† Woolf, Bill – Mathematical Reviews, American Mathematical Society
* Wu, Sheau-Huei – Datalogics, Inc.
Wulff, Robert – QUBIX
* Yugin, John – Scan Laser Printing Ltd.
† Zabala, Ignacio – Department of Computer Science, Stanford University

* * * * * * * * * * *

AN INFORMATION INTERCHANGE FORMAT FOR TEX FILES

Pierre A. MacKay
University of Washington

In an earlier issue of this journal, there was reported a proposal made by Patrick Milligan for an information interchange standard for TEX files. I remember that there were several very strong points in that proposal, but I do not have it before me now and I will probably be repeating some parts of it unconsciously. I do remember noting at the time, however, that it was in part a proposal for a tape file information interchange standard and yet made no reference to *American National Standard X3.27-1978*, "magnetic tape labels for information interchange." Since it was clear at the July 1982 TUG meeting that the problem of mutually compatible tape file formats is still very much with us, I would like to refine Patrick Milligan's suggestions by proposing that, rather than attempting to define our own unique information interchange standard, we adopt and promote the use of the ANSI standard, which has already been adopted as a Federal Information Processing Standard (see FIPS publication No. 79. 1980, October 17. U. S. Department of Commerce. National Bureau of Standards).

The name of this standard is perhaps a bit misleading in that it might suggest a concern with only the text of ANSI standard tape labels—fixed-length 80-character records containing ASCII decimal numeric and upper-case alphabetic characters. At higher levels of implementation, however, the use of these labels imposes a certain discipline in file and record format, with the result that a reference to the upper levels of implementation is, in effect, a sufficient description of a standardized file format for character files and requires little more than the agreement to convert all binary files into BigEndian hexadecimal character notation to serve as an all-purpose information interchange convention for users of TEX.

The various levels of implementation of the magnetic tape label standard are described in *ANSI X3.27-1978*, Appendix A: "Levels of Systems," and most particularly in section A3: "Distinguishing characteristics of levels of labelling." This section is meant to serve as a guide for the thorough integration of tape label processing into the basic operating system, but it can also serve as the outline for a user-level utility program if nothing better is possible. I confess to a certain missionary zeal to convert a wider range of installations toward the provision of level 3 capacities as part of the standard operating system, in the manner, for instance, of the VAX/VMS operating system, where Systems level 3 label processing is the default for all character files. I am sadly aware, however, that this conversion will take time, and that many users will have to bargain for utility programs to supplement the present inadequacies of tape label processing as they are found on most systems. It would be no small service to computing if the TEX Users Group were to contribute to the wider acceptance and use of the most effective ANSI standards. (Incidentally, the ISO standard for magnetic tape labels is virtually identical with the ANSI standard.)

For a full understanding of the content of all labels used in a Systems level 3 tape label processing utility there is no substitute for a reference to the ANSI standard itself, which can be purchased (prepaid only, and not exactly cheap) from the American National Standards Institute, 1430 Broadway, New York, N.Y. 10018. The details of special interest for TEX users are that Systems level 3 should provide

File formats:	Single file single volume, Single file multivolume, Multifile single volume, Multifile multivolume.
Labels:	VOL1 HDR1 HDR2 EOV1 EOV2 EOF1 EOF2 (Full analysis and decoding of all required fields.)
Record formats:	Fixed-length or variable-length records (A prefixed fixed-length character count field is assumed for all variable-length records. Special terminator codes are never used.)

Any Systems level 3 operation ought also to allow for the inclusion in the prescribed order of user volume labels (UVL1 through UVL9) and user header labels (HDR3 through HDR9) together with the answering EOV and EOF labels. The standard does not require that anything be done with such labels, but it is highly desirable that a tape label processing system be able to read and bypass them. A truly courteous operating system will provide a buffer from which the user can retrieve information contained in these extra-standard labels.

The content of VOL1 and HDR1 labels is pretty generally known. They provide fields for owner I.D., File I.D., File Creation date, Version number, System I.D. and a few other more esoteric matters. Two fields are best avoided or left to a well-chosen system default. A bad setting of the expiration date field can be rather a nuisance on a finicky system, and setting any of the protection fields can prevent you from reading your own tapes, let alone anyone else's. (As an example of an unfortunate system default, the TOPS-20 tape processing utility at the University of Washington sets a Volume Protection code, which inhibits transfer of files from the DEC-20 to the VAX. I have not yet discovered whether that setting can be avoided.)

The most interesting label for our purposes is the HDR2 label, since this provides a properly designed tape label processing system with all the necessary information for deblocking a tape. Among these fields is the record type field, which contains (level 3) either an ASCII upper-case 'F' for fixed-length records, or an ASCII upper-case 'D' for variable-length records. A D-type record is preceded by a four-character count field indicating the full length of that record in eight-bit bytes or tape-frames. The count is expressed in ASCII decimal digits, right-justified, and includes the length of the count field itself. Thus, an 80-character card-image may appear on tape as an 84-byte D-type record, with the first 4 bytes containing the ASCII characters '0084'. (Since the VAX/VMS operating system makes the convenient assumption that all character files are to be recorded as D-type records, a file of card-images often appears in this form. You have to specify fixed-length records to avoid the slight overhead cost of the decimal count field.)

In addition to the record type field, there is the maximum record length field and the maximum block length field, which together allow the operating system to set up its deblocking buffers efficiently. When all these fields are properly filled, there is no need to send along the usual sheaf of papers describing the tape's deblocking factors with each tape file. All the required information is contained in the labels, and a simple reference to ANSI X3.27-1978, Systems level 3, will indicate that this is so. We will, of course, need to set some reasonable limit on both block length and record length. I should think that a 2048-byte tape blocking buffer ought to be within the capacity of all operating systems that can read 9-track tape at all; it results in a quite efficient use of the tape, and conforms with ISO practice. The VAX/VMS operating system limits character records to a maximum length of 255, which, in a D-type record is 259, including the character count. This ought probably to be adopted as an absolute maximum, and perhaps a smaller maximum record length would prove more convenient on some systems. At any rate, we can easily come to some agreement about this and then get to work convincing systems programmers, with the authoritative sounding initials FIPS behind us.

The special problem that Patrick Milligan addressed was the treatment of binary files, and for these I would strongly support the use of hexadecimal coding. Communications protocols tend to take an arbitrary view of the "parity bit" in eight bit transmissions, and they can do weird things with such special characters as ASCII NUL, with or without parity. But all protocols that we are likely to be dealing with will allow the transmission of the 48-character ASCII subset which includes the ten decimal digits and the uppercase alphabetics.

I have been using BigEndian hexadecimal coding for six months now. It may be twice as slow as uncoded binaries, but it works, and it works more consistently than any alternative. For my own purpose I have found it useful to establish a very strict format for binaries. I use an 80-character fixed-length record which may be of particular interest to TEX-on-IBM users. Each 32-bit BigEndian quantity occupies a 10-column card-image field, left justified, with two trailing columns of fill. The fill columns might even be used for a simple check-sum, but I have never yet had a missed-bit error that would make this seem necessary. This is a convention which works as well over a DECNet as on magnetic tape. By choosing the time of day rather carefully—usually the hours between 2:00 A.M. and 6:00 A.M.—I have been able to send the entire DVI file resulting from the application of TEX to WEAVE.TEX across a DECNet from a 36-bit DEC 2060 to a 32-bit VAX. There are undoubtedly faster ways to achieve the same end, but I suspect that the TEX Users Group will be best served by adopting the slow but sure protocols of hexadecimal coding as the basic information interchange format for DVI, TFM and Font Raster files.