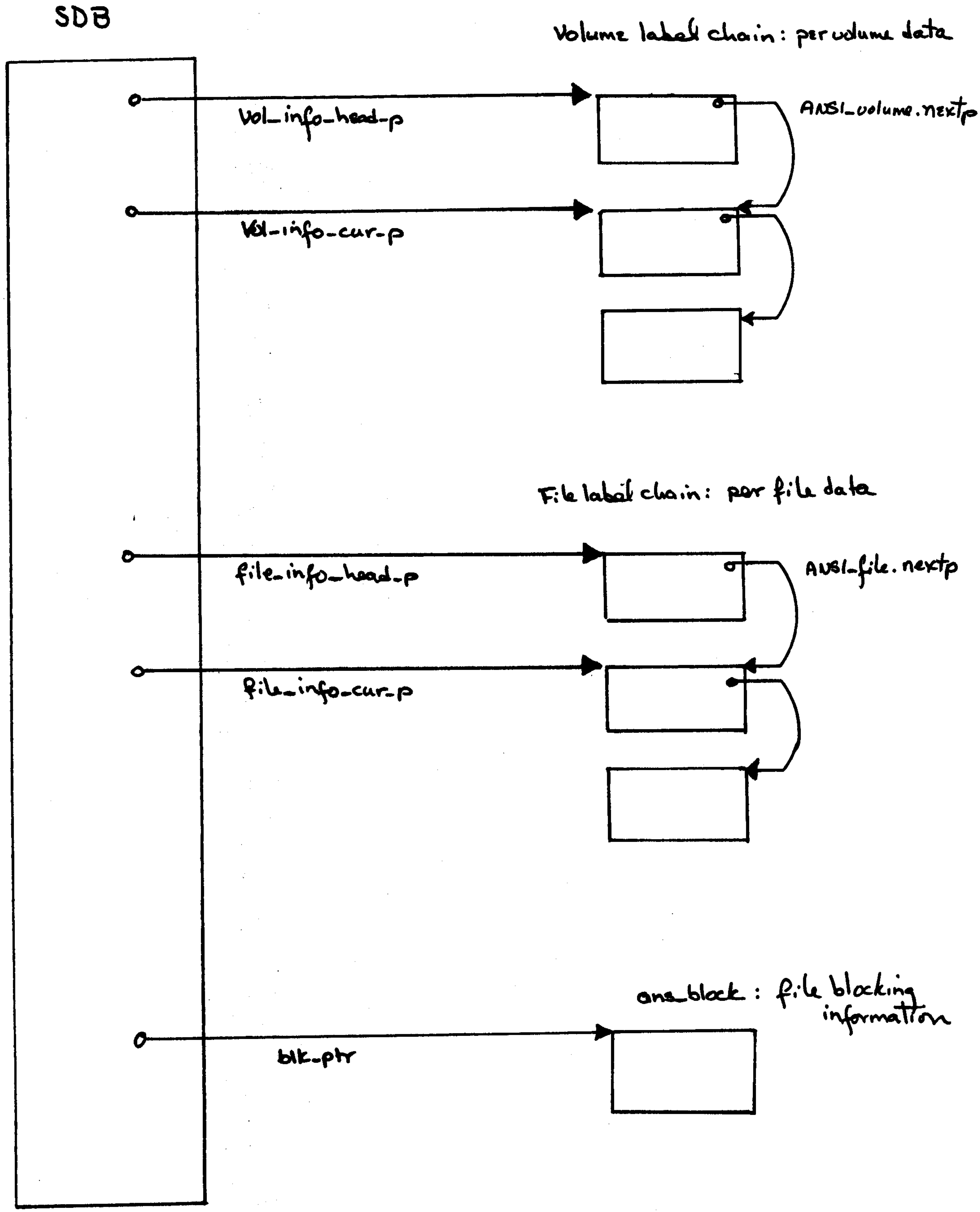
Appendix C

Volume and File Label Chain Declarations

```
/*    Volume label chain  */

dcl 1 ANSI_volume_chain based,
      2 nextp ptr,                        /* ptr to next chain entry */
      2 vol1_lbl char (80) aligned,       /* vol lbl translated to
                                             9-bit ascii (internal) */
      2 volid char (6) aligned,           /* vol serial number */
      2 next_entry char (4) aligned;      /* begin next chain entry */


/* File label chain  */

dcl 1 ANSI_file_chain based,
      2 nextp ptr,                        /* ptr to next entry on chain */
      2 filename char (17) aligned,       /* name of file */
      2 filenumber fixed bin,             /* file sequence number */
      2 section_number fixed bin,         /* tells current volume of
                                             multi-volume file */
      2 last_file_switch fixed bin,       /* set to "1" if last on vol */
      2 hdr_lbl (2) char (80) aligned,    /* the header labels, translated */
      2 next_link char (4);               /* the next entry */
```

Appendix D

Data Structure for a Stream Attachment



Data Structure for a Stream Attachment

Appendix E

Enhancements to the DIM

Enhancements to the Multics ANSI DIM

- Provision to read and write 7-track ASCII tapes.
- Provision to read DOS 360 labels.
- Provision to erase tapes.
- Provision to wind tapes back "n" records or files.
- Provision to wind tape forward "n" records or files.