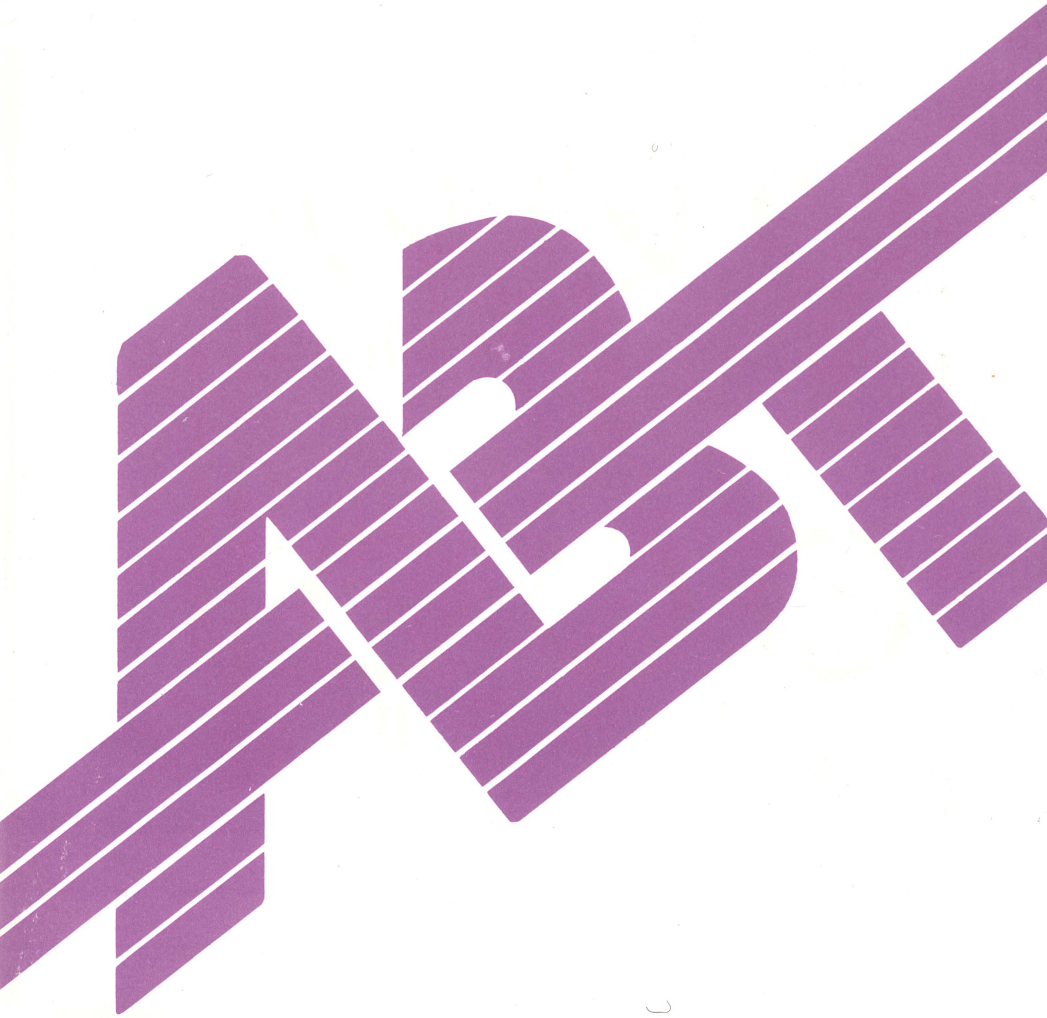
Pascal Tools I

Apple II*

Installation & Operating Instructions



Pascal Tools I
Apple II*
Installation & Operating Instructions

ADVANCED BUSINESS TECHNOLOGY, INC.

12333 Saratoga Sunnyvale Road, Saratoga, California 95070 408/446-2013

TABLE OF CONTENTS

PASCAL TOOLS I
A Tool Kit for the Serious Pascal Programmer

from
Advanced Business Technology, Inc.
and
Silicon Valley Software, Inc.

General Information

Pascal Tools I is a package of various programming aids (tools) for professional programmers using Pascal on the Apple II* which has been made available by Advanced Business Technology, Inc. Any serious Apple Pascal user will find this tool kit indispensible for the handling of large programming and other projects. These tools have been developed over time by Silicon Valley Software, Inc., as an aid to their software development of production systems on the Apple II. The programs included in Pascal Tools I are primarily directed at the handling of text files (and, for some of the utilities, primarily Pascal program source files). A companion package is available, Pascal Tools II, which concentrates primarily on the handling of binary and code files. All of the utilities in Pascal Tools I and Pascal Tools II have been proven useful and rugged in a production environment.

*Apple II is a trademark of Apple Computer, Inc.

CONTENTS OF THE PASCAL TOOLS I DISKETTE

Your Pascal Tools I diskette contains the following executable files:

XREF.CODE—Cross Reference Generator,
DIFF.CODE—Textfile Version Compare,
SPLIT.CODE—Large Textfile Splitter,
CAT.CODE—Textfile Concatenator,
MAKETEXT.CODE—Datafile into Textfile Converter,
MAKEDATA.CODE—Textfile into Datafile Converter, and
PAGE.CODE—General Purpose Pagination Utility.

In addition to these files, the following additional files are also present:

PAGEMAIN.CODE,
PAGEUNIT.TEXT, and
PAGEUNIT.CODE.

The usage of these files is explained in the manual section which describes PAGE.

GENERAL OPERATING INSTRUCTIONS

Each tool is e(Xecuted under the Apple Pascal system. The Pascal Tools I diskette can be removed during the execution of each utility. The operation of each tool is prompt driven, with "help" available on every user input by typing ? followed by return. Each tool can be aborted at the time of any prompt by hitting the escape key followed by return. It should be possible to use these tools immediately without further reading in the manual, but in order to make full use of all of their capabilities, the user is referred to the complete description of each system in the remainder of this manual.

XREF
Pascal Cross Reference Generator

CAPABILITIES

A "cross reference" listing of a Pascal program is generated. An (optional) listing of the program is followed by a list of each identifier which appears in the program and the line numbers on which it appears. Reserved words and the contents of comments and string constants are not listed among the identifiers. Source files are "included" in the program in a manner similar to the way that the Apple Pascal compiler includes files. Pascal programs of unlimited size can be cross referenced (without the use of a scratch file). For very large source programs, XREF will make multiple passes over its input to complete the cross reference. Help is available to all of XREF's prompts by typing ? followed by the return key. XREF may be aborted by typing < esc > followed by the return key to any prompt.

OPERATION

XREF is operated by e(Xecuting XREF. XREF prompts

Source file [.TEXT] -

for the name of the program to be cross referenced. At this time, the Pascal Tools I diskette can be removed from the system without affecting the operation of XREF. XREF will automatically add the suffix .TEXT to the file name provided if not supplied by the user (only .TEXT files are valid input files to XREF). The user may cross reference any Pascal source file, whether the file is a complete program or not. Next, XREF prompts

Listing file [PRINTER:] -

The user must provide the name of the output file for the program listing and cross reference listing. If the user types just return, the listing is placed on the device PRINTER:.

OPTIONS

Options may be specified to XREF at any time that user input is requested. Options are set or reset with + or - followed by the appropriate option letter. At any time, the current value of the options is displayed when help (?) is requested. The following options are recognized by XREF:

L controls listing of the program (default + L).
P controls page breaks and headings (default + P).

The L option controls whether to generate the listing of the program (or just the cross reference listing). + L causes the program to be listed. The P option controls page breaks and headings. + P causes XREF to place page headers and page breaks assuming a 66 line page. If your printing device already provides page breaks, -P is recommended. XREF will attempt to start the cross reference section at a page boundary even if automatic pagination is off by issuing a ^L to the listing device after the program listing.

NOTES

XREF will "include" source files just like the Apple Pascal compiler which appear in comments of the form

(*$1 filename*)

The files must be .TEXT files. XREF provides the .TEXT suffice automatically if it is not specified by the user. XREF will not find the include directive if it follows other directives in the comment (even though the Pascal compiler does). Also, for best results, make sure that the end of comment which causes the include is immediately followed by the end of the line (with no following characters or blanks).

It is generally best (although not necessary) to use XREF on syntactically correct Pascal programs. XREF may detect errors in incorrect Pascal programs (such as non-terminated string constants), but in general ignores most syntax errors. Incorrectly terminating comments are very disruptive to the cross reference generator.

The entire source program to be cross referenced must be on line during the operation of XREF since on very large programs more than one pass over the input is made. Remember that the Pascal Tools I diskette does not need to be on line during the operation of XREF.

DIFF
Textfile Version Compare

CAPABILITIES

A line by line comparison of two text files is made and DIFF reports the fewest lines of differences encountered or that the files are are identical. If differing lines are found between the files, or if a block of lines appears in a different position in each of the files, the system automatically resynchronizes on later lines which are identical and within a moved block of lines to report further differences. Under option control, DIFF will or will not ignore initial blanks in each line for purposes of comparison.

This tool is ideal for determining the differences between versions of a file or what changes have been made since an older copy of the file was saved. Another common use for DIFF is to compare the text output from versions of a program while the program is under development or maintenance to see what changes, if any, resulted from modifications to the program.

Help is available to all of DIFF's prompts by typing ? followed by the return key. DIFF may be aborted by typing <esc> followed by the return key to any prompt.

OPERATION

DIFF is operated by e(Xecuting DIFF. DIFF prompts

Output file for differences [CONSOLE:] -

for the name of the file on which to place the differences detected. At this time, the Pascal Tools I diskette (and if desired the Pascal system diskette) can be removed from the system without affecting the operation of DIFF. If the user types just return, the output is placed on the device CONSOLE:

DIFF then repeatedly prompts for pairs of file names to be compared. The prompt for the first file is:

First input file [.TEXT] -

and the prompt for the file to be compared to this file is:

Second input file [.TEXT] -

DIFF will automatically add the suffix .TEXT to each of the file names provided if not supplied by the user (only .TEXT files are valid input files to DIFF). Typing just return to either prompt terminates the execution of DIFF normally. The two files to be compared must both be available on line to DIFF simultaneously, however, neither the Pascal Tools I diskette nor the system diskette need to be available. Remember to replace the Pascal system diskette into its drive before terminating DIFF if you have removed it.

OPTIONS

Options may be specified to DIFF at any time that user input is requested. Options are set or reset with + or - followed by the appropriate option letter. At any time, the current value of the options is displayed when help (?) is requested. The following options are recognized by DIFF:

- B controls whether initial blanks in lines are significant (default + B).
- V controls verbosity of output of DIFF (default + V).
- Q quiet control during scanning (default -Q).

The B option controls whether initial blanks in lines are significant in line comparisons. + B indicates that the initial blanks are significant. The V option controls whether the first and last lines of matching blocks of lines are produced in the output. These lines are produced by default. The Q option can be set to inhibit the printing of the "dots" during scanning of the files. Setting + Q makes DIFF run quietly.

NOTES

DIFF only operates on files in .TEXT format. It will accept initial blanks either in the compressed or uncompressed form identically. Thus, files which are different "binary" may DIFF as identical providing the text content is the same.

There are certain limitations on the size of files which can be given to DIFF. These limitations are:

=1000 lines
=64 disk blocks.

If larger files are to be compared, use SPLIT to create smaller files for comparison.

The first line in each file is designated line 0 for compatibility with XREF and the Apple Pascal Compiler.

Typing < esc-ret > to any prompt aborts DIFF. If DIFF is terminated in this manner, rather than by return to a file name prompt, it is considered an abnormal termination and the output file is not made permanent.

PAGE
General Purpose Pagination Utility

CAPABILITIES

PAGE copies a text file input to a designated output file or device. By using the various options available through PAGE, page breaks and headers, line numbering, lines per page, lines per page break, characters per line, page paper feeding, speed of copying, and a variety of other characteristics of the copying process can be controlled. The most common usage of PAGE is for printing with page headers on fan fold paper without printing on the cracks and for printing on single sheets of paper. PAGE is simple to use for simple tasks, but is capable of accomplishing sophisticated tasks if skillfully used.

Most of the power of PAGE is available interactively by directly e(Xecuting PAGE.CODE as provided on the Pascal Tools I diskette through option settings. Because of the number of options available, a means has also been provided for users to develop their own versions of PAGE with user specified defaults (see section below on special purpose configurations).

Help is available to all of PAGE's prompts by typing ? followed by the return key. PAGE may be aborted by typing <esc> followed by the return key to any prompt.

OPERATION

This section describes the operation of PAGE as configured on the Pascal Tools I diskette. The operation may vary somewhat depending on the user configuration generated.

PAGE is operated by e(Xecuting PAGE. PAGE prompts

Output file [PRINTER:] -

for the name of the destination file or device. If the user types just return, the file name within the square brackets becomes the destination (standard configuration PAGE specifies PRINTER: as the default output, but this can be changed through reconfiguration). At this time, the Pascal Tools I diskette may be removed from the system without affecting the operation of PAGE.

PAGE will then repeatedly prompt for input file names to copy to the designated output file. The prompt is

Input file [.TEXT] -

PAGE will automatically add the suffix .TEXT to the file name provided if not supplied by the user (only .TEXT files are valid input files to PAGE). Typing just return to the input prompt terminates PAGE.

OPTIONS

Options may be specified to PAGE at any time that user input is requested. Options are set or reset with + or - followed by the appropriate option letter followed in some cases, where indicated, by an integer. At any time, the current value of the options is displayed when help (?) is requested. In the standard configuration, the default options, and their values, are

- N controls whether to number lines (set to -N).
- R controls whether to reset line number for each input (set to -R).
- W controls whether to wait before printing each page (set to -W).
- D controls whether to provide time delay, see manual (set to -D).
- P controls page breaks and headings (set to +P).
- H controls whether header is printed (set to +H).
- F controls whether filename is printed in header (set to +F).
- X controls whether page number is printed in header and what the next page number is to be (set to +X1).
- T controls whether header is printed on top or bottom (set to +T).
- U controls whether system or user header is printer (set to -U).
- L lines per page (set to +66).
- C characters per line (set to +C80).
- B lines per page break (set to +B3).

The +N option causes lines to be numbered. If +R is also set, each file will start with line number 0. If -R is set, the line numbers continue in successive files where they left off in the previous file.

Setting +W causes PAGE to prompt to the CONSOLE: and wait for the user to hit return before printing each page. This is useful for printing devices which accept only single sheets of paper. (Options may be set from the page waiting prompt.)

The +D option will slow up the output of PAGE by any amount desired by the user. This is extremely useful if you have a printing device which can not accept output as quickly as the Apple puts it out and the hardware interfaces do not "handshake." Specifying +D33, for example, provides "33 units" of delay. The default amount of delay, specified just +D, provides "25 units" of delay which is just right for printing at 45 characters per second on a daisy wheel printer (Diablo, DTC, Xerox, etc.) connected at 1200 baud. Thus, the maximum printing throughput can be achieved as compared to the alternative means of preventing overrun by connecting the device at 300 baud and only getting 30 characters output per second. Users should experiment to find the maximum speed (minimum delay units) for their devices.

Setting -P makes the notion of page completely disappear to PAGE. If +H is set, PAGE prints a header line on the top (+T) or bottom (-T) of each page. The header contains the file name (+F) and the page number (+X1) by default, but either or both can be suppressed (-F, -X). If only one is present, it is centered on the page based on the number of characters per line set (+C80 for example sets the number of characters per line to 80). If the header is printed, but it is empty, it "prints" three blank lines. The next page number generated by PAGE can be set interactively. For example, ±X7 forces the next page to be numbered 7.

If the presupplied headers are not to the user's liking, the +U option causes PAGE to prompt for a user header. The prompt is

```
Input user header on next line
(!n = pagenumber, !f = filename, !! = !)
```

The next line becomes the header line, with the following substitutions: the page number is substituted for occurrences of !n, the file name of the file being printed is substituted for occurrences of !f and one exclamation point is substituted for occurrences of two exclamation points. The user header is placed at the beginning of the line. The user may achieve centering or any kind of justification by placing blanks in the user header provided to PAGE.

In the event that other than 66 lines per page are desired, this can be reset, for example to 96 lines, with + L96. The number of lines skipped per page break can be set with the B option. Remember that if + H is set, three lines are generated for the header (possibly all blank) so the initial defaults provide for 60 lines of data per page, 3 lines of header, and 3 lines as a page break.

Spend some time experimenting with the various options and your output devices.

SPECIAL PURPOSE CONFIGURATIONS

In general, users have certain specific devices and certain more or less fixed preferences for the styles of printed output they desire. Since there are so many options to PAGE, it would be tedious to have to set up the same (nonstandard) configuration time after time. Therefore, a configuration module for PAGE is provided in Pascal Tools I. The source file for the configuration module is PAGEUNIT.TEXT which corresponds to the standard system defaults. The object code associated with this configuration unit is PAGEUNIT.CODE. Using the system linker and PAGEMAIN.CODE, PAGE.CODE can be recreated by linking PAGEMAIN.CODE (as host) to PAGEUNIT.CODE (as a library file).

In order to produce special purpose configurations of PAGE, make source changes to PAGEUNIT.TEXT as directed by the comments contained in it, use the system Pascal compiler to create a new PAGEUNIT.CODE, and then link it as described above to PAGEMAIN. Note: you may only change the body of procedure setdefaults; any change to the interface section of this unit will cause the resulting linked configuration of PAGE to operate incorrectly.

NOTES

When producing special purpose configurations of PAGE, be sure to keep the original version. Also, it may be useful to have several special purpose configurations available for various devices or tasks.

CAT
Textfile Concatenator

CAPABILITIES

CAT accepts a series of .TEXT files as input and produces a .TEXT file as output which is the concatenation of the input files. The file produced by CAT has the same "edit environment" as the first file concatenated to the output. The files produced by CAT may be too large to edit, but can be made into more manageable size if needed using SPLIT. Help is available to all of CAT's prompts by typing ? followed by the return key. Entering <esc> followed by return to any prompt aborts CAT.

OPERATION

CAT is operated by e(Xecuting CAT. CAT prompts

Output file [.TEXT] -

for the file to be the destination of the concatenated source files. The Pascal Tools I diskette (as well as the Pascal system diskette if desired) may be removed from the system at this time without affecting the operation of CAT. The output must be a .TEXT file. CAT automatically adds the .TEXT suffix if it is not provided by the user. The output file must be "on line" when it is specified and remain in its place for the remainder of the operation of CAT.

CAT successively prompts for input file names until the user inputs an empty file name (by just typing < cr > to the prompt). The input files must be .TEXT files, with the suffix provided automatically by CAT if not provided by the user. Diskettes may be shuffled into drives before each input file is specified, but the output file must not be moved. In the event that the input file cannot be opened for reading, CAT prompts again for input without having altered the output file. If there is a write error on the output file (almost always caused by a full diskette), the operation of CAT is aborted. Entering < esc-ret > to abort CAT will prevent the output file from becoming permanent. Be sure to return the Pascal system diskette to its drive before terminating CAT if you have removed it.

NOTES

When creating a .TEXT file with the editor, an "extra" (hidden) < cr > is inserted at the end of the file. CAT does not remove this extra <cr> so that after concatenating files, it appears that an extra return is between files. This "extra" blank line can be avoided when using the editor by removing characters from the end of the file until the "end" position of the editor's cursor is at the end of the last nonempty line.

SPLIT
Large Textfile Splitter

CAPABILITIES

SPLIT accepts as input a .TEXT file, presumably too large to edit and divides it up into a series of .TEXT files of a size specified by the user. Each file produced by SPLIT has the same "edit environment" as the source file. Large .TEXT files arise from program output (including the output of XREF) and from the use of the concatenation tool, CAT. The files produced by SPLIT can be reassembled using CAT, which is the exact inverse operation of SPLIT. Help is available to all of SPLIT's prompts by typing ? followed by the return key. SPLIT may be aborted by typing < esc > followed by the return key to any prompt.

OPERATION

SPLIT is operated by e(Xecuting SPLIT. SPLIT prompts

Input file [.TEXT] -

for the file to be split. The Pascal Tools I diskette may be removed from the system (as well as the Pascal system diskette if desired) at this time without affecting the operation of SPLIT. Only .TEXT files are accepted. SPLIT automatically adds the .TEXT suffix if it is not provided by the user. The input file must be "on line" when it is specified and remain in its place for the remainder of the operation of SPLIT. Next SPLIT prompts

Number of blocks per output file [30] -

to determine the number of blocks per output file. Entering just < cr > sets the output size to 30 blocks which is a large, but editable file, for the Apple Pascal Editor. Any even file output file size from 4 to 50 blocks will be accepted (.TEXT files must be even in length and, since the format does not place data in the first 2 blocks, the minimum meaningful size is 4 blocks).

SPLIT successively prompts for output file names until the input file is exhausted. The output files must be .TEXT files, with the suffix provided automatically by SPLIT if not provided by the user. Diskettes may be shuffled into drives before each output file is specified, but the input file must not be moved. Be sure to return the Pascal system diskette to its drive if you have removed it before SPLIT terminates.

NOTES

When creating a .TEXT file with the editor, an "extra" (hidden) <cr> is inserted at the end of the file. SPLIT does not add this extra <cr> so that when editing a file which was produced by SPLIT (other than the very last file produced), a "jump end" edit command leaves the cursor at the right end of the final line.

MAKETEXT
Datafile to Textfile Converter

CAPABILITIES

MAKETEXT accepts as input a file consisting of "printable" characters, either in .TEXT format or not and produces a file which is in .TEXT format. Files must be in .TEXT format in order to be edited using the system text editor, to be compiled, to be SPLIT, and for various other utilities. If the input is already in .TEXT format, MAKETEXT attempts to compress the file by inserting the leading blank count abbreviation "DLE count" at the beginning of each line. If the input file is not in .TEXT format, MAKETEXT will interpret any character code as the end of line indicator, under user control. See the NOTES section for a definition of .TEXT format and datafile formats.

This tool is useful for transporting files to the Apple Pascal system which originated on other systems. Another (and probably more important) use for this tool is as an easy means of getting around the fact that Pascal I/O is extremely slow. Users may write their programs using "block write" with a straightforward file representation (instead of the rather messy textfile format) and convert the files to be "editable" only if necessary using MAKETEXT. Since files are printable, copyable, etc., as datafiles, this format may suffice in most situations as long as a means exists to get textfile format if necessary.

Help is available to all of MAKETEXT's prompts by typing ? followed by the return key. Entering < esc > followed by return to any prompt aborts MAKETEXT.

OPERATION

MAKETEXT is operated by e(Xecuting MAKETEXT. MAKETEXT prompts

Input file -

for the file to be converted to .TEXT format. The Pascal Tools I diskette may be removed from the system at this time without affecting the operation of MAKETEXT. The input file may be a .TEXT file or not, and no ending is provided automatically for the user. In the event that the file is not a .TEXT file, MAKETEXT prompts for the character code to be recognized as the end of line character code.

ASCII character code to be recognized as end of line [13] -

MAKETEXT expects the user to enter either a decimal integer in the range of 1 to 127 or to enter just a return, which provides the default value 13 which is the standard ASCII representation for "carriage return." Finally, MAKETEXT prompts

Output file [.TEXT] -

for the name of the output file which must end in .TEXT. MAKETEXT provides the .TEXT suffix automatically if it is not provided by the user.

NOTES

MAKETEXT determines whether its input file is in .TEXT format or not strictly by the file name provided by the user.

Files produced by MAKETEXT are in .TEXT format which has the following characteristics:

- The file name ends in .TEXT.
- The first two blocks of the file are set to zeros. This is interpreted by the editor as a reasonable standard edit environment.
- The file is an even number of blocks long.
- Each block pair, starting with blocks 2/3 consists of full lines followed by some number of trailing zeros which pad out the end of the block pair.
- A line consists of the DLE character (character code 16), followed by a byte indicating 32 + the number of initial blanks, followed by a string of (nonzero, non chr(13)) characters, followed by the return character (character code 13).
- The final block ends in an additional return character followed by at least one zero (to be compatible with the text editor).

If the input file has a name which ends in .TEXT, MAKETEXT ignores the first two input blocks of the file. The initial DLE and blank count on each line are interpreted if present but are not required. In all other ways, MAKETEXT presumes that the above format is followed.

In general, when writing a file from a Pascal program, it will be in .TEXT format providing its name ends in .TEXT. However, standard Pascal I/O does not put blank count abbreviation at the beginning of lines. Therefore, running MAKETEXT on .TEXT files produced by Pascal programs with many initial blanks may result in substantial file compression. Moreover, Pascal I/O does not prohibit the placement of zero characters in the middle of a file which will disrupt the operation of MAKETEXT (as well as the system text editor).

If the input file has a name which ends in any other character sequence than .TEXT, MAKETEXT will assume it is a datafile with the following characteristics:

- Data begins at the start of the file.
- Data is a stream of characters, not containing the zero character, terminated either by the end of the last block in the file or by a zero character in the last block of the file.
- Some character code specified by the user is interpreted as the end of line character.

When writing a file using Pascal I/O with a name ending in other than .TEXT, this is the format that is produced.

MAKEDATA
Textfile to Datafile Converter

CAPABILITIES

MAKEDATA accepts as input a file in .TEXT format and produces a file which is in a rather standard datafile format. Files must be in .TEXT format in order to be edited using the system text editor, to be compiled, to be SPLIT, and for various other utilities. MAKEDATA will generate any character code in the output datafile as the end of line indicator, under user control. See the NOTES section for a definition of .TEXT format and datafile formats.

This tool is useful for transporting files from the Apple Pascal system to other systems. In particular, if you wish to copy a file to a device which prefers to get the newline character (ASCII character 10), or some other character, as the end of line indicator, this tool makes that conversion convenient. Another (and probably more important) use for this tool is as an easy means of getting around the fact that Pascal I/O is extremely slow. Users may write their programs using "block read" with a straightforward file representation (instead of tne rather messy textfile format). If input data is to be prepared using the text editor, rather than by another program written by the user, MAKEDATA will convert the file to the form expected by the program.

Help is available to all of MAKEDATA's prompts by typing ? followed by the return key. Entering < esc > followed by return to any prompt aborts MAKEDATA.

OPERATION

MAKEDATA is operated by e(Xecuting MAKEDATA. MAKEDATA prompts

Input file [.TEXT] -

for the file to be converted from .TEXT format. The Pascal Tools I diskette may be removed from the system at this time without affecting the operation of MAKEDATA. The input file must be a .TEXT file, and MAKEDATA will provide the .TEXT suffix automatically if necessary for the user. Next MAKEDATA prompts

ASCII character code to be generated as end of line [13] -

MAKEDATA expects the user to enter either a decimal integer in the range of 1 to 127 or to enter just a return, which provides the default value 13 which is the standard ASCII representation for "carriage return." Finally, MAKEDATA prompts

Output file -

for the name of the output file which must not end in .TEXT.

NOTES

Files processed by MAKEDATA are in .TEXT format which has the following characteristics:

- —The file name ends in .TEXT.
- —The first two blocks of the file do not contain data and are ignored.
- —The file is an even number of blocks long.
- —Each block pair, starting with blocks 2/3 consists of full lines followed by some number of trailing zeros which pad out the end of the block pair.
- —A line consists of the DLE character (character code 16), followed by a byte indicating 32 + the number of initial blanks, followed by a string of (nonzero, non chr(13)) characters, followed by the return character (character code 13). The DLE and blank count are optional.
- —If the file has been produced by the text editor, or a system compatible with it, the final block pair will have an "extra" blank line at its end. MAKEDATA passes this blank line on to the datafile.

In general, when writing a file from a Pascal program, it will be in .TEXT format providing its name ends in .TEXT. Pascal I/O does not, however, prohibit the placement of zero characters in the middle of a file which will disrupt the operation of MAKEDATA (as well as the system text editor).

The output file produced by MAKEDATA will be a datafile with the following characteristics:

- Data begins at the start of the file.
- Data is a stream of characters, not containing the zero character, terminated either by the end of the last block in the file or by a zero character in the last block of the file.
- Some character code specified by the user is generated as the end of line character.

This format is similar, but not exactly the same as the file format generated when writing a file using Pascal I/O with a name ending in other than .TEXT. The only difference is in the end of file determination which in this format is indicated by a zero byte (or last block end). Pascal I/O maintains a byte count for the final block in the directory entry for the file to indicate end of file.

DISCLAIMER

OF ALL WARRANTIES AND LIABILITY

Advanced Business Technology, Inc., makes no warranties with respect to this manual or with the software described in this manual. It is sold or licensed "as is." The entire risk as to its performance is with the buyer. Should the programs prove defective following their purchase, the buyer (and not Advanced Business Technology, Inc., its distributor, or its retailer) assumes the entire cost of all necessary servicing, repair, or correction and any incidental or consequential damages resulting from any defect in the software, even if Advanced Business Technology, Inc., has been advised of the possibility of such damages.

Advanced Business Technology, Inc.
12333 Saratoga-Sunnyvale Road
Saratoga, California 95070

ADVANCED BUSINESS
TECHNOLOGY, INC.